applied computing

VCE UNITS 1 & 2

7E

SIGN IN & SUCCEED
NELSONNET ACCESS CODE INSIDE!

GARY BASS

SELINA DENNIS

THERESE KEANE

ANTHONY SULLIVAN

NELSON
A Cengage Company

Cengage Learning

COPYRIGHT NOTICE

Copyright in this work is owned by Cengage Learning Australia ("the work"). A condition of purchase of this electronic version of the work is that you agree to respect the copyright in the work, abide by the Copyright Act 1968 and specifically agree not to transfer, sell, assign, misuse, copy or transmit an electronic or other version of the work to any third party.

Please note: This product is accompanied by a licence (single user, network or adoption) governing the terms and conditions of its use.

This is a legal agreement between the you, (the "Customer") and Cengage Learning Australia Pty Limited (ABN 14 058 280 149) (the "Licensor") which provides the terms and conditions of this non-exclusive licence and the limited warranty for the Product. Use of the Product indicates an acknowledgement that the Customer has read and agreed to be bound by the terms and conditions of this Agreement. If you do not agree to these terms and conditions, return the Product to the place of purchase within 15 days of the date of purchase (with proof of purchase) for a full refund

1. **Licence Grant**
   You do not receive title to the Product. Copyright in the Product (which includes all images, photographs, video, animations, audio, music and text incorporated in the Product, including all of the accompanying printed material) is owned by the Licensor and/or its suppliers and is protected by Australian copyright laws. The Licensor grants you a non-exclusive licence to use the Product subject to the restrictions and terms set out in this Agreement.

2. **A Licence allows you to:**
   Use the Product on your computer. The Customer represents that they shall in no way place the Product in the public domain or in any way compromise our copyright in the Material. You agree to take reasonable steps to protect our copyright.

3. **You may not:**
   Alter, modify, translate, reverse engineer, decompile, or adapt the software or create derivative works based on the Product. Make further copies by any means technological, electronic, digital whatsoever without the written permission of the Licensor. Rent or transfer all or any part of your rights under this Agreement. Remove or alter any copyright or other proprietary notice or label attached to the software.

4. **Termination**
   Any failure to comply with the terms and conditions of this agreement will result in the automatic termination of this licence. Upon termination of this licence for any reason, the Customer must destroy or return to the Licensor all copies of the software and accompanying documentation.

5. **Warranties**
   To the extent permitted by law, the Licensor's liability for any breach of the warranty or any term implied by law into this licence is limited to the lowest cost of replacing the goods, acquiring equivalent goods or having the goods repaired.

applied computing

VCE UNITS 1 & 2

7E

GARY BASS

SELINA DENNIS

THERESE KEANE

ANTHONY SULLIVAN

NELSON
A Cengage Company

Applied Computing VCE Units 1&2
7th Edition
Gary Bass
Selina Dennis
Therese Keane
Anthony Sullivan
Mark Kelly
ISBN 9780170440806

Senior publisher: Eleanor Gregory
Project editor: Kathryn Coulehan
Editor: Nick Tapp
Proofreader: Kelly Robinson
Indexer: Max McMaster
Visual designer: James Steer
Cover design: Chris Starr, MakeWork
Text design: Leigh Ashforth, Watershed Art & Design
Permissions researcher: Debbie Gallagher
Production controller: Karen Young
Typeset by: DiacriTech

Any URLs contained in this publication were checked for currency during the production process. Note, however, that the publisher cannot vouch for the ongoing currency of URLs.

Acknowledgements
Extracts from the VCE Applied Computing Study Design (2020–2023), are reproduced by permission,© VCAA. VCE is a registered trademark of the VCAA. The VCAA does not endorse or make warranties regarding this study resource. Current VCE study Designs, past VCE exams and related content can be accessed directly at www.vcaa.vic.edu.au

PHOTOCOPYING OF BOOKS IS RESTRICTED UNDER LAW

© 2020 Cengage Learning Australia Pty Limited

Copyright Notice
This Work is copyright. No part of this Work may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without prior written permission of the Publisher. Except as permitted under the *Copyright Act 1968*, for example any fair dealing for the purposes of private study, research, criticism or review, subject to certain limitations. These limitations include: Restricting the copying to a maximum of one chapter or 10% of this book, whichever is greater; providing an appropriate notice and warning with the copies of the Work disseminated; taking all reasonable steps to limit access to these copies to people authorised to receive these copies; ensuring you hold the appropriate Licences issued by the Copyright Agency Limited ("CAL"), supply a remuneration notice to CAL and pay any required fees. For details of CAL licences and remuneration notices please contact CAL at Level 11, 66 Goulburn Street, Sydney NSW 2000, Tel: (02) 9394 7600, Fax: (02) 9394 7601
Email: info@copyright.com.au
Website: www.copyright.com.au

For product information and technology assistance,
in Australia call **1300 790 853**;
in New Zealand call **0800 449 725**

For permission to use material from this text or product, please email
aust.permissions@cengage.com

ISBN 978 0 17 044080 6

Cengage Learning Australia
Level 7, 80 Dorcas Street
South Melbourne, Victoria Australia 3205

Cengage Learning New Zealand
Unit 4B Rosedale Office Park
331 Rosedale Road, Albany, North Shore 0632, NZ

For learning solutions, visit **cengage.com.au**

Printed in Singapore by 1010 Printing International Limited.
1 2 3 4 5 6 7 23 22 21 20 19

Contents

Unit 1

9780170440806

Unit 2

9780170440806

Preface

This seventh edition of *Applied Computing VCE Units 1 & 2* incorporates the changes to the VCAA VCE Applied Computing Study Design that took effect from 2020.

This textbook looks at how individuals and organisations use, and can be affected by, information systems in their daily lives.

We believe that teachers and students require a text that focuses on the **Areas of Study** specified in the **Study Design**, and that presents information in a sequence that allows easy transition from theory into practical assessment tasks. We have therefore written this textbook so that a class can begin at Chapter 1 and work their way systematically through to the end. Students will encounter material relating to the **key knowledge** dot points for each **Outcome** before they reach the special section that describes the Outcome. The Study Design outlines key skills that indicate how the knowledge can be applied to produce a solution to an information problem. These Outcome preparation sections occur regularly throughout the textbook, and flag an appropriate point in the student's development for each Outcome to be completed. The authors have covered all key knowledge dot points for the Outcomes from the Applied Computing VCE Units 1 & 2 course.

Our approach has been to focus on the key knowledge required for each school-assessed Outcome, and to ensure that students are well prepared for these; however, there is considerable duplication in the Study Design relating to the knowledge required for many of the outcomes. We have found that, with an Outcomes approach, we are sometimes covering the same material several times. For example, knowledge of a problem-solving methodology is listed as key knowledge for many different outcomes. In these cases, we have tried to provide a general coverage in the first instance, and specifically apply the concept to a situation relevant to the related outcome on subsequent encounters.

The authors assume teachers will develop the required key skills with their students within the context of the key knowledge addressed in this textbook and the resources available to them.

We have incorporated a margin column in the text that provides additional information and reinforcement of key concepts. The margin column also includes activities related to the topics covered in the text, and consideration of issues relevant to the use of information systems.

Outcome features are included at several points in the book, indicating the nature of the tasks that students are to undertake in the completion of the school-assessed Outcome. The steps required to complete the Outcome are listed, together with advice and suggestions for approaching the task. The output and support material needed for submission are described. Sample tasks and further advice relating to the outcomes are available at https://www.nelsonnet.com.au.

The chapters are organised to present the optimum amount of information in the most effective manner. The text is presented in concise, clearly identified sections to guide students through the text. Each chapter is organised into the sections described on pages vii–viii.

9780170440806

About the authors

Gary Bass teaches VCE Applied Computing at Year 11 and Year 12 in an online course environment at Virtual School Victoria. Previously he has taught VCE Physics, as well as developing and delivering middle school ICT courses. Gary has presented at DLTV DigiCON and the annual IT teachers conference on many topics, including 'Pop-up Makerspace', 'Big Data requires huge analysis – data visualisation', 'AR + VR = Mixed reality', and 'Marshall McLuhan – Medium is the message'.

Selina Dennis is a Software Development and English Language teacher for the Department of Education and has been heavily involved in past and present Computing Study Designs. Selina has a Bachelor of Arts and Science in Computer Science and Linguistics from the University of Melbourne, and has a particular interest in Computational Linguistics. She spent several years in California in the computing industry as an Engineering Manager and Director of Engineering before entering teaching.

Associate Professor Therese Keane, Deputy Chair of the Department of Education at Swinburne University, has worked in a variety of school settings, where she has taught IT and led in K–12 education as the Director of ICT. Her passion and many achievements in the ICT in Education and Robotics space have been acknowledged by her peers in the numerous national and state awards she has received. Therese has presented numerous seminars and workshops for teachers involved in the teaching of IT. She has written several textbooks in all units of Senior IT in Victoria, and in VCE Information Technology since 1995. Therese's research interests include the use of technology in education, gender inequalities in STEM-based subjects, robotics in education and computers in schools for teaching and learning purposes. Therese is involved with the FIRST LEGO League as the Championship Tournament Director for Victoria, and is a lead mentor for the RoboCats – an all-girl robotics team that participates in the FIRST Robotics Competition.

Anthony Sullivan is a Curriculum and Learning Specialist at Monash College, where he is responsible for creating assessment and learning materials for accounting and computing subjects as part of the Monash University Foundation Year program. Before this, Anthony had been teaching business and computing subjects for more than 25 years. He has taught in both government and non-government settings in Australia and taught computing and information technology courses in schools in Asia and the United Kingdom. Anthony has also been a VCE Examination Assessor, a member of the committee that reviewed and wrote the previous study design for VCE Computing, and has written a range of commercial resources related to VCE Computing. He has presented at a number of conferences and professional development events and has also presented at student examination preparation sessions.

9780170440806

How to use this book

KEY KNOWLEDGE

The key knowledge from the VCAA Applied Computing VCE Units 1 & 2 Study Design that you will cover in each chapter is listed on the first page of each chapter. The list includes key knowledge specified in the Outcome related to the chapter.

FOR THE STUDENT

The first page of each chapter includes an overview of the chapter's contents so that you are aware of the material you will encounter.

FOR THE TEACHER

This section is for your teacher and outlines how the chapter fits into the overall study of Applied Computing, and indicates how the material relates to the completion of Outcomes.

CHAPTERS

The major learning material that you will encounter in the chapter is presented as text, photographs, screenshots and illustrations. The text describes in detail the theory associated with the stated outcomes of the Applied Computing VCE Units 1 & 2 Study Design in easy-to-understand language. The photographs show hardware, software and other objects that have been described in the text. Illustrations are used to demonstrate concepts that are more easily explained in this manner.

Throughout the chapter, glossary terms are highlighted in bold, light-blue text and you can find their definitions at the end of the chapter, in **Essential terms**.

MARGIN COLUMN

The margin column contains further explanations that support the main text, weblink icons, additional material outside the Study Design and cross-references to material covered elsewhere in the textbook. Issues relevant to Applied Computing that you can discuss with your classmates are also included in the form of 'Think about Applied Computing' boxes (right).

3.1

THINK ABOUT APPLIED COMPUTING

Project management tools are useful to find the perfect number of people needed on a task so it is finished as quickly as possible without anyone being idle. Use software to develop a Gantt chart to plan the baking of a cake. Assume you can use as many cooks as you want.

CHAPTER SUMMARY

The chapter summary at the end of each chapter is divided into two main parts to help you review each chapter.

Essential terms lists the glossary terms that have been highlighted throughout the chapter.

Important facts is a list of summaries, ideas, processes and statements relevant to the chapter, in the order in which they occur in the chapter.

9780170440806

TEST YOUR KNOWLEDGE

These are short-answer questions that are provided to help you when reviewing the chapter material. The questions are grouped, and identified with a section of the text, to allow your teacher to direct appropriate questions based on material covered in class. Teachers will be able to access answers to these questions at **https://www.nelsonnet.com.au**.

APPLY YOUR KNOWLEDGE

Each chapter concludes with a set of questions requiring you to demonstrate that you can apply the theory from the chapter to more complex questions. The style of questions reflects what you can expect in the end-of-year examination. Teachers will be able to access suggested responses to these applications at **https://www.nelsonnet.com.au**.

PREPARING FOR THE OUTCOMES

This section appears at points in the course where it is appropriate for you to complete an Outcome task. The information provided describes what you need to do in the Outcome, the suggested steps to be followed in the completion of the task and the material that needs to be submitted for assessment.

NELSONNET

The NelsonNet student website contains:

- multiple-choice quizzes for each chapter, mirroring the VCAA Unit 3 & 4 exam.
- additional material such as spreadsheets and infographics.

A weblink page is also provided for all weblinks that appear in the margins throughout the textbook. This is accessible at the student website at **https://nelsonnet.com.au.**

The NelsonNet teacher website is accessible only to teachers and it contains:

- answers for the **Test your knowledge** and **Apply your knowledge** questions in the book
- sample SACs
- chapter tests
- practice exams.

Please note that complimentary access to NelsonNet and the NelsonNetBook is only available to teachers who use the accompanying student textbook as a core educational resource in their classroom. Contact your sales representative for information about access codes and conditions.

9780170440806

Outcomes

OUTCOME	KEY KNOWLEDGE	LOCATION
Unit 1 Area of Study 1 Outcome 1	**Data analysis** On completion of this unit the student should be able to interpret teacher-provided solution requirements and designs, collect and manipulate data, analyse patterns and relationships, and develop data visualisations to present findings.	
Data and information	• types and purposes of qualitative and quantitative data	p. 3
	• characteristics of data and information	p. 5
	• sources, methods and techniques for acquiring and referencing primary and secondary data and information	p. 5
	• interpretation of information for communication and decision making	p. 11
	• factors affecting the quality of data and information, such as accuracy, bias, integrity, relevance and reliability	p. 12
	• characteristics of data types and data structures relevant to selected software tools	p. 15
	• procedures for the legal and ethical collection and use of data and information, such as using consent forms	p. 18
	• techniques for protecting data and information from misuse, such as de-identifying personal data and the use of physical and software security controls	p. 23
Approaches to problem solving	• structural characteristics of spreadsheets and databases, such as cells, fields, records and tables	p. 72
	• types and purposes of data visualisations suitable for educating, entertaining, informing and persuading audiences	p. 39
	• functional and non-functional requirements of solutions, constraints and scope	p. 59
	• design tools for representing the functionality and appearance of databases, spreadsheets and data visualisations, such as annotated diagrams and mock-ups	p. 63
	• formats and conventions suitable for databases, spreadsheets and data visualisations	p. 68
	• software functions and techniques for efficiently and effectively manipulating, validating and testing data to develop databases, spreadsheets and data visualisations	p. 77
Interactions and impacts	• Australian Privacy Principles relating to the acquisition, management and communication of data and information including non-identification of individuals (Principle 2), information only being held for its primary purpose (Principle 6) and the security measures used to protect personal information (Principle 11)	p. 28
	• ethical issues arising from the acquisition, storage and use of data and information	p. 31
Key skills	• acquire and reference data and information from primary and secondary sources, taking into account legal and ethical considerations	
	• analyse the selected data, and discuss the relationships and patterns identified	
	• interpret solution requirements, constraints and scope	
	• interpret designs using appropriate design tools to represent the functionality and appearance of databases, spreadsheets and data visualisations	
	• use software, and select and apply functions, formats, conventions, data validation and testing techniques to efficiently manipulate data and create data visualisations	
	• compare and interpret data visualisations	

9780170440806

OUTCOME	KEY KNOWLEDGE	LOCATION
Unit 1 Area of Study 2 Outcome 2	**Programming** On completion of this unit the student should be able to interpret teacher-provided solution requirements to design, develop and evaluate a software solution using a programming language.	
Digital systems	• functions and capabilities of key hardware and software components of digital systems required for processing, storing and communicating data and information	p. 110
Data and information	• characteristics of data types	p. 124
	• types of data structures	p. 126
Approaches to problem solving	• features of functional and non-functional solution requirements, constraints and scope	p. 120
	• design tools for representing the functionality and appearance of solution designs such as data dictionaries, mock-ups and pseudocode	p. 134
	• naming conventions for solution elements such as files, functions, methods and variables	p. 136
	• processing features of a programming language	p. 152
	• characteristics of internal documentation	p. 164
	• formatting and structural characteristics of input and output such as file formats	p. 166
	• testing and debugging techniques to ensure software solutions meet requirements such as test tables and test data	p. 169
	• techniques for evaluating the efficiency and effectiveness of software solutions	p. 177
	• project plans to coordinate and monitor the tasks, including sequencing and time allocation to create software solutions	p. 118
Key skills	• analyse solution requirements to develop a software solution	p. 120
	• select and use appropriate design tools to represent solution designs	p. 133
	• use a range of data types and data structures	pp. 124, 126
	• develop a software solution using appropriate processing features of a programming language	p. 152
	• design and apply suitable testing and debugging techniques using appropriate test data	p. 169
	• evaluate the efficiency and effectiveness of the software solution to meet requirements	p. 177
	• document and monitor project plans using software	p. 118

9780170440806

OUTCOME	KEY KNOWLEDGE	LOCATION
Unit 2 Area of Study 1 Outcome 1	**Innovative solutions** On completion of this unit the student should be able to, in collaboration with other students, analyse, design, develop and evaluate an innovative solution to an identified need or opportunity involving a digital system.	
Digital systems	• components of digital systems	p. 195
	• types of digital devices used for a range of current and emerging applications such as smart phones, smart refrigerators and virtual assistants	p. 195
	• emerging trends in digital systems and the importance of innovation to organisations, such as improving efficiency and effectiveness of customer service and maintaining competitiveness	p. 195
	• functions and capabilities of digital systems used by individuals and organisations, such as assistive technologies, financial services, global positioning system (GPS) devices, robotics and traffic management	p. 208
Data and information	• techniques for collecting data to determine user needs and requirements, such as interviews and surveys	p. 246
Approaches to problem solving	• techniques for documenting the development of solutions	p. 242
	• solution specifications such as functional and non-functional requirements, constraints and scope	p. 247
	• characteristics of creative and innovative solutions	p. 251
	• design tools and techniques for representing solution designs, such as mock-ups, pseudocode, sitemaps and storyboards	p. 252
	• functions and techniques for developing innovative solutions	p. 267
	• techniques for validating and testing solutions	p. 261
	• evaluation criteria and techniques for evaluating the efficiency and effectiveness of innovative solutions	p. 272
	• tools and techniques for coordinating and monitoring projects, such as Gantt charts	p. 242
Interactions and impact	• goals and objectives of digital systems	p. 218
	• economic issues involving emerging technologies, such as access, deskilling, job loss, misuse and sustainability	p. 219
	• the impact of current and emerging technologies, such as automation, cyberbullying and the decline of physical human interactions and interpersonal skills	p. 224
	• key legislation and how emerging technologies are affected by: the *Copyright Act 1968*, the *Health Records Act 2001*, the *Privacy Act 1988* and the *Privacy and Protection Act 2014*	p. 234
	• ethical issues arising from the development of emerging technologies	p. 239

9780170440806

OUTCOME	KEY KNOWLEDGE	LOCATION
Key skills	• investigate a problem, need or opportunity and identify potential users and purpose	p. 251
	• propose a range of methods to collect data for analysis	p. 246
	• analyse and document solution requirements to develop an innovative solution	p. 247
	• select and use appropriate design tools for generating solution designs	p. 257
	• develop an innovative solution using appropriate digital systems	p. 267
	• document the development of the innovative solution	p. 242
	• design and apply suitable validation and testing techniques	p. 262
	• identify and discuss potential legal and ethical issues affecting the development of an innovative solution	p. 239
	• apply evaluation criteria and evaluate the efficiency and effectiveness of an innovative solution to meet a need or opportunity	p. 272
	• document, monitor and modify project plans using a Gantt chart	p. 242
Unit 2 Area of Study 2 Outcome 2	**Network security** On completion of this unit the student should be able to respond to a teacher-provided case study to examine the capabilities and vulnerabilities of a network, design a network solution, discuss the threats to data and information, and propose strategies to protect the security of data and information.	
Digital systems	• applications and capabilities of LANs, Wide Area Networks (WANs) and Wireless Personal Area Networks (WPANs)	p. 284
	• functions and characteristics of key hardware and software components of networks required for communicating and storing data and information	p. 290
	• strengths and limitations of wired, wireless and mobile communications technology, measured in terms of cost, data storage options, data transfer rate, reliability and security	p. 297
	• technical underpinnings of intranets, the internet and virtual private networks	p. 305
	• design tools for representing the appearance of networks	p. 308
	• security threats to data and information, such as improper credential management, malicious software, outdated versions of software and weak passwords	p. 309
	• technical underpinnings of malware that can intentionally threaten the security of networks, such as denial of service attacks on websites, spyware, viruses and worms	p. 310
	• data and network protection strategies, such as authentication techniques and symmetric and asymmetric encryption methods	p. 313
	• preventative practices to reduce risks to networks, such as application of firmware, disaster recovery plans, operating system updates, software malware updates and staff procedures	p. 317
	• technical underpinnings of intrusion detection systems (IDS) and intrusion prevention systems (IPS)	p. 319
	• the role of ethical hacking	p. 324

9780170440806

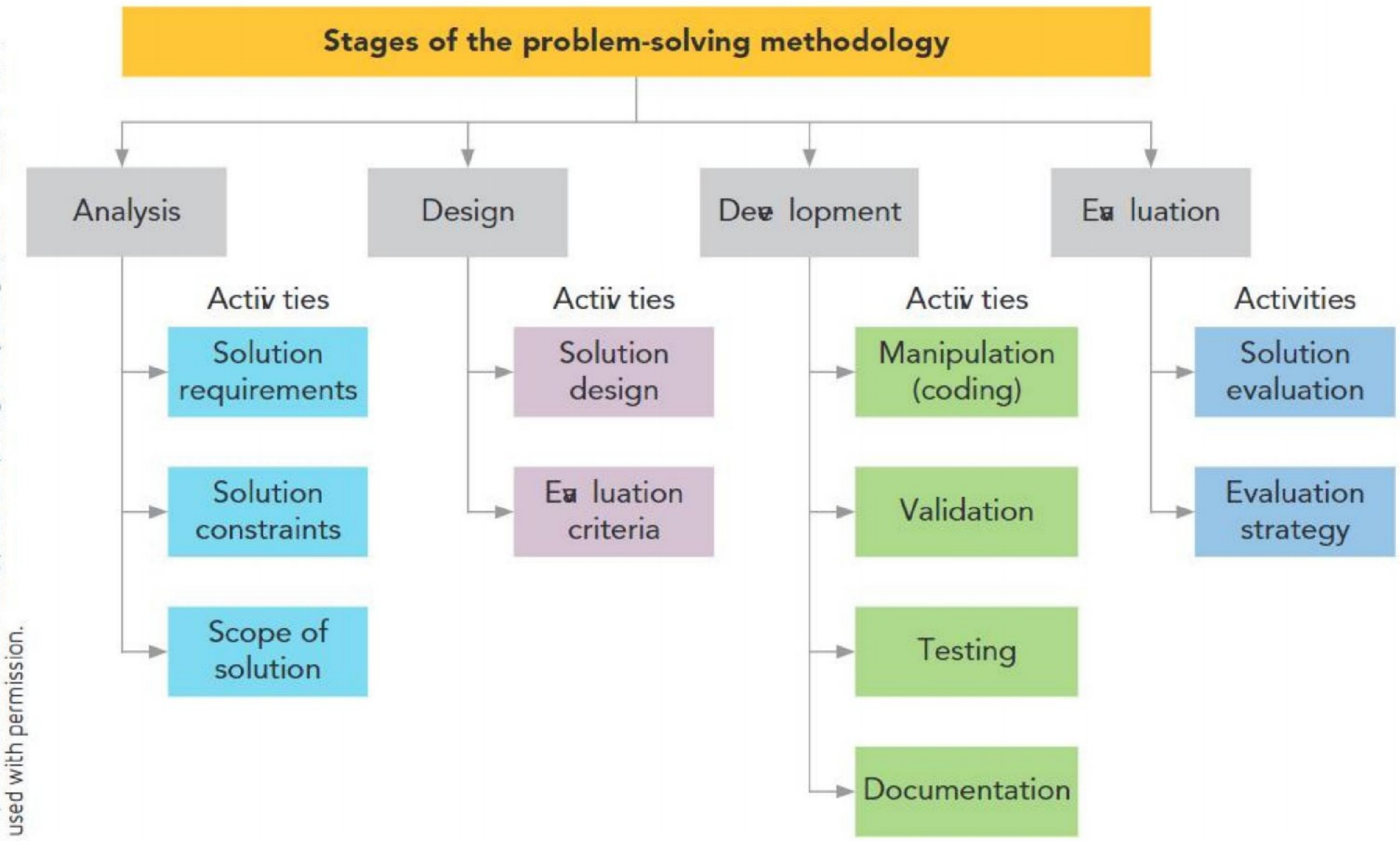
OUTCOME	KEY KNOWLEDGE	LOCATION
Interactions and impacts	• risks and benefits of using networks in a global environment	p. 324
	• key legislation that affects how organisations control the storage and communication of data and information: the *Health Records Act 2001*, the *Privacy Act 1988* and the *Privacy and Data Protection Act 2014*	p. 326
	• ethical issues arising from data and information security practices	p. 329
Key skills	• identify and describe the applications and capabilities of different networks	p. 284
	• examine the impact of common network vulnerabilities	p. 309
	• design a network solution with wireless capability	p. 297
	• identify and evaluate threats to the security of data and information	p. 309
	• propose and justify strategies to protect the security of data and information within a network	p. 313
	• identify and discuss possible legal and ethical issues arising from ineffective data and information security practices	p. 324

Reproduced from the VCE Applied Computing Study Design (2020–2023) © VCAA; used with permission.

9780170440806

Problem-solving methodology

When an information problem exists, a structured problem-solving methodology is followed to ensure that the most appropriate solution is found and implemented. For the purpose of this course, the problem-solving methodology has four key stages: analysis, design, development and evaluation. Each of these stages can be further broken down into a common set of activities. Each unit may require you to examine a different set of problem-solving stages. It is critical for you to understand the problem-solving methodology because it underpins the entire VCE Applied Computing course.



Reproduced from the VCE Applied Computing Study Design (2020–2023) © VCAA; used with permission.

FIGURE 1 The four stages of the problem-solving methodology and their key activities

Analyse the problem

The purpose of analysis is to establish the root cause of the problem, the specific information needs of the organisation involved, limitations on the problem and exactly what a possible solution would be expected to do (the scope). The three key activities are:

1. identifying solution requirements – attributes and functionality that the solution needs to include, information it must produce and data needed to produce this information
2. establishing solution constraints – the limitations on solution development that need to be considered. Constraints are classified as economic, technical, social, legal and related to usability
3. defining the scope of the solution – what the solution will and will not be able to do.

9780170440806

Design the solution

During the design stage, several alternative design ideas based on both appearance and function are planned and the most appropriate of these is chosen. Criteria are also created to select the most appropriate ideas and to evaluate the solution's success once it has been implemented. The two key design activities are:

1. creating the solution design – it must clearly show a developer what the solution should look like, the specific data required and how its data elements should be structured, validated and manipulated. Tools typically used to represent data elements could include data dictionaries, data structure diagrams, input–process–output (IPO) charts, flowcharts, pseudocode and object descriptions. The following tools are also used to show the relationship between various components of the solution: storyboards, site maps, data flow diagrams, structure charts, hierarchy charts and context diagrams. Furthermore, the appearance of the solution, including elements such as a user interface, reports, graphic representations or data visualisations, needs to be planned so that overall layout, fonts and their colours, for example, can be represented. Layout diagrams and annotated diagrams (or mock-ups) usually fulfil this requirement. A combination of tools from each of these categories will be selected to represent the overall solution design. Regardless of the visual or functional aspects of a solution design, at this stage a test for the design to ultimately ensure the solution is functioning correctly must also be created
2. specifying evaluation criteria – during the evaluation stage, the solution is assessed to establish how well it has met its intended objectives. The criteria for evaluation must be created during the design stage so that all personnel involved in the task are aware of the level of performance that ultimately will determine the success or otherwise of the solution. The criteria are based on the solution requirements identified in the analysis stage and are measured in terms of efficiency and effectiveness.

Develop the solution

The solution is created by the developers during this stage from the designs supplied to them. The 'coding' takes place, but also checking of input data (validation), testing that the solution works, and the creation of user documentation. The four activities involved with development are:

1. manipulating or coding the solution – the designs are used to build the electronic solution. The coding will occur here and internal documentation will be included where necessary
2. checking the accuracy of input data by way of validation – manual and electronic methods are used; for example, proofreading is a manual validation technique. Electronic validation involves using the solution itself to ensure that data is reasonable by checking for existence, data type and that it fits within the required range. Electronic validation, along with any other formulas, always needs to be tested to ensure that it works properly

9780170440806

3 ensuring that a solution works through testing – each formula and function, not to mention validation and even the layout of elements on the screen, needs to be tested. Standard testing procedures involve stating what tests will be conducted, identifying test data, stating the expected result, running the tests, stating the actual result and correcting any errors
4 documentation allowing users to interact with (or use) the solution – while it can be printed, in many cases it is now designed to be viewed on screen. User documentation normally outlines procedures for operating the solution, as well as generating output (such as reports) and doing basic troubleshooting.

Evaluate the solution

At some time after a solution has been in use by the end user or client, it needs to be assessed or evaluated to ensure that it has been successful and does actually meet the user's requirements. The two activities involved in evaluating a solution are:

1 evaluating the solution – providing feedback to the user about how well the solution meets their requirements, needs or opportunities in terms of efficiency and effectiveness. This is based on the findings of the data gathered at the beginning of the evaluation stage when compared with the evaluation criteria created during the design stage
2 working out an evaluation strategy – creating a timeline for when various elements of the evaluation will occur and how and what data will be collected (because it must relate to the criteria created in the design stage).

9780170440806

Key concepts

Within each VCE Applied Computing subject are four key concepts whose purpose is to organise course content into themes. These themes are intended to make it easier to teach and make connections between related concepts and to think about information problems. Key knowledge for each Area of Study is categorised into these key concepts, but not all concepts are covered by each Area of Study. The four key concepts are:

1 digital systems
2 data and information
3 approaches to problem solving
4 interactions and impact.

Digital systems focus on how hardware and software operate in a technical sense. This also includes networks, applications, the internet and communication protocols. Information systems have digital systems as one of their parts. The other components of an information system are people, data and processes.

Data and information focuses on the acquisition, structure, representation and interpretation of data and information in order to elicit meaning or make deductions. This process needs to be completed in order to create solutions.

Approaches to problem solving focuses on thinking about problems, needs or opportunities and ways of creating solutions. Computational, design and systems thinking are the three key problem-solving approaches.

Interactions and impact focuses on relationships that exist between different information systems and how these relationships affect the achievement of organisational goals and objectives. Three types of relationships are considered:

1 how people interact with other people when collaborating or communicating with digital systems
2 how people interact with digital systems
3 how information systems interact with other information systems.

This theme also looks at the impact of these relationships on data and information needs, privacy and personal safety.

9780170440806

Unit 1

INTRODUCTION

VCE Unit 1 of Applied Computing looks at how software tools such as databases and spreadsheets can be used to create visualisations of data. Students also study programming languages.

Throughout the unit, students will apply the stages of the problem-solving methodology. They will plan and monitor the progress of the tasks using project-management concepts. Different types of data will be acquired and manipulated in database and spreadsheet software.

There are two outcomes to be completed in Unit 1.

Area of Study 1: Data analysis

OUTCOME 1 Your teacher will provide you with solution requirements and designs for which you need to gather and organise appropriate data, analyse it and present the findings as data visualisations. As part of the solution development you will need to validate your data and apply appropriate formats and conventions to the data visualisations.

Area of Study 2: Programming

OUTCOME 2 You will be provided with a set of program requirements by your teacher. You are to use these specifications to design, develop and evaluate a solution created in a programming language. You will monitor the progress of your programming project, although you do not need to use project-management software.

Shutterstock.com / REDPIXEL.PL

9781170440806

CHAPTER

1 Data analysis

KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

Data and information

- types and purposes of qualitative and quantitative data
- characteristics of data and information
- sources, methods and techniques for acquiring and referencing primary and secondary data and information
- interpretation of information for communication and decision making
- factors affecting the quality of data and information such as accuracy, bias, integrity, relevance and reliability
- characteristics of data types and data structures relevant to selected software tools
- procedures for the legal and ethical collection and use of data and information, such as using consent forms
- techniques for protecting data and information from misuse, such as de-identifying personal data and the use of physical and software security controls

Interactions and impacts

- Australian Privacy Principles relating to the acquisition, management and communication of data and information such as non-identification of individuals (Principle 2), information only being held for its primary purpose (Principle 6) and the security measures used to protect personal information (Principle 11)
- ethical issues arising from the acquisition, storage and use of data and information

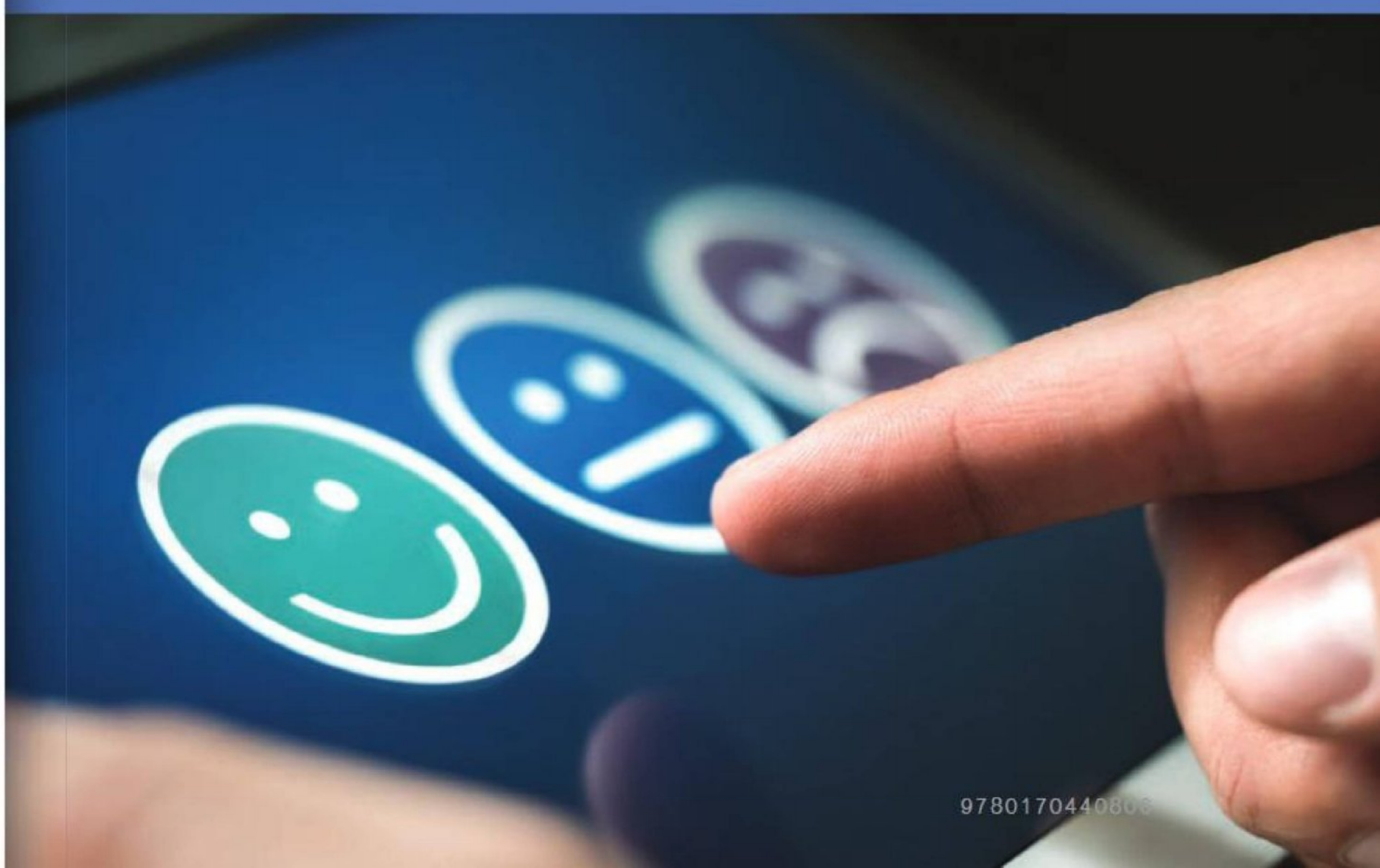
VCE Computing Study Design (2019) © The Victorian Curriculum and Assessment Authority (VCAA). Used with permission.

FOR THE STUDENT

In this chapter, you will respond to a teacher-provided analysis of requirements and designs to identify and collect data in order to present your findings as data visualisations.

FOR THE TEACHER

This chapter is based on Unit 1, Area of Study 1, and, together with Chapter 2, provides the key knowledge required to complete Unit 1, Outcome 1. At the end of Chapters 1 and 2, students should be able to interpret teacher-provided solutions requirements and designs, collect and manipulate data, analyse patterns and relationships, and develop data visualisations to present findings.



Shutterstock.com/Tero Vesalainen

97801704408

Understanding research

Some people consume research, and others produce research. Consumers of research spend a lot of time reading other people's research rather than conducting their own. On the other hand, producers of research investigate or explore an area that has relevance to them, interpret their **data** and then communicate their findings.

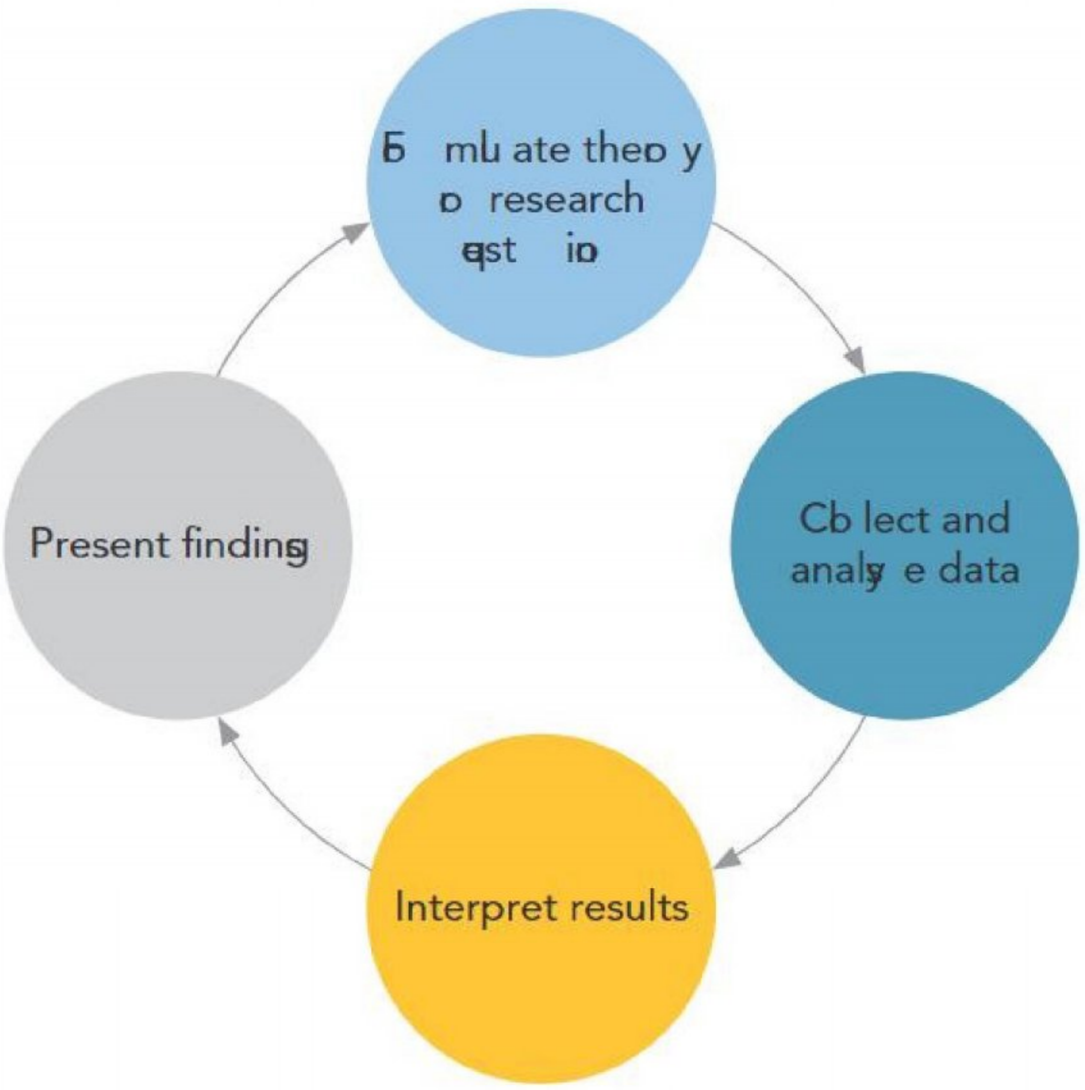


FIGURE 1.1 Producers of research often start with a theory or a research question.

Theories are usually general statements that describe something, provide an explanation of why something happens, and can be applied to predict what will happen in the future. Theories are, in principle, falsifiable or disprovable; that is, they contain information about the sorts of events that, if they were to happen, would show the theory to be false.

Some research questions are tied closely with theories. **Research questions** assist researchers to narrow the focus of the topic of the investigation. For example, 'Is therapeutic exercise of benefit in repairing cognition impairment in octogenarians?'

Hypotheses, on the other hand, are based on probabilities about what will happen according to the applied theory. Theories are tested by using data-collection tools such as surveys and/or interviews, and then the results of the study will either confirm or disprove the hypothesis.

Types of research

Investigating or researching topics of interest may be undertaken using quantitative or qualitative research methods.

Quantitative data is measurable and specific, and is therefore relatively easy to chart or graph. At a simplistic level, quantitative data gathering is based on verifying a research question through the use of statistics and data that is largely numerical, while qualitative data provides a more in-depth understanding.

9780170440806

The following is an example of quantitative data:

> 56.0 per cent said a lack of role models was a significant or moderate obstacle to their career advancement.

Professionals Australia (2018). *All Talk: Gap between policy and practice a key obstacle to gender equity in STEM – 2018 Women in STEM Professions Survey Report.* August 2018, page 32.

SPSS and MiniTab are statistical software packages used to analyse quantitative data.

When data has been gathered using surveys, focus groups, observation or other methods, quantitative data can be analysed by using software such as Excel, the Statistical Package for the Social Sciences (SPSS) and Minitab. This takes time and often involves hours of data entry, depending on the complexity of the data-gathering instrument. For data gathering, online surveys such as SurveyMonkey, Qualtrics, Google Forms and Microsoft Forms allow users to create surveys and manage the collection and analysis of quantitative data.

Qualitative data is about qualities or attributes, and is much harder to measure than quantitative data. You can gather qualitative data using instruments such as interviews, focus groups, video footage and observation. Generally, qualitative data needs to be recorded accurately and transcribed at a later stage. Online survey software permits qualitative data to be entered through text boxes.

The analysis of qualitative data is quite different from that of quantitative data. With quantitative data, the researcher looks for themes or patterns through the use of numbers, while with qualitative data, the researcher establishes rich descriptions and finds themes through reading the text and classifying these themes. The following example of qualitative data gathering is more descriptive:

NVivo is an example of a software package designed to help with the analysis of qualitative data. It lets a researcher sort and classify qualitative (non-numerical) information, identify relationships within it and examine whether it supports particular hypotheses.

> Survey respondents noted that a lack of role models and the lack of women both in the workplace and in senior roles were issues that impacted them. Respondents also noted that women in senior roles were not necessarily always positive role models and that in male-dominated workplaces and professions, career support and advancement for women could often depend on positive male role models.

Professionals Australia (2018). *All Talk: Gap between policy and practice a key obstacle to gender equity in STEM – 2018 Women in STEM Professions Survey Report.* August 2018, page 32.

Advantages and disadvantages of quantitative and qualitative data

Participants are more willing to be part of a quantitative study as it is less demanding of them. Often, quantitative studies use surveys, which can capture a large sample. Having a large sample size provides statistical validity, and helps to accurately reflect the characteristics, attitudes or views of the population. Data is interpreted, relationships are identified and findings are then communicated. Conversely, because surveys do not have a provision for probing the participants further, the answers provided do not have as much depth and are at times superficial. If too much information were provided, researchers would be overwhelmed by the amount of data collected and would not be in a position to analyse it. This can also be due to time and budgetary constraints.

Qualitative research provides for rich, in-depth studies of participants. Researchers can ask further questions, especially if something of interest arises. Generally, qualitative studies are small, and provide a narrative description of a sample group. Data-gathering tools can include interviews and focus groups. However, because the sample size is small and the sample is not very random, conclusions may not generalise readily to a larger sample size. Findings may be peculiar to a particular sample.

Data and information

The terms 'data' and 'information' are often used interchangeably. Data refers to the raw, unorganised facts, figures and symbols. Data can also mean ideas or concepts before they have been refined. In addition to text and numbers, data also includes sounds and images (still and moving).

Information is produced when data is manipulated into a meaningful and useful form. This can be achieved by organising the data and presenting it in a way that suits the needs of the intended audience. The information produced can be used to inform, entertain or persuade an audience. When information makes it possible to identify individuals, protecting their privacy becomes a consideration.

Technically speaking, a datum is a single item of data; however, the term 'data' is commonly used and accepted as both the singular and plural forms of the word.

Primary and secondary data and information

There are many sources of data and information, and many methods and techniques used to collect it. These include:

- what people say in interviews, focus groups, questionnaires/surveys, personal histories, biographies;
- images, audio recordings and other audio-visual materials;
- records generated for administrative purposes (e.g. billing, service provision) or as required by legislation (e.g. disease notification);
- digital information generated directly by the population through their use of mobile devices and the internet;
- physical specimens or artefacts;
- information generated by analysis of existing personal information (from clinical, organizational, social, observational or other sources);
- observations;
- results from experimental testing and investigations; and
- information derived from human biospecimens such as blood, bone, muscle and urine.

National Health and Medical Research Council.
National Statement on Ethical Conduct in Human Research 2007 (updated 2018). Page 33.

Sources

Primary sources of data provide a firsthand account of a person, object, event or phenomena. Many of the methods and techniques outlined in the National Statement on Ethical Conduct in Human Research are sources of primary data. Questioning them or surveying their opinions can provide different insights and more in-depth data than using information from **secondary sources**. The data will often be more up-to-date and can provide more unusual and important insights into issues, especially at the immediate local level, than secondary sources, which often present overall conclusions and general summaries.

Techniques and methods

Collecting data is usually done through methods such as surveys and interviews. While the results of surveys are easy to present graphically, interview results often can only be presented as written summaries and conclusions. However, both require analytical discussions to interpret their meaning.

Surveys contain a range of questions that relate directly to the research question being investigated. Participants may answer these questions by selecting a response from a list of alternatives, such as A/B/C/D, selecting multiple options from a given list, using a Likert scale (either 1 to 5, very low to very high, strongly agree to strongly disagree), or in short or extended text-based responses.

Interviews are usually conducted face to face, sometimes in groups, and, depending on the number of interviews being conducted, it can take a substantial amount of time to conduct and analyse the responses. A major feature of an interview is the opportunity for in-depth follow-up and clarification questions. These are not possible with surveys, which are often answered in private. Interviews are very useful for eliciting people's feelings, attitudes and opinions, which are too complex to easily record in a survey.

A group interview is also known as a focus group.

Data collection methods

Before we can produce information, we first must start with data. Methods such as surveys, interviews or observation provide a means of capturing data. Other ways to collect data electronically include using sensors, such as traffic cameras and satellites, and online sources, such as websites or data logs. The data collected can be used for a variety of purposes, including describing, predicting and improving processes within an organisation or for research.

Surveys

Surveys are common methods used to collect data. They can provide data about what the respondents think is true, or their preferences for consumer goods and political parties. A survey can be a quick way of gathering large amounts of data. Surveys need to be carefully designed, otherwise the participants' responses may not provide suitable data to analyse, rendering them useless. Questions used in a survey must be carefully worded so that the response will provide meaningful and useful data without the need for further clarification.

Focus groups

A **focus group** is the meeting of a small group of individuals who are guided through a discussion by a researcher, similar to a group interview. The group is carefully selected to fit a particular demographic and so the researcher can obtain the necessary data through a guided discussion that probes the participants' attitudes about the topic. Focus groups often comprise between five and 12 people and the discussion is loosely structured to encourage ideas to flow.

Interviews

Interviews are used to elicit people's opinions and beliefs. They can be used to gather data for research projects. Interviews are usually conducted with one or more participants in a quiet and relaxed atmosphere. They should be recorded, with the interviewee's permission, with easily used and unobtrusive audio or video equipment. Writing down the responses during the interview is not helpful to the interviewer or the interviewee. These records are research data in themselves, but may also be transcribed later. Collating and analysing information can be difficult and time-consuming and may require the use of someone with expertise. There are many interview styles that can be used:

- structured interviews, which follow a set list of questions
- semi-structured interviews, which follow a list of issues to be explored
- unstructured interviews, which involve spontaneous generation of questions and where the interview is driven by the interviewee rather than the interviewer.

Open-ended and closed questions

Questions used on a survey and during an interview can be open-ended or closed. **Closed questions** limit the responses available to the respondent (Figure 1.2, page 8). They include 'Yes/No' boxes, multiple-choice questions, and scales on which attitudes and beliefs are measured using responses such as 'strongly agree', 'agree', 'disagree' or 'strongly disagree'.

Closed (or closed-ended) questions are generally considered to be quantitative in nature. They are called 'closed' because the range of answers the participant can choose is limited. They are considered to be quantitative because the response options can be converted to numbers. For example:

How often do you wash your car?

5 I always wash my car.
4 I sometimes wash my car.
3 I occasionally wash my car.
2 I wash my car once in a while.
1 I never wash my car.

Each of these options can have a value placed next to it. However, we do not talk in numbers and we shouldn't create surveys that only have numbers. Surveys should be thought of as a conversation between the person asking the questions and the person answering them.

Open-ended questions do not limit the answers that the respondent can give (Figure 1.3, page 8). They should be worded so that the responses received are capable of correct interpretation. For instance, if you asked the question, 'How do you feel about the widespread use of computer games?', the responses would probably be too broad to be usefully categorised and analysed. The wording must therefore limit the scope of the possible responses to specific areas of interest: 'How has the playing of computer games affected your child's school results?' Open-ended questions also allow for follow-up questions, which are called probing questions, such as 'Why?' or 'Please give an example'. Such questions tend to elicit more detail.

Open-ended questions are 'open-ended' because participants are free to answer in any manner they choose. Unlike closed questions, there are no response options specified. They are qualitative because responses are considered and measured by feel rather than by numbers.

THINK ABOUT APPLIED COMPUTING 1.1

What types of charts would you use to display the aggregated responses to closed questions?

Closed questions are easier to develop, quicker to administer and answer, easier to collate and analyse, and can provide a large and balanced sample; however, they may not be useful for complex issues. In this case, open-ended questions may be needed as they elicit greater detail in the responses, can bring forth unusual ideas and can show links between various aspects of the issues.

Closed questions
1 How long have you shopped at this store?
2 How many times per week do you go shopping?
3 How much do you spend per week?
4 Which of the following sources of information most influences your purchasing habits?
• Advertising pamphlets delivered to the home
• Newspaper advertisements
• Television promotions
• Recommendations from friends
5 Do you use a computer? • Yes • No
6 What time of day do you normally go shopping?

FIGURE 1.2 Closed questions should be designed to elicit short, straightforward answers.

Open-ended questions
1 What is your opinion of the games available from this store?
2 How influential do you think the advertising campaign has been?
3 What are some of the errors in data entry that you have observed?
4 Describe the most frustrating experience you have had when using the computer system.
5 What are some of the problems you experience in receiving information on time?
6 What changes would you recommend to improve the billing system?

FIGURE 1.3 Open-ended questions try not to limit the answers the respondent can give.

Observation

Observation is a way of understanding the world around us as well as developing an understanding of existing processes. Using our senses (sight, smell, hearing and touch) we are able to pick up detailed information about our environment. However, as a method of data collection, observation is more than just looking or listening, as we can be selective about what we perceive to be most useful to us. Researchers engaged in observation attempt to learn what life is like for someone in a particular setting, while they themselves remain outsiders. While observing, they make careful notes of what they see, and record all accounts including conversations and interactions. Observation generally takes place in community settings, such as classrooms, or in locations believed to have some relevance to the research questions. Observation is unlike other forms of data-collection tools, as the researcher approaches participants in their own environment rather than having the participants come to the researcher.

Referencing primary sources

Once primary data has been gathered, details need to be carefully recorded to enable appropriate **referencing**.

Interviews

For an interview, the following details need to be documented so that an interested person can go back to the source for checking, clarification and further information:

- name of interviewee
- date of interview
- place of interview
- qualification to be an interviewee – that is, whether the interviewee is a **stakeholder** in the issue and/or an expert about it
- organisation to which the interviewee belongs (if relevant)
- contact information for interviewee – phone number, address, email address, online chat handle
- how the interview was conducted; for example, in person, by phone, email or online chat
- name and contact details of interviewer.

Surveys

If you want to cite an individual response to a survey, you need to record these details:

- name of respondent
- when the survey was completed
- title of survey
- organisation to which the survey belongs (if relevant)
- how the survey was conducted – paper/online.

Observation

For observation, the following details need to be recorded:

- the name of the person or group of people observed
- when the observation was conducted (date/time)
- where the observation was conducted.

Examples of referencing

Citations in a document help readers to find the source of the information and also assist students to avoid **plagiarism**. There are many ways to cite sources, such as providing footnotes or in-text citations, or listing sources at the end of the document in a bibliography or reference list.

Footnotes

Footnotes are listed at the bottom of the page on which a citation is made. Some academic disciplines prefer to use footnotes (notes at the foot of the page) to reference their writing. Although this method differs in style from the 'author, date' system, its purpose – to acknowledge the source of ideas, data or quotations without undue interruption to the flow of the writing – is the same. Footnotes are usually sequenced: series of numbers above the text (superscript) are placed in the appropriate part of the text to indicate the cited work and are matched at the bottom of the page to the footnote. A footnote lists the author, title and details of publication, in that order. For example, in a 2018 report, *Optimising STEM Industry–School Partnerships: Inspiring Australia's Next Generation Final Report*, the paragraph shown in Figure 1.4 appears in the introduction.

> To meet the challenges of the future, as noted in the ISA 2030 Plan, governments, educators and industry must work more closely together to maximise and amplify the impact of industry investment and ensure students and teachers keep up with the rapid pace of change in STEM disciplines.[1]
>
> ---
>
> 1 Innovation and Science Australia 2017, *Australia 2030: prosperity through innovation*, Australian Government, Canberra, p. 33.

FIGURE 1.4 An example of a footnote. A superscript number is inserted in the body of the text, and the full reference is provided at the bottom of the same page.

APA

The American Psychological Association (APA) created a style guide to assist with academic writing such as essays, books and other publications. The APA style is widely used and is the reference style that students are expected to use in VCE Applied Computing. Citations within the text, and their corresponding source details in a reference list at the end of the work, are necessary elements of the APA style. These show the reader where ideas and research have come from. Typically, when referencing using APA, the author's surname and the date of the publication are featured in the text. If quoting directly from the source, then the page number is also included. Figure 1.5 shows an example.

> The Theory of Individual Differences (Trauth et al., 2004) draws on previous research on gender and IT and emphasises the differences within genders rather than between them. The theory 'looks at men and women as individuals, who experience a range of different social-cultural influences which shape their inclination to participate in IT in a variety of ways' (Trauth et al., 2004, p. 116).
>
> …
>
> Trauth, E.M., Quesenberry, J.L., & Morgan, A.J. (2004). *Understanding the under representation of women in IT: Toward a theory of individual differences.* Paper presented at the SIGMIS 2004 Conference on Computer Personnel Research, Tucson, Arizona, USA.

FIGURE 1.5 An example of APA-style citation. The author's name, date of publication and sometimes the page number are inserted in the body of the text; the full reference is provided in the reference list at the end of the document.

Within the main body of the document, any reference to the publication will need to be cited. As illustrated in Figure 1.5, when using the APA style, the format usually follows author–date–page, where the author's last name, the year of publication and the page number of the quote are referenced. These are all separated by commas, and are placed within parentheses following the text. The page number is preceded by a lower case 'p' with a period (full stop) after it; for example, '(Trauth et al., 2004, p. 116)'.

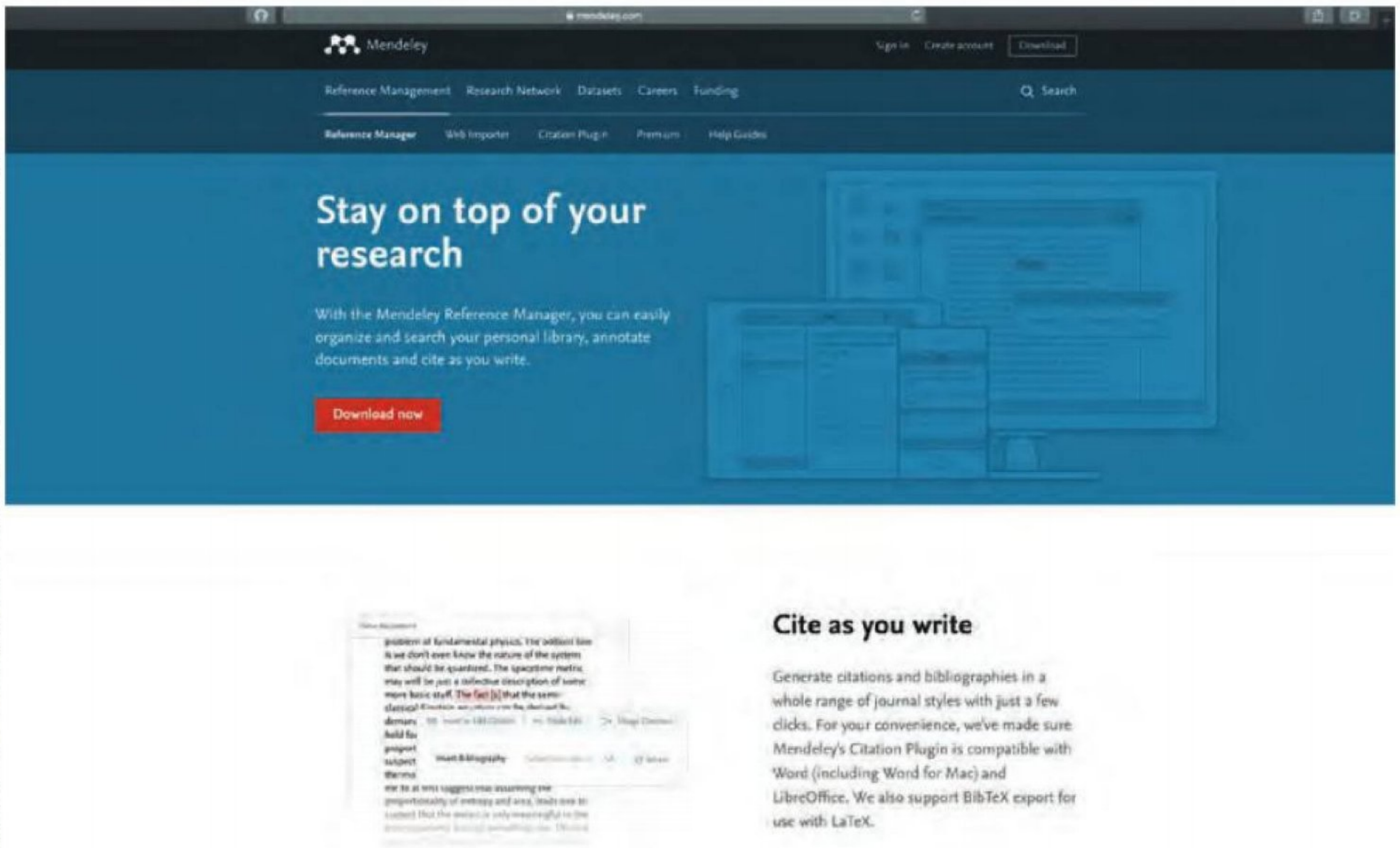
For more detailed information, visit the APA website.

Reference management software

Reference management software is widely available and can be used to manage citations and reference lists. An example of such software is EndNote. EndNote is similar to a database and is used to keep all references in one place. It also integrates with Microsoft Word, so that references can easily be inserted into the text, as shown in Figure 1.6. Alternatively, Microsoft Word has its own built-in referencing capabilities that will allow you to create a reference list and manage your sources without the need for additional software.

For more information, visit the EndNote website.

FIGURE 1.6 EndNote integrates easily into Microsoft Word.



https://www.mendeley.com/reference-management/reference-manager

FIGURE 1.7 Mendeley can be used by students to assist with referencing.

The free Mendeley website, shown in Figure 1.7, generates citations in the APA style. It can assist students to generate references in the appropriate format.

Interpretation of information for communication and decision making

Data and information can be collected, stored and interpreted for a particular purpose or activity, including research. The findings or results may be of significant benefit to the participants or their relatives, as in the case of the analysis of human biological specimens; conversely, it is worth noting that there could be psychological, social or legal risks involved.

The purpose of disseminating outcomes from the interpretation of the information is to make a contribution to knowledge. Sometimes, the interpretation of findings assists with decision making to better inform participants about options available. Common ways to disseminate findings are through publications such as journals, books, conference presentations, data visualisations, creative works and performances.

'The approach taken to communicating findings and results should ... adhere to the ethical principles of justice, respect and beneficence ... including consideration of the values and preferences of traditional custodians, such as Aboriginal and Torres Strait Islander peoples.'

National Health and Medical Research Council. *National Statement on Ethical Conduct in Human Research 2007 (updated 2018)*. Page 38.

Quality of data and information

One aspect of identifying relevant data from a given data set is ensuring that the data is usable. To be usable, data must be relevant, accurate, free from bias and reliable.

Relevance

To produce usable information, data must be **relevant**. For example, if a computing department in an organisation is evaluating PC-only software, then surveying people who only use an Apple device is not useful. The data collected from Apple users would not be relevant to the overall data collection. Data also needs to be processed while it is current, because decision making should be based on current data, not outdated data.

Accuracy

Data that is collected must be **accurate**, otherwise the insights will be poor and decisions based on this data could cause integrity issues. Transcription is often a cause of error. Transcription errors occur when the person entering the data misreads the information through, for example, a lapse in concentration or being interrupted, or presses the wrong key. It is easy to make a mistake when entering a large amount of data, particularly numbers with many digits that may not contain spaces or punctuation to signify thousands. Clearly, if the data collected is incorrect, the information produced from it will be incorrect. If data has been gathered from a primary source, it is a good idea to check the data against the source. If data has been gathered from a secondary source and is suspect, it is worthwhile verifying the data using other secondary sources.

Freedom from bias

Bias can easily creep into data and make the information processed from it unreliable. Several influences can lead to the introduction of bias into data; these include vested interests, timing and small sample sizes. There can also be bias through sorting and bias through graphic representations.

Vested interest

Bias can enter data if the respondent to a survey or interview has a **vested interest** in the outcome of the research. A common example is seen in the use of celebrities who are paid to promote particular products in commercials or social media. It would be unreasonable to trust their assertions that one product is better than others simply because they are celebrities; they are only saying what they have been paid to say and may not necessarily be providing an independent judgement that has been derived from research or experience.

Timing

The timing of the data collection may also introduce bias. Suppose, for example, you plan to survey a sample of the population for their views about Australia becoming a republic. The data you gather may be biased if, just before the survey is conducted, a royal tour takes place and there is extensive media coverage about the royal family. The timing of the data collection would introduce bias because it coincides with a significant event that could influence people's responses.

Note too that bias is not restricted to data gathered from surveys or during interviews. For example, suppose that Qantas needed to decide whether to schedule two new weekly flights to New York. The decision could depend on the demand for existing flights. If the airline collected data from bookings made over a four-week period just before or during a significant event, such as the New York marathon, the data gathered would be biased. Such data should not be relied on for making this decision because the influence of this event on customer demand is irregular and unlikely to occur again.

Small sample size

Choosing a sample size that is too small may also create bias. The sample size and composition must be suitable for the purpose of the data collection. Usually, a larger sample size leads to greater **precision**, provided the sample composition is suitably representative of the target population. The sample size must be big enough to make any conclusions drawn, and information produced, credible. For example, if you wanted to determine whether or not the school uniform should be changed, the data would not be reliable if you only surveyed students in one class. Not only is this sample too small to be representative of the student body; it would also fail to include other stakeholders, such as parents and school administrators. Similarly, if you wanted to gather sales data over a four-day period to predict monthly sales at a fish and chip shop, this time scale would not be sufficient to make a prediction. For instance, by collecting data on the four Mondays in the month, you may be selecting the quietest trading days in the week. If you picked the four Fridays in the month, you may be picking the busiest trading days. When selecting a sample size, you need to ensure that it is representative of the whole population.

Bias through sorting

The way in which you sort lists can introduce bias, although this is frequently unavoidable. A classroom teacher often consults a class list that is sorted alphabetically; for example, to select students for special tasks. The list is biased towards students whose surnames appear early in the alphabet and are thus at the top of the class list. If you need to hire an electrician and consult an online directory, it is more likely that you will pick an early entry than one from the second page of listings. Bias of this type is difficult to avoid, so it is preferable to educate the user to recognise that the output has built-in bias and to encourage strategies to overcome that bias.

Bias in graphic representations

Bias can occur through your choice of typeface or the scale or size chosen for graphics. Graphic representations should be sized proportionally to avoid overstating or trivialising the importance of one of the **variables** involved. For example, in Figure 1.8, a teacher has created a graph to show the heights of her students. The graph does not really give a clear picture of how the heights vary. The bars are all of similar height, so it is difficult to see the differences between them. In contrast, the different heights in Figure 1.9 are more distinct. This has been achieved by decreasing the scale of the vertical axis. The variation in the student heights looks much greater, even though the data has not changed. This graph makes it easier to see the small differences.

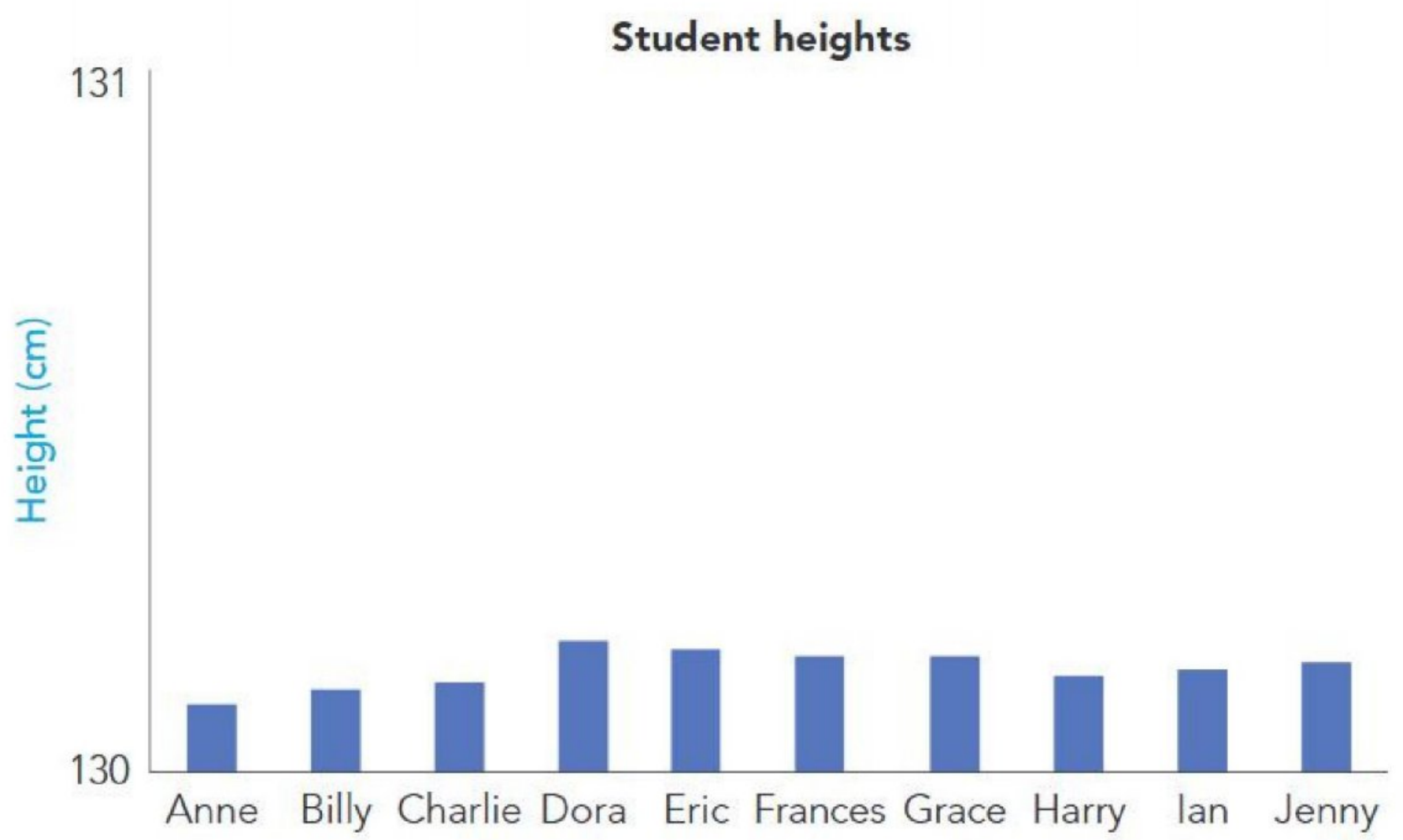


FIGURE 1.8 Column graph with little variation

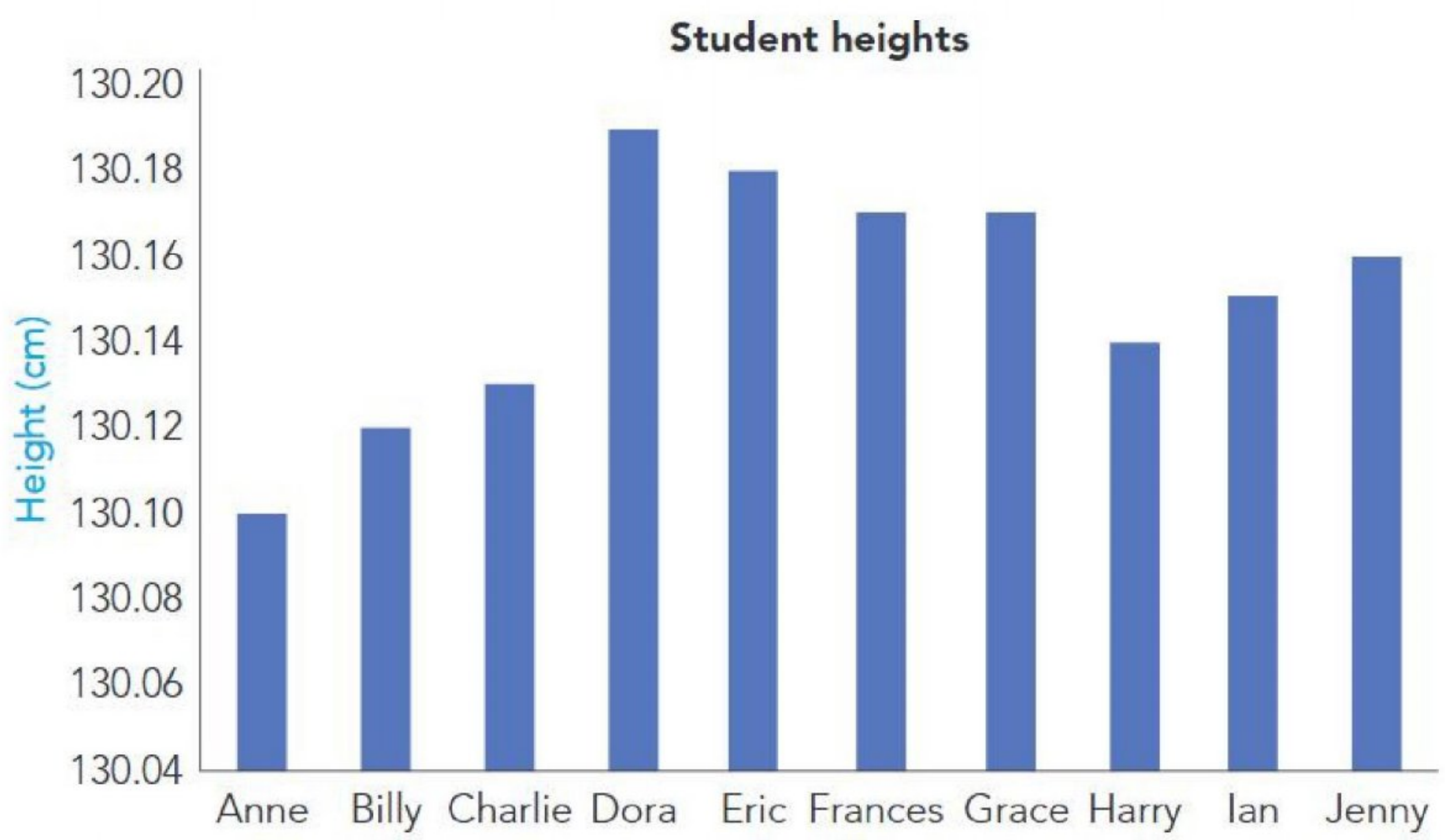


FIGURE 1.9 Column graph with greater variation

THINK ABOUT APPLIED COMPUTING 1.2

Find three reliable sources and describe in one paragraph the type of data they produce.

THINK ABOUT APPLIED COMPUTING 1.3

Create a list of five organisations in Australia that provide data sets for public use. They may be government departments or private organisations, or a mix of both.

Reliability

The internet has made it easier for people to communicate their views and present information in a format that is easily accessible to others. There are many personal websites, homemade videos, wikis, podcasts, vodcasts and a plethora of unchecked information on the internet, and some of the views presented may not be widely accepted or proven to be accurate. Alternatively, there are many sources on the internet that *are* **reliable**, such as the World Health Organization (WHO), the Smithsonian Institution, the Australian Bureau of Statistics, UN Data, Google Public Data Explorer, Eurostat and universities.

Integrity

Data integrity refers to the quality of the data. An information system's purpose is to transform data into useful information. It is therefore important that the data input has integrity, because the information produced is based on the data itself. For example, if a customer database contains the wrong postal details for a customer, this is incorrect data. When a data set contains these types of errors, it loses its integrity. The more errors the data set contains, the lower its integrity. There are many ways in which the quality or integrity of data can be measured.

Data types and data structures relevant to selected software tools

Many different software tools can be used to create data visualisations; most, however, are derivatives of either a spreadsheet tool or a database tool. Data needs to be separated into distinct fields or columns that only have one data type in them. **Data types** are particular forms that an item of data can take. Common data types are text, numeric, currency, date/time and Boolean. Each of these is discussed in the section that follows Table 1.1.

Area of Study 1 concentrates on students presenting their work in ways that include database, spreadsheet and data visualisations solutions. This section will concentrate on data types and structures relevant to databases and spreadsheets.

The data held by a database management system (DBMS) must be in a format suitable for storage. Typically, each field will be created to hold a specific type of data. This ensures data consistency in each field and assists with validation, because a field will refuse to accept data of the wrong type.

TABLE 1.1 Most common data types used in databases

Data type	Characteristics	Examples
Text	Alphanumeric	Name, address, postcode and phone number
Numeric	Numbers only (see below for different forms)	Any number to be used in a calculation
Currency	Numbers, but in dollar amounts formatted with $ symbol and including .00 at the end as a default	Any number used to represent a financial value; usually only applied to a total, rather than values in a list
Date/Time	A variation of numbers formatted to represent a date and/or time	Any date can be used in calculations
Boolean	Represents one of two states, True/False; can be used in logical tests	Also represented as Yes/No, On/Off and 0/1

Text (string)

A text data type consists of a '**string** of characters'. Text may also be referred to as a string, and the terms are sometimes used interchangeably to refer to the same data type. This type of field holds a mix of characters (letters, numbers, special characters), also referred to as alphanumeric. Names and addresses are considered to be text data. Also, postcodes and telephone numbers are normally formatted as text – even though they contain numerals – because they may contain spaces and are not intended to be used in mathematical calculations. It is also more efficient to store the values as text rather than as a large numeric value.

Numeric: integer, floating point

The numeric data type refers to numbers that are to be used for a calculation of some kind. For example, the quantity of an item purchased might need to be multiplied by its price in order to calculate a total amount payable. Numeric fields can be separated into two distinct categories: integer and floating point.

Integer

An **integer** is a number without a fractional or decimal component – in other words, a whole number. Integers include negative numbers.

Floating point

A **floating-point number** is a number with a fractional or decimal component, such as a decimal number. Where decimal numbers are required, such as when dealing with financial transactions or percentages, then the floating-point data type is used. Examples of the floating-point data type are '3.17', '–6.890' and '0.001'.

A floating-point number can be split into two parts: the mantissa and the exponent. The **mantissa** holds all of the digits in the number, with a decimal point after the first digit that is not zero. The **exponent** holds the power of 10 by which the mantissa must be multiplied to regain the original number. For example, 0.00317 can be expressed as 3.17×10^{-3}, in which case the mantissa is 3.17 and the exponent is –3.

Date

Strictly speaking, a date format is another variation on the numeric data type. The value of a date is normally based on the number of days since the 'zero' day built into the operating system. For example, the day '1' might be displayed as 01 January 1900, while '42673' would be displayed as 30 October 2016. Calculations can be performed on dates, which can be handy when considering the difference between them. Dates can be formatted to show a combination of years, months, days, hours, minutes and seconds, depending on the needs of the user. In terms of time, they can also display 12- and 24-hour clocks.

Character

A **character data type** holds a single letter, number or symbol. Numbers stored as a character data type cannot be used in mathematical calculations. This is a text field that will only accept a single alphanumeric character. It is used when there are multiple options for a value, but they can be represented with a single character to make data entry easier and to save storage space. For example, small, medium and large sizes of the same type of wooden box might be entered as 'S', 'M' or 'L', respectively.

Boolean

A **Boolean data type** can hold one of only two possible values, usually 'true' or 'false'. A Boolean data type is known as a logical data type. It is often used in conditional statements to test if a condition or value is true or not, but it may also represent Yes/No or On/Off.

THINK ABOUT APPLIED COMPUTING 1.4

In theory, a Boolean data type only requires 1 bit of memory space. Find out why it actually needs to take up 1 byte of memory.

Data structure

A **data structure** is a particular way of organising a collection of data items. Using data structures helps to improve the efficiency of the data-handling process. Three common data structures are arrays, records and files.

Array

An **array** is a collection of data items that are generally of the same data type. Each item in the array is allocated an address. Table 1.2 illustrates an array data structure. The structure is named 'Number' and it holds six data items – all integers. The value of Number (0) is 40 and that of Number (5) is 67.

TABLE 1.2 Array data structure

Number	
Index	Value
0	40
1	34
2	56
3	21
4	12
5	67

Record

A **record** is a collection of data items of different data types. Each element of a record is generally called a field. The record in Table 1.3 consists of six fields and involves data of different data types.

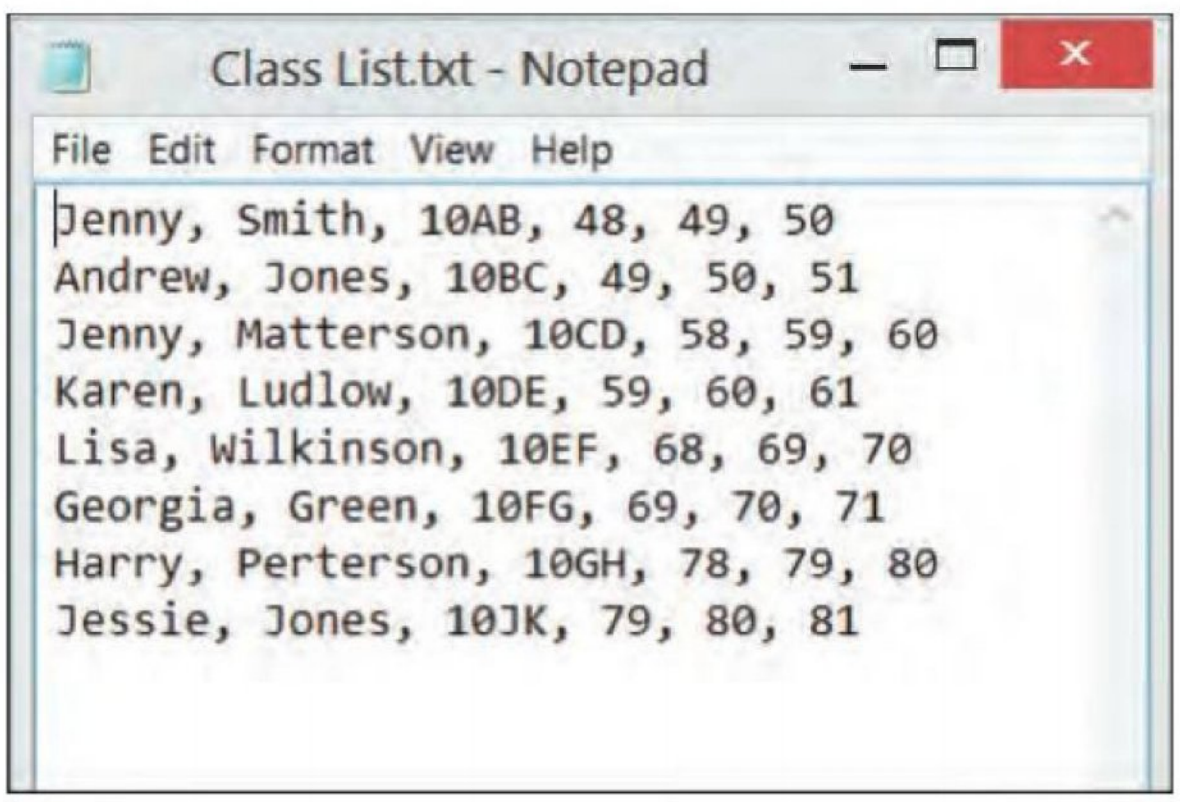
TABLE 1.3 Record data structure

Customer ID	Name	Phone	VIP	Quantity	Cost
01901	Karen	0406348XXX	T	5	3.99

> **1.5**
> **THINK ABOUT APPLIED COMPUTING**
>
> Referring to Table 1.3, how many different data types can you identify?

File

A file data structure can hold numerous data items, arrays or records. A file is saved separately from the software program that utilises the file. Large data sets can be read from, or written to, a file for storage purposes. The file in Figure 1.10 is an example of a **comma-separated value (CSV) file**.




```
Jenny, Smith, 10AB, 48, 49, 50
Andrew, Jones, 10BC, 49, 50, 51
Jenny, Matterson, 10CD, 58, 59, 60
Karen, Ludlow, 10DE, 59, 60, 61
Lisa, Wilkinson, 10EF, 68, 69, 70
Georgia, Green, 10FG, 69, 70, 71
Harry, Perterson, 10GH, 78, 79, 80
Jessie, Jones, 10JK, 79, 80, 81
```

FIGURE 1.10 A collection of records related to student results

Seeking permission

Because of privacy laws such as the Australian Privacy Principles (APPs), it is necessary to seek permission to collect any data or information that involves people. For example, you should obtain permission (consent) to photograph or video individuals or groups. The organisation or individual seeking permission needs to let the people photographed or videoed know the purpose of the photographs or video and what they may be used for. Permission is usually provided in written form and is often referred to as 'consent'. Permission must be sought because a photograph or video image in which an individual can be identified is considered to be personal information. Pictures of people can be used in advertisements, or for marketing purposes, and sometimes pictures can be used thoughtlessly and depict people in a false light.

Participants in any research need to be informed about what the research entails. They need to know what they are required to do and how much of their time it will take, and how often they will be required. All details of the research need to be given so that participants can make an informed decision to participate. For example, when researchers want to conduct a survey, they need to specify how much time it will take (for example, 15 minutes) and how many times the survey will need to be completed. Participants need to know whether the survey will be paper-based or electronic, and how they will get access to it. All these details need to be explicitly stated so that each participant has a clear understanding before agreeing to take part. The researchers cannot put pressure on the participants or use coercion or provide financial or other incentives for them to participate. Participation in research needs to be voluntary and informed.

In universities, research that involves people or animals cannot begin until researchers obtain **ethics** clearance. Each university has an ethics committee. These committees follow the guidelines set out by the National Health and Medical Research Council (NHMRC) to ensure research is carried out ethically and with integrity. In order to obtain clearance from the ethics committee, the researcher/s must demonstrate that they have followed correct procedures and processes before collecting data to ensure that all risks have been addressed, and that it is established that participation is informed and voluntary.

NHMRC
Australian Code for the Responsible Conduct of Research

The *Australian Code for the Responsible Conduct of Research*, 2018 (the 2018 Code) 'establishes a framework for responsible research conduct that provides a foundation for high-quality research, credibility and community trust in the research endeavour' (NHMRC, 2018, p. 1). This code sets out good research practices that researchers in Australian universities must adhere to. These practices cover important principles such as honesty, fairness, transparency, respect and accountability.

Consent forms

One method of obtaining permission for research purposes is to use consent forms. The **informed consent** of all research participants is required. This means that before agreeing to participate in research, they are aware of what the research involves, the time required from them and any risks that may arise. Participation in research not only has to be informed; it must also be voluntary – that is, participants are not pressured to be involved, and have the capacity to make their own decisions based on their understanding of the research.

You must obtain consent when interviewing or observing participants, creating surveys or collecting any type of data.

Consent forms should have the following information clearly listed:

- the title of the project
- the name of the researcher
- what the project is about and why it is being undertaken
- what is required from the participants in terms of time, effort, resources and costs
- the rights and interests of the participants – that they freely consent to be involved in the research and can withdraw at any time without having to provide a reason
- a statement of whether the participant's identity will be preserved
- statements of confirmation, such as:
  - 'I have been informed of and understand the purposes of the study.'
  - 'I have been given an opportunity to ask questions.'
  - 'I understand I can withdraw at any time without prejudice.'
  - 'Any information that might potentially identify me will not be used in published material.'
  - 'I agree to participate in the study as outlined to me.'

The form should also have space for the participant's name, their signature and the date.

Name of school:

Project title:

Investigator(s):

1 I consent to participate in the project named above. I have been provided a copy of the project consent information statement to which this consent form relates and any questions I have asked have been answered to my satisfaction.

2 In relation to this project, please circle your response to the following:

• I agree to be interviewed by the researcher	**Yes**	**No**
• I agree to allow the interview to be recorded by an electronic device	**Yes**	**No**
• I agree to make myself available for further information if required	**Yes**	**No**
• I agree to complete surveys asking me about [Insert topic]	**Yes**	**No**

3 I acknowledge that:

a my participation is voluntary and that I am free to withdraw from the project at any time without explanation;

b the project is for the purpose of research and not for profit;

c any identifiable information about me which is gathered in the course of and as the result of my participating in this project will be **(i)** collected and retained for the purpose of this project and **(ii)** accessed and analysed by the researcher(s) for the purpose of conducting this project;

d my anonymity is preserved and I will not be identified in publications or otherwise without my express written consent.

By signing this document I agree to participate in this project.

Participant name: ....

Signature and date: ....

FIGURE 1.11 Sample consent form

Participation information statements

The most common way to obtain informed consent is to provide a **participant information statement**, written in plain English, along with a consent form. This provides participants with information about the research and also provides the scope to answer questions that participants may have. Once they have read and understood the information statement, they are asked to sign the consent form to indicate their agreement to participate. When developing a participant information statement, consider the following.

- Use language your audience can easily understand. The statement should be suitable for a 12-year-old reading age. If technical terms are required, make sure they are clearly explained the first time they are used.
- Write in a conversational style, as if you were speaking to the participant.
- Language used should be clear, concise, invitational, culturally appropriate and logically set out.
- Use pronouns, such as 'I', 'we' and 'you'. This encourages the use of active voice and will be clearer to the reader. For example: 'You will be asked to participate'.
- Use reader-friendly formatting so that your document is easy to read.
- Ask others to read and edit your document.
- Include a statement on how the data will be stored.
- State where and how the research findings will be published.
- Give further information about the project, such as whom to contact.

Even if the data-collection technique is an online survey and the researchers do not meet the participants, respondents must still be informed about the research, and asked to consent online before completing the survey. They may be asked to consent by accepting and selecting the 'I agree to participate' option. If they choose the 'I do not wish to participate' option, then they do not get access to the survey. See Figure 1.12 for an example of an integrated online information statement and consent form.

English

Technology Disconnect

The transition from secondary school to university is filled with expectations that may or may not be met once you start your tertiary studies. In particular, expectations with respect to the use of technologies for learning and teaching - both by teachers/lecturers and you as a student - is one of the key areas of interest for many universities. Please be assured that your responses will be kept completely confidential. This project aims to discover what your expectations were for technology use at university while you were still in secondary school, and compare that to the reality you have encountered since you started your university studies. We would like to invite you to tell us what you thought you would find, and what you have actually found to be the use of technology for learning and teaching at university. The study is a voluntary, anonymous, online survey that should take you no longer than 15 minutes to complete. While we encourage you to complete the whole survey, you may stop at any time and only those responses you have already answered will be used in the study.

The anonymous findings will be collated before being used:

- to set strategic directions around the use of technology for learning and teaching
- to inform future investment in technologies for learning and teaching
- to inform professional development for staff in the use of technologies

It is critical to have the student voice guiding the above so we strongly encourage you to participate. If you would like further information about any aspect of the project before agreeing to participate, please do not hesitate to contact us:

[redacted]
Associate Professor ([redacted]
[redacted]
-[redacted]

Any queries about your participation in this project may be directed to the Primary Investigators listed above. If you have any concerns or complaints about the conduct of this project, please contact:
Research Ethics Officer, [redacted], [redacted]
F[redacted]

By clicking the button below, you acknowledge that your participation in the study is voluntary, you are 18 years of age, and that you are aware that you may choose to terminate your participation in the study at any time and for any reason.

Please note that this survey will be best displayed on a laptop or desktop computer. Some features may be less compatible for use on a mobile device.

I consent, begin the study

I do not consent, I do not wish to participate

FIGURE 1.12 An example of an integrated online information statement and consent form. Note that personal information has been redacted for privacy reasons.

9780170440806

The consent information statement should be clearly labelled as such (or quite similar) and contain key pieces of information as follows:

Project title
Give the project a full title.

Investigators and other project personnel

List clearly all researchers.

Introduction to project and invitation to participate
Both introduce the project and invite participation.

What this project is about and why it is being undertaken
Sufficiently explain what the project is about, its aims, and why it is being conducted.

Project and researcher interests
Researcher and project interests should be sufficiently disclosed as applicable; for example, project is partly, mainly or wholly to satisfy the requirements for a student's VCE subject.

What participation will involve – time, effort, resources, costs
Give sufficient clear detail as to what is being asked of participants – voluntary consent to their time, effort, supply of information/body tissue/records/personal effects, etc. Avoid language that can be read as orders or directives (i.e. not 'You will do this or that'; but better as 'We will ask you', etc.) and presumptuous language (e.g. 'Dear Participant').

Participant rights and interests – risks and benefits/contingencies/back-up support
Outline realistically any potential risks (minimal or otherwise) and what preventative, minimisation or redress arrangements are in place. If some research questions or issues can be considered particularly sensitive, give sample questions or topics as an indication of the information that will be discussed or requested. Describe any benefits pertaining to individual participation or more generally. Avoid grandiose claims.

Participant rights and interests – free consent/withdrawal from participation
Participation should be voluntary, and free from any coercion or perceived coercion. Detail on this matter should be clear – that an individual is free to participate or not, and the circumstances. If, for example, the participants are students, patients or employees, it will help to clarify that their decision whether or not to participate will have no bearing on their results, treatment or employment (in some cases this may need further explanation, such as details about recruitment).

A statement about the participant's right to withdraw participation, data or material contributed, ordinarily without question or explanation, needs to be included.

Don't forget to outline how valid consent is to be obtained (by signed consent form, completion and return of an anonymous survey, any witnessing procedure if applicable, etc.). Often it helps to highlight or bold this part.

Participant rights and interests – privacy and confidentiality
Give clear information about secure arrangements for data access, collection, use, retention and/or disposal. This needs to comply with mandatory Australian Privacy Principles.

If signed consent forms are required, state whether they will be stored separately from any data collected and who will have access to them.

Remember, people are increasingly concerned about data access and data matching. Clear information will help allay any concerns.

Research output
Outline intended or anticipated publication or reporting of research findings. If need be, reiterate or refer to privacy arrangements for confidentiality/anonymity. Offer to make available any report or article or summary, where appropriate, and indicate how this will occur.

Further information about the project – whom to contact
You will need to nominate at least one person to contact regarding further information about the research activity or participation in the project.

FIGURE 1.13 Information that should be included in a participant information statement

Privacy

When undertaking research, it is very important to uphold the privacy of the participants. Privacy is a fine balance between their interests and those of researchers. Privacy laws attempt to stop inappropriate intrusion into the lives of individuals. Often, however, the problem is not the collection of data, but how the data is used or misused by people entrusted with it. To maintain privacy, and to **de-identify** the data, personal identifiers such as names and birthdates that are associated with individuals need to be removed so that information cannot be traced or identified.

Care must be taken when de-identifying data. The example of de-identified data in Table 1.5 only removes the names and dates of birth of survey participants. Depending on the remaining data and the information generated, this may be insufficient to protect

TABLE 1.4 Data that has been collected but not de-identified

Given name	Surname	Height (cm)	Weight (kg)	Sex	Birthday	Postcode	Age	Active
Charlotte	Micelli	145	48	F	22/10/2009	3103	11	No
Candace	Fock	153	40	F	10/10/2009	3105	11	Yes
Abdul	Mohammad	160	37	M	19/10/2008	3122	12	Yes
Eric	Dorian	142	38	M	12/12/2009	3040	11	Yes
Jaspreet	Singh	148	41	M	31/12/2008	3041	12	No
Sienna	Megane	152	42	F	2/12/2009	3103	11	Yes
Elie	Wu	144	43	M	6/01/2009	3122	11	No
Jo	Wu	149	47	F	6/01/2009	3122	11	Yes
Hamish	DiPaola	154	37	M	2/02/2009	3044	11	No
Jacinda	Black	143	46	F	4/04/2009	3111	11	No

TABLE 1.5 Data that has been de-identified. Personal information such as birthdates, first names and surnames has been removed so that data cannot be identified or traced.

Participant	Height (cm)	Weight (kg)	Sex	Postcode	Age	Active
ID1001	145	48	F	3103	11	No
ID1002	153	40	F	3105	11	Yes
ID1003	160	37	M	3122	12	Yes
ID1004	142	38	M	3040	11	Yes
ID1005	148	41	M	3041	12	No
ID1006	152	42	F	3103	11	Yes
ID1007	144	43	M	3122	11	No
ID1008	149	47	F	3122	11	Yes
ID1009	154	37	M	3044	11	No
ID1010	143	46	F	3111	11	No

individual research participants from being identified (even accidentally) by at least some users of the information. This is particularly likely if the users know who the members of the sample group are, or if they are familiar with the wider population that is being sampled. Users of the information may be able to identify a survey participant even after his or her name and date of birth have been removed. This could be caused by the presence of a characteristic known to be rare in the sample group or wider population, because this helps to narrow possibilities. Combinations of rare characteristics would amplify this effect, so that users of the information could make an intelligent guess as to a person's identity.

Physical and software security controls

Data must be stored in a way that preserves confidentiality and meets all legal requirements. Stored data can be protected with both physical and software-based controls, such as backing up of data and shredding of confidential documents.

Data needs to be securely retained and then securely disposed of or destroyed when no longer needed. How long the data or material need to be retained depends on the type of data. The minimum timeframe for data to be retained for research purposes is five years from the date of any published or reportable outcome based on the data. For some specific types of research, other timeframes may be required. For data such as identifiable health research data, the minimum period may need to be seven years. For identifiable health research data involving children, the data may need to be kept until the individuals involved turn 25, or five years after any research outcome based on the data is published or made available for release, whichever is the longer.

Physical security controls

Steps need to be taken to ensure that personal information is not inappropriately accessed. Hard copies of personal information need to be secured, and workspaces need to be designed to facilitate privacy. Consideration should be given to what measures are taken to control access. For example, swipe cards allow particular personnel into certain workspaces. Logs are kept of staff movements and their access to particular rooms, and security and alarm systems are used as a mechanism to control entry to the workplace. Those areas of an organisation where there is a greater need for access to personal information, such as human resources departments, should be physically segregated from other areas.

If the organisation requires an attendant to greet clients, consideration needs to be given to the placement of computers and computer monitors so that clients cannot see what is on the screen. Other measures include having designated areas for visitors so that they cannot see forms or hear telephone calls where personal information could be shared. Other physical controls include the provision of secure storage spaces near workstations to secure documents temporarily, and a place for securing physical files containing personal information. If files are placed in lockable cabinets, there needs to be consideration given to who has access and how access to keys is controlled.

Encryption

Encryption is the process of translating data into a code that can only be read by authorised users. To read an encrypted file, you must have access to a secret key that you use to **decrypt** the data. **Unencrypted data** is also known as 'plain text'. Data that has been collected for research purposes needs to be secured so that only authorised people have access. Encryption is one way to ensure that the data is secured. Encryption methods need to be reviewed periodically to ensure that they continue to be relevant and effective. Areas where encryption could be used include:

- databases that contain personal information
- servers, backups and cloud storage services
- mobile devices such as smartphones and tablets
- data storage devices such as flash drives and external hard drives
- data transferred on internal networks or over the internet – for example, using email or shared files.

Backups

Backups form an essential step in data management. Regular backups protect against a number of risks, including human error, computer crashes and software faults. Critical data files or data that are used regularly should be backed up frequently.

It is not unusual for someone to accidentally delete an important file or edit a document and later realise that some important information was removed. One strategy to minimise such loss of data by an authorised user is to maintain a backup system. Important files inadvertently lost can be retrieved from the backup media.

A **full backup** copies all of the files from a device to a storage medium. It can take considerable time and is usually performed once over a time period (such as a week, fortnight or month). A **differential backup** copies only those files that have been changed since the last full backup. Restoration of data would involve restoring files from the full backup and then from the differential backup. An **incremental backup** is similar to a differential backup, the difference being that it uses more than two backup media, while a differential backup uses only two media. An incremental backup only copies files that have been changed since the last incremental backup. It is the most complicated strategy from which to restore files since it requires restoration from a full backup and then from a series of incremental backups. It is a good practice to clearly label all backup media so that you know when the backup was made and what is on it.

Location of backup files

Once you have created backups, where do you put them? Ideally, your backups should be stored in a location that is safe from theft and damage caused by extremes of temperature or disasters. Most small businesses have a fireproof and waterproof safe in which valuable company documents are stored. This might also be used to store backups. It is preferable, however, to store backups at a remote location, perhaps even in the cloud. This means that if there is a large natural disaster, such as a flood or an earthquake, the backups will be safe.

One last point to remember is to ensure that backups actually work when you want to restore the data. It is important to test the effectiveness of your backup files by running a

The word 'cloud' is used because the internet was originally, and still is, represented by a cloud in network diagrams. An example of cloud computing is Google's Gmail, which uses cloud-computing processing powers and storage facilities.

disaster recovery simulation. If files cannot be restored from the backup or the system refuses to recognise them, it is better to discover this before a real emergency.

Cloud-computing companies provide offsite storage, processing and computer resources to individuals and organisations. These companies are typically third party (i.e. not one of the individuals or organisations directly concerned with the data) and they store data to a remote database in real time. The internet provides the connection between this database and the user's computer. One of the advantages of cloud storage is the ability to access data from any location that has internet access, which eliminates the need to carry a USB or a hard drive to retrieve and store data. The ability to share files with other people and to collaborate simultaneously, such as by using Google Docs, is also an advantage. Finally, if something were to happen to the computer, such as a fire or natural disaster, and the data on it were to be destroyed, having the data saved offsite in the cloud would prevent the data from being lost.

Google Docs, Sheets, Slides and Forms allows users to create, edit and collaborate with documents, spreadsheets, presentations and surveys (Figure 1.14). Users can edit files using a Google application and work on them at the same time as others, so they can read or make edits simultaneously.

FIGURE 1.14 A selection of applications from Google's suite of office tools

Usernames and passwords

A username is usually identifiable as belonging to a particular person and can be easily remembered. **Usernames** are uniquely assigned to users. **Passwords** are set by the user and should be known only to that user. To maintain high levels of security, user passwords should:

- be at least eight digits long
- include non-alphabetical characters
- not be easily guessed (for example, a favourite pet's name is not suitable)
- be changed every month.

Some network policies force users to change their passwords on a regular basis, and do not allow passwords to be repeated.

Password protection can also be placed on individual files to protect them against being viewed by unauthorised people, as shown in Figure 1.15.

FIGURE 1.15 Password protection has been placed on the file, preventing access to it unless a password is entered.

Firewall

Intrusion prevention and detection systems are designed to protect computer networks by effectively identifying and responding to known attack profiles. A **firewall** is one such mechanism to control incoming and outgoing network traffic. A firewall is a server and software combination that filters the information coming through an internet connection into an organisation's internal network. Any packet of data that is flagged by the filters as unwanted is not allowed through.

The filters used by a firewall may include examining the IP address of computers that request information from an internal server, blocking all access to certain domain names, banning certain protocols (e.g. file transfer protocol, mail protocol or Telnet protocol) from accessing particular servers, and searching for certain words and phrases included in packets of information, and blocking transfers that contain them.

A firewall can also be used to restrict employees' access to sensitive information. For example, a firewall can be used to stop some personnel from accessing the payroll database.

Most firewalls use two separate network interface controllers (NICs); one is connected to the internal network and the other to the outside world. Material can only move from one card to the other through the CPU of the server computer that is acting as the firewall. While the data or information is being checked for authenticity, it is also examined for viruses and other malicious codes. Everything that comes in from outside is examined for danger.

Malware protection

Malware is malicious software that includes spyware, adware, Trojan horses, worms and viruses. Spyware and adware use cookies to track the internet sites that a user might visit. Trojan horses can leave your computer open to others to read your personal information by creating backdoor access to your system. Viruses and worms can hijack your system to send multiple emails to others or perform other acts of mischief. Both can use up essential system resources, which may result in the computer freezing.

Network administrators usually require workstations to run virus protection software. The antivirus software is often updated automatically via the network. A firewall is also useful to block malware from sending personal information over the internet. Anti-adware programs should also be run on workstations.

Whitelisting and blacklisting

Whitelisting and blacklisting are ways of controlling the content, applications or entities that are allowed to run on, or access, a device or network. A **whitelist** is a list of those programs and IP or email addresses that are permitted to access your system. Anything that is not on the list is blocked. System administrators create such lists based on the needs and requirements of their users.

A **blacklist**, on the other hand, is a list of those entities that may not access your system. Many antivirus programs rely on a set of blacklists. Blacklists work on the premise that everything that does not appear on the list is permissible, whereas whitelists are more conservative and take the position that any address or domain not on the list is unsafe.

Both whitelists and blacklists can prevent unsafe material from accessing your system, but whitelists provide more protection.

Many cyberattacks start with a phishing or spoofing email that attempts to obtain users' personal information.

Australian Privacy Principles

Originally, the *Privacy Act 1988* (Cwlth) only dealt with the handling of data by Australian Government agencies. Many people criticised this limitation because it seemed that private organisations were not required to apply even the most basic of safeguards on data they collected. Even worse, there were no regulations preventing non-government organisations from collecting data by any method and using it for any purpose without the consent of the people whose private details were concerned. In particular, the rapid growth of electronic transactions, especially over the internet, led many people to demand some sort of legal protection from those who might gather data about internet browsing habits. The government was keen to encourage the development of electronic commerce while protecting the confidentiality of consumers and increasing public confidence in electronic transactions. These amendments have now been incorporated into the *Privacy Act 1988* (Cwlth) and are the most significant changes to privacy laws since the inception of the legislation.

Mandatory data breach notification amendments have also been passed.

There have been several additional powers included within this Act since 1988, but its main purpose has remained unchanged. The *Privacy Act 1988* (Cwlth) was amended by the *Privacy Amendment (Enhancing Privacy Protection) Act 2012*. This came into effect in 2014. As part of this Act, the Australian Privacy Principles replaced the National Privacy Principles and the Information Privacy Principles so that Australia now has one set of privacy principles. The Australian Privacy Principles (APPs) generally apply to Australian Government agencies. They do not apply to local councils or state or territory governments. Some states have their own privacy laws. For example, Victoria has the *Privacy and Data Protection Act 2014* (PDPA).

The changes to the Privacy Act include 13 privacy policy principles known as the Australian Privacy Principles (APPs) (Table 1.6). The APPs were devised to set out the standards, rights and obligations for collecting, handling, holding, accessing, using, disclosing and correcting personal information. The APPs oversee the handling of personal information by:

- Australian and Norfolk Island government agencies
- all private health service providers
- businesses that have an annual turnover of $3 million or those that trade personal information.

It should be noted that Section 6 of the Privacy Act defines 'personal information' as 'information or an opinion about an identified individual, or an individual who is reasonably identifiable'. This might include a person's name and address, medical records, bank account details, photos, videos and even information about what an individual likes, their opinions and where they work. Another important aspect of the Privacy Act refers to 'sensitive information'. Sensitive information is defined in the glossary and includes health information. The Privacy Act provides a higher level of privacy protection to sensitive information than to other personal information.

Of particular interest for this study are Principle 2, 'Anonymity and pseudonymity', Principle 6, 'Use or disclosure of personal information', and Principle 11, 'Security of personal information'.

APP 2, 'Anonymity and pseudonymity'

Australian Privacy Principle 2 provides individuals dealing with organisations the option of using a different name or a **pseudonym** in relation to a particular matter. This measure is in place so that individuals cannot be identified. In addition, individuals can also remain anonymous. For example, when an individual calls an organisation, often a message states that the call will be recorded for training purposes. If the individual objects, the call is not recorded. At a later date, when staff from the organisation receive training, the names of the individuals whose voices have been recorded must be changed in order to protect their identity when these real examples are used.

APP 6, 'Use or disclosure of personal information'

Australian Privacy Principle 6 states that the information that is being held is in line with the primary purpose it was intended for. Information cannot be used for a secondary purpose unless the holders of the information have received consent from the individuals concerned. For example, if a sporting organisation collected information about their players for the purpose of organising competitions and making the details available to the coaches and captains, then that would be the primary purpose of the data. However, if the sporting organisation wanted to provide the details of their members to a shop that specialised in sporting merchandise, then it could not do so, as this would be using the information for a purpose for which it was not intended.

APP 11, 'Security of personal information'

Australian Privacy Principle 11 refers to measures taken to actively secure personal information held, and also considers whether those who hold this personal information are permitted to retain it. Reasonable steps need to be taken to protect the information from misuse, interference and loss, as well as unauthorised access, modification or disclosure. Additionally, once there is no longer a need for any purpose, reasonable steps need to be taken to destroy or de-identify the personal information held.

TABLE 1.6 Australian Privacy Principles

APP 1	**Open and transparent management of personal information** Ensures that APP entities manage personal information in an open and transparent way. This includes having a clearly expressed and up-to-date APP privacy policy.
APP 2	**Anonymity and pseudonymity** Requires APP entities to give individuals the option of not identifying themselves, or of using a pseudonym. Limited exceptions apply.
APP 3	**Collection of solicited personal information** Outlines when an APP entity can collect personal information that is solicited. It applies higher standards to the collection of 'sensitive' information.
APP 4	**Dealing with unsolicited personal information** Outlines how APP entities must deal with unsolicited personal information.
APP 5	**Notification of the collection of personal information** Outlines when and in what circumstances an APP entity that collects personal information must notify an individual of certain matters.
APP 6	**Use or disclosure of personal information** Outlines the circumstances in which an APP entity may use or disclose personal information that it holds.
APP 7	**Direct marketing** An organisation may only use or disclose personal information for direct marketing purposes if certain conditions are met.
APP 8	**Cross-border disclosure of personal information** Outlines the steps an APP entity must take to protect personal information before it is disclosed overseas.
APP 9	**Adoption, use or disclosure of government related identifiers** Outlines the limited circumstances when an organisation may adopt a government related identifier of an individual as its own identifier, or use or disclose a government related identifier of an individual.
APP 10	**Quality of personal information** An APP entity must take reasonable steps to ensure the personal information it collects is accurate, up to date and complete. An entity must also take reasonable steps to ensure the personal information it uses or discloses is accurate, up to date, complete and relevant, having regard to the purpose of the use or disclosure.
APP 11	**Security of personal information** An APP entity must take reasonable steps to protect personal information it holds from misuse, interference and loss, and from unauthorised access, modification or disclosure. An entity has obligations to destroy or de-identify personal information in certain circumstances.
APP 12	**Access to personal information** Outlines an APP entity's obligations when an individual requests to be given access to personal information held about them by the entity. This includes a requirement to provide access unless a specific exception applies.
APP 13	**Correction of personal information** Outlines an APP entity's obligations in relation to correcting the personal information it holds about individuals.

Office of the Australian Information Commissioner website, www.oaic.gov.au, Creative Commons Attribution 3.0 Licence

Ethical dilemmas

Everyone wants the benefits of digital systems; however, intended and unintended negative effects can impinge upon people's rights. As a result, those who design, control and use digital systems have a responsibility to consider the real and potential negative effects and to eliminate or lessen them as much as possible. Sometimes, even this may not be enough to justify the proposed collection, creation or storage of data and information. It is important to take into account legal objections and ethical considerations when creating, acquiring and storing data and information. The purpose for collection needs to be clear. This also needs to be articulated in the participant information statements and the consent forms provided to the people from whom information will be sought. Questions of storage also need to be resolved so that data and information cannot be accessed by unauthorised users, and to specify the period of time for which it will be held.

Ethics are the accepted moral standards that govern the ways in which we behave. These standards may be common across a particular society or specific to a single organisation. They apply to activities that are questionable even if they are not illegal and are distinct from laws. Ethics often provide us with a set of guidelines for appropriate behaviour. If we choose to ignore these guidelines, we may not be committing a crime, but we may be sacked by an employer or shunned by society. For example, the impact of violent video games on children has long been debated. Some people have voiced their concerns that video game writers should not include animated violence in their games because it has a negative impact on children. A system of classification exists for games, similar to television and film classifications, but there is no legal restriction preventing these games from being created. These examples demonstrate how ethics hinge on society's values and standards. In the video game example, there are two competing principles. On the one hand, some would argue that protecting children from video games that might be harmful is the right thing to do. Others would argue that it is more important to maintain freedom of expression. Often, then, questions of ethics become debates over which of two principles is more important. Such conflict may be said to constitute a **dilemma**. This is especially the case when the consequences of action are open to debate or interpretation.

The standards or guidelines that determine whether an action is good or bad are what we call ethics. Ethics govern, among other things, the use of data collection. Ethical principles or guidelines often have an accompanying law, but the ethical principle is usually broader and the law applies only to certain circumstances or certain applications of the principle.

Anonymity of data may be of particular concern when deciding what must be removed to de-identify data sufficiently to protect all its potential users. It is also important when reporting personal information anonymously or using pseudonyms in a newspaper report, for example.

For example, it is ethical to obtain permission to publish photos of people on websites or in promotional material. Sometimes, people may object to their images being used for these purposes. The purpose for taking the photo and how it is intended to be used need to be made clear. Ethically it is wrong to use a photo for a different purpose from that for which it was originally collected. Similarly, when using data-collection tools such as surveys and interviews, it is important to reassure participants that the data provided, within the limits of the law, will remain anonymous and that others will not be able to identify their individual comments. It is not simply that it is important to put participants' minds at rest regarding their concerns about protecting privacy; it is also important to ensure that their privacy is in fact protected (and to ensure that non-participants in the larger group of which the sample is supposed to be representative are not put at unacceptable risk of suffering as a result of mistaken identification).

In addition to legislation such as the *Copyright Act 1968* and the *Privacy Act 1988* (see Chapter 6), many organisations have policy documents that mandate such things as who has access to different areas of the information system, and the penalties for misuse of information within an organisation. Ethical, legal or social tensions may arise within an organisation through a lack of clarity in such policy documents.

For example, a situation could arise in which an employee needs to perform a task on the network that requires approval from their manager. If the manager were away on sick leave, the employee might take it upon themselves to access the manager's account in order to get the task done. Although such an action is not illegal, it is unethical and it could result in disciplinary action for the employee.

The six steps outlined below for handling ethical dilemmas act as a framework that can be used within an organisation to solve legal, ethical or social tensions. This framework will provide support and guidance in making a decision when presented with a dilemma. Once the organisation has solved its problem, it may wish to update any policies concerned to ensure that the process for resolution is clearer.

1 Identify the problem: What decision has to be made and what facts are required?
2 Identify the stakeholders: Who are they? What interests do they have? What power do they have? Who is vulnerable? How are the vulnerable to be protected?
3 Identify possible alternatives: What options are available? What are the likely consequences?
4 Identify ethical standards: Are there any applicable laws? Are there any morals or standards that could be applied? Is there a precedent?
5 Evaluate options: Identify strengths and weaknesses. Identify the option that causes least harm. Can the decision be reversed?
6 Make a decision: Select the preferred option. Justify the option. Communicate the result of the decision to all stakeholders.

1

CHAPTER SUMMARY

Essential terms

accurate correct in all details and free from errors

array an indexed set of related elements. All elements within an array have a numeric index and are of the same data type

backup the process of copying files from an information system to some type of storage device to guard against possible data loss

bias prejudicial or unreasoned judgement

blacklist a list of applications, email addresses, IP addresses and websites that cannot access your network

Boolean data type a logical data type; Boolean data can hold one of only two possible values, usually true or false

character a data type representing a single letter, number, symbol, punctuation mark or space

closed (or closed-ended) questions queries that are restricted in the range of options provided so that only specific answers are elicited

comma-separated value (CSV) file a file type that stores data in a tabular format in plain text, with table columns separated by commas (delimiters)

data unprocessed, unorganised and distinct facts or ideas; in addition to text and numbers, data also includes sounds, images and video

data integrity the quality of the data

data structure a way of organising data so it can be used in an effective way

data types different forms that variables and data may take, that determine the data that a variable may contain, and how the data or variable may be manipulated

decrypt to translate encrypted data back into ordinary text that can be read by anyone

de-identify to remove information from data so an individual cannot be identified

differential backup used in conjunction with a full backup; only files that have been altered since the last full backup are copied; restoration requires the full backup to be restored first, followed by files from the differential backup

dilemma when people must choose between two (or more) equally desirable (or undesirable) options; for example, between allowing the sale of violent video games to preserve freedom of expression and banning their sale in order to protect children from possible harm

encryption the process of encoding or changing data so that an unauthorised user who reads the data would not be able to understand it

ethics accepted moral standards that guide behaviour; these standards may be common across a particular society or specific to a single organisation, and they apply to questionable activities over and above any legal requirements; ethics often provide us with a set of guidelines for appropriate behaviour

exponent the power of 10 by which the mantissa in a floating-point number must be multiplied to regain the original number

firewall hardware and software that restrict access to data and information on a network

floating-point number a number with a fractional or decimal component

focus group a small group of individuals who are carefully selected, so that each one fits a particular demographic and the researcher can obtain the necessary data through a guided discussion that probes the participants' attitudes about the topic

footnote a reference that is listed at the bottom of the page on which the citation is made

full backup copying all chosen files to a backup device; it can be slow to perform, but is the easiest and quickest form of backup from which to restore data

9780170440806

incremental backup similar to a differential backup in that it works in conjunction with a full backup, and only backs up files that have been altered since the last incremental backup, but it uses more than two backup media; it is the most complicated strategy from which to restore files

information processed, organised and value-added data, which can be paper-based (hard copy) or digital (soft copy)

informed consent a necessity for all participants before agreeing to take part in research; participants must be informed of what the research involves, the time commitment expected and any possible risks that may arise

integer a number without a fractional or decimal component

interview a conversation, usually between two people, in which questions are asked and answers are given

malware short for 'malicious software'; programs designed to infiltrate and cause damage, disruption or access to a device or network without the user's knowledge or consent; includes viruses, worms, Trojan horses, adware, spyware, logic bombs and keyloggers

mantissa all of the digits in a floating-point number, with a decimal point after the first digit that is not zero

observation a way of finding out about the world around us by using our five senses

open-ended questions queries that allow people to answer in the manner they wish

participant information statement a document that provides participants with information about the research in an unbiased way, and also provides the scope to answer questions that the participant may have

password a secret term used to identify the user

plagiarism passing off someone else's work as your own

precision being exact

primary sources original, unprocessed data and resources; that is, information that has not been processed, analysed or interpreted in any way, such as interviews, speeches, emails, debates and meetings; primary data usually comes from stakeholders

pseudonym a fictitious name that is given to a person, or that is chosen by a person, to hide or protect their identity

qualitative data collected data that is based on subjective data-collection techniques such as interviews, focus groups, video footage and observation

quantitative data collected data that is measurable and specific; quantitative data gathering is based on verifying a theory through the use of statistics and data that is largely numerical

record an indexed set of related elements. All elements within a record have a field index and may be of different data types

referencing citations in a document that assist readers to know where to locate the source of an original idea or quote in a piece of work, and assist students to avoid plagiarism

relevant appropriate to the discussion

reliable able to be trusted

research question a question that enables a researcher to narrow the focus of the topic of the investigation

secondary sources sources of information that has been processed, interpreted or analysed in some way by other people, such as textbooks, websites, magazines, newspapers and TV programs

stakeholder an individual or group who either has an interest in, or is affected by, the decisions and actions of an organisation

string a data type representing a set or sequence (collection) of characters

survey usually a set of questions that ask for a response from a list of alternatives, such as A, B, C, D, or from a range, such as 1–5 or very low to very high; surveys can easily be given to many people, and are easily processed and analysed using computer-based methods because the answers can be recorded as numbers

theory a general statement that describes something, provides an explanation of why something happens and can be applied to predict what will happen in the future

unencrypted data data that is not protected by encryption and can thus be read by anyone; also known as 'plain text'

username the name given to the user on a computer or computer network

variable in programming, a key word, phrase or symbol that represents a value that may change

9780170440806

vested interest arises when an individual, group or organisation has a strong personal interest because there is an advantage to be gained

whitelist a list of those programs and IP and email addresses to which the administrator has granted access to a system or network

Important facts

1. **Data** must be relevant to produce usable information. Data needs to be processed while it is current because decision making should not be based on outdated data.
2. Data that is entered into a computer must be **accurate**. Transcription is often a cause of error. Transcription errors occur when the person entering the data misreads the information through, for example, a lapse in concentration or being interrupted, or presses the wrong key.
3. **Interviews** are usually done one-to-one, but can sometimes be done in groups, and can take a substantial amount of time. A major feature of an interview is the opportunity for in-depth follow-up and clarification questions that cannot be done with surveys, which are often answered in private. Interviews are very useful for eliciting the feelings, attitudes and opinions of people that are too complex to easily record in a survey.
4. **Bias** can infiltrate data if the respondent to a survey or interview has a **vested interest** in the outcome of the research, if the timing of the data gathering is inappropriate, or if the chosen sample size is too small.
5. **Timing of events** needs consideration when collecting data as it can cause skewed results, which can lead to inaccurate or misleading conclusions.
6. **Sample size** must relate to the purpose of the data collection. Generally, a larger sample size leads to greater precision.
7. There is a plethora of unchecked information on the internet; however, some of the views presented may not be widely accepted or proven. Sources cited should be **reliable**.
8. The **American Psychological Association (APA)** created a style guide to assist with academic writing such as essays, books and other publications. The APA style is widely used and is the style that students are expected to reference.
9. **Privacy** is a fine balance between the interests of researchers and those of the participants. Privacy laws attempt to stop inappropriate intrusion into the lives of individuals. Often, however, the problem is not the collection of data, but how the data is used or misused by people entrusted with it.
10. Researchers must ensure that data and materials generated and collected as part of their research, regardless of the format, are stored securely in a durable and accessible form. Stored data can be protected with both physical and software-based controls, such as **backing up of data** and **shredding of confidential documents**.
11. **Cloud-computing** companies provide offsite storage, processing and computer resources to individuals and organisations. These companies are typically third party and they store data to a remote database in real time.
12. To verify users' rights to access a network, security features are required. A system of establishing **usernames** (or user IDs) and **passwords** allows for the identification and authentication of each user.
13. The ***Privacy Act 1988* (Cwlth)** was amended by the Privacy Amendment (Enhancing Privacy Protection) Bill in 2012. This came into effect in 2014. As part of this Act, the Australian Privacy Principles (APPs) replaced the National Privacy Principles and the Information Privacy Principles so Australia now has one set of privacy principles. The APPs apply to Australian Government agencies.
14. **APP 2, 'Anonymity and pseudonymity'**, offers individuals dealing with organisations the option of using a different name or a pseudonym in relation to a particular matter.
15. **APP 6, 'Use or disclosure of personal information'**, states that the information that is being held is in line with the primary purpose it was intended for. Information cannot be used for a secondary purpose without consent from the individuals concerned.
16. **APP 11, 'Security of personal information'**, states that reasonable steps need to be taken to protect personal information an organisation holds from misuse, interference and loss, and from unauthorised access, modification or disclosure. The organisation also needs to destroy or de-identify personal information in certain circumstances.

9780170440806

TEST YOUR KNOWLEDGE

Review quiz

Data and information

1 What can occur if information is produced from incorrect or incomplete data?

2 Why is it important to ensure that data is accurate?

3 What are the properties of usable data?

4 What is a common cause of inaccurate data?

5 How can the accuracy of a primary source be determined?

6 With an example, explain the importance of timeliness in ensuring the quality and usability of data.

7 What influences the introduction of bias into data?

8 What is the difference between quantitative and qualitative data? Provide two examples.

9 What strategies could be used when gathering quantitative data?

10 Provide an example of referencing based on the APA style.

11 Why is it important to obtain permission when collecting data?

12 What is the purpose of consent forms?

13 Why is encryption important in data security?

14 How do usernames and passwords protect data?

15 What makes a strong password? Provide an example of a very strong password.

16 Describe a strategy for backing up data.

17 What is the difference between a full backup and an incremental backup?

18 Why is it important to secure data when conducting research?

Interactions and impacts

19 How do the Australian Privacy Principles affect the individual?

20 Under which legislation do the Australian Privacy Principles fall?

21 Why is it important to de-identify personal data?

22 What is an ethical dilemma in the context of data collection for research purposes?

9780170440806

APPLY YOUR KNOWLEDGE

Street traffic

1 Milorad believes that there is too much traffic in his street, particularly on weekdays – in the morning and evening. Milorad lives in a small residential street, but cars use this street as a short cut to avoid traffic lights and the main roads. Milorad wants to do some research to support his theory about the traffic on his street, and present a case to the local council.

- a Clearly state the topic Milorad will investigate.
- b What type of data will Milorad need to collect to assist in his investigation?
- c Identify an appropriate data-gathering technique Milorad could use.
- d Justify the selected data-gathering technique.
- e How will Milorad keep the data safe?
- f Does Milorad need to get permission to conduct his research and, if so, from whom?
- g What tools will Milorad use to interpret the results?
- h What types of relationships and patterns is Milorad looking for?
- i How will Milorad present the data to the local council?

Internet usage

2 Go to the UN Data website and select 'Datamarts'. Then find the World Telecommunication/ICT Indicators database from the list of all the data sets provided. Select to view the data on the percentage of individuals using the internet.

- a Filter the data to show only the value of countries using the internet in 2010–14 from 10 countries. It is best to select from a mixture of countries to highlight the differences between them in levels of access to the internet.
- b Copy and paste this data into a spreadsheet program.
- c Create a column chart and a scatter diagram chart to depict these statistics.
- d Discuss which graphic representation best conveys the data and why.

UN Data

9780170440806

CHAPTER

2 Data visualisations

KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

Approaches to problem solving

- structural characteristics of spreadsheets and databases, such as cells, fields, records and tables
- types and purposes of data visualisations suitable for educating, entertaining, informing and persuading audiences
- functional and non-functional requirements of solutions, constraints and scope
- design tools for representing the functionality and appearance of databases, spreadsheets and data visualisations, such as annotated diagrams and mock-ups
- formats and conventions suitable for databases, spreadsheets and data visualisations
- software functions and techniques for efficiently and effectively manipulating, validating and testing data to develop databases, spreadsheets and data visualisations

Reproduced from the VCE Applied Computing Study Design (2020–2023) © VCAA; used with permission.

FOR THE STUDENT

If you can imagine the amount of data that is generated every day just by using social-networking tools, you might also be able to imagine that there is someone, somewhere, who is looking through a mountain of data for meaning. Data visualisation is the process by which we take large amounts of data and process it into effective graphical representations that will meet the needs of users or clients. These representations can take the form of charts, graphs, spatial relationships and network diagrams. In some cases, the data visualisation might involve interactivity and the inclusion of dynamic data that allows the user to deduce further meaning from the visualisation.

FOR THE TEACHER

This chapter introduces students to the knowledge and skills needed to use software tools to access authentic data from repositories and present the information in a visual form.

The key knowledge and skills are based on Unit 1, Area of Study 1. If a data visualisation is effective, it reduces the effort needed by readers to interpret the information. This chapter takes students through the different types of visualisations, and then uses a case study to explore some of the tools available to process data from various sources and repositories, such as the domestic energy consumption records, the Bureau of Meteorology and the Australian Bureau of Statistics.

Shutterstock.com/solarseven

9780170440806

What is a data visualisation?

Data visualisation results from using software tools to access and select data from large data repositories, and then present that data in graphical form. This can be in the form of graphs and histograms, charts and maps, or spatial relationship diagrams and network diagrams. This type of graphic representation helps the viewer to identify patterns and relationships in vast amounts of data that would not be identifiable in the raw data. Data visualisations can be still (**static**) or moving, or updated regularly or in real time (dynamic). Emerging examples using virtual reality (VR) and augmented reality (AR) allow viewers to be immersed within the data to observe patterns and relationships.

Science Photo Library/Sovereign/ISM

FIGURE 2.1 The human brain has a preference for visual information. The primary visual cortex (yellow) and the secondary visual cortex (pink) are highlighted here. The visual cortex receives and processes visual input from the retina.

Virtual reality and augmented reality are discussed in Chapter 6.

While there are many ways to convey information based on data, a visual method is the most effective. Extensive research has consistently supported the proposition that humans gain better understanding by visual communication. The human brain has developed to interpret visual data quickly and accurately.

There are two main aspects of visualisation communications:

- *Information visualisation* deals more with ideas and concepts.
- *Data visualisation* offers a graphic representation of complex **data sets**.

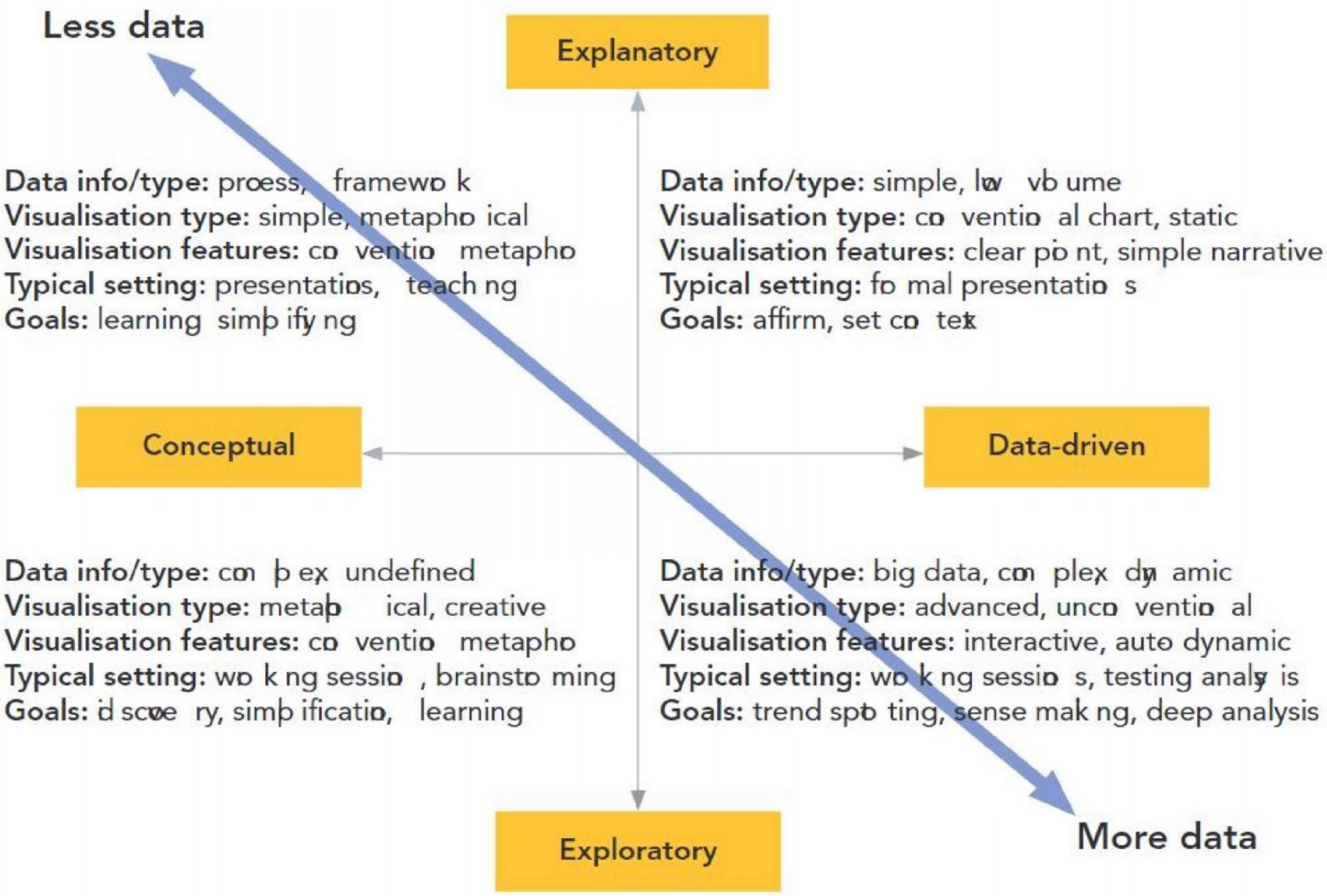


FIGURE 2.2 Data and information may be visualised in several ways.

Purposes of data visualisation

There are many purposes for a data visualisation. A successful chart could satisfy many needs by simultaneously:

- educating to develop understanding
- entertaining to amuse and distract or provoke a response
- informing to convey information
- persuading to encourage adoption of a point of view or opinion.

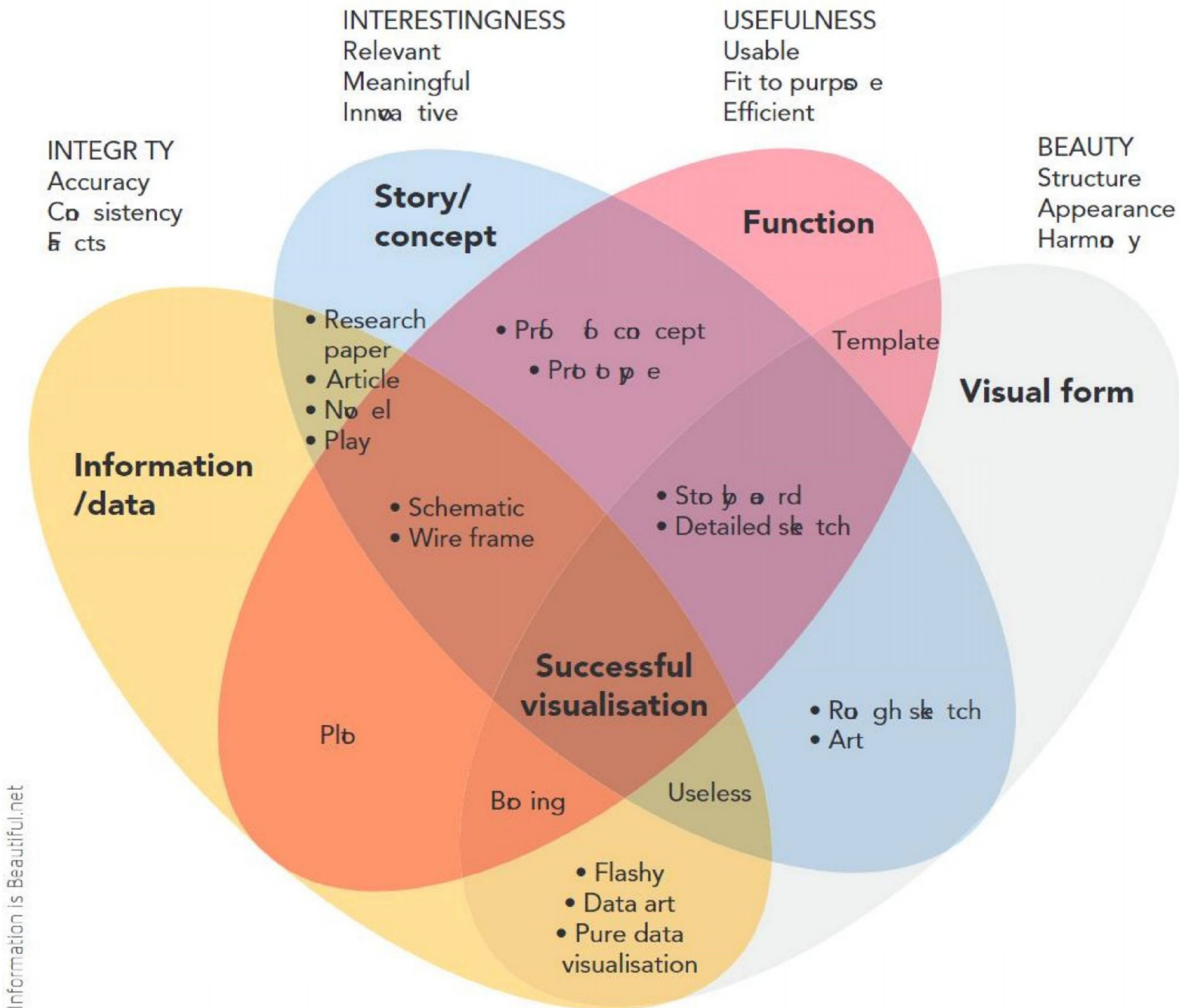


FIGURE 2.3 What makes a good visualisation?

Categories of data visualisations

Data visualisations can be loosely described in two categories:

- *explanation* – to convey an existing point of view. This style may also be persuasive, inviting the viewer to reconsider a previously held point of view
- *exploration* – allows for some user choice. The views are selected from a range of options. The combinations of views, while necessarily limited, are not predetermined. The viewer may locate and further interrogate a data item of interest.

Explanatory

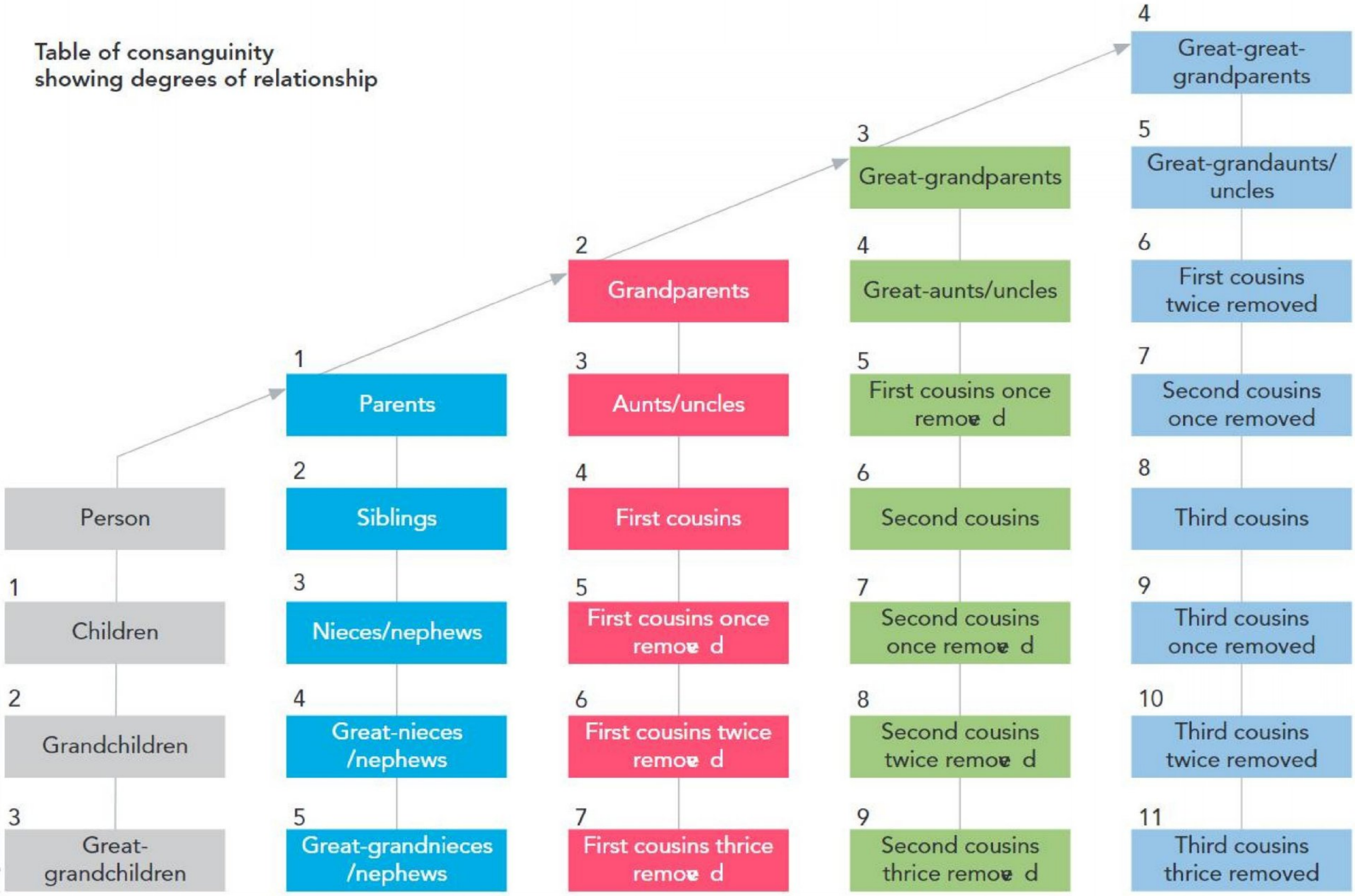
An explanatory data visualisation sets out to explain a point of view. Features of this type of visualisation are that it is:

- clearly expressed
- simply laid out
- based on known facts.

Spelling varies according to whether AU/UK or US English is being used. In this text we apply the AU/UK spelling 'visualisation', rather than the US spelling 'visualization'. Curiously, when abbreviated, the AU/UK 'dataVis' is consistent with both spellings, while the alternative 'dataViz' is not.

Exploratory

Exploratory visualisations allow users to select a value or category to explore connections and relationships. An example is shown in Figure 2.6, which is an on-screen interactive visualisation of student progress and results. The relationships between submitted work, progress and grades for the entire class group can be combined onto one screen. The hover tooltip reveals further details on each record. As further records are added, the display is recalculated and updated.

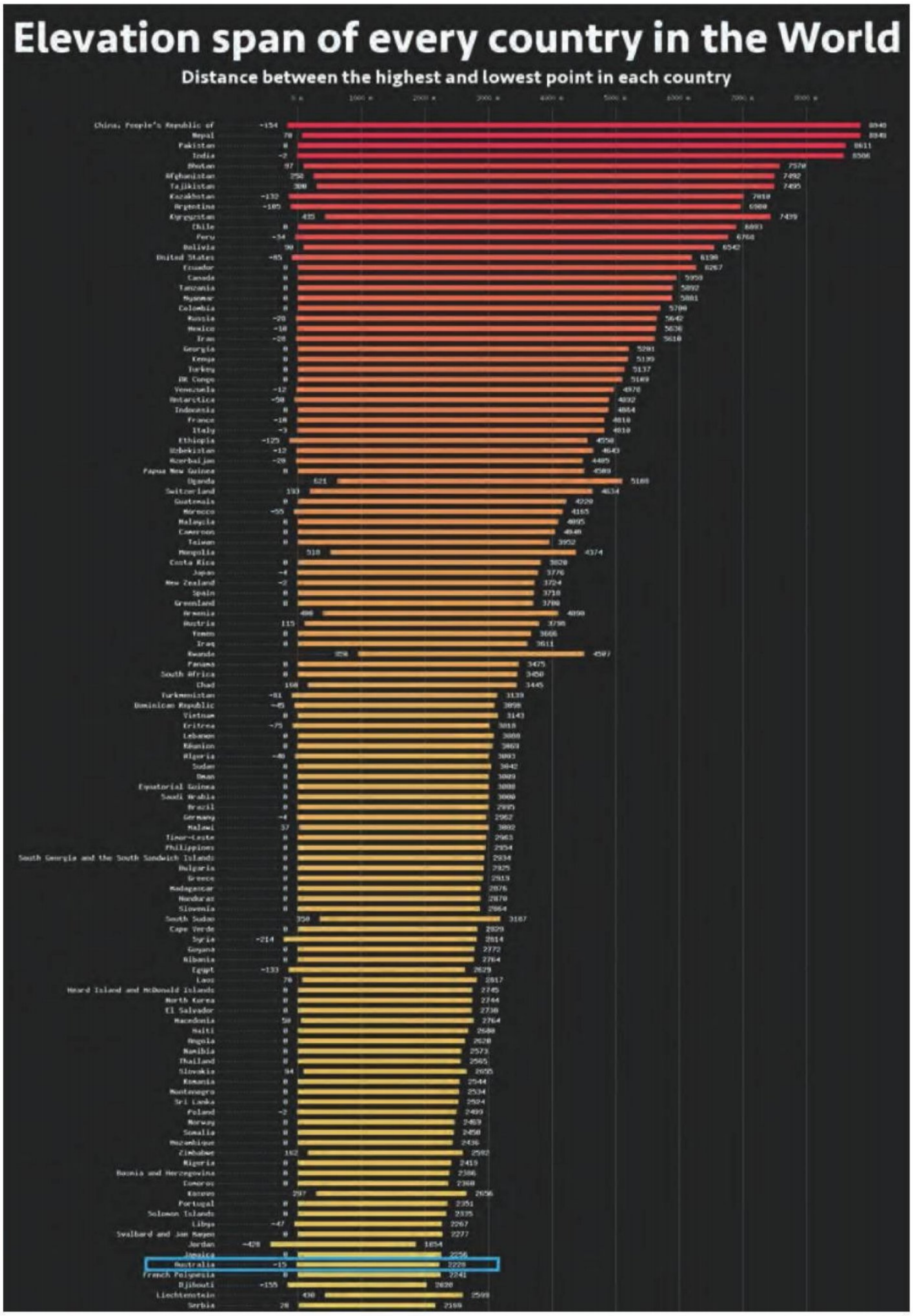


Adapted from https://commons.wikimedia.org/wiki/File:Table_of_Consanguinity_showing_degrees_of_relationship.png Creative Commons 3.0

FIGURE 2.4 There is very little number data here. Consanguinity shows the degree of separation from a person of interest across a family's generations, known as 'how many times removed'.

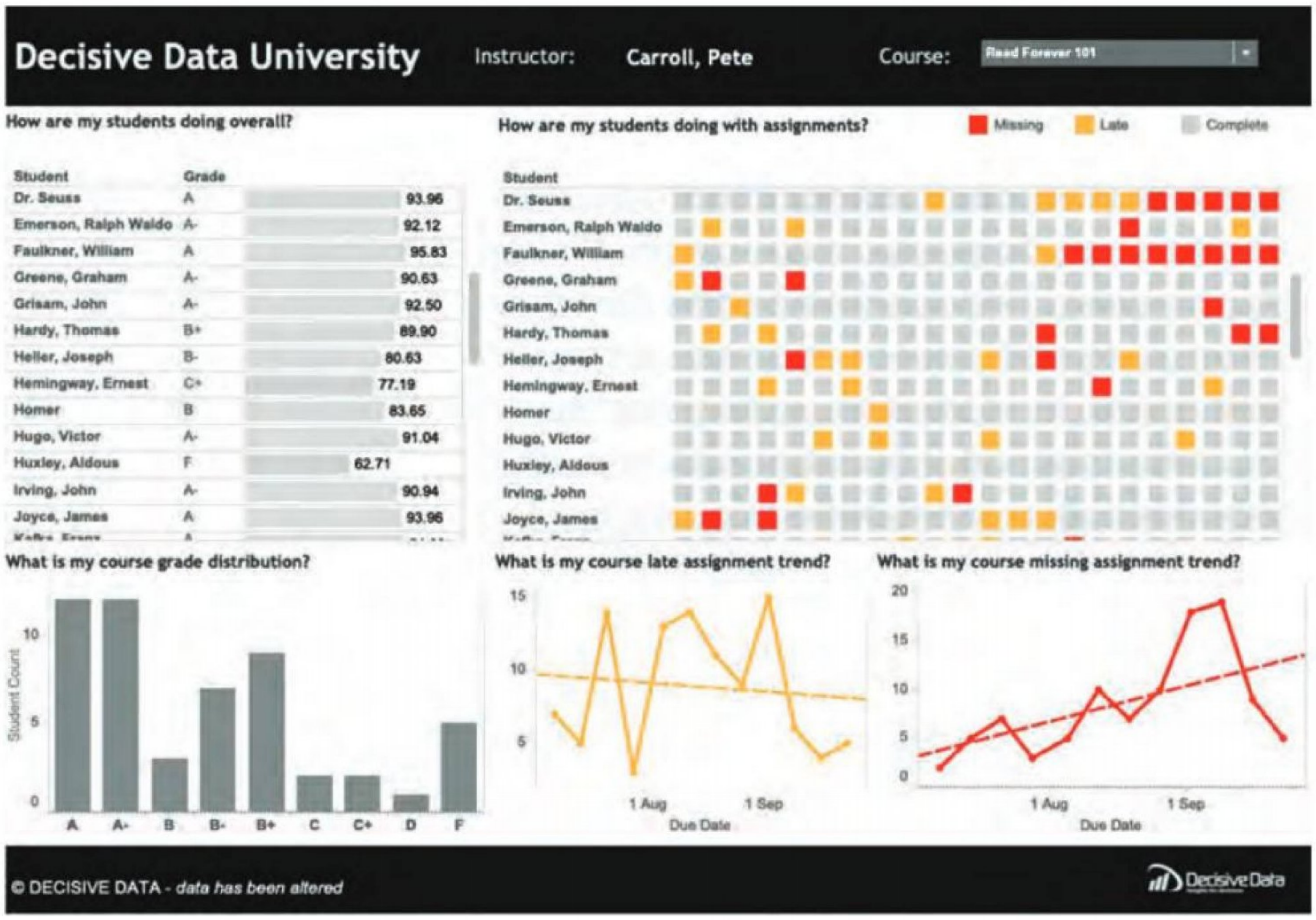
Informing by demonstration or exhibition

Demonstration or exhibition visualisations allow for demonstration of relative data values and comparison between categories. On-screen, these may be interactive to allow for user input and direction.



https://www.visualcapitalist.com/wp-content/uploads/2018/09/elevation-spans.html, used with permission

FIGURE 2.5 Complex data sorted and shown in descending data order. Australia is indicated by contrasting blue outline box.



https://www.decisivedata.net/k12-classroom-evaluation

FIGURE 2.6 An interactive dashboard with data visualisation. This Tableau demonstration has hover tooltips for each element to provide further detailed information.

Types of data visualisation

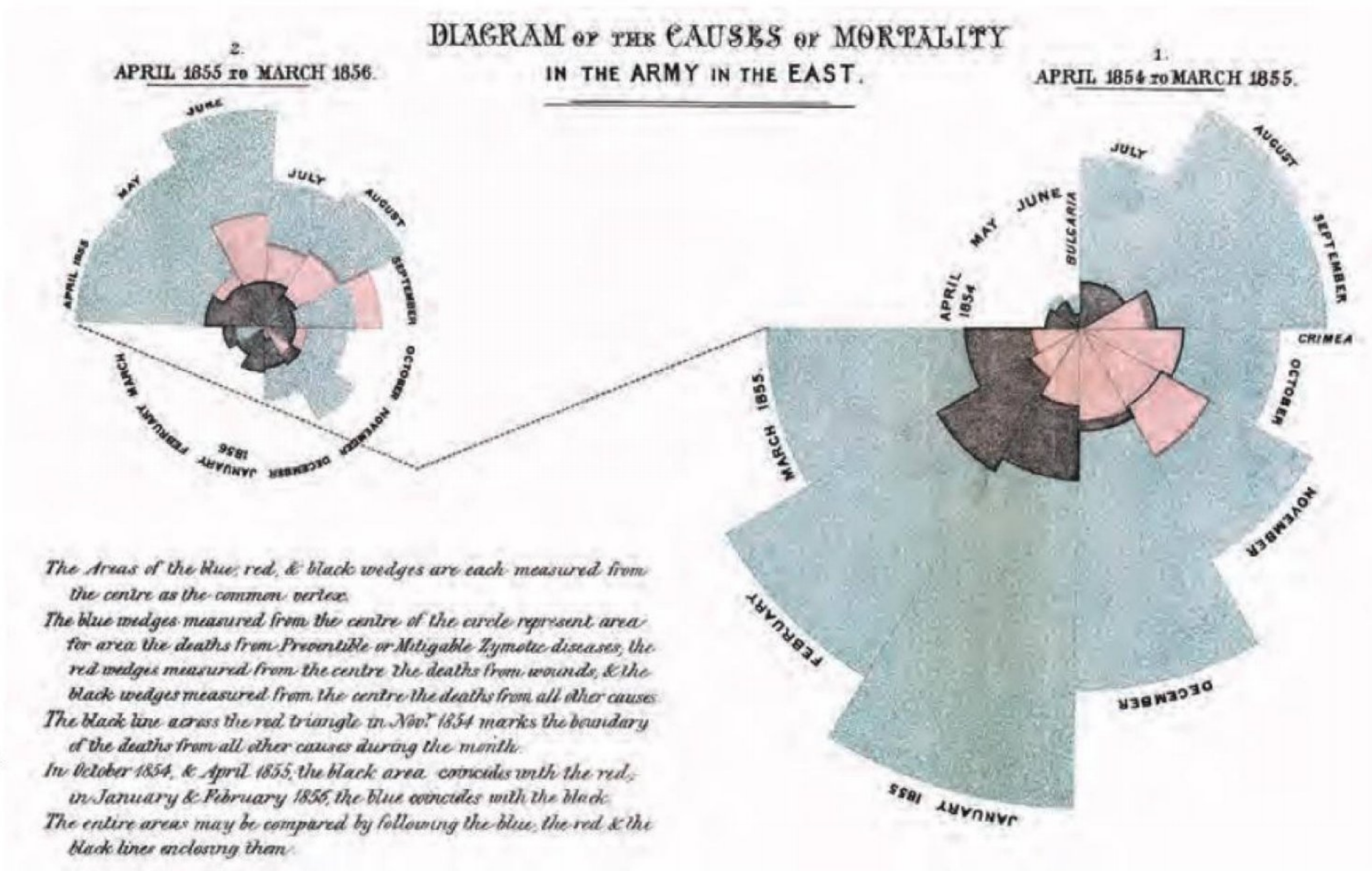
Data visualisation is a relatively new field of computing. As more and more data sets are made available, the number of individuals and organisations performing analysis on this data is also increasing. The variety of visualisations being used is endless. Data is being manipulated and processed into new forms of visualisations every day.

Any graphic that displays and explains information, whether data or words, can be described as a 'data visualisation'. Data visualisations can take many forms. The elements used to convey the message can include:

- graphics
- images
- video
- audio
- animations
- data
- text
- charts.

The first charts were produced in response to a need to convince others about a situation. The visual presentation was a means of showing numbers that were otherwise too difficult to follow and whose significance was therefore lost.

Florence Nightingale, for example, was not only a nurse, but a statistician and a social reformer. She made use of a **rose chart** or **coxcomb chart** – a combination of stacked bar and pie chart used to convey proportion and amount in one image (see Figure 2.7). She used this to convince hospital administrators to change hygiene practices during the British war in the Crimea (1853–56).



'Diagram of the causes of mortality in the army in the East' by Florence Nightingale, in *Notes on Matters Affecting the Health, Efficiency, and Hospital Administration of the British Army*, 1858

FIGURE 2.7 Florence Nightingale provided persuasive rose chart analysis arguing that disease caused more deaths than wounds in the Crimean War (1853–56).

CASE STUDY

Chart with a story

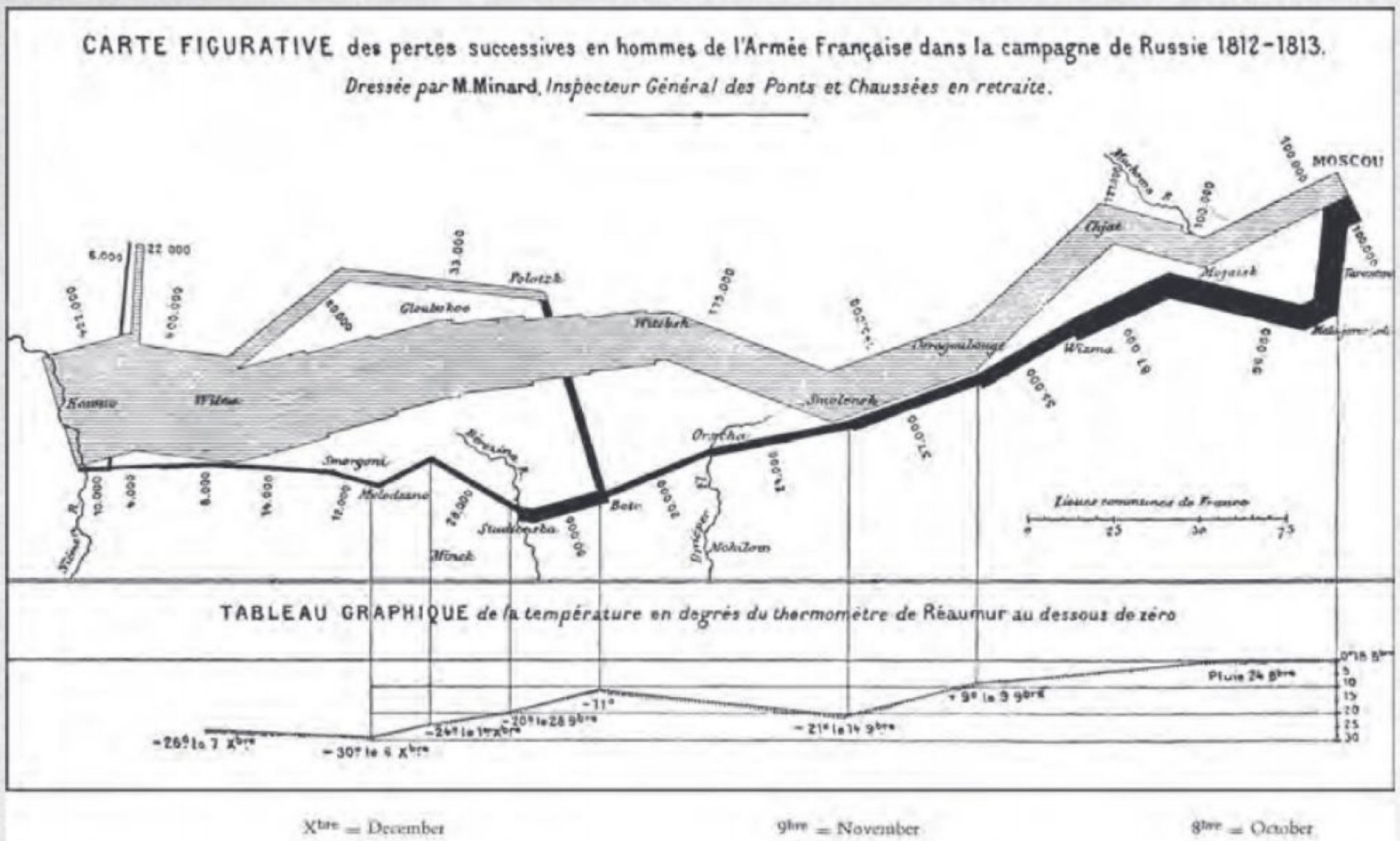
Widely regarded as the best graph ever drawn, Charles Minard's visual representation of Napoleon's march to Moscow (Figure 2.8) gives the user information on several levels at once:

- geographical
- climatic
- numerical.

The top line represents the troops moving (from left to right) towards Moscow in 1812–13. The figures on the side of the line show the dramatic fall in troop numbers, and the corresponding thinning of the line shows the fall visually. It can be seen that each time Napoleon crossed a river, he lost large numbers of troops. The line at the bottom of the graph shows temperature. As the temperature continued to drop, still more troops were lost.

The black line shows the troops retreating (from right to left). One can only speculate about how different history might have been if Napoleon had had Minard in his midst during his troop campaign in Russia.

Thirty years later, Matthew Henry Phineas Riall Sankey used this visualisation for energy flow. This type of graph came to be called a **Sankey diagram**. These band graphs are routinely used for illustration of flows.



Charles Minard, Inspector General of Bridges and Roads in retirement, Paris, 20 November 1869

FIGURE 2.8 Charles Minard's map showing the French army's disastrous Russian campaign of 1812. The graphic represents six types of data in just two dimensions: the number of Napoleon's troops; distance; temperature; the latitude and longitude; direction of travel; and location relative to specific dates.

Charts

A **chart** (also known as a graph) is a method of displaying data visually, where the data set is represented as symbols in the chart. Many spreadsheet applications have charting (or graphing) capabilities. Features of a chart can include a title, axis, scale or grid, data labels and a legend.

Charts are often used to visualise numerical data. There are a range of chart types, and each type is used for different purposes. A bar chart can be used to compare different items, a pie chart to show each data item as a proportion of the population, line charts are useful for showing the trend in a data item over time and histograms are useful for grouping data, then showing the frequency of each group. New types of charts are regularly developed that have not been seen previously as animation and presentation technologies allow greater functionality.

Which chart to choose?

It is your choice how best to present the data you have selected to present, illustrate and support a statement of your findings. There is no best way, but there are other, better ways of presenting the data and information.

In making your decision on how to present the data, consider how you will:

- convey a message or meaning
- identify trends or specific information
- illustrate an aspect worthy of attention
- provide an overall view of the data.

Different types of charts have advantages over others for achieving clearer communication. Figure 2.9 shows different types of charts used for different purposes, but these only scratch the surface in terms of types of data visualisations.

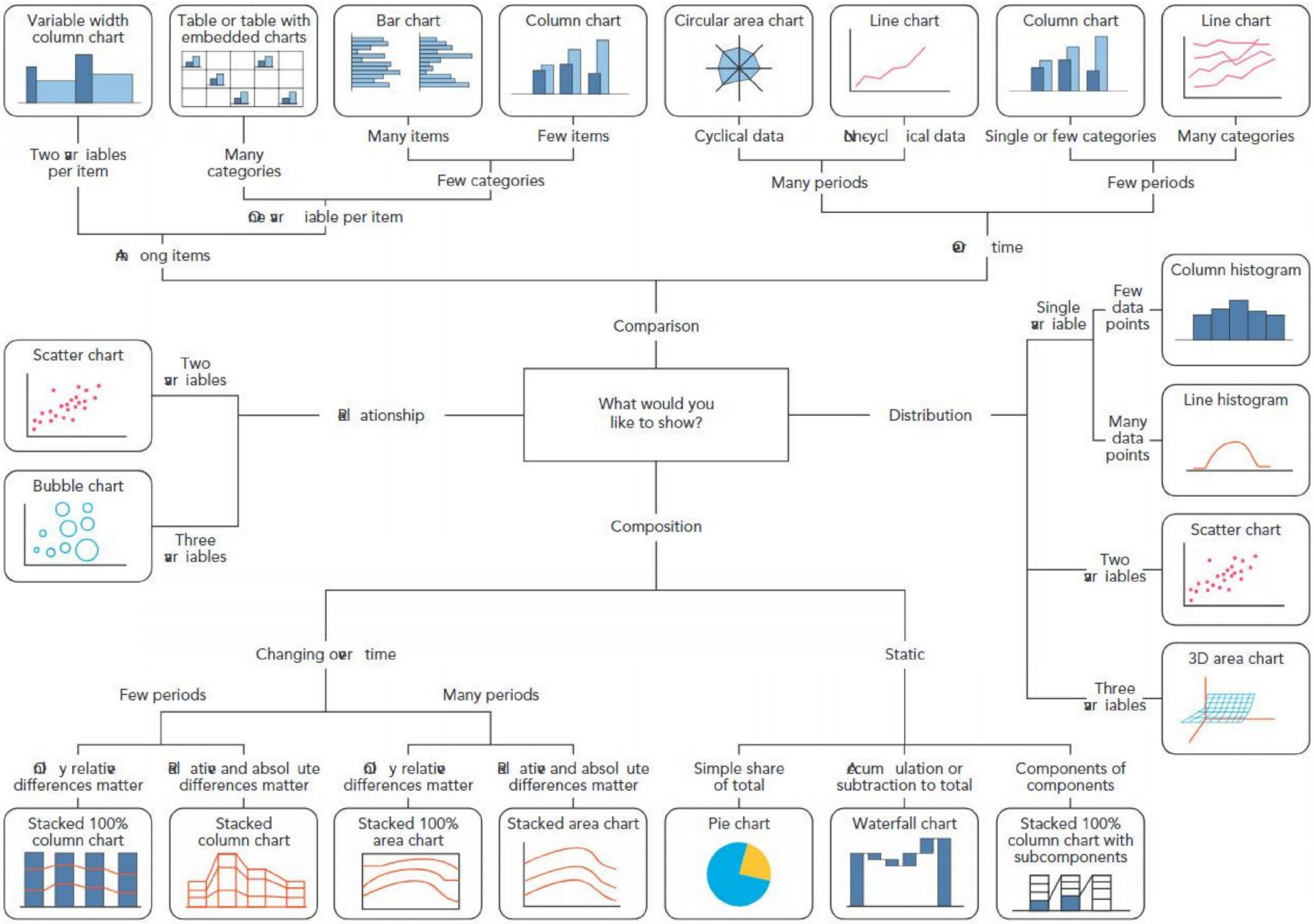


FIGURE 2.9 Some common types of data visualisations; network diagrams and maps are not shown.

Advanced Presentations by Design: Creating Communication that Drives Action, 2nd Edition, Andrew Abela, May 2013, Pfeiffer

Qualitative Chart Chooser

Qualitative data presents some difficulties as charts using standard axes do not easily display such data in a meaningful way. The Qualitative Chart Chooser website has a gauge to determine which chart may be appropriate.

Maps

Interactive 3D globe

A popular method of displaying geographical data is to use **map-based visualisations**. These types of visualisation are often called **geospatial visualisations**. Geospatial data is data that is related to the geographical location covered. Data could be related to population, roads, rivers, climate, mobile phone towers or any other characteristic of the area. Many geospatial visualisations are dynamic and allow the user to zoom in or out or navigate over an area. Geospatial visualisations are becoming more popular because they are a very powerful tool that allows the data to be brought to life through visualisation. As a range of data

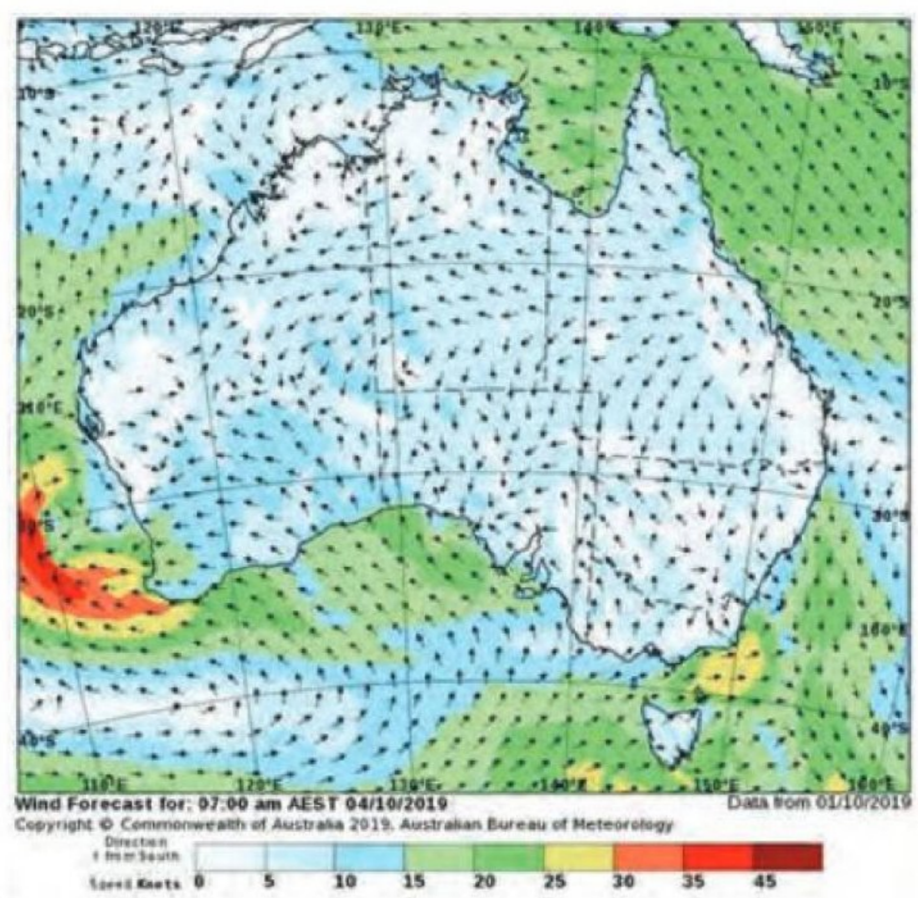
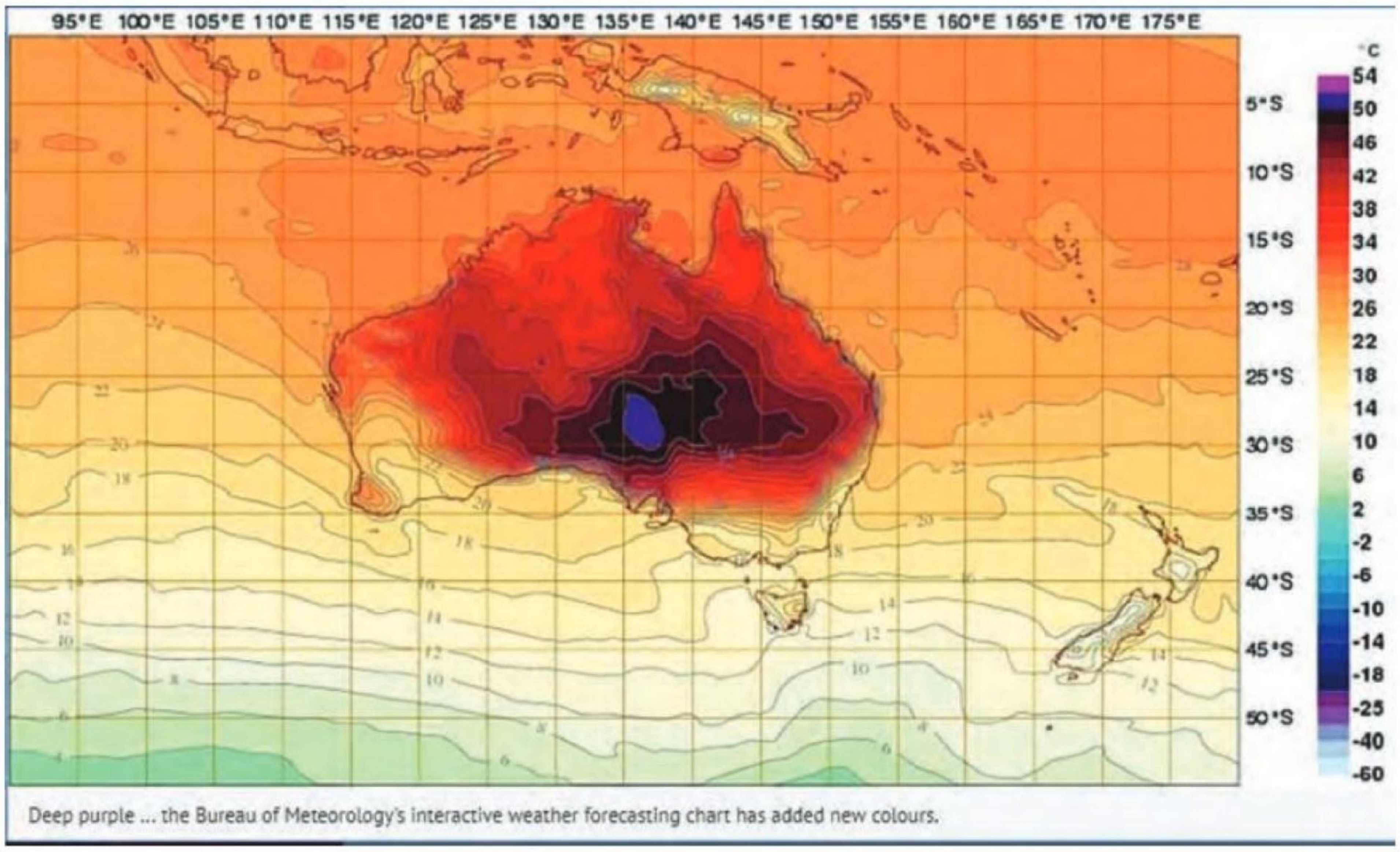


FIGURE 2.10 This map from the Bureau of Meteorology (BoM) shows predicted wind direction and speed.

Bureau of Meteorology www.bom.gov.au CC BY 3.0 AU

can be overlaid with a geographical location, the uses for these types of visualisations are many. They have been commonly used for agricultural, environmental, mining and urban planning purposes, but the list is endless.



Bureau of Meteorology, www.bom.gov.au CC BY 3.0 AU

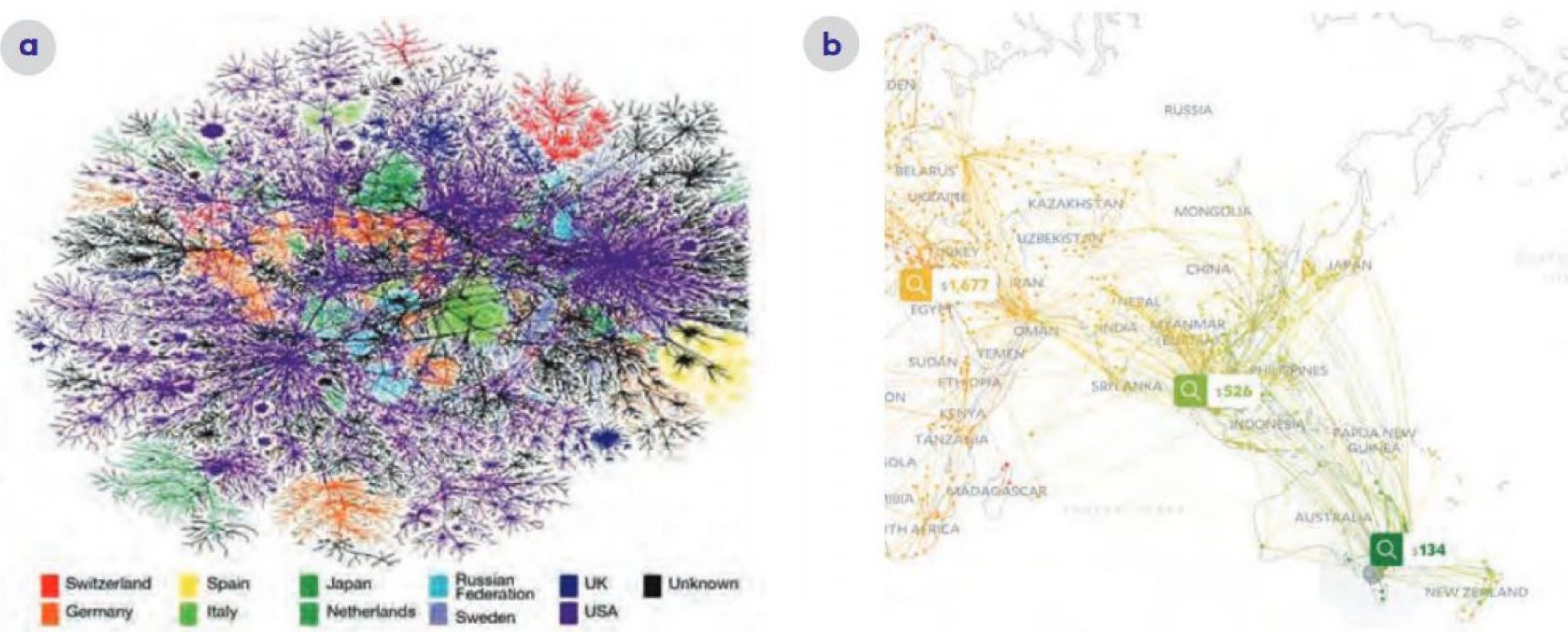
FIGURE 2.11 An example of geospatial visualisation. The legend for BoM weather forecast maps was modified in 2013 to include mauve for temperatures above 50°C.

Geospatial visualisation

Networks

Network visualisations show relationships between different data items and relationships between different data sets. Finding relationships within and between data sets has been an area of increasing interest in recent years as more data has become publicly available from both government and private organisations around the world.

A network visualisation might show the frequency with which individual players might pass the ball to each other in a football game or the number of people who travel on a public transport system each day. Network visualisations are also used to represent the layout of computer networks and public transport systems. Figure 2.12 shows examples of network visualisations.



UC Regents/B. Huffaker Image created using graph layout code provided by B. Cheswick and H. Burch.

MIT Senseable City Laboratory, greatescape.co

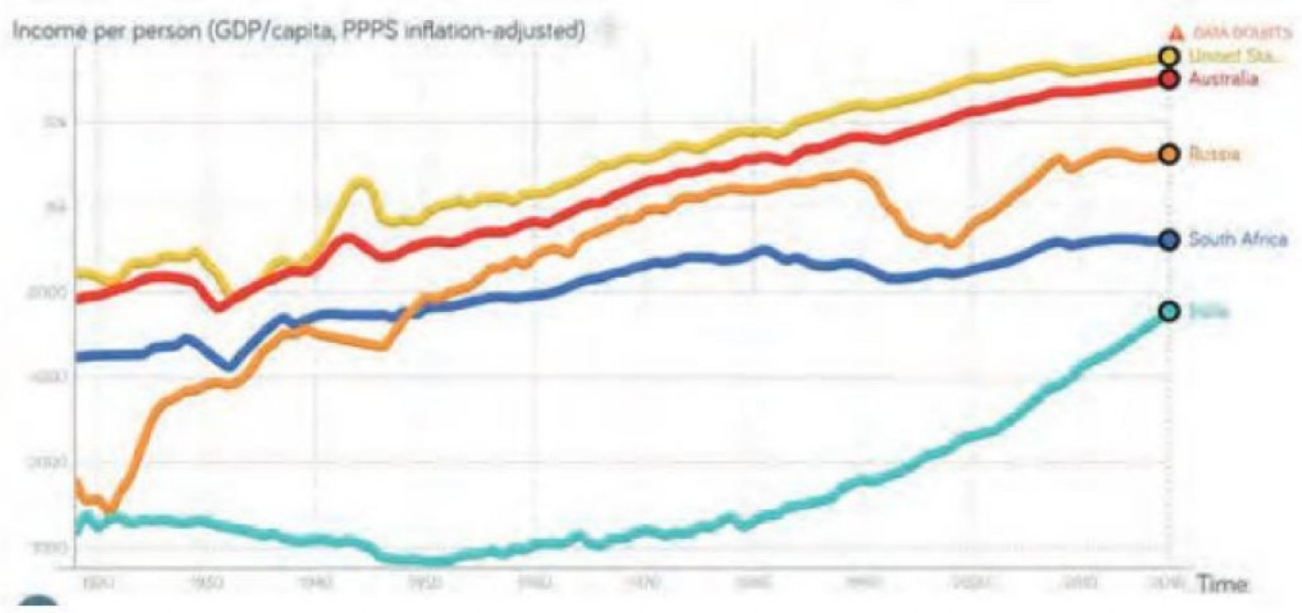
FIGURE 2.12 (a) No map exists of the entire internet, but these lines show the paths an email may take across some of the largest networks; (b) network visualisation of low-cost flight connections from Melbourne

9780170440806

Time series

Time visualisation represents a data item or data set over a period of time. Some time-based visualisations will show historical data, while others capture live data to provide real-time information. It is also possible to display the dimension of time by adding motion or animation to create a dynamic data representation.

The data could also be related to a timeline or time series. Timeline data may relate to individual items or events and show the order in which the items or events occurred over a time period. Time series data may relate to the same data item and show the variations or changes in the item over a time period.



Gapminder based on World Bank, A. Maddison, M. Lindren, IMF & more. Creative Commons 4.0 licence.

FIGURE 2.13 Time visualisation: screenshot from gapminder.com showing GDP for several countries from the time period 1918–2018

FlowingData

THINK ABOUT APPLIED COMPUTING 2.1

1. Using a search engine, find three examples of user-flow diagrams.
2. Go to the FlowingData website and find three examples of interactive animations.

Flow

Flow visualisations involve representing data that illustrates the flow pattern of one or more data items. This could be the pattern of customer movements through a supermarket or the series of pages a user would visit on a website to complete a transaction (user-flow diagram).

Flow visualisations are also used for scientific purposes to visualise the flow patterns of objects that are normally invisible, including air and water. The data for this visualisation (Figure 2.14a) was collected during testing using a wind tunnel, and the data converted to an image. The basketball court heatmap (Figure 2.14b) represents the location of players during the game and can be used to inform tactical and strategic planning.

a

b

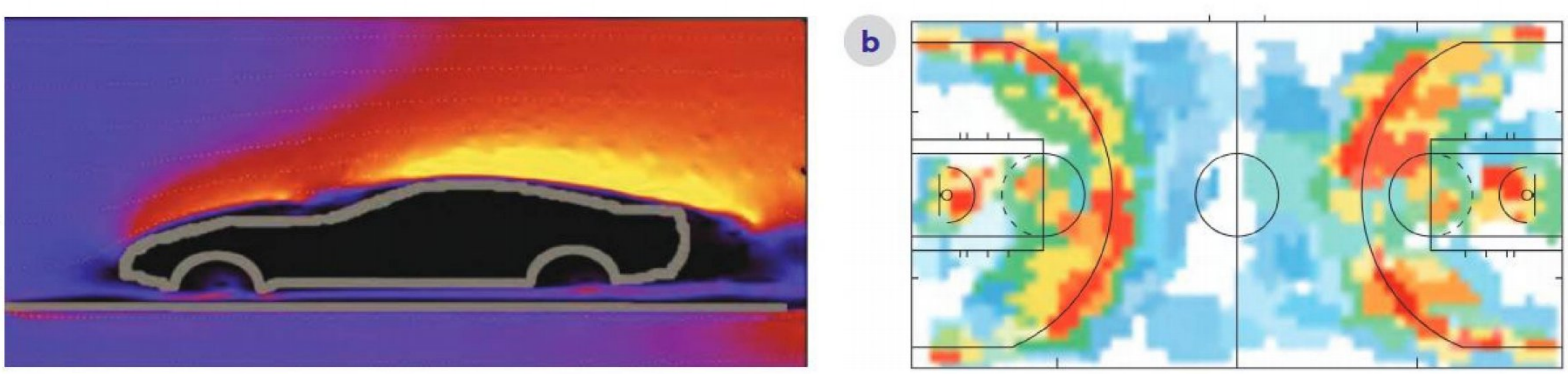


Image created using the Wind Tunnel app ©Algorizk

Shutterstock.com/Ilya Kovsnik

FIGURE 2.14 (a) Flow visualisation of air speed; (b) heatmap of movement of players on a basketball court

Matrix

Matrix visualisations can be used to show the composition of individual items in the sample size. In this regard, they can be considered similar to pie charts. Matrix visualisations often divide the display area up into grids (similar to cells in a spreadsheet). Different sections of the display area are then used to represent the proportion of individual (or groups) of data items.

Matrix diagrams can be used to compare different data items, similar to a scatter chart. Again, the display area is divided in a grid or table format. Figure 2.15 shows a number of matrix visualisations, each of which represents a county in the United Kingdom. The colours represent the percentage of people who voted for a particular political party in an election.

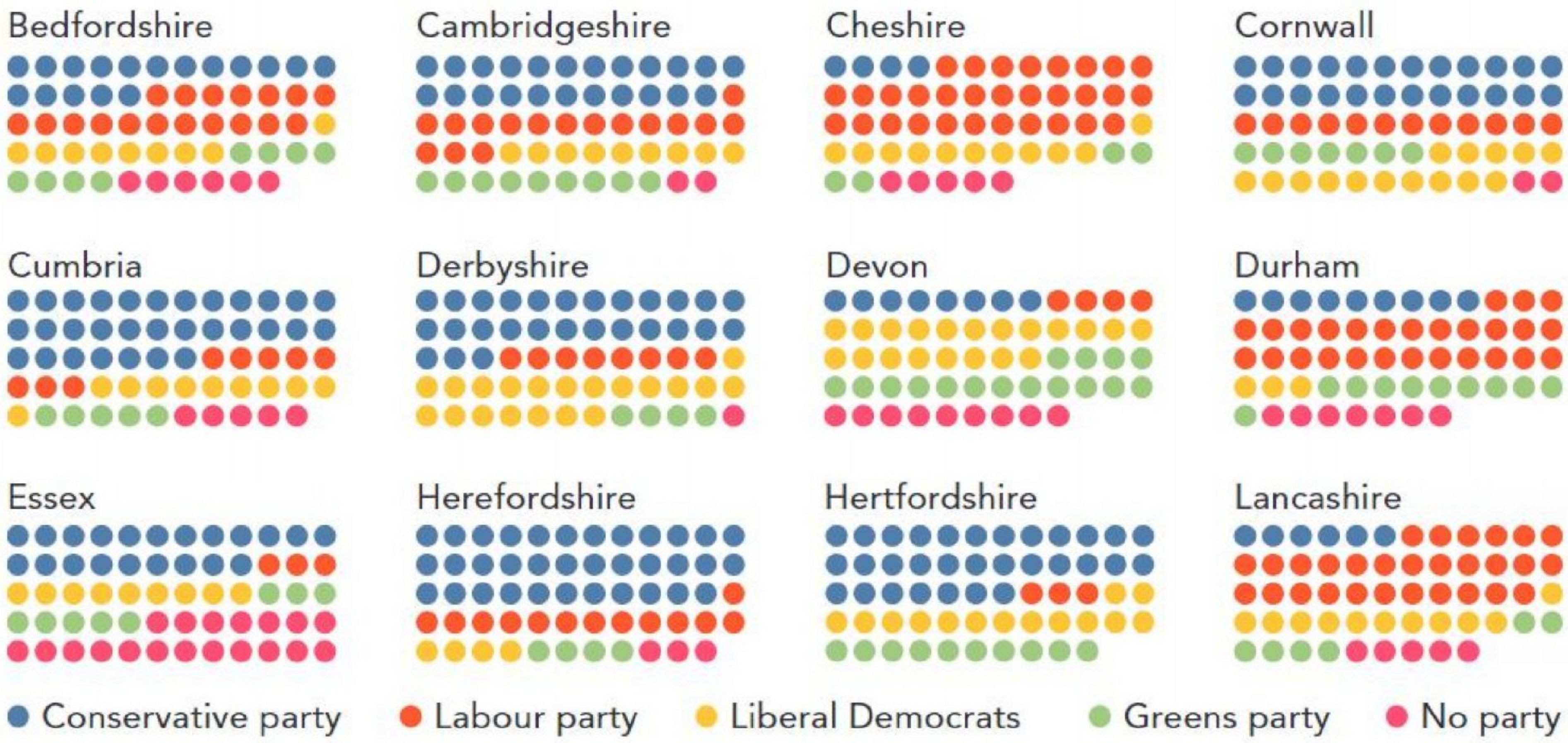


Diagram created using template by Christian Behrens, The Data Visualisation Catalogue

FIGURE 2.15 Matrix visualisation

A quick word about truthful graphs

Any graph type should be considered if it can accurately and truthfully convey insight or a message. There are, however, several graphs that consistently mislead and confuse.

'If the only thing worse than a pie chart is lots of them, then I think I've discovered that the only thing worse than lots of them, is half of one.'

Tufte, E. R. (1983). *The Visual Display of Quantitative Information*. Cheshire, Connecticut: Graphics Press.

Pie charts

A **pie chart** shows numerical proportions of a whole entity. If a pie chart is to be truthful then it should only be used with fewer than six categories and be presented with a 2D aspect.

When choosing to use a pie chart, ensure the parts add to 100 per cent, and that the pieces of pie are ordered from smallest to largest, beginning at 0° (the top of the circle), and moving clockwise.

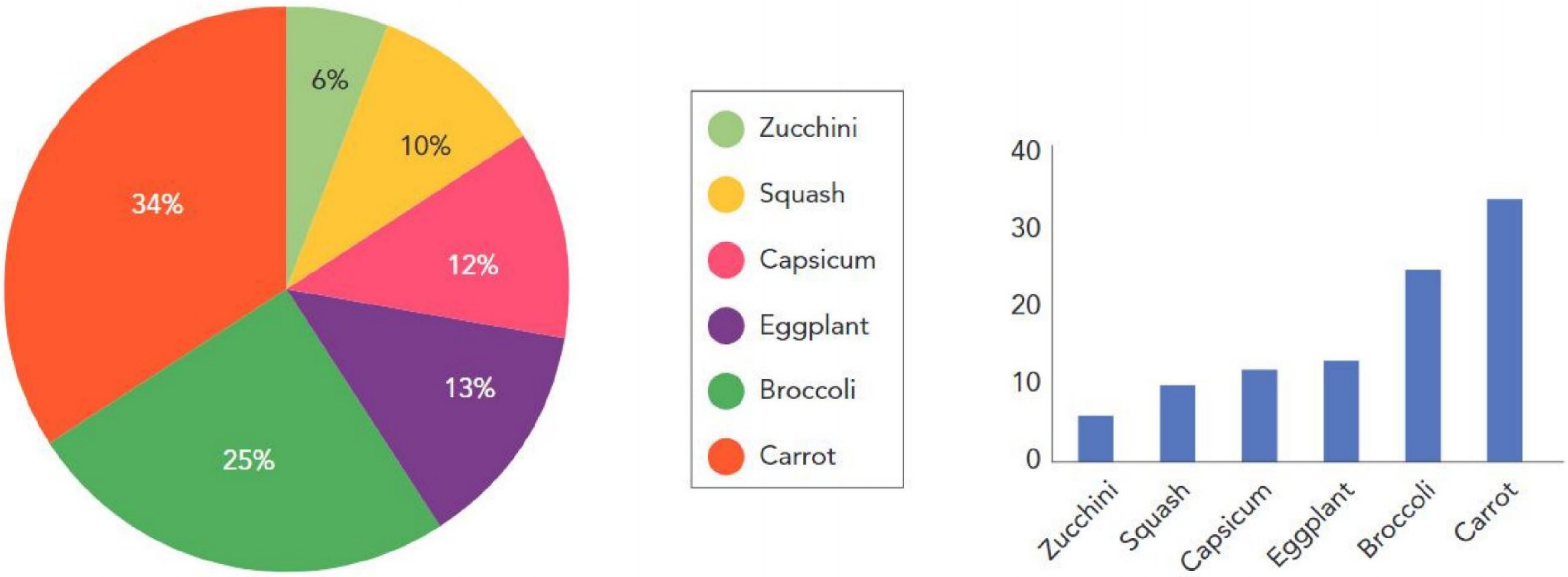


FIGURE 2.16 An example showing the limitations of bar charts and pie charts for quick accurate comparisons. A limitation of the bar chart is that it is not immediately apparent that it represents a total of 100%, so each bar is a proportion of the whole.

Three-dimensional pie charts can be misleading. Three dimensions represented on two-dimensional media must distort the information contained by the graphic. These distortions may misrepresent the data and create perceptions that are wrong in fact.

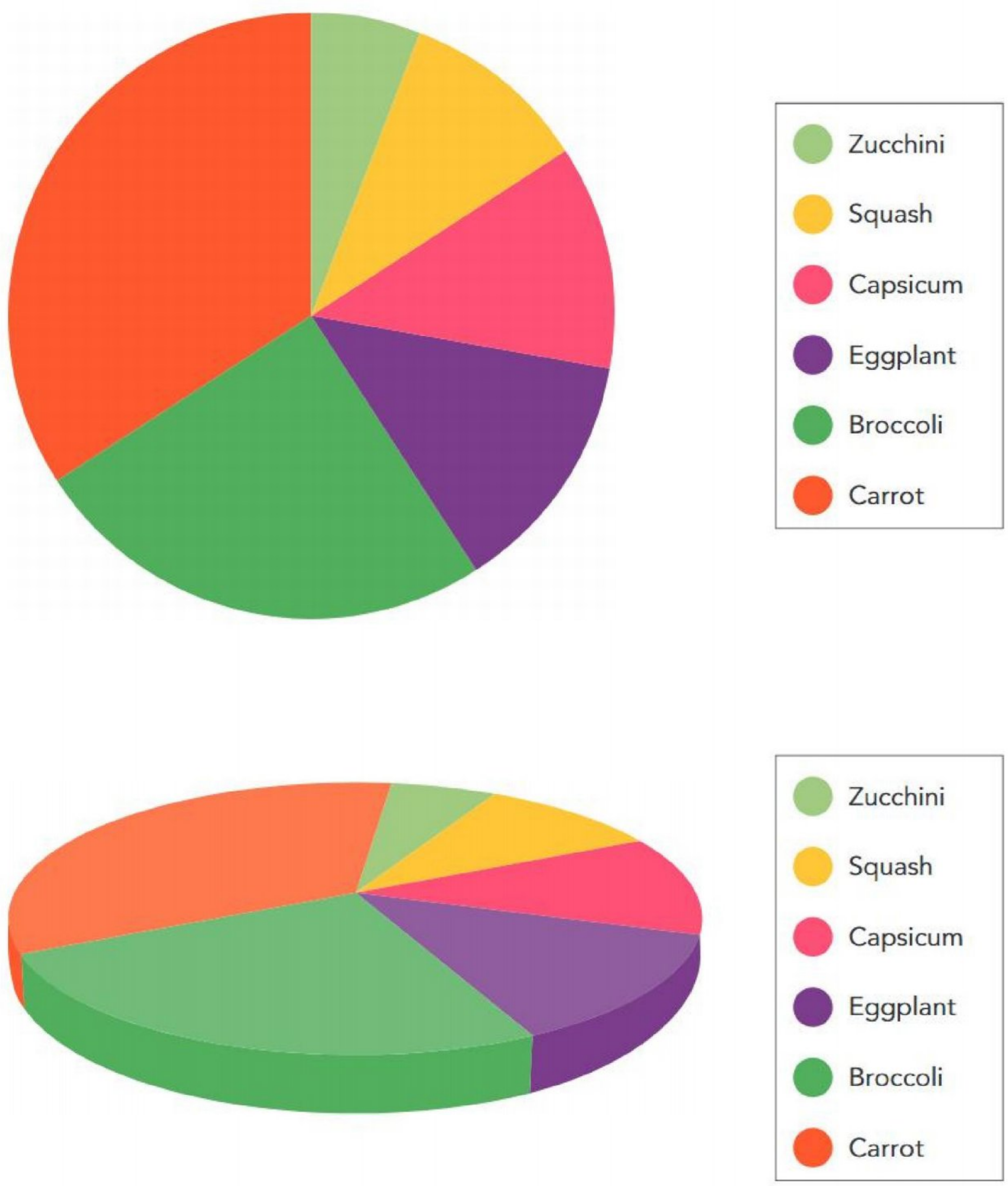


FIGURE 2.17 The 3D pie chart appears to show carrot (orange, 34%) is smaller than broccoli (green, 25%). The angle to the eye is larger for the sector at the lower edge of the visual. This distortion creates a false impression.

The most glaring 3D to 2D distortion is the mapping of Earth's globe in two dimensions, where areas of countries are stretched or compressed. Africa and Australia are consistently reduced in size, and Europe and North America are enlarged. This creates a wrong impression about distances and relative sizes. Different arrangements of the chart (known as projections) can achieve better understanding (Figure 2.18).

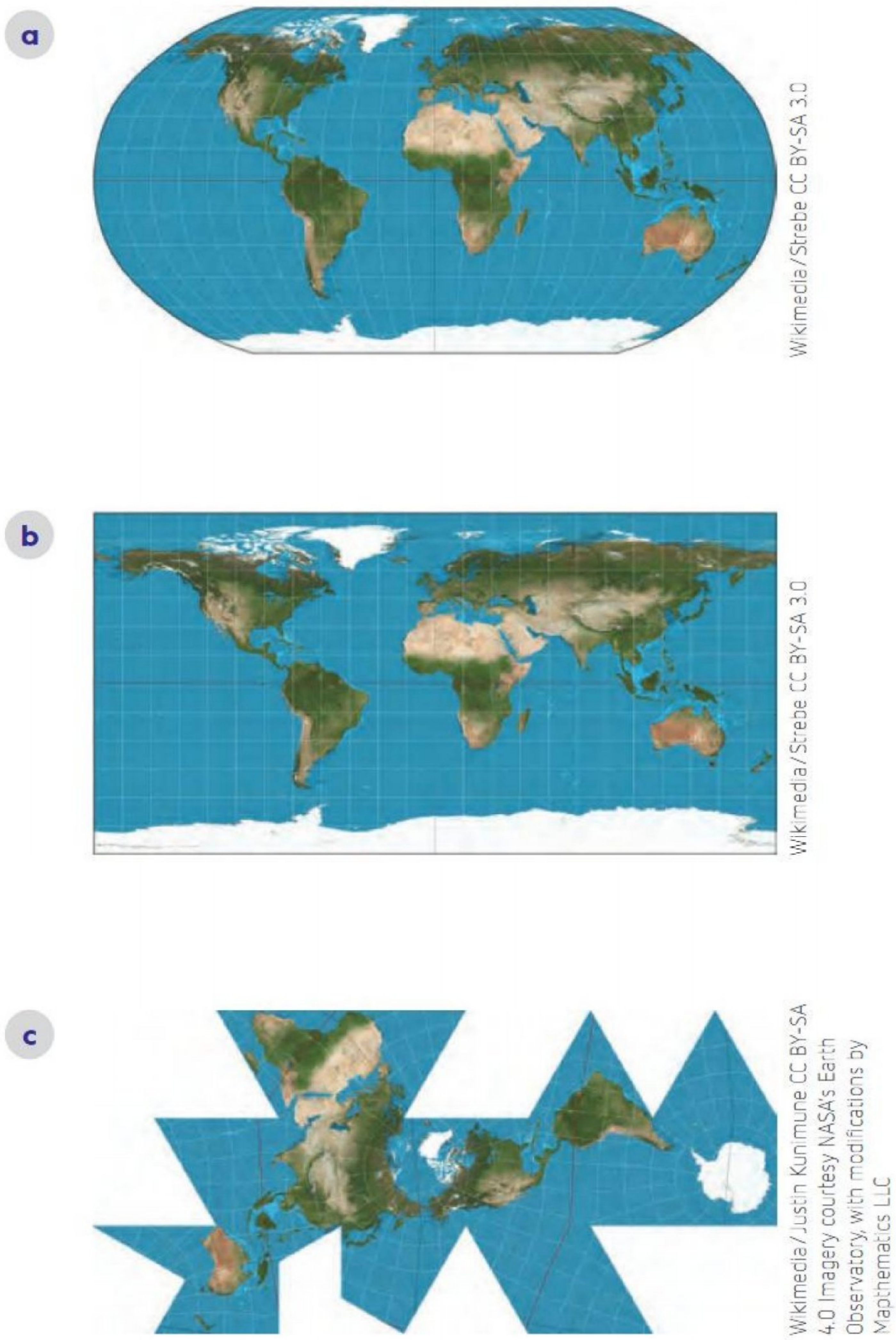


FIGURE 2.18 (a) A Robinson projection was designed to keep types of distortion (direction, distance, shape and size) low across the majority of the map. It becomes more distorted at the poles. Its purpose is to be visually appealing. (b) A plate carrée (flat square) projection has noticeable distortion. (c) A dymaxion projection has less distortion and there is no 'right way up'. This reduces the bias towards the Northern Hemisphere inherent in the other map types. However, it can be confusing to use.

Plot all the data

When summary calculations of data are made, the details are lost. The various forms of averages (mean, mode, median, variance, standard deviation) all calculate a central 'representative' value. If a trend line or average is the only information quoted, then significant details may be disguised or obscured and the result may just be misleading.

The famous example by Frank Anscombe illustrates the significance of seeing all the data (Figure 2.19). The descriptive statistics, mean, variance and regression trend line are very similar – to the point of being identical – yet the data sets are completely different.

In the hands of novice users, statistical packages can produce many meaningless statistics. Mathematicians would know that trend lines do not apply in every case. The use of trend lines is not appropriate for non-linear relationships and outliers.

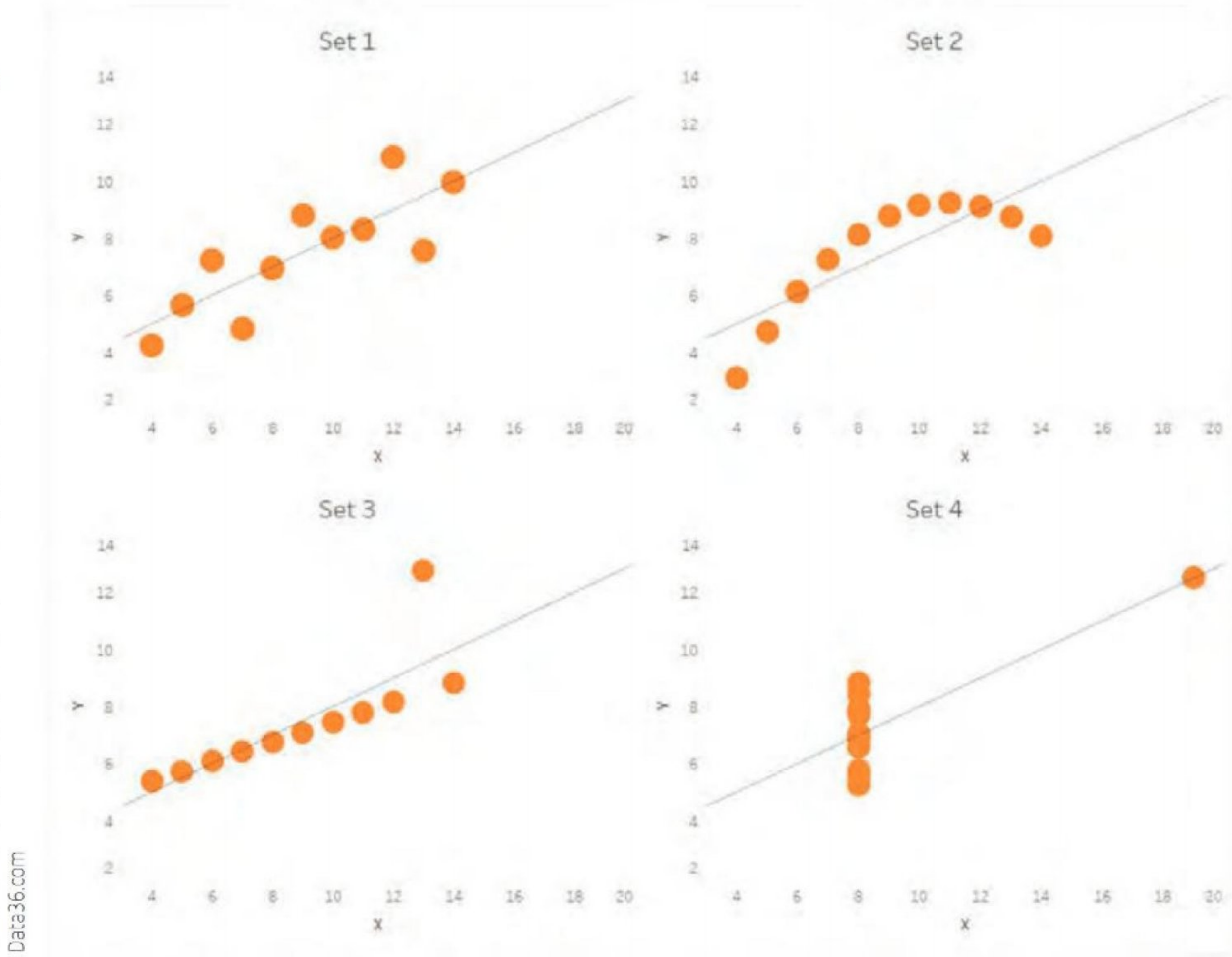


FIGURE 2.19 The descriptive statistics (mean = 9, variance = 3.32) and regression trend lines are identical for all four data sets graphed. When the data is plotted as a scatter plot, you can see that the data sets are completely different.

Animation can show development over time

How couples met

On-screen visualisations have a clear advantage over static or print. See the weblink 'How couples met' to see the relative frequencies of the manner in which future couples first met over the decades from the 1960s to the 2010s. In Figure 2.20 you can see the movement of the *Met online* category from nearly last in 1960 to number one in 2010.

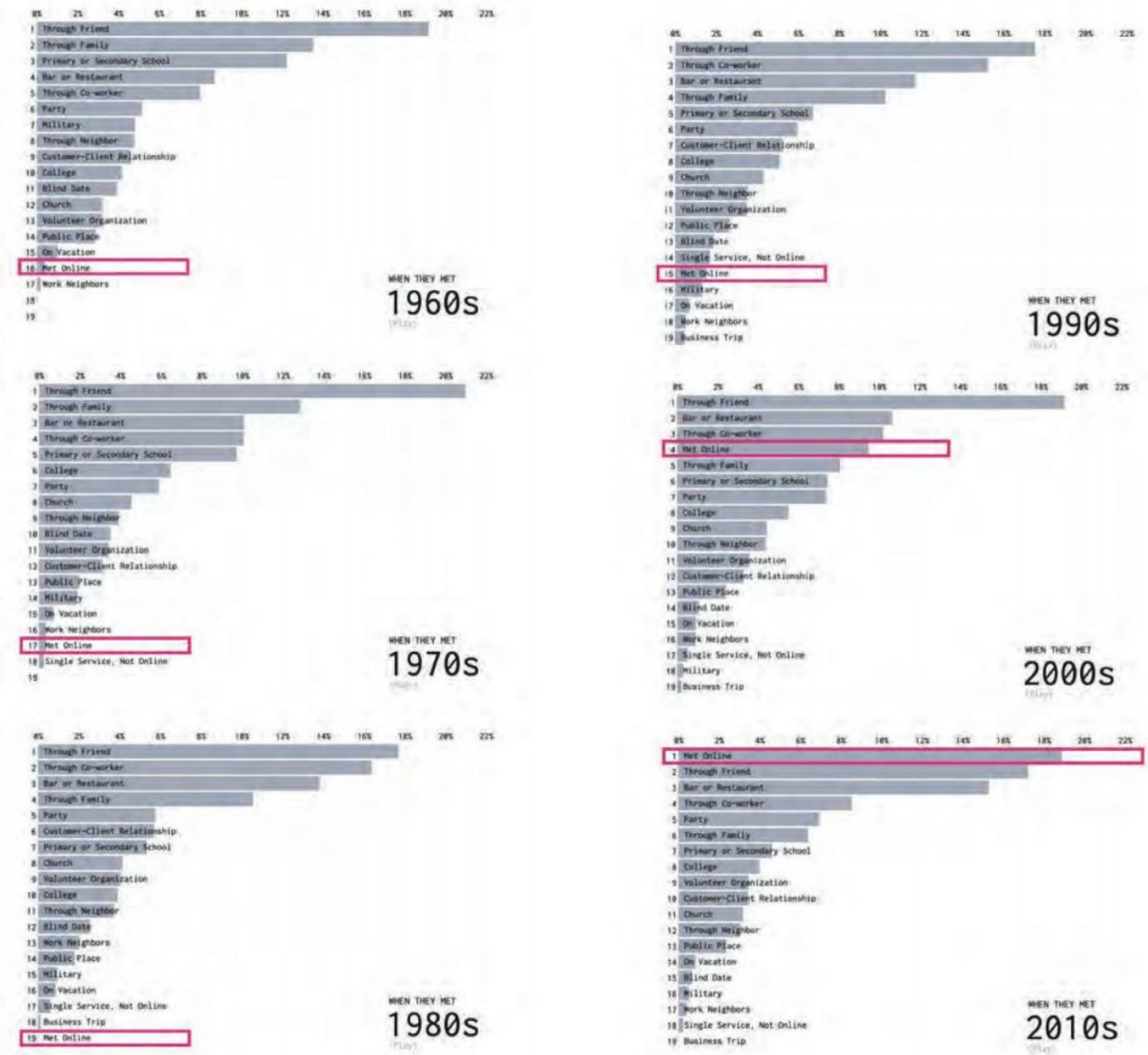


FIGURE 2.20 How couples met, 1960–2017

Graphs: Nathan Yau, Flowing Data, https://flowingdata.com, Data: Rosenfeld, Michael J., Reuben J. Thomas, and Sonia Hausen. 2019 How Couples Meet and Stay Together 2017 fresh sample. [Computer files]. Stanford, CA: Stanford University Libraries.

Dynamic data visualisations

A **dynamic data visualisation** is one where the appearance of the infographic can be changed (see Figure 2.22). This can be done by the user selecting alternate views, which may have been previously processed; in this case, it is also known as an interactive visualisation. Changing the appearance of the infographic can also be achieved by updating the data, or by the data being continually updated so the latest information can be included in the presentation.

Immersive media is becoming more widespread and acceptable when providing a 'lived' experience. The viewer, rather than observing a 2D or flat image, can get in between the data points in a virtual reality (VR) 360° data model, 3D or 360° graphs, which can be seen through a VR viewer.

Shutterstock.com/Gorodenkoff

FIGURE 2.21 Geospatial data can be used to create 3D visualisations that multiple users can view through a VR viewer.

Infogram

2.2

THINK ABOUT APPLIED COMPUTING

1 Search the web for the latest VR/AR data visualisations.
2 Go to the Infogram website to view several examples of VR data visualisations. How might VR graphics be more effective than 2D graphics?

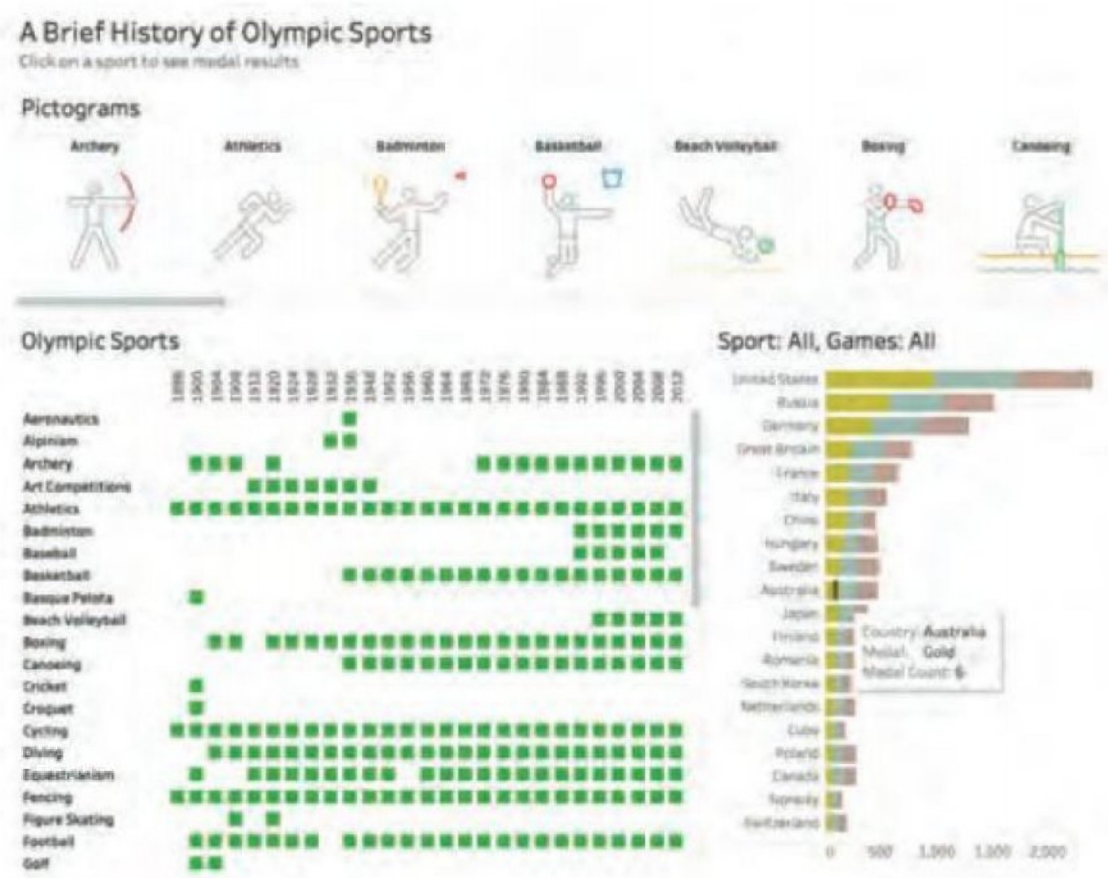
Guidelines for creating charts

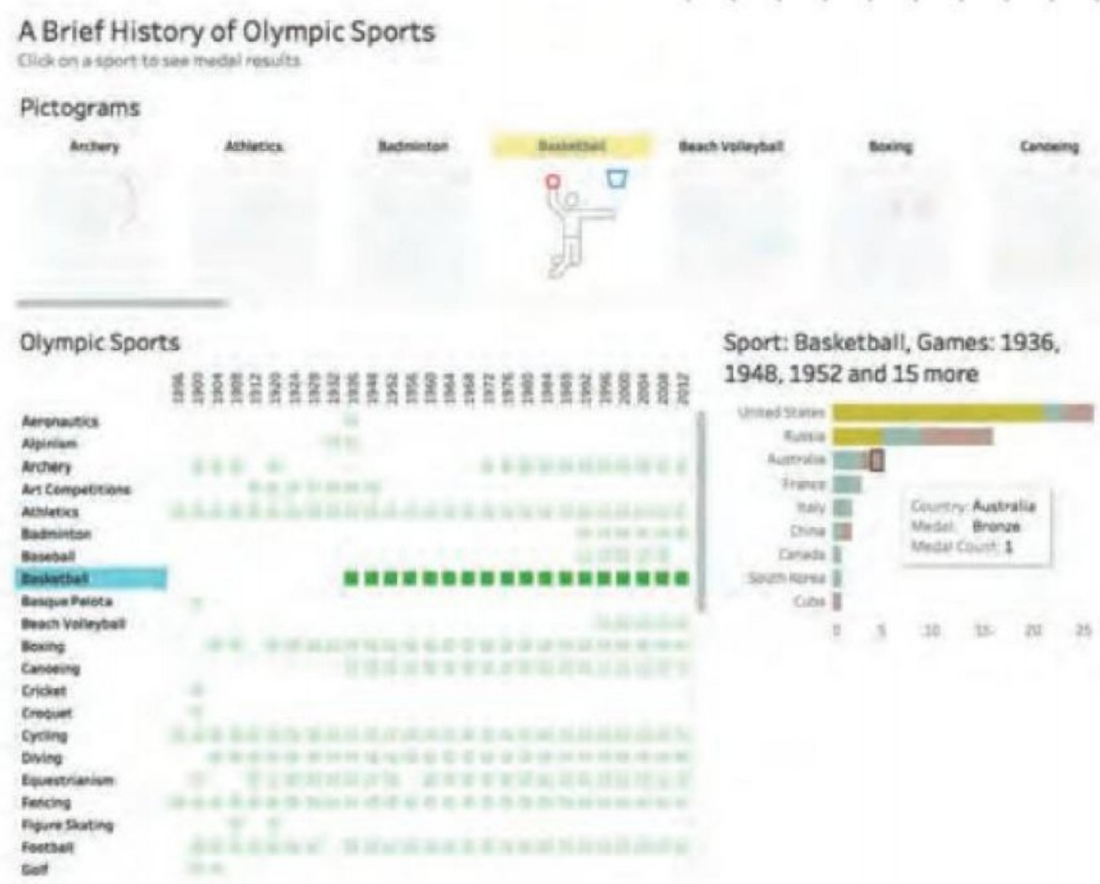
You should follow some simple rules with charts:

- They must have titles.
- The *x*-axis and the *y*-axis must be labelled.
- Use a key (or legend) if more than one set of data is provided on the same graph.
- Include author identification and/or the source of the data, the date and a filename (if appropriate).
- Include the unit of measurement on the relevant axis.
- Label each segment of a pie chart.
- Arrange segments of a pie chart (starting at 12 o'clock position) from smallest to largest.
- Consider including absolute figures as well as percentages.
- Choose colours that match the information being discussed.
- Use graphs to show trends or relationships between values on each axis.
- With pie charts, if possible, limit the number of items represented in a chart to five or six.

Choose and use your software carefully

The software chosen to create your charts will be capable of reading, processing and displaying your data set. Some data in raw form needs to be 'cleaned' or wrangled in order for the chart to be displayed. For example, some elements may have text instead of numbers recorded, or have empty fields. Sometimes what appears as numbers has been entered as text (or string) and will need to be converted. If the raw data is presented in a table format (rows and columns), the headings will need to be verified so each of the column data is known to be as it has been assumed. A chart can be created once all the data has been verified and cleaned ready for processing. Microsoft Excel, Tableau and Wolfram Mathematica are readily available software applications that can provide tools to construct many types of charts. These applications are downloaded and installed to a single computer. Online tools such as Plotly and DE3 require registration to save the graphic. Combinations of D3.js, R and Python would be considered industry standard; however, these are well beyond expected VCE skills and knowledge.





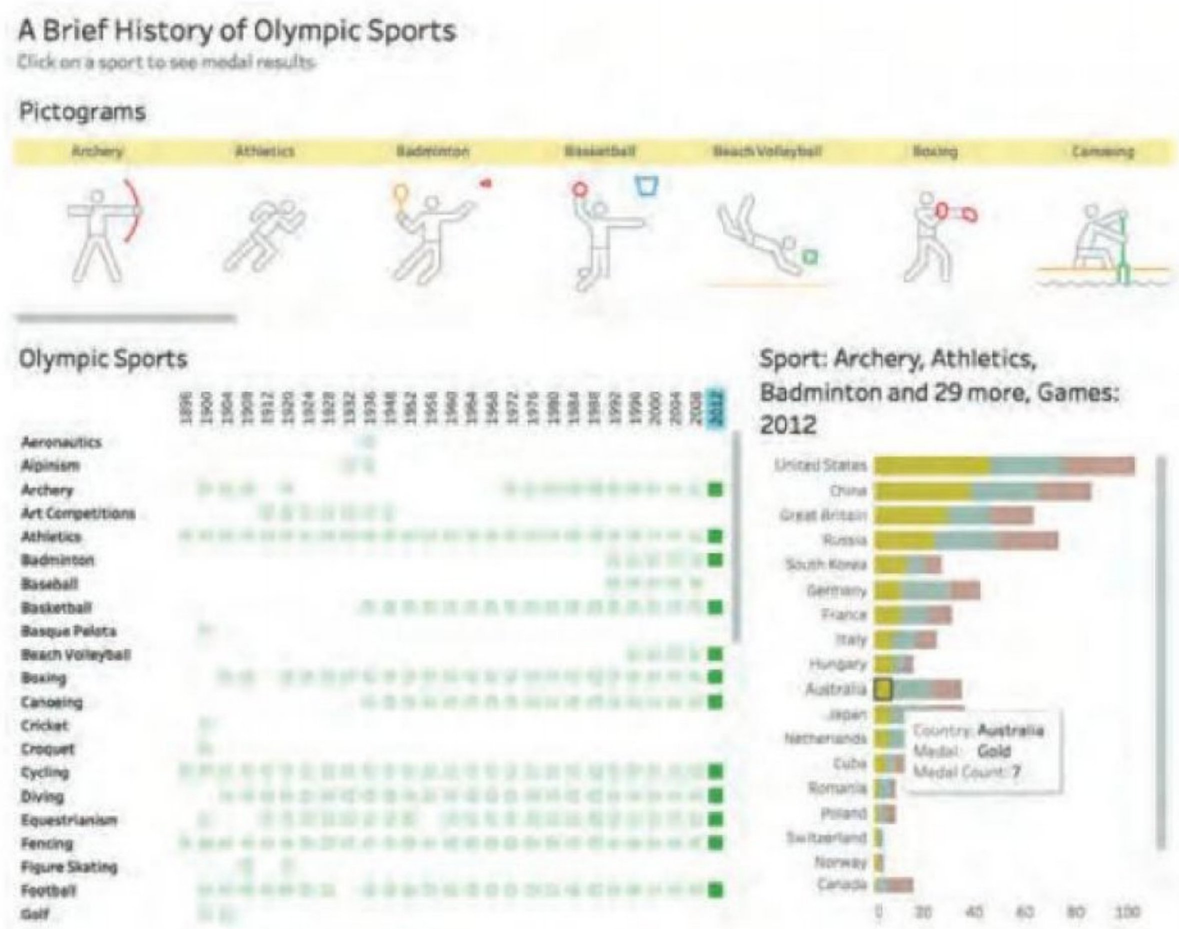


FIGURE 2.22 Dynamic data visualisation has a menu that allows choice of data to be viewed. Interactive features included: Tooltip hover, Menu buttons, Year button, Country button.

Social Science Data Collection Stanford University

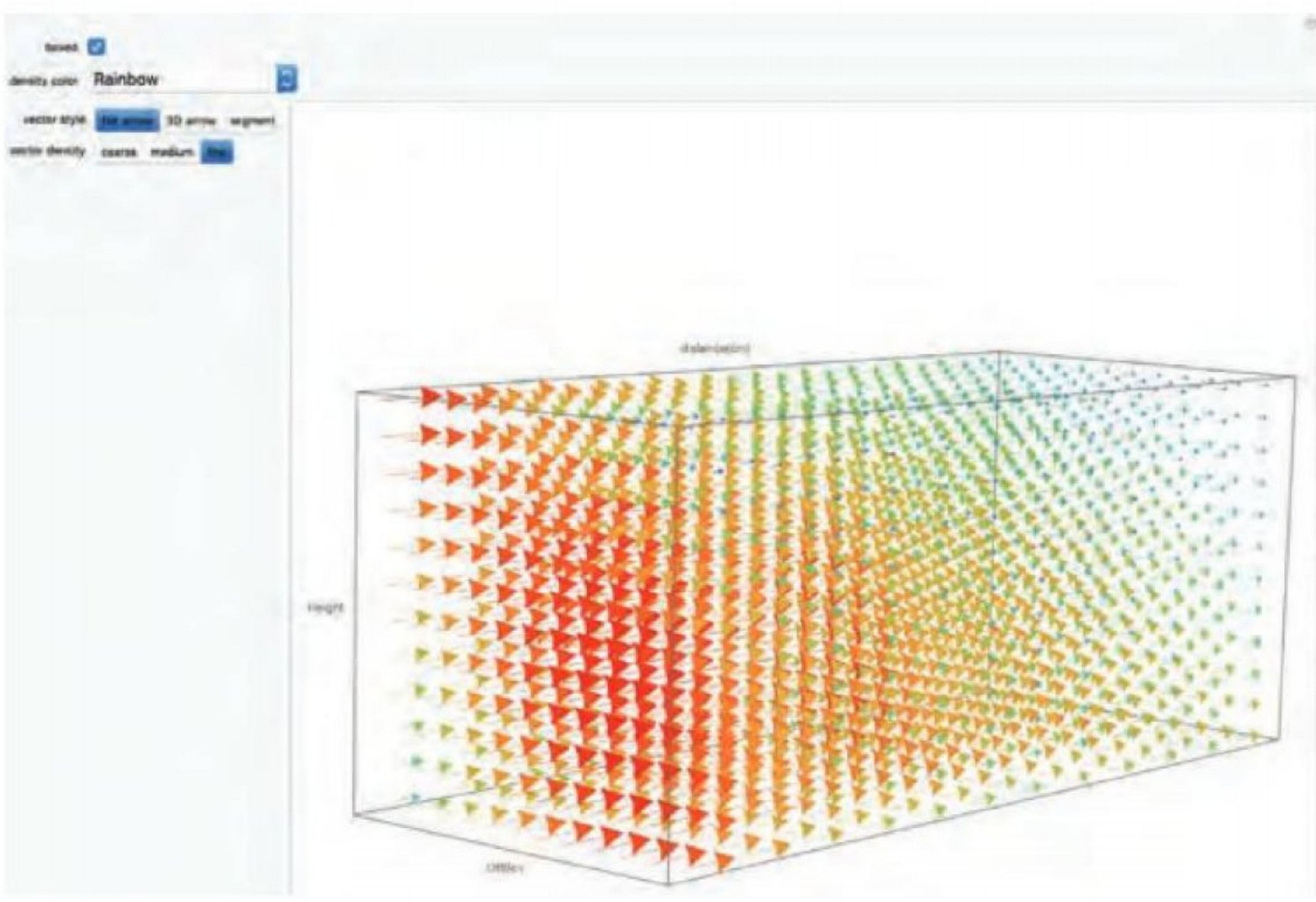


FIGURE 2.23 A Mathematica interactive data visualisation showing air speed near a circular fan (90°) in a model wind tunnel. The air speed is fastest near the fan blades, slowest at the centre and edges. Menu choices alter the view displayed.

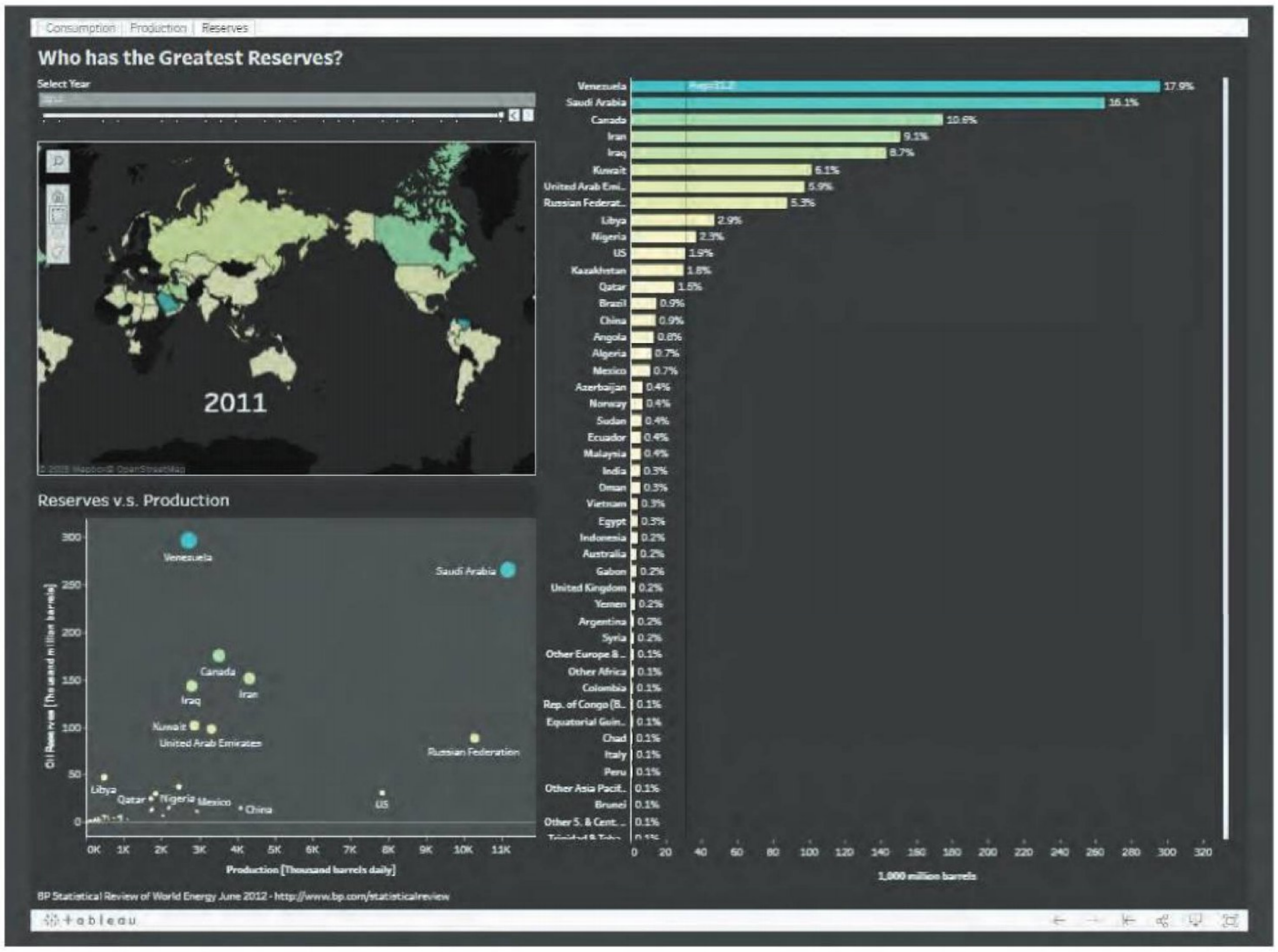
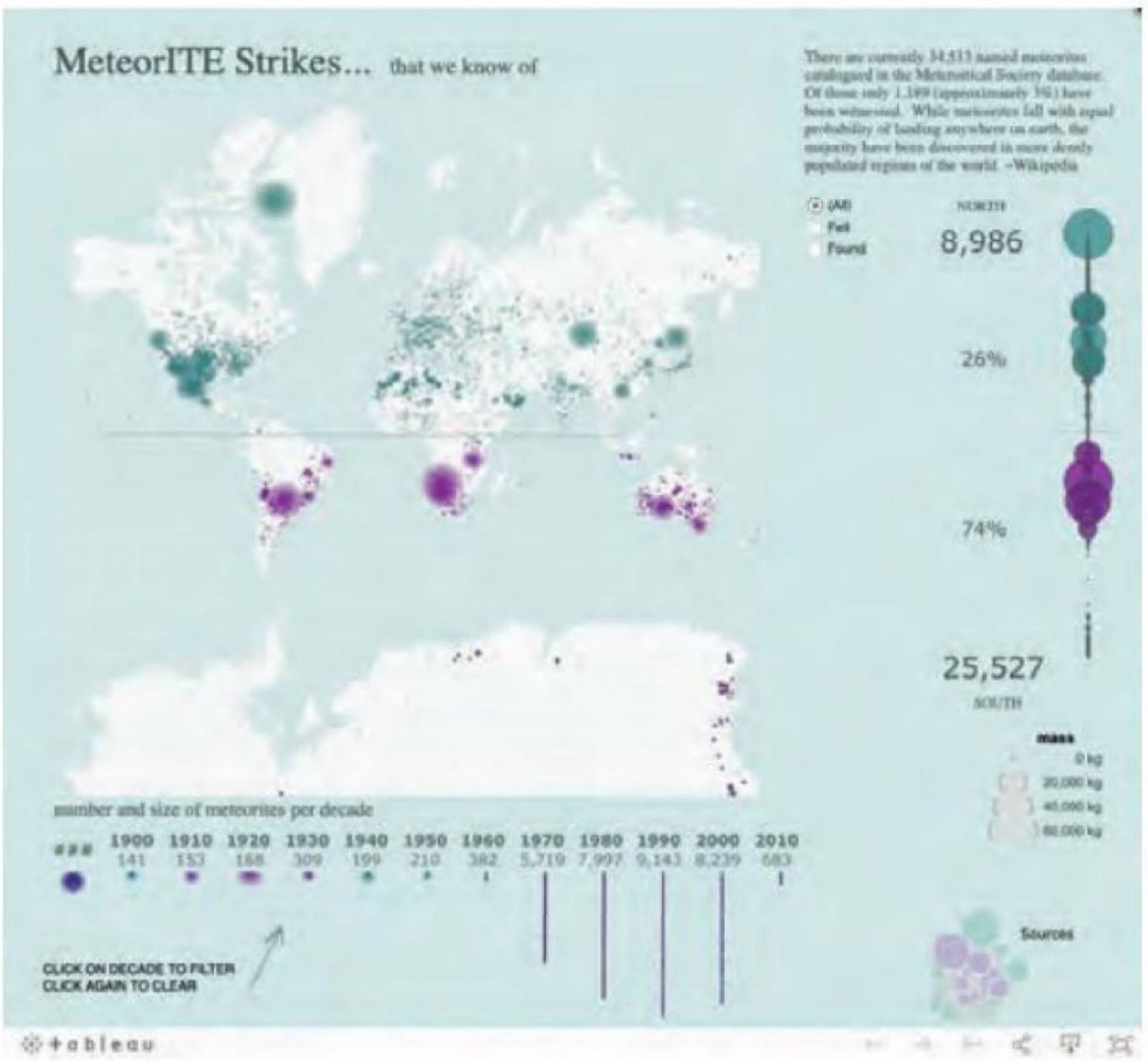
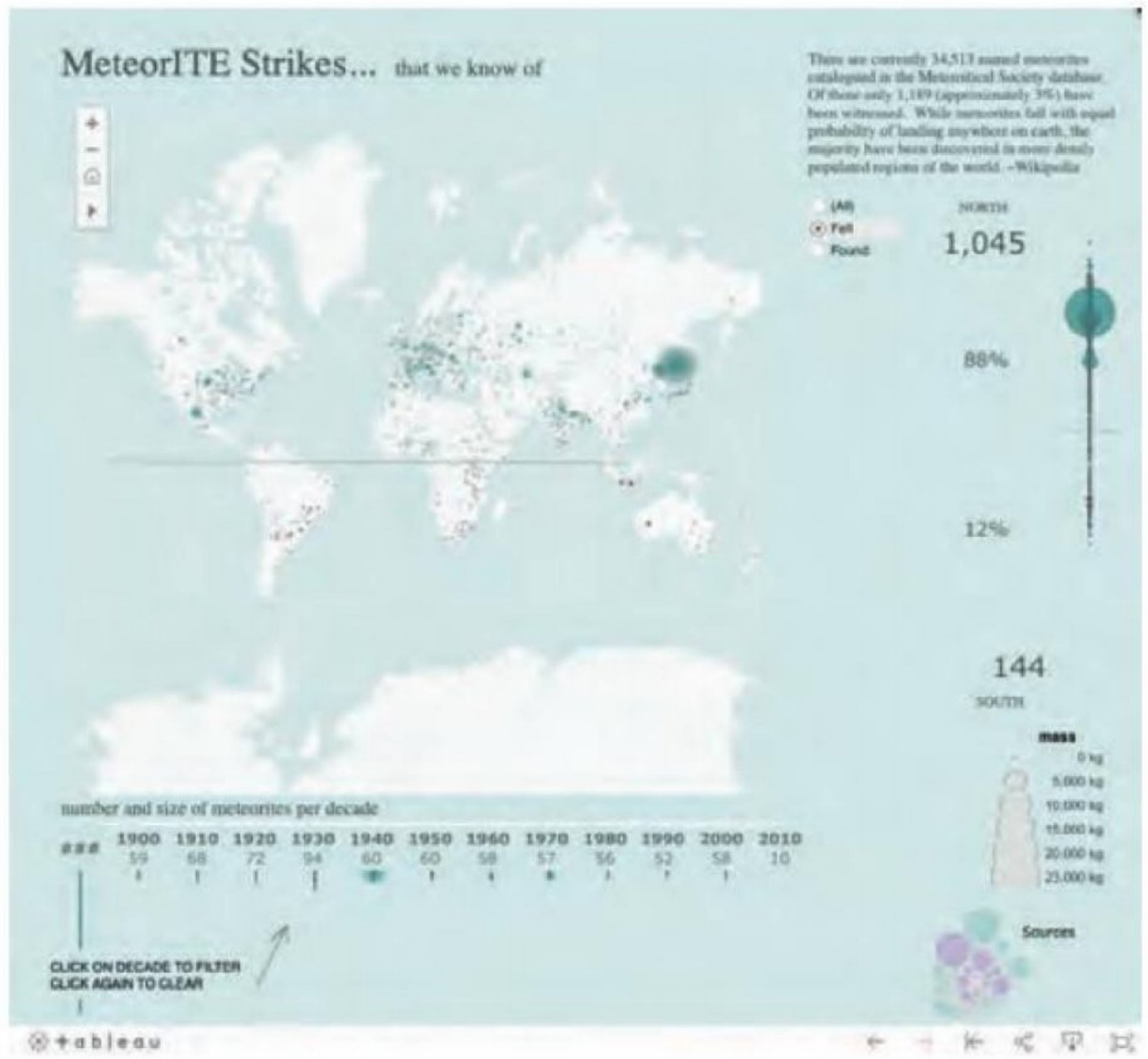


Tableau.com

FIGURE 2.24 Tableau Review of World Energy 2012. These two views illustrate how different displays can be chosen using a menu. Each element has a hover tooltip to provide more detailed information about the country chosen. A country can be chosen by reference to the map, or to the country listing bar chart, or the scatter plot. Colour is used to indicate amount of resources.





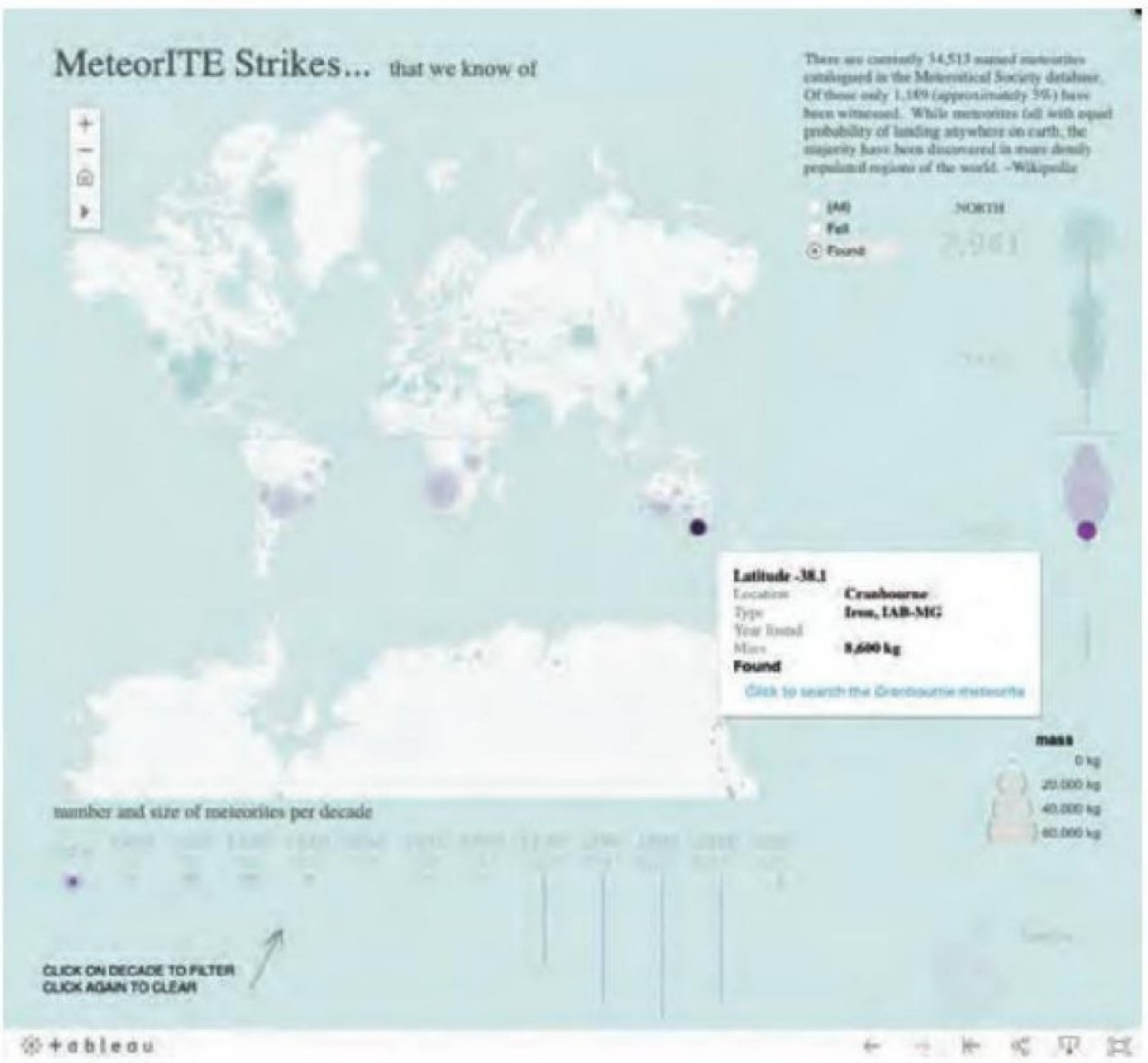


Tableau.com

FIGURE 2.25 Meteorite strikes radio button filter and tooltip hover, with URL links to data sources incorporated within the graphic

Smart meter data

CASE STUDY

Victorian electricity subscribers can apply to their provider to obtain a record of their electricity consumption. Smart meters installed at most premises record consumption every 30 minutes. Consumers can analyse their consumption patterns and consider whether to change their electricity use behaviour.

Many consumers have adopted a time-of-use electricity plan. Certain 'peak' and 'off-peak' time periods are designated through the day. The pricing is higher or lower at these times to encourage or discourage consumption. Note: every retailer has a different pricing structure so making comparisons becomes very difficult, unless you have access to data visualisation software.

Several retailers have seen the confusion deliberately created by many of the existing retail players and have provided 'real time' calculation of consumption and pricing to gain a marketing edge.

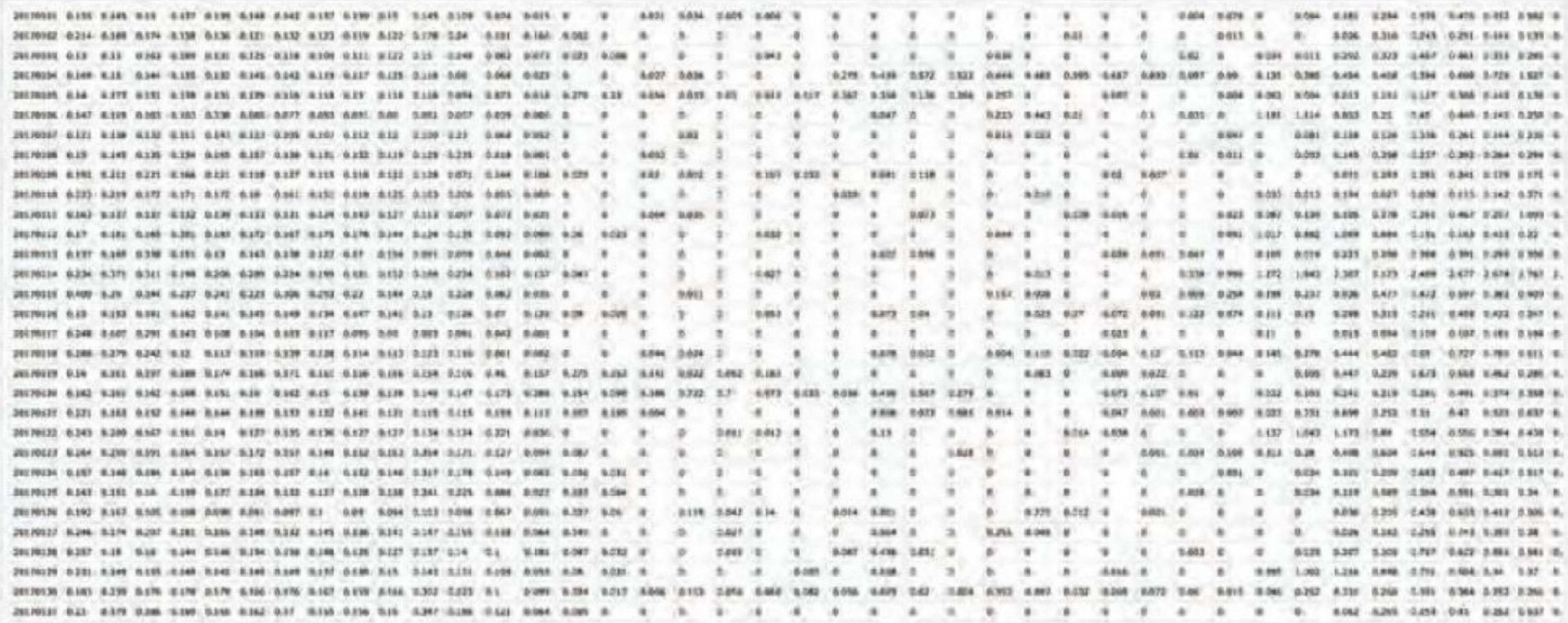


FIGURE 2.26 Raw data supplied by an electricity provider. Electricity consumption is recorded by the smart meter every 30 minutes.

Ad SampleElectricity Consumption.csv

2.3

THINK ABOUT APPLIED COMPUTING

Ask if you can get your household's electricity consumption file downloaded from your electricity supplier. Your school can also acquire this data. Alternatively, go to NelsonNet to download the sample file *SampleElectrictyConsumption.csv*.

Explore the data to discover any 'dodgy' (unvalidated) data, which needs to be 'fixed' (corrected). Unvalidated data is the contents of any cell that will not be plotted by software (i.e. not a number).

Step 1: Plot all the data

The only way to get an idea of what you have is to make a scatter plot of the data.

Step 2: Look for patterns

'What about ...?' is the key question to begin your investigation of the raw data. There may be 'odd' plots or **outliers**. These may be items of interest for further understanding. Each plot corresponds to a date and time. Peaks will indicate the largest amount of consumption. Individual dots may not show a pattern.

Step 3: Plot each series

On the same axes, plot each 24-hour day (48 readings on each of 365 days). Calculate the total consumption for each day.

Step 4: Plot each day as a bar chart

Alternatively, a stacked bar chart could show the consumption each day and the total each day. The tooltip hover feature will allow inspection of items of interest.

Note: For those consumers with solar PV panels, it will be possible to have 'negative' consumption indicated. Consider what this means for calculations. It will be necessary to consider those consequences for total consumption and for billing purposes.

Design principles for data visualisations

Design principles are guidelines to help you enhance the **appearance** and functionality of solutions. Graphic solutions that are displayed on-screen need to be easily understood and accessed with minimal time and effort.

To communicate effectively, graphic solutions need to be clear and functional. You need to ensure that facts are obvious and your message is unmistakable. Your solution must be carefully designed, taking the design principles into account.

The **usability** and **accessibility** principles shown in Figure 2.27 are discussed in the following section, in terms of how they may apply to your solution.

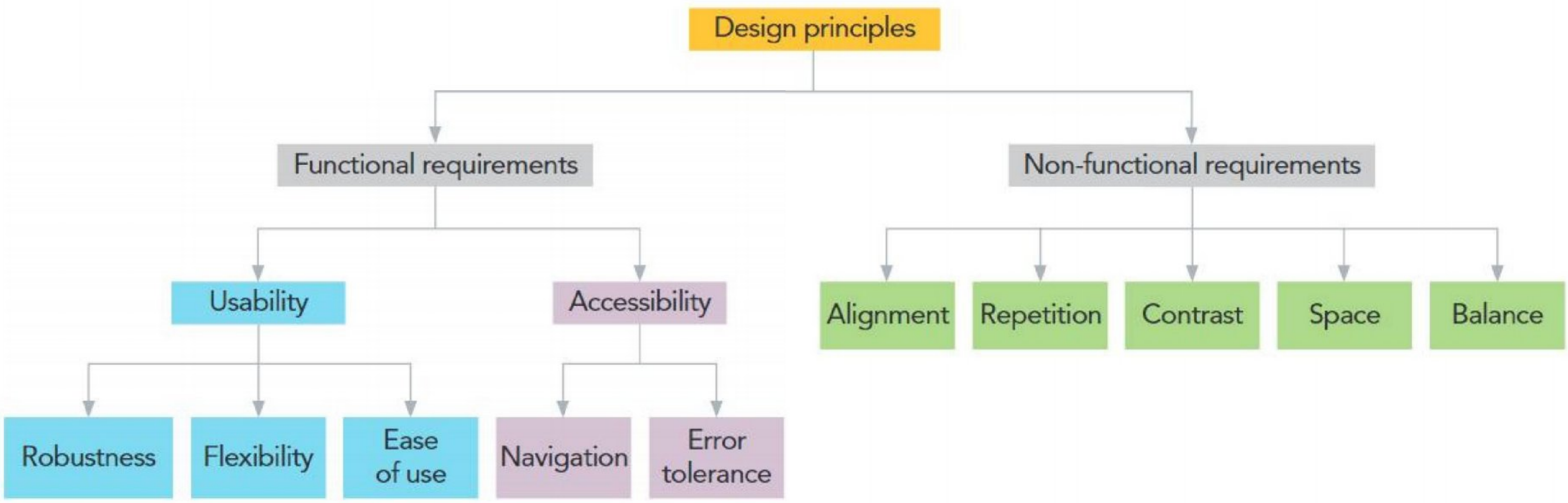


FIGURE 2.27 Design principles are accepted characteristics that contribute to the functionality and appearance of solutions. For VCE Applied Computing, the principles that relate to functional requirements are usability and accessibility, and non-functional requirements are appearance.

Functional requirements

Functional requirements relate to what the data visualisation should do. Typically, functional requirements will describe a behaviour or function.

Usability

All graphic solutions that represent data need to be easy to use, flexible and robust. The user should be able to acquire the required information easily and efficiently. When designing your data visualisation for usability, consider the following.

- Can the user understand the graphic form and the information it conveys?
- Can the user quickly identify the purpose and interpret the critical data?
- Is the graphic form accurate and a true reflection of the data that has been analysed?
- Have you saved the final data visualisation in a format that maximises legibility, and in a size that will be quick to load?
- Have you chosen sensible dimensions? For example, you may want to think carefully before designing wider than most website body panes (around 800 px) for a browser presentation or 16:9 ratio for computer screens and (most) projectors.
- Have you chosen easily readable font sizes?
- Have you tested the data visualisation in multiple browsers to check that it works? If plug-ins are necessary, these must be included with instructions.

Accessibility

A user should be able to access meaning and relevant information from a graphic solution efficiently and easily. Users should not have to puzzle over the meaning or placement of text and symbols used.

To help users along, use simple and relevant labels and commonly understood symbols and units. For example, use '$' to represent money.

When using your graphic solution, users should be able to navigate their way through text and images in a standard way; from top to bottom or left to right. A 2006 eye tracking study found that people used an F-shaped reading pattern for websites. This reading behaviour should be considered during visual design of web pages and graphics; for example, when arranging chart and dashboard elements. Use size, colour or style if you literally want readers' eyes to move a certain direction.

F-Shaped Pattern For Reading Web Content by Jakob Nielsen (April 16, 2006). https://www.nngroup.com/articles/f-shaped-pattern-reading-web-content-discovered/

FIGURE 2.28 F-shaped reading patterns can be modified by choosing page elements and emphasis carefully.

Do not make anything so small or hard to read that it goes unnoticed. Important facts should be treated with appropriate formats and conventions that highlight why they are crucial.

Essentially, make it easy for users to glean meaning immediately. When you are creating an infographic, it can never be *too* obvious.

Non-functional requirements

Non-functional requirements describe how the data visualisation appears rather than what it does.

Alignment

Alignment refers to arranging text, images and objects vertically or horizontally in either straight lines or correct relative positions. Horizontal alignment can be either left, right, centre or full justified. Full justification refers to the text being aligned on both the left and right margins with spacing distributed evenly across the line to achieve this. Vertical alignment can be top, middle or bottom.

9780170440806

Using alignment cleverly gives your text, images and objects a sense of order and organisation that helps to communicate your message clearly. For example, putting something in the centre of the page vertically and horizontally suggests immediately that it is of central importance. Putting something in a small font size at the bottom right would seem to be of little importance.

Smart use of alignment also helps to imply relationships between different elements of your solution. You will find alignment tools in all graphics software packages.

Repetition

Repetition refers to the use of the same or similar visual elements repeatedly within a graphic solution. It is used to unify elements of a layout, and is achieved by repeating patterns, textures, fonts, colours and page elements. On a graphic solution, repetition usually means that each section or module uses similar headings, colours, font styles or other visual cues so that it is easier for the user to understand the relationships when switching between elements within the solution. For example, if you have interactive and dynamic charts, each chart style needs to be consistent when the chart switches between views.

Contrast

Contrast refers to the visual difference in colour or tone between objects (both text and images) in a graphic solution. Greater contrast will make objects appear to stand out more from one another. If there is not enough contrast between two objects, they may appear to blend into each other, making it difficult for the user to see each of them clearly. Contrast between the background of your graphic solution and text should make the information clearly visible and legible.

The use of white space can enhance contrast around objects within your graphic solution.

Space

Space refers to the areas around and between objects – text and images. If your visualisation is cluttered it may be difficult to follow. You may want to include lots of detailed information in your solution, but it is important to put as much space between objects as is necessary for them to be distinguished as separate and navigated through in the correct order to guide user behaviour.

Balance

Balance refers to the symmetrical arrangement of items so that they appear balanced to the left and right. Some asymmetrical elements have different weightings that achieve an off-centre balance.

Constraints

Any factor that limits or restricts the data visualisation may be considered a **constraint**. Constraints can be broken down into categories: economic, technical, social, legal and usability.

Economic

Economic constraints include both time and budget. Either a lack of time or a small or restricted budget may result in a re-evaluation of the user's requirements, or a re-evaluation of how the requirements can be achieved. The deadline by which the user or client needs to have the solution operational will define the time available to design and develop the solution. The longer the time available, the more time there is to complete an in-depth analysis and detailed designs, and to develop advanced features of the solution.

Technical

Technical constraints are constraints related to the hardware and software available for the project. Available hardware and software, memory and storage capacity, processing and transmission speeds, and security concerns are all examples of possible technical constraints.

For example, the data collection may be incomplete, with gaps in the data sequence, or may have errors due to collection errors or transcription errors.

Chart size limitations

The width of any graphic will be approximately one screen, or 800 px. The axes labels need to be chosen carefully to ensure clarity.

Software limitations

The computer hosting the data may not be able to manipulate, calculate or redraw quickly. After about five seconds, users begin to notice the computer is 'running slowly'. Big data may be defined as data sets that cannot be accommodated within RAM on host computers. The data sets accessed in this course are unlikely to approach that size. For example, in research and industry, data sets typically from 100 MB up to 1 GB file size may be considered small. Excel, for example, is difficult to use with file sizes greater than 10 GB.

Excel, which is used for data manipulations and some simple visualisations, is limited to 255 series and 16 variables (columns) and some number precision values. An Excel chart is essentially unlimited, with up to 1 048 576 rows by 16 384 columns and most other functions limited only by available computer memory. Tableau is used for data visualisation, and has limitations dependent on the host computer.

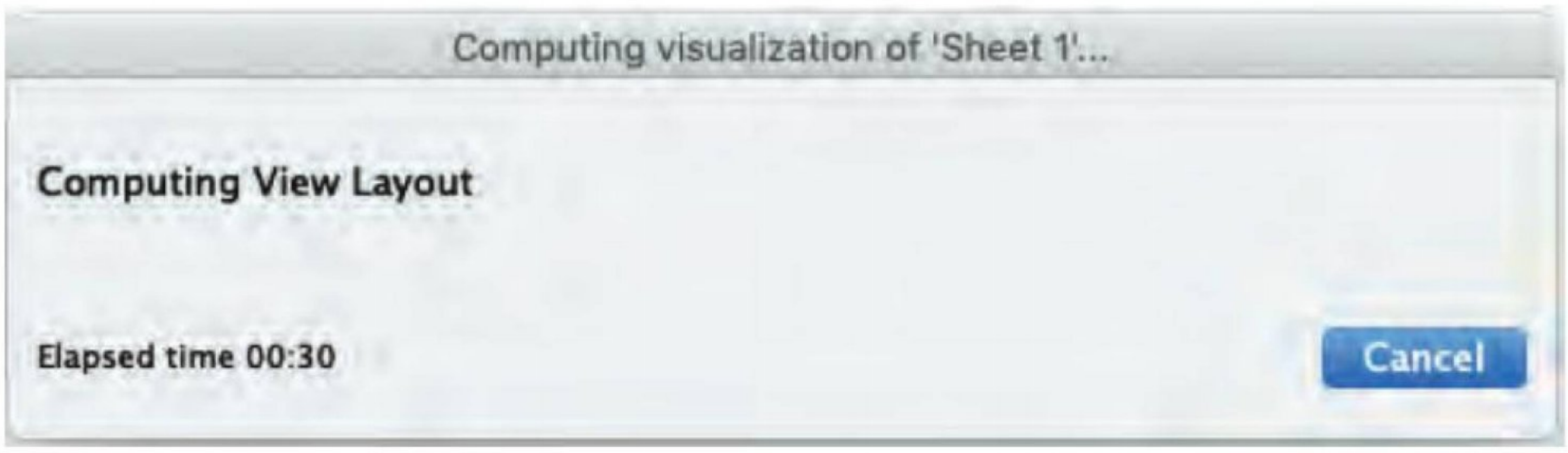


FIGURE 2.29 Tableau measured the time it took to refresh the display when rearranging more than 100 000 data points on a 4 GHz i7 CPU with 32 GB RAM. Other devices may be slower.

Social, legal and usability constraints

Non-technical constraints relate to areas other than hardware and software. The user's level of expertise (social) is an example where a solution, when being developed for users with little digital systems expertise, may restrict some of the requirements that would involve complex manoeuvres to complete.

When considering usability, all visual solutions that represent data need to be easy to use, flexible and robust. The user should be able to acquire the required information easily and efficiently.

Legal requirements are another type of non-technical constraint. Privacy laws may restrict features linked to displaying personal data in the solution, or to collecting data from the devices of someone using your solution. Copyright laws may restrict features that allow other users to upload content to the solution.

Scope of a solution

The scope of a solution refers to the work that must be done to complete that solution. Scope will be informed by the functions and features of the solution. Scope is documented in a scope statement, which may be referred to later, in order to make decisions as the solution develops. Managing scope is about what is and is not included in the solution scope.

Design tools

Design tools are used to represent the functionality and appearance of your solution. Normally, the solution is first sketched on paper. Several design tools can be used to represent the design of graphic solutions; however, **annotated diagrams** are the most apt for planning a solution and presenting it.

Software such as spreadsheets or databases can assist with the manipulation of data to develop graphs and charts. This software has inbuilt search, sort and filter functions that enable users to work quickly and with minimal effort, thereby maximising **effectiveness**.

While there are a wide range of design tools that may be used to design the appearance of a solution, in VCE Applied Computing we shall consider:

- appearance design tools
  - layout diagrams
  - storyboards
  - annotated diagrams and mock-ups
- functionality design tools
  - input–process–output (IPO) charts
  - flowcharts.

Appearance design tools

Layout diagrams

A layout diagram can be used to plan how the visualisation will appear. The diagram can be hand drawn or computer generated but, if computer generated, it should not use the same software as the solution. The purpose of a layout diagram is to illustrate how the final solution will appear. Once completed, the layout diagram can be inspected to see if the type of visualisation chosen is appropriate for the data set.

A number of alternative layout diagrams can be created to represent the data set in a variety of types of visualisations, to help decide which type to use in the final solution.

Storyboards

Storyboards can be used to demonstrate how a dynamic data visualisation may be animated or updated, or how user interactions may affect the appearance of the data visualisation.

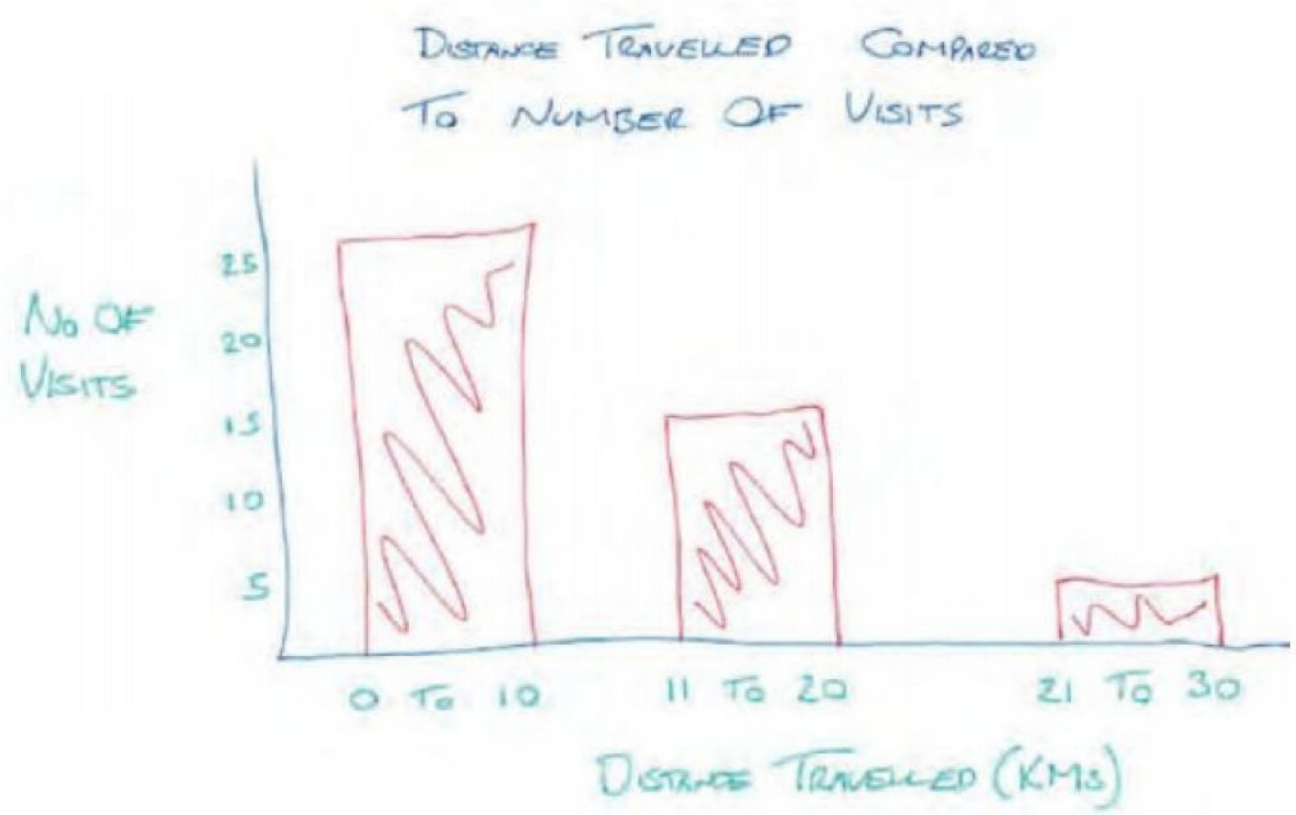


FIGURE 2.30 A hand-drawn layout diagram set out as a column graph

Annotated diagrams and mock-ups

Wire frames are annotated diagrams that provide a visual depiction of how your visualisation solution should look (Figure 2.31).

The designs indicate features such as differences in font size, colour and positioning of objects. The placement of objects must be planned so that a balanced, visually appealing and clear effect is attained.

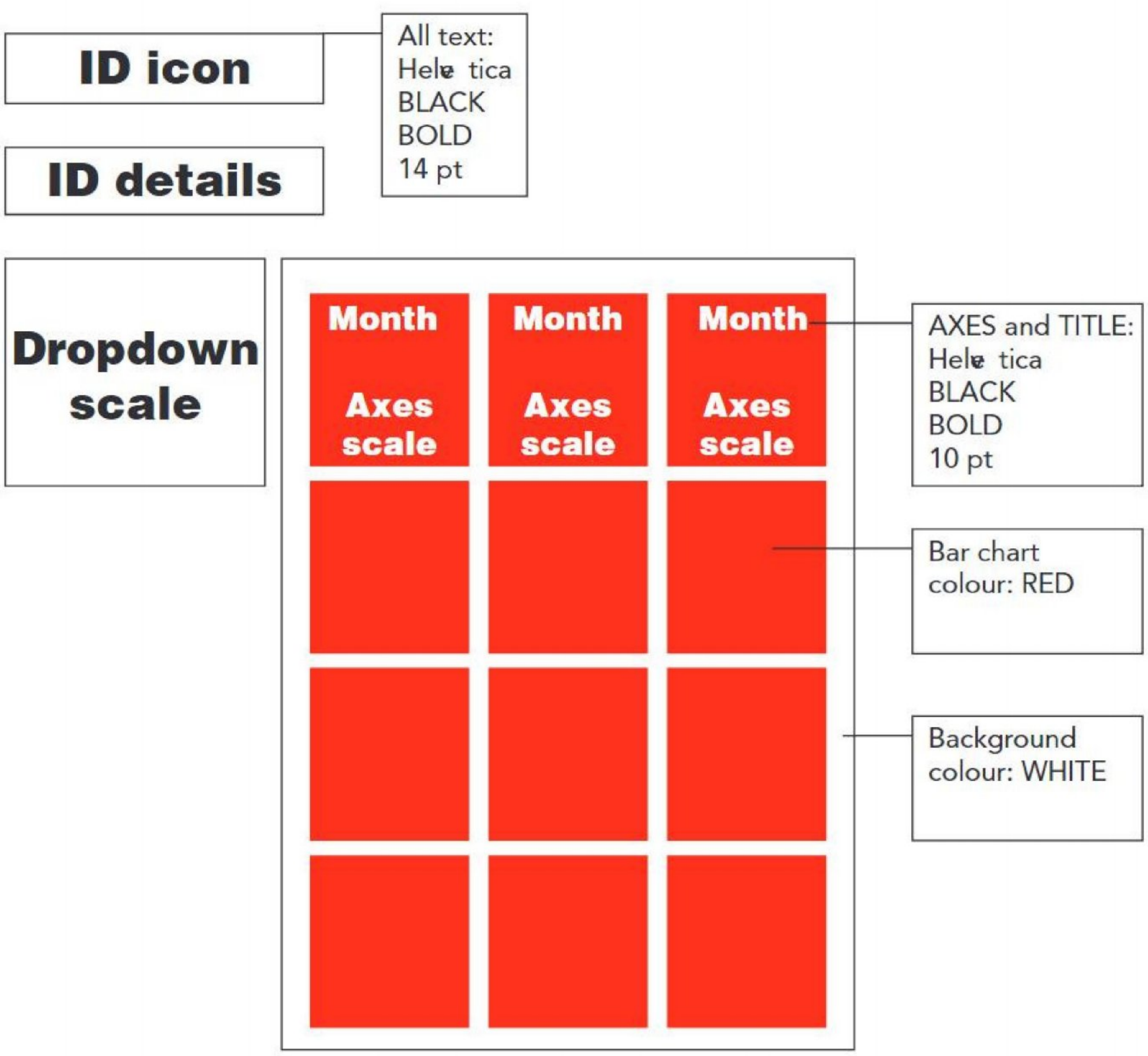


FIGURE 2.31 An example of an annotated diagram for the Daily energy bar chart and for the Energy dashboard

2.4 THINK ABOUT APPLIED COMPUTING

1. Hand draw one of the charts from this chapter as an annotated wire-frame diagram similar to Figure 2.31. (You will not be able to identify specific typefaces when you do this.)
2. Label the headers, axis titles, labels and chart type on the hand-drawn visualisation.

The sketch is usually 'rough' in black and white, and contains structure and placement detail.

Mock-ups are usually full-colour, fully featured 'fakes' that give an indication of what the final visualisation may look like.

What is the difference between a wire frame and a mock-up?

A wire frame is typically a rough sketch drawn in a single colour that provides layout information: location, spacings, and content details listing colour, style, font and size (if known).

A mock-up provides format information. Colour, size, font and style details are shown on a full-colour 'fake' or facsimile that creates a 'look and feel' of the final visualisation solution.

The annotated diagram should clearly indicate:

- the shape or type of graphic solutions
- titles and layout of the diagram
- formats and conventions that are used in graphic solutions.

Before using an annotated diagram, you should consider the data set that will be used as the basis of the graphic solution. The amount and type of data, and purpose for developing a graphic solution of the data, will help you decide on the type of graphic solution to create.

Functionality design tools

IPO charts

An **IPO chart** (Table 2.1) is used during the design stage to clearly identify the solution's input and output, and the processing steps required to transform the data into information. By completing an IPO chart, the developer gets a sense of how much formula development work might occur during the manipulation stage of the project. An IPO chart can be used to show how data is processed into meaningful information. An IPO chart, also called a 'defining diagram', identifies what data is required for the solution (input), what information the solution needs to produce (output), and the processing manipulation activities required to transform the data into information, or the function of the solution. In the case of creating a graphical solution, the input requires an understanding of what data is needed and where it is coming from, the process focuses on the functionality of the solution and the output refers to the graphical solution that is being created.

TABLE 2.1 An example of an IPO chart for investigation into Energy consumption in 2017

Input (data)	Processing (calculations)	Output (information)
Daily 30-minute readings for energy consumption	Tally the value for each day	Daily total
Weekly energy consumption	Tally the results for each week 1–52 for 2017	Weekly total
Monthly energy consumption	Tally the results for each month	Monthly total
Amount of solar energy generated daily	Tally the negative values	PV solar energy
Energy bill for each month *Reminder: Do not include PV Solar values*	Tally the energy bill using peak (3 p.m.–9 p.m.)/off-peak tariff (other times)	Energy bill totals

Flowcharts for producing a visualisation

This flowchart (Figure 2.32) indicates the procedural stages that must be completed when creating a data visualisation.

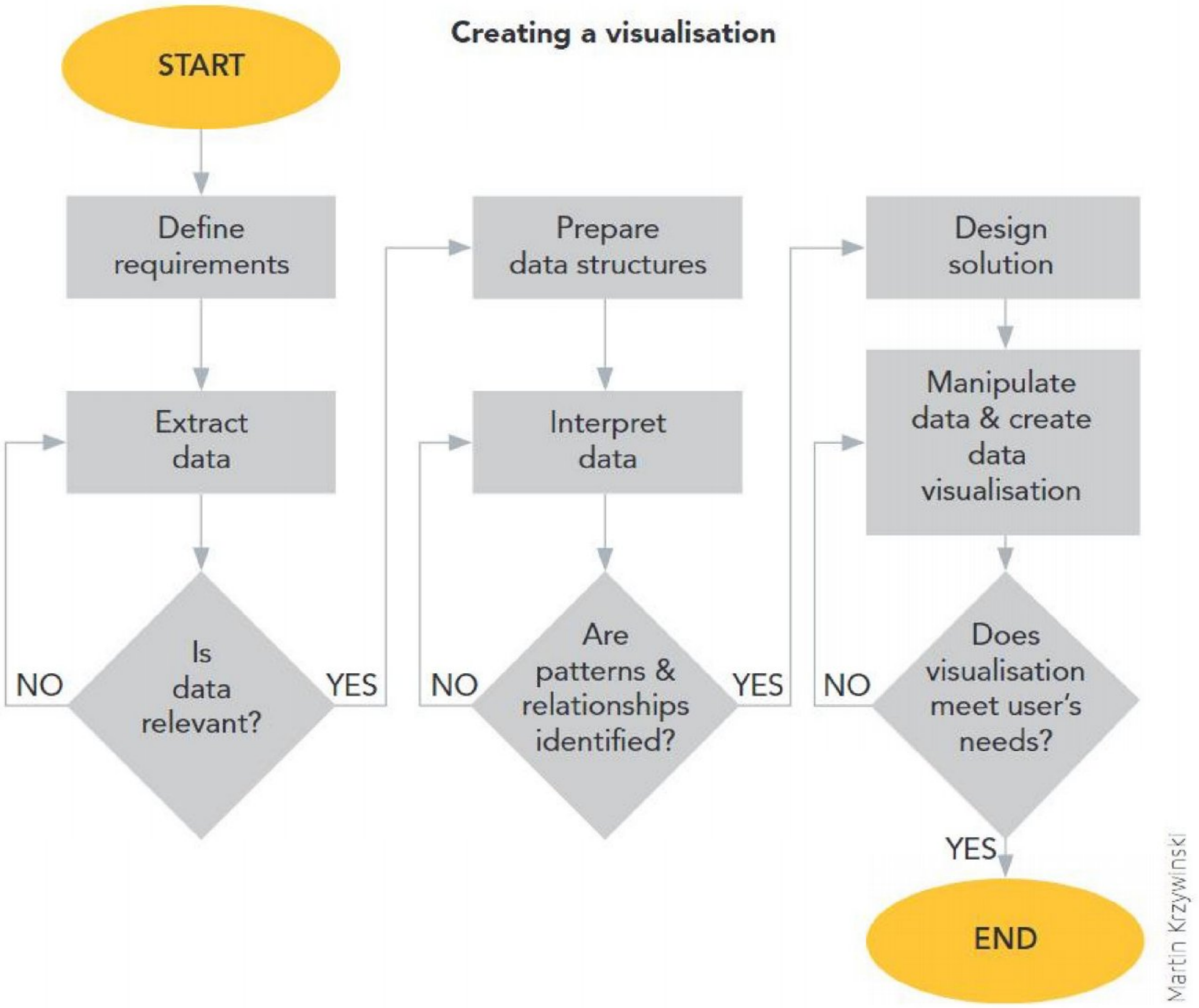


FIGURE 2.32 Flowchart on how to make a visualisation

Using temporary guidelines

When planning your solution, manage the elements of your data visualisation by drawing a visual 'grid' (Figure 2.33). This will help you to work out where you can place items in your solution, how much space you have, and what kind of alignment works best for each element.

This approach can be applied to database input forms and reports, and to spreadsheet layout(s) for workbooks with multiple worksheets, as well as data visualisation graphic solutions, which often have multiple charts and text combined in a single dashboard screen.

Figure 2.33 is a generic visual grid that can be adapted for database, spreadsheet and visualisation designs. By adopting a template that can be repeatedly applied, you introduce consistency into the look of your solutions, and improve your **efficiency** by reducing the time taken in arranging and rearranging objects for your solutions.

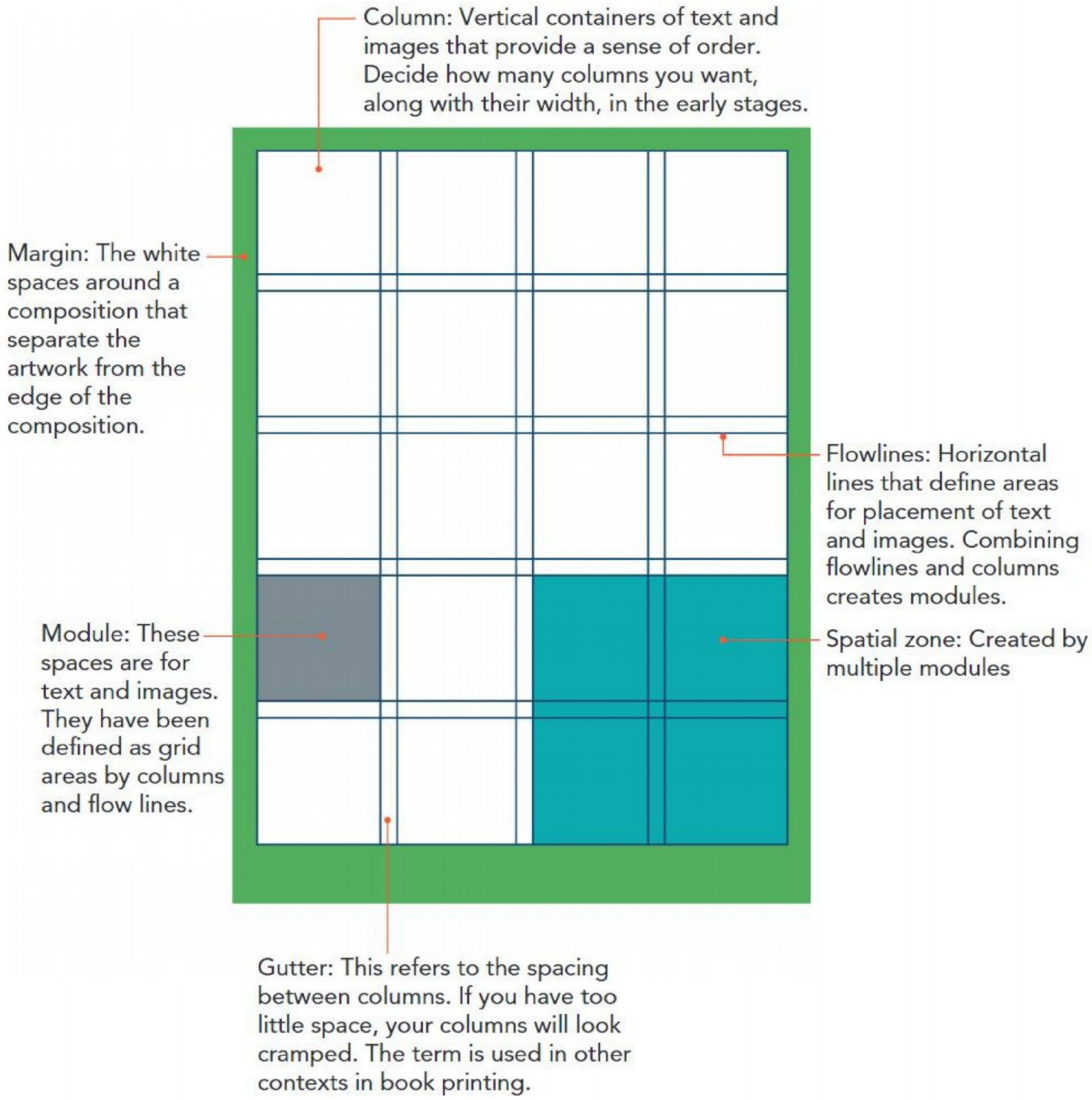


FIGURE 2.33 The visual grid will not just help you with alignment; it also helps with balance, space and accessibility.

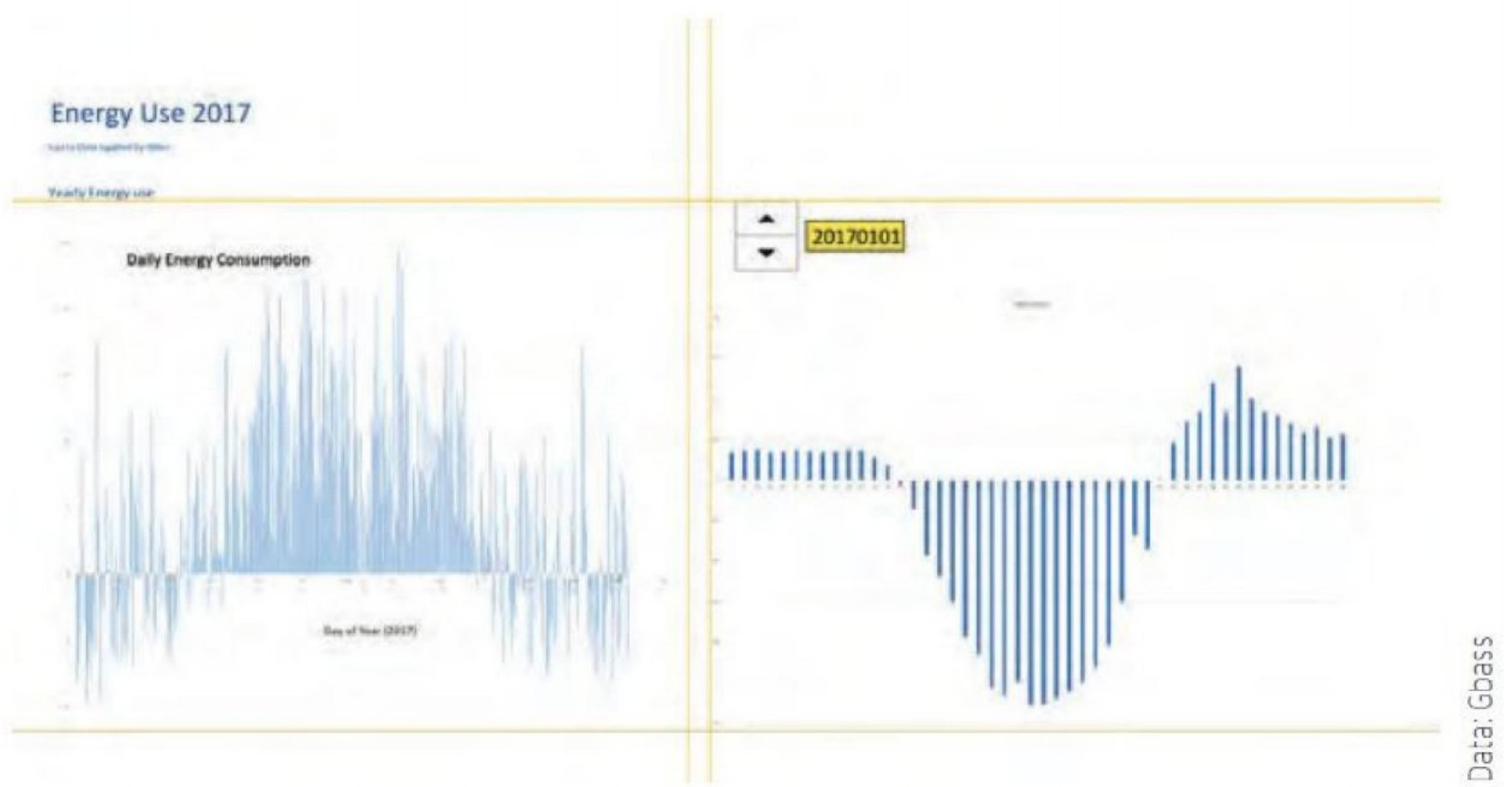


FIGURE 2.34 Imagine the layout of your visualisation with a grid overlay to align each element. Heading, label, legend/key and axes can have their priority changed by placement or text emphasis (size, colour, style).

Formats and conventions

THINK ABOUT APPLIED COMPUTING 2.5

Study a number of data visualisations you find effective. Try to identify the formats and conventions that have been used to make the solution stand out.

ISO 8601 **Date standard** specifies dates should be represented in the form YYYY-MM-DD.

The hyphens can be omitted to use space more efficiently. For example, 2 April 2020 becomes 20200402.

Time can also be represented in a similar way using HH-MM-SS. Hours are represented using a 24-hour clock. For example, 2:34 p.m. becomes 1434.

Dates and times in these formats are easily comparable and can be sorted.

Formats are related to the font type and size selected, background colour used and any other change of appearance. **Conventions** are general rules that are followed when using a particular format. When we present data in a table, often the column headings are in bold, the text left aligned and the numbers right aligned. When presenting data in a website, we use clearly named hyperlinks to make it easy for the user to find the required information. When addressing a letter, there are standard rules (or conventions) which should be followed. For example, the stamp goes in the upper right-hand corner, the return address goes in the upper left-hand corner while the 'to' address is centred on the envelope (Figure 2.35).

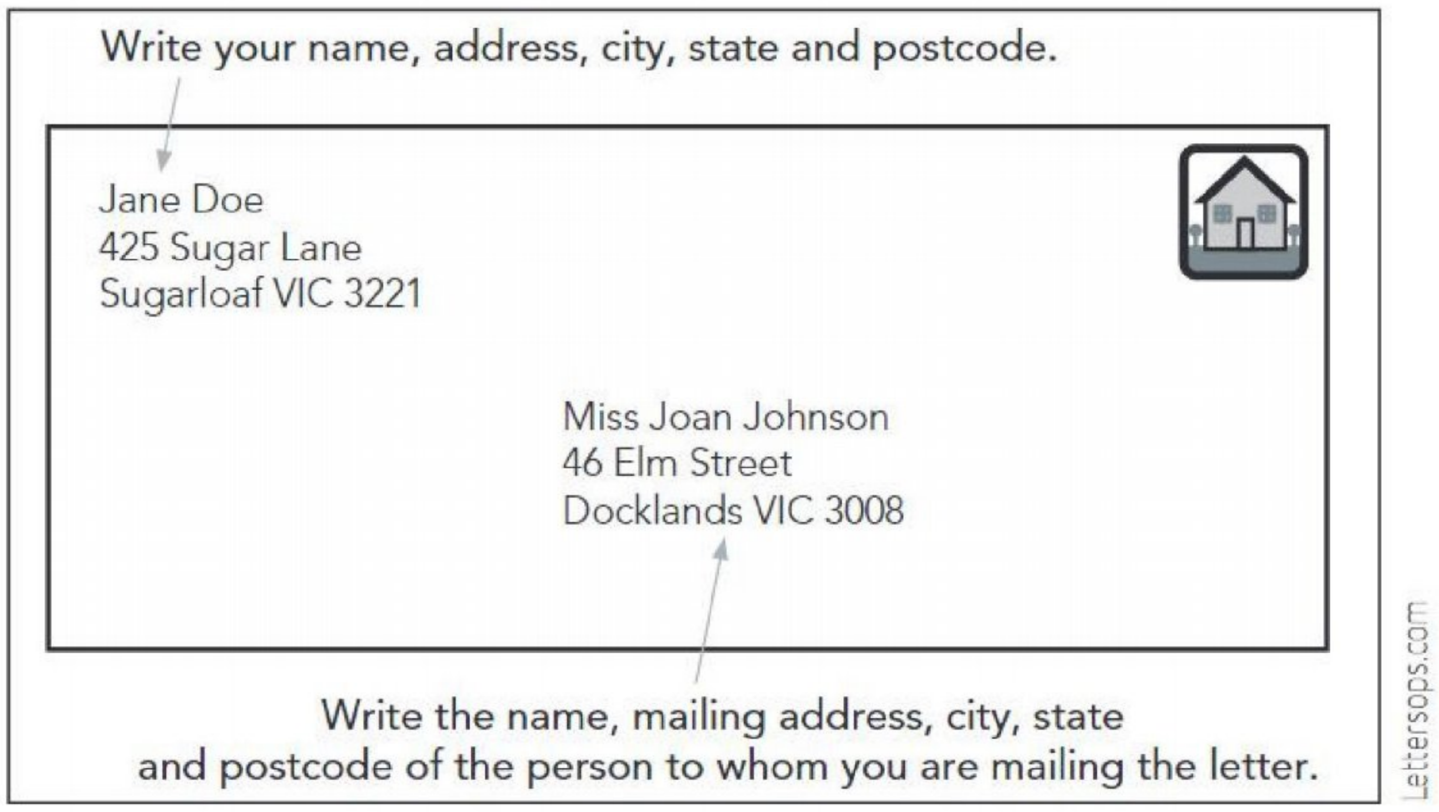


FIGURE 2.35 Conventions followed when addressing an envelope

With data visualisations appearing in so many different formats, there really is a range of conventions that would apply to each format.

Some common conventions that would improve the effectiveness of the visualisation would include:

- clearly title the visualisation and explain its purpose
- label axes if appropriate
- use a key or legend to identify different data items
- include the name of the author and the source of the data
- identify the units of measurement
- choose colours that match the information being discussed.

Filenaming

The International Organisation for Standardisation (ISO) established date formats in 1988. These references can be included in filenames. For example, 20200407dataVisVer2.xlxs

Data visualisations

In addition to the basic design principles you should follow when creating your data visualisation, there are a number of useful formats and conventions suitable for graphic solutions, such as titles, text styles, lines and arrows, sources of data and legend, and colours and contrasts.

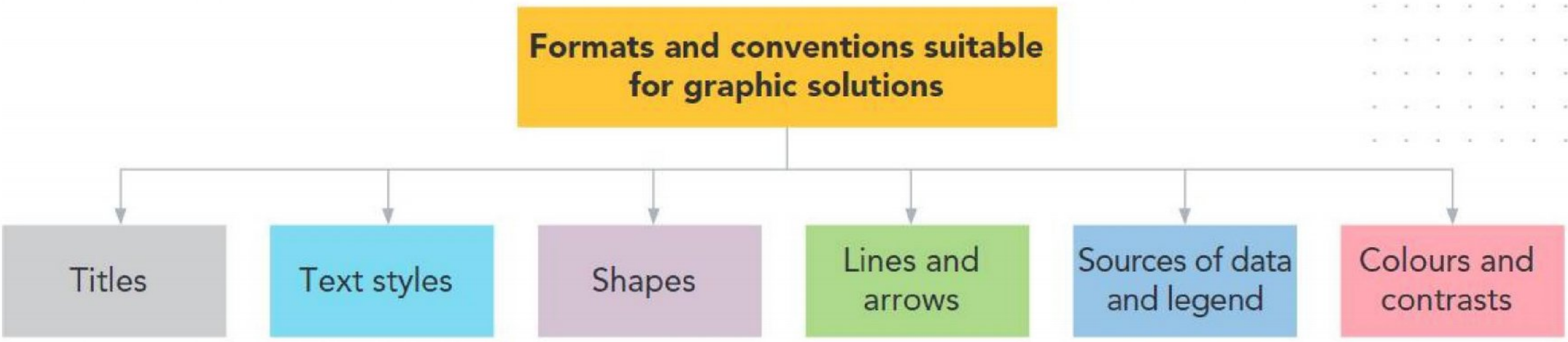


FIGURE 2.36 Formats and conventions

Titles

In the simplest of terms, adding a title to a document makes it a dominant element. Titles are generally styled as headings, with type that is bold and larger than the body text or subheadings. Titles make an impression. Titles should be concise, to-the-point and easy to say. Your title should be in larger text than the rest of your solution – perhaps at least 20pt if the body text is 10pt. Location could be left justified at the top, or centred for greatest emphasis. Bear in mind the F-shaped reading patterns of uncommitted readers.

Text styles

When we discuss text styles, we are essentially talking about fonts. A font is a typeface (such as Times New Roman, Arial or Calibri) plus its attributes (such as 20pt, bold, red). You may already know a few standard, familiar typefaces, such as:

- Times New Roman, a serif typeface. Serif typefaces have tiny marks or 'tails' on the end of the horizontal and vertical strokes of each letter. Serifs are used in books for body text and especially for long passages of text. This paragraph is in a serif font.
- Arial, a sans serif typeface. Sans serifs do not have the serifs on the strokes of each letter. They work best for short paragraphs, large headings and online, but not for long passages of printed text. The 'Think about Applied Computing' box (right) is in a sans serif font.
- Courier New, a slab serif typeface, often used in programming. Slab serifs are best used when the focus is on function and not appearance. Slab serif fonts ensure your characters are legible and unmistakable.

THINK ABOUT APPLIED COMPUTING 2.6

Compare these dates and times using local AU/UK, US, JP formats and ISO8601:

US format: MM/DD/YY

AU/UK format: DD/MM/YY

Japanese format: YY/MM/DD

a 20th of November 2021
b 11th of February 2020
c 11:59 p.m.
d 12 a.m.
e 12 p.m.

Serif fonts	Sans serif fonts
Courier New	Arial
Georgia	Calibri
Times New Roman	Helvetica
	Verdana

FIGURE 2.37 Fonts with and without (sans) a serif offer different emphasis and may be suitable for titles, axis labels, value labels and digits. Generally, sans serif fonts are clearer for readability.

There are other typeface styles, such as handwriting, script and decorative. You will know decorative fonts such as Impact because it is the typeface predominantly used in memes online. Comic Sans, a casual script or handwriting typeface, is also well known.

When you are choosing text styles for your graphic solution, keep things simple. Use a few well-chosen typefaces, perhaps three at most, and use bold, italic, colour and point size to set out heading levels and distinguish between different types of text.

Text styles will apply some contrast while promoting a streamlined, professional appearance. However, using many different typefaces in one space can be untidy and overwhelming.

Remember: less is still more – bigger is not always better. Really think about what needs to be emphasised the most and what needs to be highlighted. Not everything needs to be bold, italic and 40pt.

Lines and arrows

A line is a versatile visual element that uses only length and width. Lines can be:

Solid	
Dashed	
Dotted	
Broken	
Double	
Thick	
Thin	
Curved	
Freeform	

- Bold or thick lines work well for emphasis and for representing a structure within a space. The thicker the line, the more it will draw the eye to the space, but the more crowded and boxed in it will look, so use a thick line carefully.
- Light and fine lines can suggest technical details but also retain a sense of minimalism. Line thicknesses you might use range from:

0.25pt
0.5pt
0.75pt
1pt
1.25pt
1.5pt
1.75pt
2pt

You can use lines in your visualisation as borders or containers for sections of text, images or logos.

THINK ABOUT APPLIED COMPUTING 2.7

1. Identify the differences between Pascal case, camel case and Hungarian case naming conventions.
2. Hungarian case is no longer recommended when hyperlinks are used. Why not?
3. Spaces are discouraged in filenames and variable names. Why?

Charts

While there are very few mandatory rules, there are some strong recommendations for effective charts:

- Simplicity is preferred over complexity.
- Use colour tone rather than colour variety.
- Label axes with meaningful names, include units or denominations.
- Careful choice of labelling may reduce the need for legends and keys.
- In most cases, begin value axes at zero.
- Cite the sources of the data on the chart.
- If choosing a pie chart, for less than six categories, clearly label each sector with name and value. Ensure the pieces add to 100%.
- If choosing bubble chart comparisons, it is the area that is the comparison value, rather than the diameter.
- When plotting all the data, ensure there is clarity by highlighting the focus of interest.

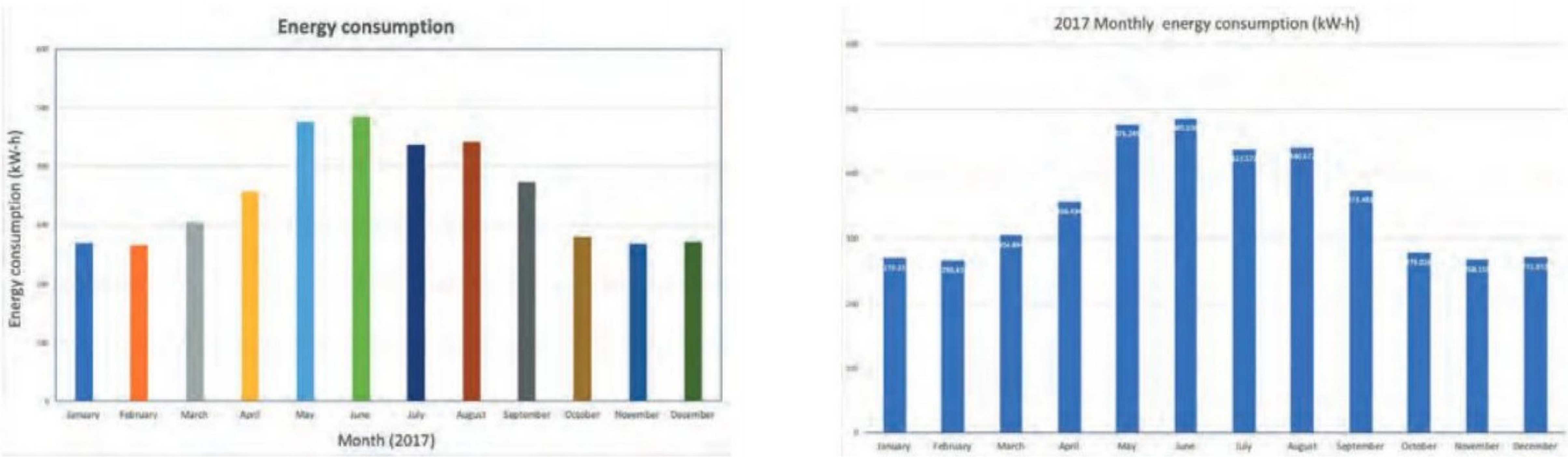


FIGURE 2.38 Simplify charts by reducing borders, gridlines and legends. Labelling is often duplicated without improving understanding – for example, different colours for different months when the months are already labelled.

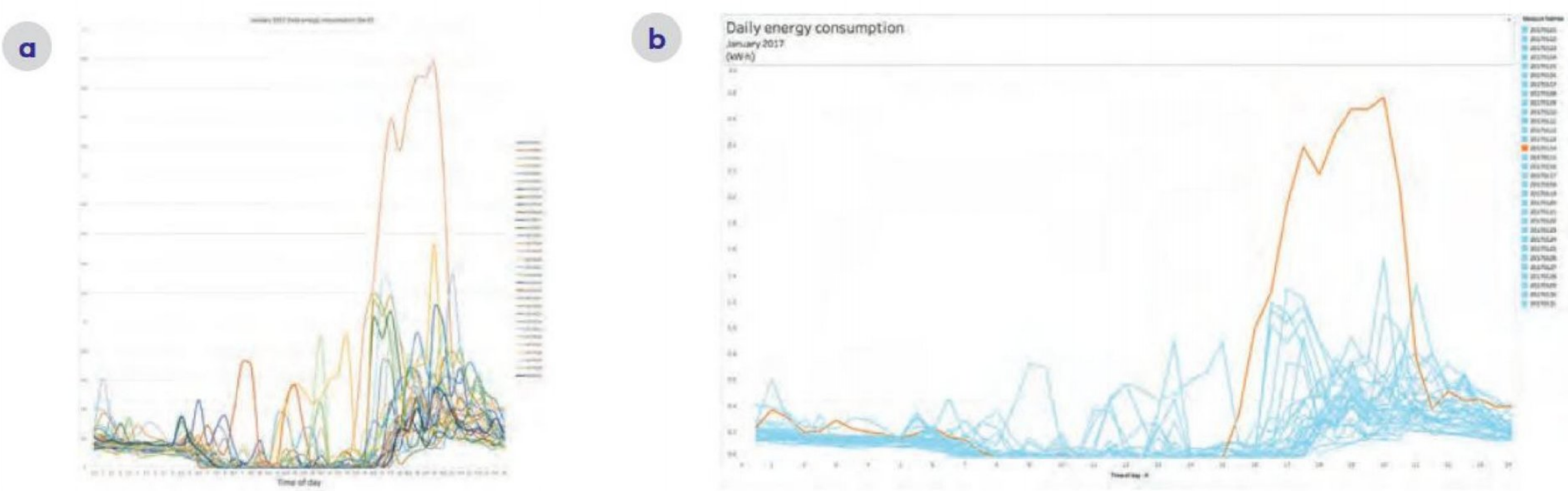


FIGURE 2.39 Line charts showing daily changes for January. (a) Excel default showing a colour for each of 31 days, individual colour highlight can be achieved using Developer Controls. (b) Tableau allows simple choice of day to highlight. Both allow tooltip hover to provide specific values for data points.

Structural characteristics of database and spreadsheet software

Before we consider the formats and conventions of database and spreadsheet applications, we will refresh our understanding of the features of spreadsheets and databases. There is a terminology that is assumed when dealing with data, so we need to be sure we have a shared understanding of those terms.

TABLE 2.2 Terminology for organised data in database and spreadsheet applications

Cell	The intersection of a row and a column, holds one item of information
Field	An item of information. Column headings correspond to a field name
Record	Data about one specific object is stored in records, and held in rows
Table	Consists of rows and columns, or records and fields

a

StudentID	ST20-351
LastName	White
FirstName	Sam
Initial	S
Age	18
Subject	Specialist Maths

b

StudentID	LastName	FirstName	Initial	Age	Subject
ST20-351	White	Sam	S	18	Specialist Maths
ST20-352	Black	Jess	J	19	Data Analytics
ST20-353	Green	Jo	J	17	Software
ST21-354	Gray	CJ	C	19	Visual Communitaion
ST21-355	Apricot	Lou	L	18	Legal Studies

FIGURE 2.40 Database views. (a) Record view shows fields and values for one record. (b) Table view shows records in each row and fields in each column. Note: there are no reference co-ordinates for each field value in the table view.

	A	B	C	D	E	F
1	StudentID	FirstName	LastName	Initial	Age	Subject
2	ST20-351	Sam	White	S	18	Specialist Maths
3	ST20-352	Jess	Black	J	19	Data Analytics
4	ST20-353	Jo	Green	J	17	Software Development
5	ST21-354	CJ	Gray	C	19	Visual Communitaion
6	ST21-355	Lou	Apricot	L	18	Legal Studies
7						
8						

FIGURE 2.41 Illustration of a spreadsheet table holding records as rows, with columns as fields and values in cells

While a spreadsheet has just a table view, a database can show a table view or a record view that shows each field value for a record.

Spreadsheet cell references and names

Cell references

Spreadsheets can be used for many purposes that involve calculations. For example, spreadsheets can be used for account keeping or stock control, to present budget information as part of a project, or to store and manipulate data from surveys or results of an investigation.

The rows and columns collectively are called a worksheet. A number of worksheets can be included in a spreadsheet file. Specific locations in a spreadsheet are known as cells. The reference convention is to list column, then row. An alternative is to use R1C1 notation, where rows and then columns are listed. For example, cell A1 would be [1,1]. There are advantages in using RC notation.

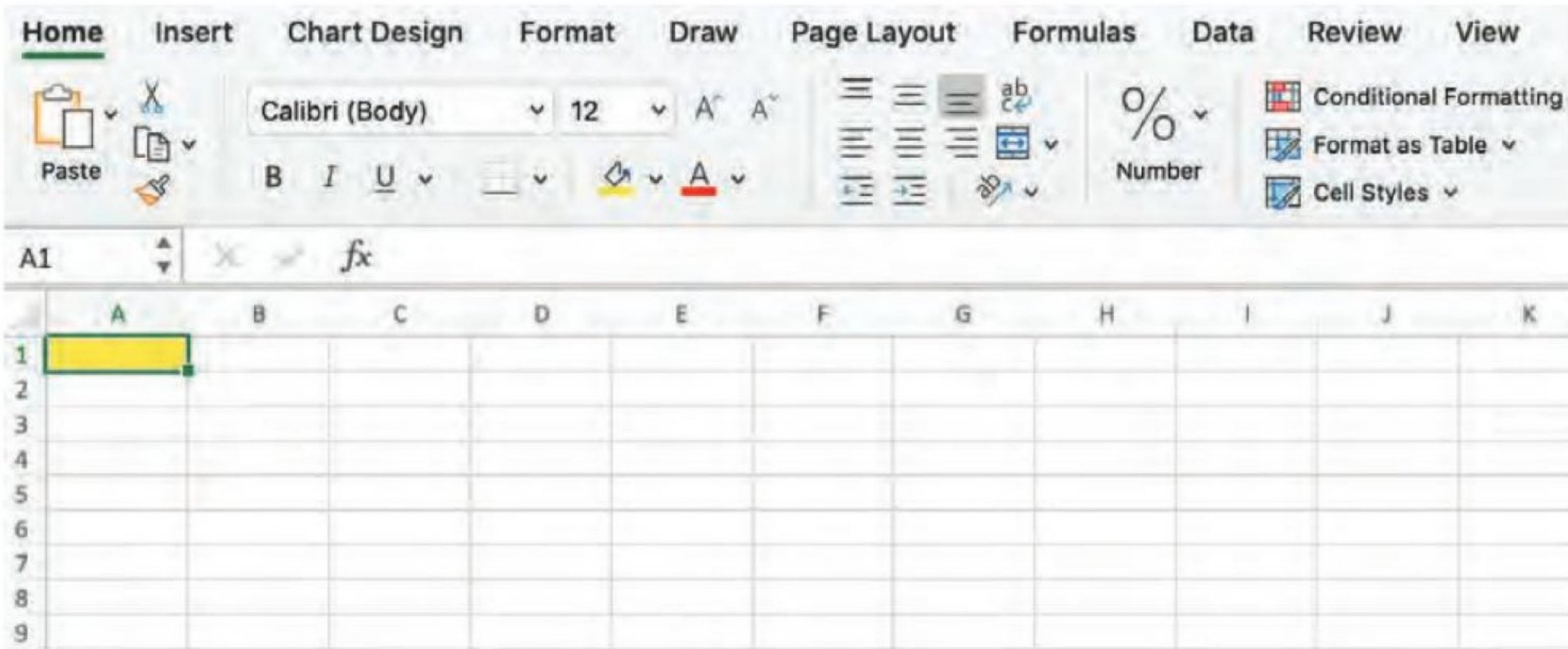


FIGURE 2.42 Excel spreadsheet layout. For example, A1 is the first top left cell.

By default, cell references are relative to the location of the cell. For example, if you refer to cell A2 from cell C2, then it is referencing the same row (Row 2) and two cells to the left (C to A).

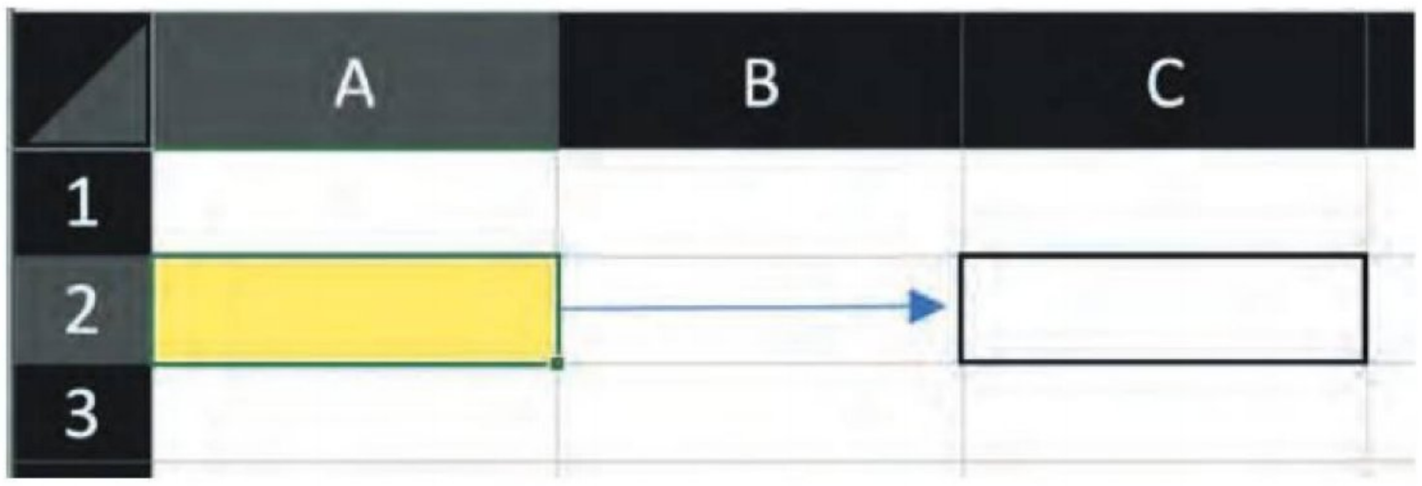


FIGURE 2.43 Relative positions of cells A2 and C2

Cell references can be also be absolute. When cells are copied, the cell reference can maintain the relative reference.

Alternatively, an absolute requirement can be imposed. The absolute reference is shown by the dollar symbol ($). There are three possible absolute reference combinations: two mixed references ($A1 and A$1) and one absolute reference (A1).

Cell naming

In Excel, a name can be given to a cell or range of cells. For example, instead of using the cell reference A1 or A1:C1, simply use the name you have assigned to it. When referencing those cells in a formula, just use the assigned name: =SUM(VALUES) instead of =SUM(A1:C1).

THINK ABOUT APPLIED COMPUTING 2.8

Search the web for information about the advantages of using R1C1 and A1 notation. Consider:

- relative references
- formulas in tables.

9780170440806

TABLE 2.3 Summary of reference types when a formula containing a cell reference is copied one cell down and two cells to the right (from A1 to C2)

If the reference is:	It changes to:
\$A\$1 Absolute column Absolute row	\$A\$1 The reference is absolute
A\$1 Relative column Absolute row	C\$1 The reference is mixed
\$A1 Absolute column Relative row	\$A3 The reference is mixed
A1 Relative column Relative row	C1 The reference is relative

There are several methods for creating name ranges in Excel.

Method 1: Use Define Name

Select by highlighting the range you want to name. Go to Formulas > Define Name.

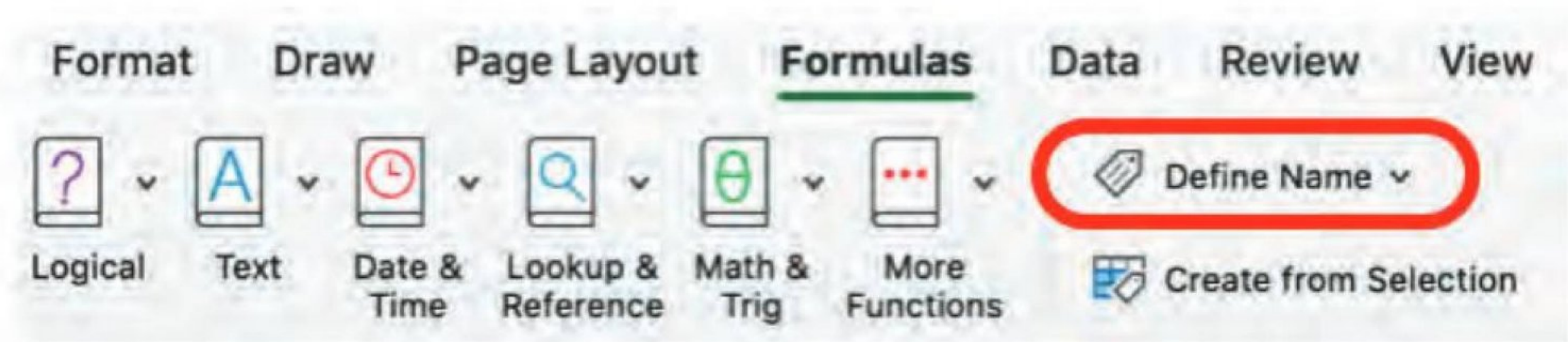


FIGURE 2.44 Create a named cell range under Formulas > Define Name.

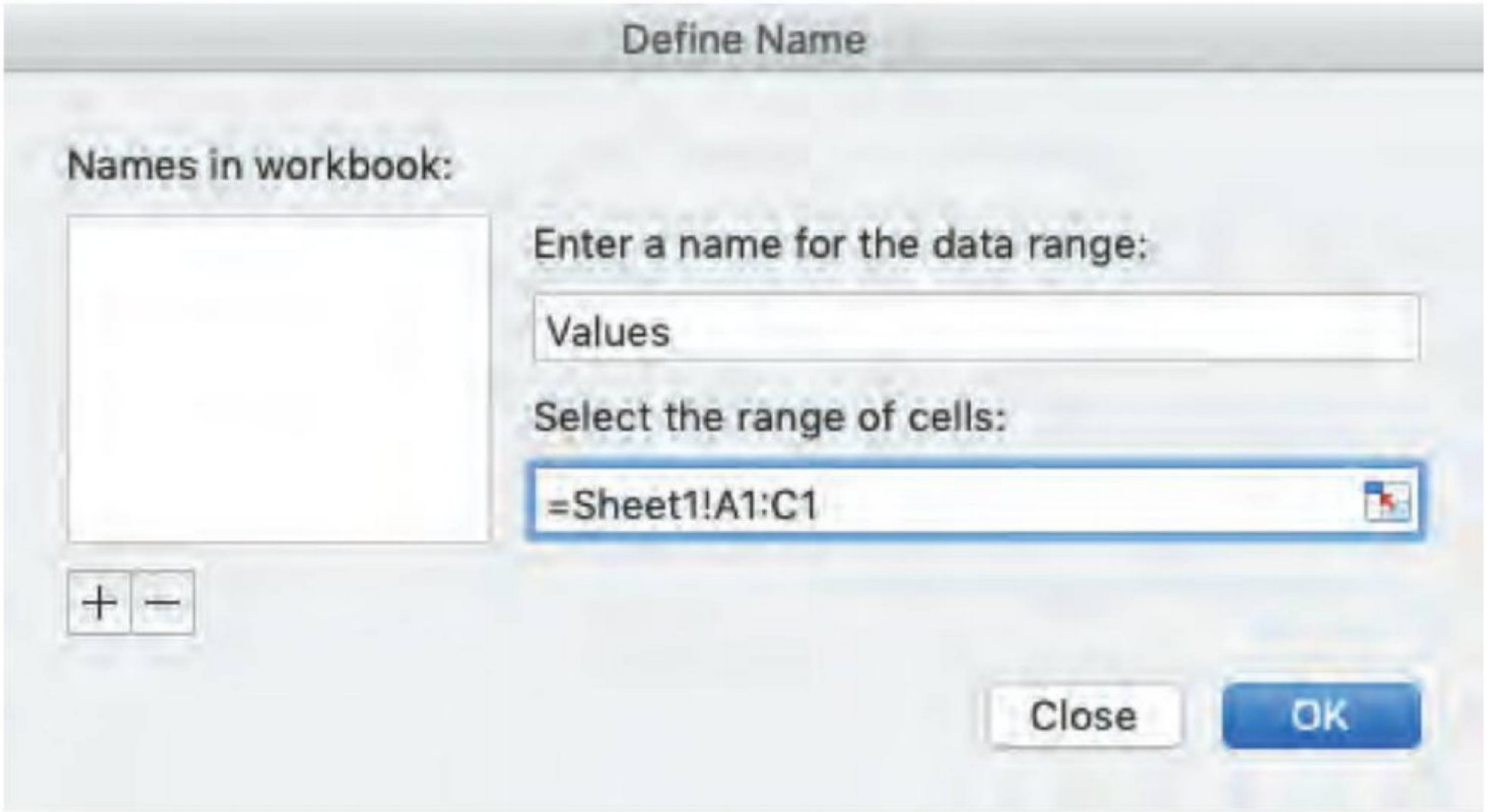


FIGURE 2.45 Enter the chosen name and range.

Method 2: Use the Name Box

Select by highlighting the range of cells that you want to name. Go to the Name Box at the left of the Formula bar, then type the name you have chosen for those cells. Note that the name created by Method 2 will apply to the entire Workbook. If you want a local name that only applies to this worksheet, use Method 1.

FIGURE 2.46 Create a named range by typing directly into the Name Box.

Method 3: Use Create from Selection

Go to Formulas > Create from Selection. Then select one or more of the options.

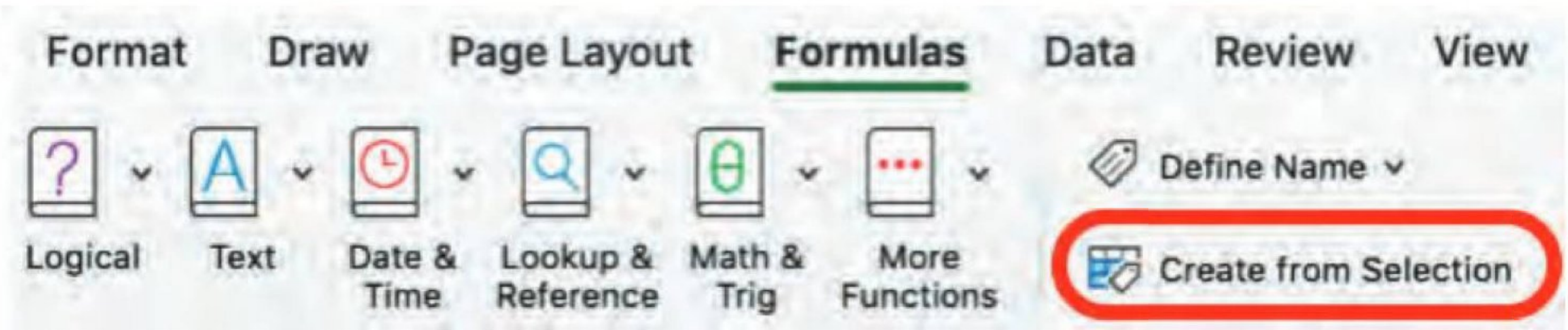


FIGURE 2.47 Create a named cell range under Formulas > Create from Selection.

FIGURE 2.48 This method is useful when you have Column Headers. If you choose several boxes, then several names will be chosen from the field names.

A spreadsheet is not a database

Many of the functions of a spreadsheet can achieve similar results to a database. For example, each spreadsheet row behaves like a record; data and records can be sorted, searched and filtered; and data can be extracted from other sources. The main difference is the size of the data set that can be manipulated. Spreadsheets require the data set to be held in the RAM of the local computer. For Excel, this is a limit of around several million records or approximately 10 GB, depending on the memory specifications of the computer. Database applications have no limit to the size of the files that are accessed. The efficiency of the operation – for example, search or sort – will be affected by the complexity of the instruction and the number of records to be processed. Processing times would be measured in hours or days in these instances.

In many business operations, where the file size is small and the number of records is only a few hundred thousand, spreadsheets are used instead of dedicated database applications. The opportunity for error is greatly increased if cells are left 'open' or inexperienced users gain access to the master file.

Databases

Formats and conventions for database activities are not well defined. The basic rule is for ease of use and predictability. Every database developer has their own style. The more experience you gain with database operations, the more you can adapt to the style of the database developer. Consistency is the strength of any convention, though exceptions may be permitted, if the reasons are documented and recorded.

Some common conventions for databases are:

- Adopt a pattern and apply the rules consistently to tables, views/reports, columns and keys.
- Camel case is a simple naming convention to apply, where the first word is lower-case and the other words start with upper-case – forExampleCamelCaseImprovesReadability.
- Include the type in the name (see Table 2.4).

TABLE 2.4

Table	tbl_EnergyUse
Report	rpt_EnergyConsumption
Primary key	date_id
Foreign key	location_id

Naming conventions are a matter of choice. Your task is to consistently apply whatever scheme you decide on, and document what you are doing. When maintenance is required some months later, you may be required to revisit the database scheme. To avoid frustration and wondering what the developer was thinking, a documented naming approach is recommended.

Spreadsheets

The first SSRB standard recommends that a spreadsheet should '*always be considered from a user's perspective rather than a developer's perspective*'. BPMS1-1 Clarity of purpose must be a primary consideration for the developer when creating workbooks, sheets and components within a sheet.

An overall approach for every spreadsheet is recommended to adopt 'purpose-based formatting'. In our situation, this will require distinguishing between sheet types and worksheet cell ranges.

The main ideas for spreadsheets include:

- naming sheets with descriptions of purpose
- using fill colour to define areas of sheets
- using font colour to indicate the content of a cell
- using range names to describe the content or use of the range being named (see cell naming on pages 73–5)
- aligning data with the same type consistently down rows and across columns
- using number formats that are right aligned
- clearly indicating or labelling all units and denominations.

In 2003, the **Spreadsheet Standards Review Board (SSRB)** was established to maintain the 'Best Practice Spreadsheet Modelling Standards (BPMS)'. The full documentation is available as a free download from the weblink.

Best Practice Spreadsheet Modelling Standards

Workbooks should be set to calculate automatically, as pressing the F9 key for manual calculations is not widely known.

If you are developing advanced interactive models, then the range naming prefixes will become useful. Less complex spreadsheets will not require as many categories because the number of range names will not be required.

TABLE 2.5 Generally, it is recommended that every workbook contains a range naming key to prevent any potential confusion in understanding the naming system used in the workbook.

Range Type/purpose	Description/purpose	Prefix
Row Array	Single row, multiple-column, single-area array	RA_
Column Array	Single column, multiple-row, single-area array	CA_
Block Array	Single area, multiple-cell, non-row, non-column array	BA_
Base Cell	Single cell base cell (for INDEX function or OFFSET function reference, etc.)	BC_
Lookups	Names a lookup table range on a lookups sheet	LU_
Hyperlink Cell Reference	Hyperlink cell reference	HL_
Check Box Cell Link	Check box cell link	CB_
Drop Down Box Cell Link	Drop down box cell link	DD_
List Box Cell Link	List box cell link	LB_
Option Button Cell Link	Option button cell link	OB_
Spin Button Cell Link	Spin button cell link	S_
Scroll Bar Cell Link	Scroll bar cell link	SB_

Source: Adapted from 'BPMC 9-6 Range Naming Prefixes', www.ssrb.org/resources/standards/ce/sma9/range_naming

Advanced use of range names allows use of an INDEX function to dynamically update the worksheet data. Advanced Excel tutorials are beyond the scope of this textbook; however, you could search 'Excel INDEX function dynamic Name Range' on the web to discover details about advanced Excel features.

A word of caution about using macros

In many cases, the use of **macros** can greatly improve the capabilities, functionalities and user-friendliness of spreadsheets. However, they should be used with extreme care as they are not forgiving if not recorded or written correctly. Macros are likely to inadvertently cause irreparable damage to a spreadsheet. Backups and saved versions are essential when developing macros.

A macro recorder may be employed to create simple automated execution of a sequence of instructions. No changes can be made to any of the objects (sheet names, range names, charts, etc.) if the macro is to operate reliably. Macros are not recommended for calculations. Macros will be slower than worksheet formulas and not well understood by other developers and users.

High frequency trading (HFT) accounts for 25% of the Australian (Stock) Market, according to the Australian Securities and Investments Commission (ASIC) 2018.
HFT relies on algorithm-driven systems designed to automatically execute quick trades in an effort to profit from short-term price movements. HFT quick trades may be completed in microseconds (millionths of a second). In attempts to leverage advantage, trading offices have updated all connections between computers to use fast fibre-optic switches, and moved office locations, sometimes by as little as 500 m, to gain a few microseconds of advantage.

Software functions

Modern software provides developers with many tools to select and shape the data in order to allow display and reporting of chosen trends, highlights and insights. Many of these functions are regularly applied by advanced users as the 'standard treatment' of manipulating or 'wrangling' data prior to reporting or display visualisations. Further processing of data creates summaries and aggregations which can provide historical trends and allow accurate predictions. Many visualisations can be arranged to be dynamically updated. This can occur automatically to provide a minimal time delay between data capture, manipulation and report being displayed. While most data is not time sensitive or volatile, many financial transactions can be manipulated to advantage if up to the second data analysis is available.

Extracting data

Data that is relevant to specific purposes can be obtained from data repositories such as databases and spreadsheets.

Spreadsheet software

Many data sources provide data in a file format that is compatible with existing spreadsheet software applications.

AFL Victoria

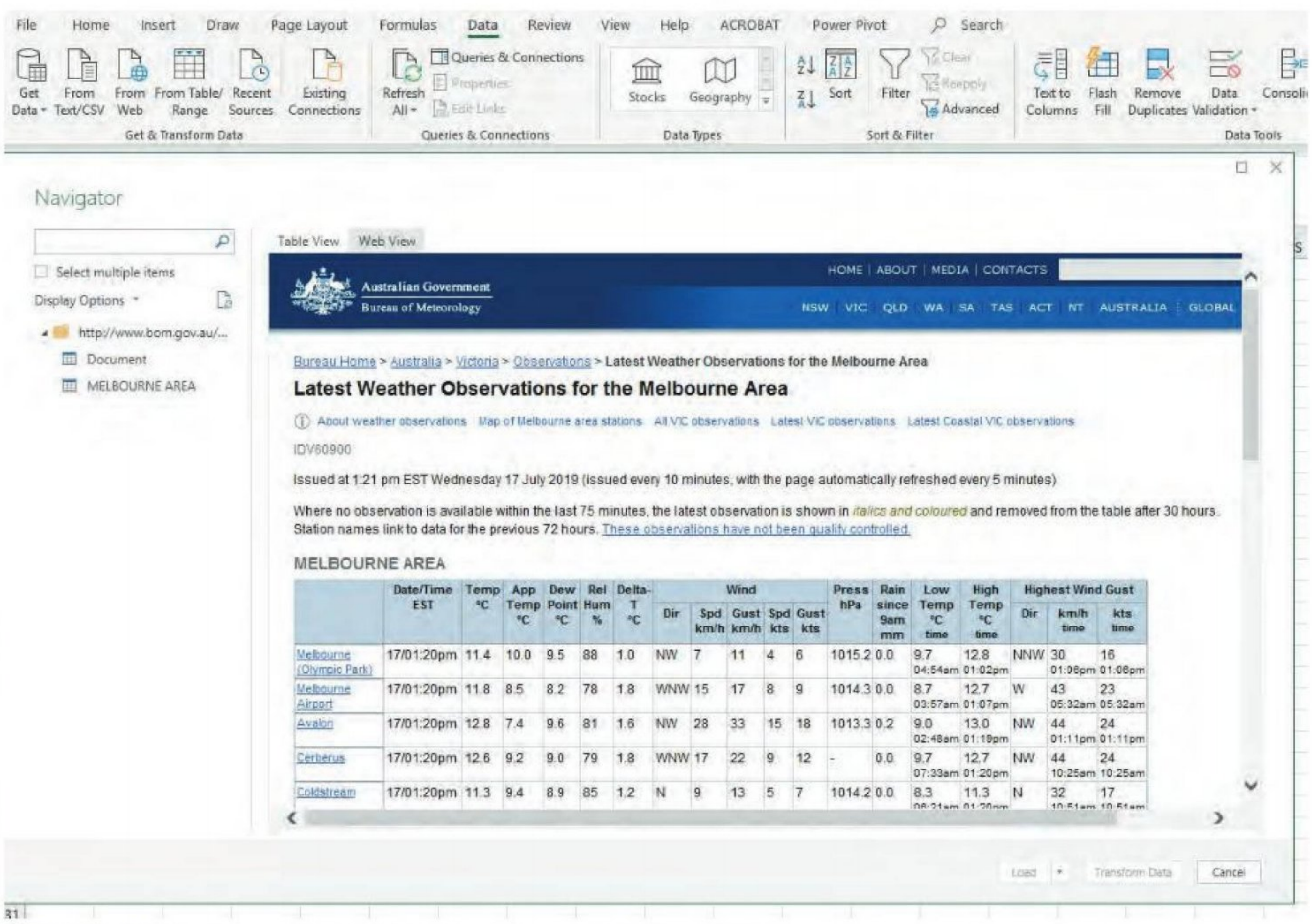
Figure 2.49 shows a data set available from the AFL Victoria website, containing all of the details and locations of AFL teams and programs in Victoria. The data set is available in CSV file format that is compatible with spreadsheet software.

	A	B	C	D	E	F	G	H	I
1	Name	Address	Suburb	Postcode	State	Category	LGA	Region	
2	AFL Auskick - Aberfeldie	Aberfeldie Primar	Essendon	3040	VIC	AFL Auskick	Moonee Valley (	North and West Metropolitan Region	
3	AFL Auskick - Airport West	Hansen Reserve	Airport West	3042	VIC	AFL Auskick	Moonee Valley ;	North and West Metropolitan Region	
4	AFL Auskick - Ajax	Princes Park	Caulfield South	3162	VIC	AFL Auskick	Glen Eira	Southern Metro	
5	AFL Auskick - Alexandra	5665 Maroondah	Alexandra	3714	VIC	AFL Auskick	Murrindindi	Hume	
6	AFL Auskick - Alfredton	Alfredton Reserve	Alfredton	3350	VIC	AFL Auskick	Ballarat	Grampians	
7	AFL Auskick - Allansford	Allansford Recrea	Allansford	3277	VIC	AFL Auskick	Program		
8	AFL Auskick - Alphington/Fairfield		Northcote	3070	VIC	AFL Auskick	Yarra ; Darebin	North and West Metropolitan Region	
9	AFL Auskick - Altona	Altona Football C	Altona	3018	VIC	AFL Auskick	Hobsons Bay	North and West Metropolitan Region	
10	AFL Auskick - Altona Meac	Comden Reserve	Altona Meadows	3028	VIC	AFL Auskick	Wyndham ; Hob	North and West Metropolitan Region	
11	AFL Auskick - Amateurs	Queens Park Oval	Newtown	3220	VIC	AFL Auskick	Greater Geelong	Barwon S/W	
12	AFL Auskick - Anakie	2155 Ballan Roac	Anakie	3221	VIC	AFL Auskick	Surf Coast , Gre	Grampians, Barwon S/W	
13	AFL Auskick - Anglesea	Ellimatta Reserve	Anglesea	3230	VIC	AFL Auskick	Surf Coast	Barwon S/W	
14	AFL Auskick - Apollo Bay	Apollo Bay Recrea	Apollo Bay	3233	VIC	AFL Auskick	Colac-otway	Barwon S/W	
15	AFL Auskick - Apollo Parkways		Greensborough	3088	VIC	AFL Auskick	Nillumbik ; Ban	North and West Metropolitan Region	
16	AFL Auskick - Ararat	Richardson Oval	Ararat	3377	VIC	AFL Auskick	Ararat Rural	Grampians	
17	AFL Auskick - Ararat North	Blake St	Ararat	3377	VIC	AFL Auskick	Program		

Copyright © Department of Planning and Community Development (Vic), Creative Commons Attribution 3.0 Australia

FIGURE 2.49 Data set downloaded into spreadsheet software

Spreadsheet software also contains a feature known as a web query, which allows data to be acquired from online data sources. One limitation of a web query is that the spreadsheet software needs to recognise the data.



Screenshot from www.bom.gov.au

FIGURE 2.50 Example of a web query in spreadsheet software

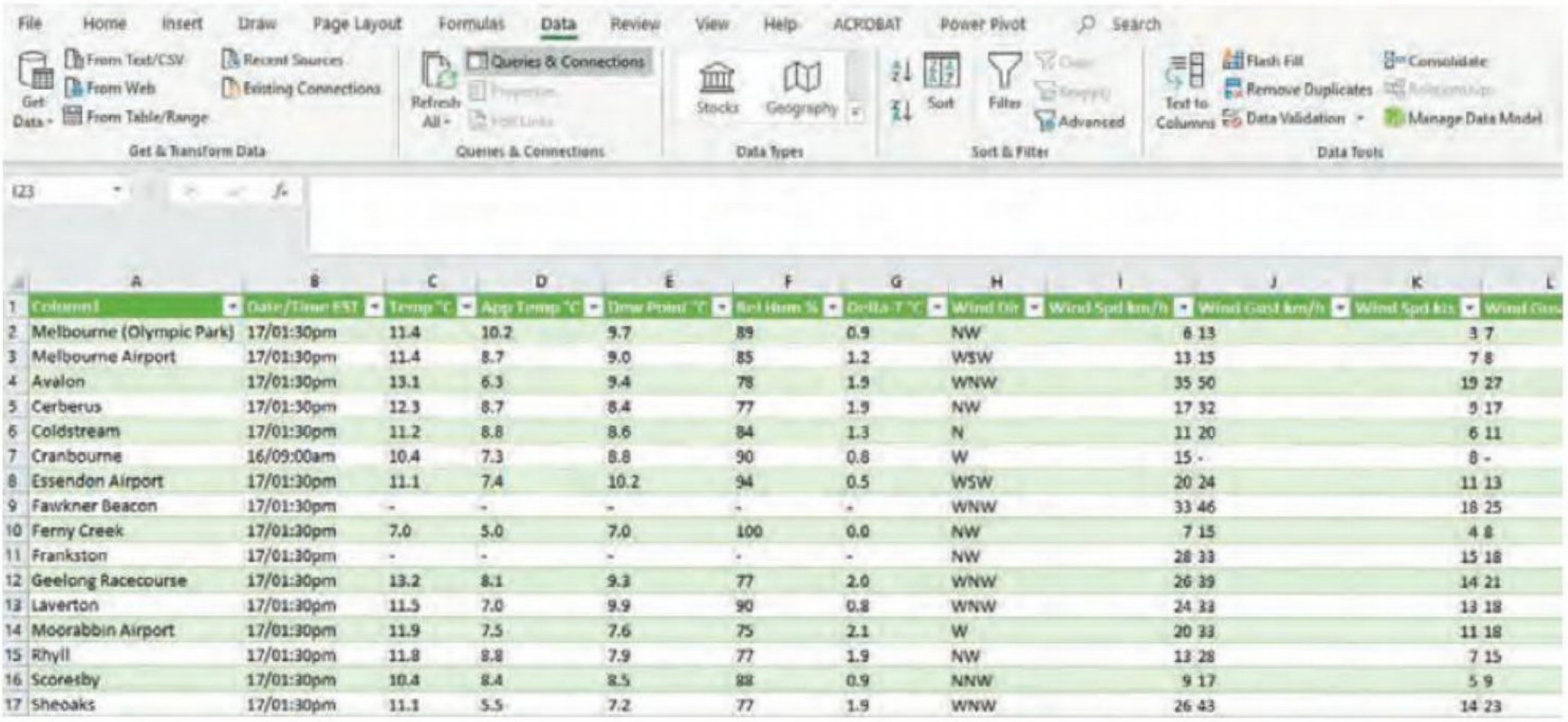


FIGURE 2.51 Data imported into spreadsheet software using a web query

Database software

Database software can also be used to acquire data from data sources. Using the external data function, data can be imported from a range of sources and file formats, including HTML documents, Excel, text and XML files, as well as external databases.

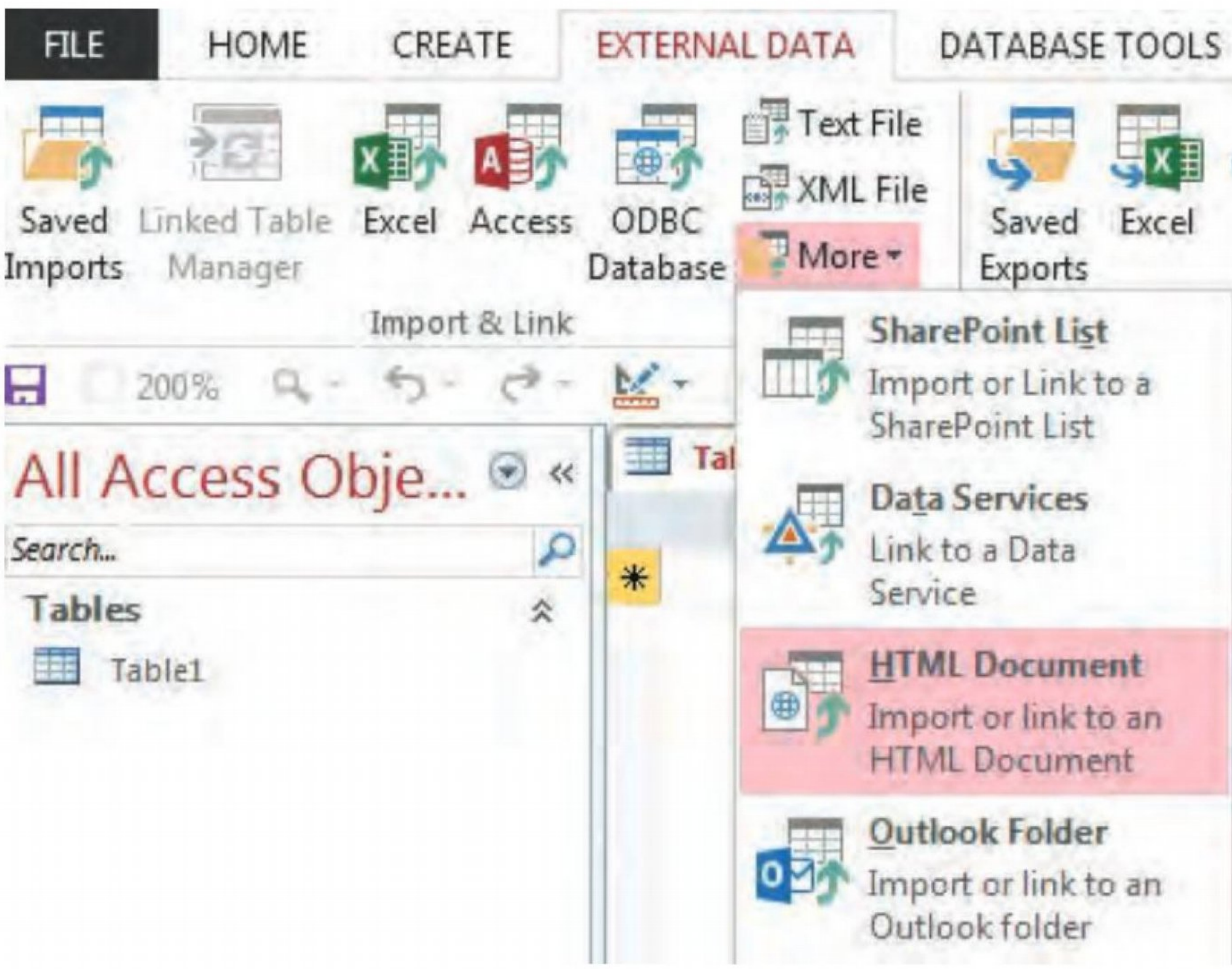


FIGURE 2.52 Options to import external data into database software

Programming languages

Some programming languages have the capability to acquire data from external sources.

Python

Python is a free programming language. It comes installed as part of macOS and must be downloaded and installed on WIN OS. There are many free libraries available that provide the ability to graph and chart data.

TABLE 2.6 Graphics plotting tool libraries for Python

Matplotlib	This was the first data visualisation library for Python. Many other libraries make use of these functions during analysis; for example, pandas and Seaborn
Seaborn	A modern display style with improved colour palettes and formats that uses the old style matplotlib
ggplot	Copied across from ggplot2 for the R plotting system and based on concepts from *The Grammar of Graphics*. ggplot works closely with pandas
Bokeh	A Python native library that does not rely on other libraries, also based on *The Grammar of Graphics*. There are no default settings, so every element is required to be defined
Pygal	Create interactive plots that can be embedded in HTML to be displayed by a web browser. Suitable for small data sets
Plotly	Python can access the online plot.ly to create interactive graphics and offers a greater range of chart options
Geoplotlib	A dedicated library that creates maps

Source: wiki.python.org

Python can fetch (extract) data from websites. Applications can be created in Python that will fetch data from a website based on the page's HTML code.

Using the 'Inspect Element' command in a browser, you can view the HTML code of a web page. The area that the data is located in the code can be identified and then imported into another application. Once data has been acquired, it can be imported into other applications, such as database or spreadsheet software.

Figure 2.53 shows a sample of Python code that will read all of the HTML code from Google.

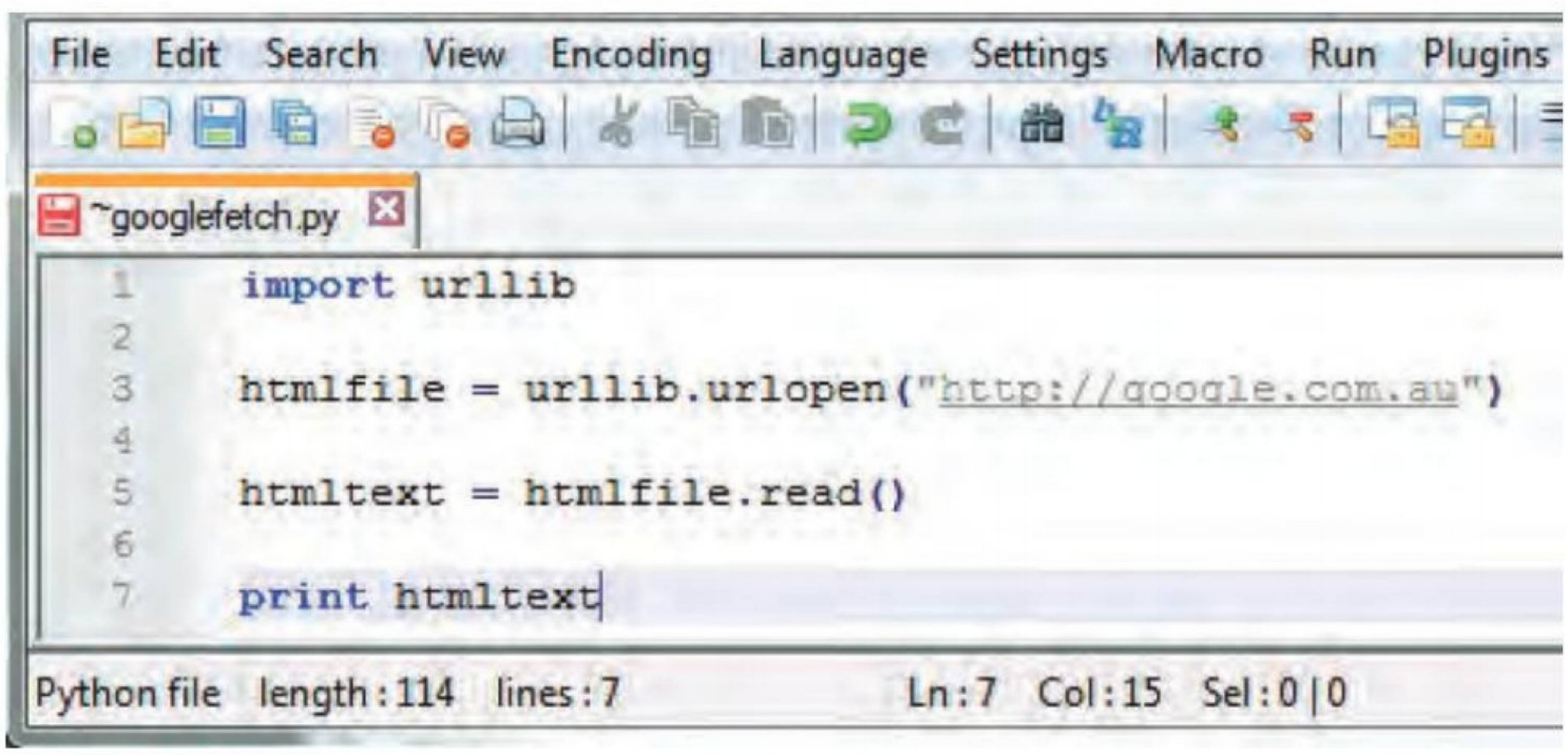


FIGURE 2.53 Sample Python code reading HTML code from Google

Mathematica

Mathematica uses Wolfram Language, which has inbuilt functions to display data acquired in real time from online 'databins' using the data drop website. Mathematica uses online processors so the local machine does not do the computation. Every possible graphing style and function is built in with default settings. Detailed documentation with examples is provided on every feature.

Wolfram Data Drop collects data in real time from a variety of user-defined sources

What is the difference between Python and Mathematica (and R)?

Python is free and open-source. Mathematica is a proprietary application and has an annual cost associated with using the local installed program and the online Wolfram cloud. Matplot is a competing proprietary application that is used in engineering studies, while Mathematica is used in mathematics and physics studies. R programming language is also free and open-source. R requires a deep programming knowledge and is considered too advanced for introductory secondary school purposes. Whether you choose a detailed programming approach with Python or a 'black box' approach with Mathematica will depend on a number of local factors, including:

- student familiarity with coding Python or Wolfram Language
- time available to devote to understanding a new language
- intended further study. For example, will the knowledge be useful in 'later' studies?
- teacher preference. Students are often guided by the amount of support that their school and teacher can provide.

Mathematica and Wolfram Language are available to all Victorian students and teachers at no cost. Further details are available online. Web search 'mathematica free Victoria Wolfram software'.

Validation of data

To make the solution accurate and reasonable, all data that is used should be validated. This means that the original data should be manually checked for illegal data types, for reasonableness, for correct spelling, to ensure that data fall within a correct 'range' or that any codes that are used are consistent (follow similar style) and reasonable (similar codes relate to similar products).

It is easy for data entry errors to occur during the input phase, particularly if a large amount of data is involved. Data should therefore always be validated before any processing occurs.

There are two types of **validation**: manual and electronic validation.

Manual validation occurs when the data entered are checked for accuracy by a person rather than by a machine. Proofreading is one manual validation technique. When you proofread data, you look for transcription errors. The data entered should be compared with the source document to ensure that they match. Any differences observed must be followed up.

Data may be validated by computer if a validation function is contained in the software or built into the solution to a problem. **Electronic validation** techniques, sometimes called machine-validation techniques, ensure the accuracy of data and are built into software, such as spell checkers. Types of electronic validation include range checking, existence checking and data type checking.

A feature of modern software is the inbuilt functions that provide assurance that the data entered will be within predetermined ranges. This validation is simple to incorporate into your design. Usually a warning to the data entry operator will ensure the data is rejected, reviewed and correctly entered. This will create a high level of confidence in the quality of the data.

Validation applies to both databases and spreadsheets when entering data. Data visualisation software is designed to use data from other sources, so validation may already have occurred. **Filtering** of existing data is more likely in that case.

Spreadsheets

Note that on 1 July 2019, SSRB was superseded by the Best Practice Modelling Guidelines (BPMG) by Modano.

Spreadsheet Standards Review Board (SSRB)

Modano

In Excel, forms can be established when there is repetitive data entry. The cells can be formatted to ensure the data conforms to expected ranges. Certain values are disallowed.

Alternatively, a range of cells can be identified for a particular function. (See margin note and weblinks to gain access to further details.)

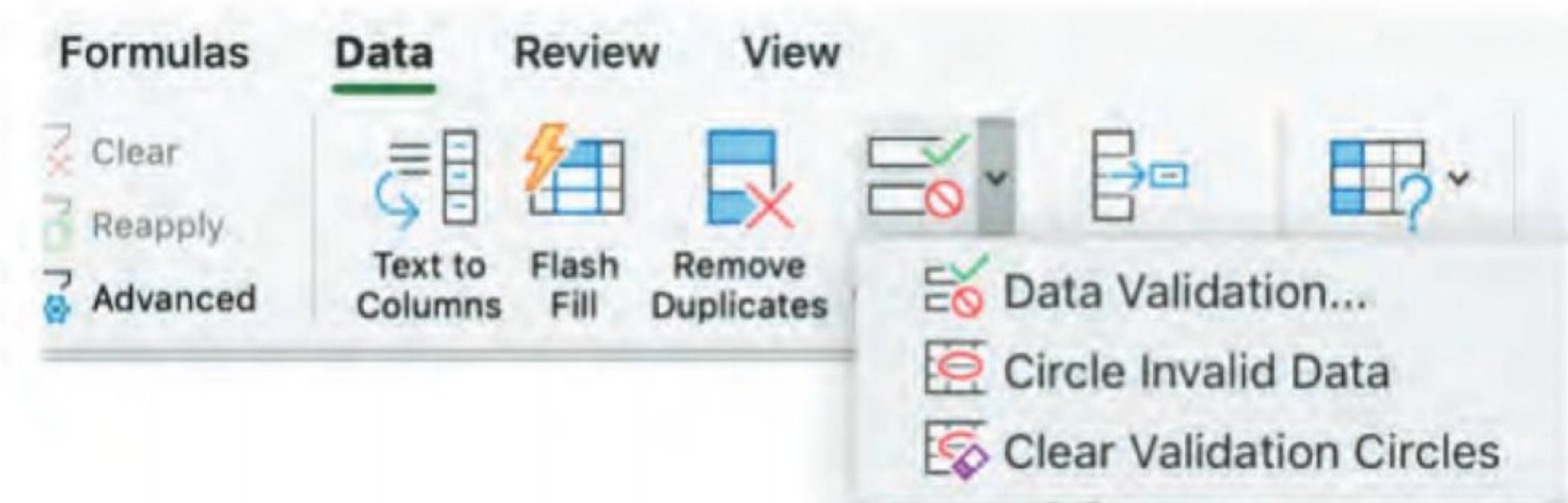


FIGURE 2.54 In Excel, choose Data > Data Validation. You are then prompted to insert a restriction, which will be applied to any attempt to place data in that cell.

Web search 'Excel data forms'. The most reliable advice will begin with URL 'support.office.com'. However, there many other tutorials and comments that you may find useful.

Excel for Windows has additional functionality that includes ActiveX and data entry forms. It should be noted that a spreadsheet is *not* a database. A table of records, where each row is a record and each column is a variable or category, may be sufficient for many purposes when there are fewer than a million records. As well, Visual Basic is a scripting language that allows developers to further enhance spreadsheets.

Databases

There are several database software applications relevant to VCE Applied Computing, with others available at a significant cost.

Free database applications include:

- LibreOffice BASE
- mySQL (part of WAMPP/XAMPP install, which also includes PHP and WordPress servers)
- Python-mySQL (with prior experience)
- Python-R (very advanced)

Database applications for purchase include:

- Microsoft Access
- FileMaker
- Wolfram Language – Mathematica.

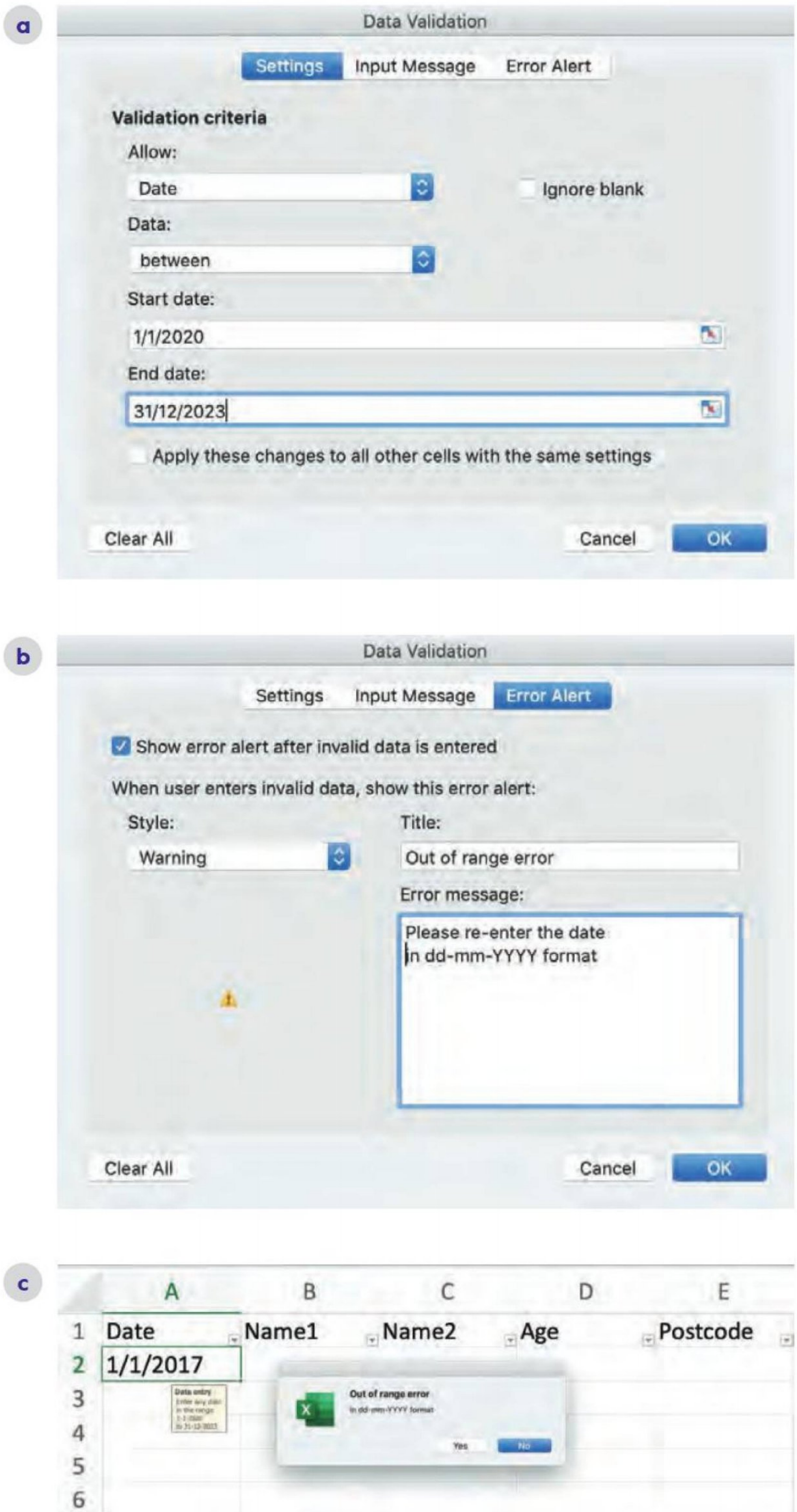


FIGURE 2.55 Data validation settings and error warning message generated when data is outside specified range. (a) Validation settings. (b) Warning message. (c) Message appears when invalid data is entered into protected cell.

2.9

THINK ABOUT APPLIED COMPUTING

Steps

1. Open a new Excel file.
2. Choose a small range of cells.
3. Apply data validation rules to those cells for date, time, number, etc.
4. Test that your data entry conditions work as expected.

Questions

1. What happens when you applied validation rules to existing data already entered into cells?
2. How may this assist your data manipulation and preparation for data visualisation charts?

9780170440806

The following examples will be in FileMaker, but the ideas apply equally to any database.

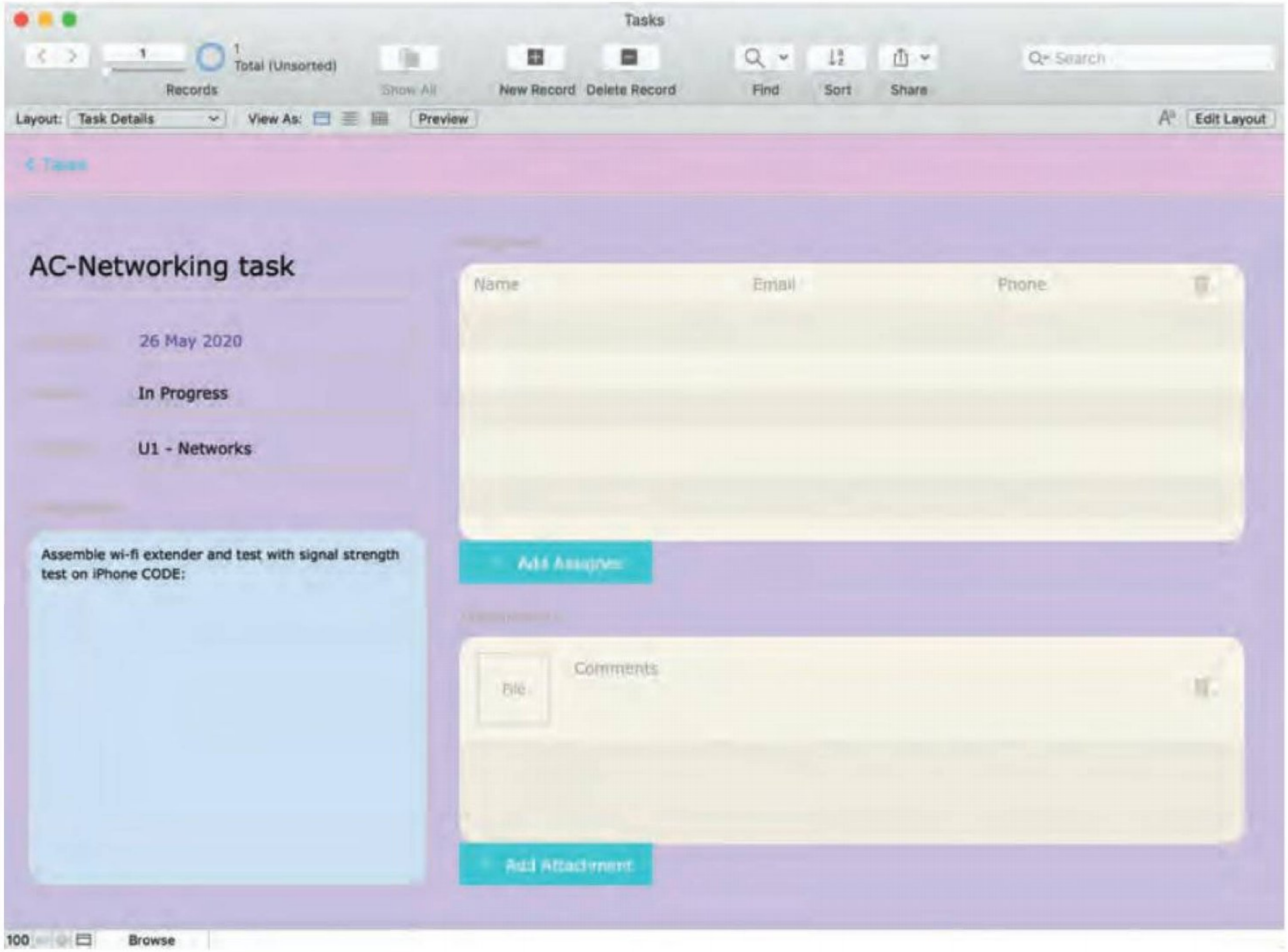


FIGURE 2.56 Database entry screen: validation for date, drop-down menu for category, open field for comment

Once the data has been successfully entered, reports can be requested from the data set. The simplest report is to list all records and all fields. The strength of the database is the ability to filter on one field and assemble a report using a selection of the other fields.

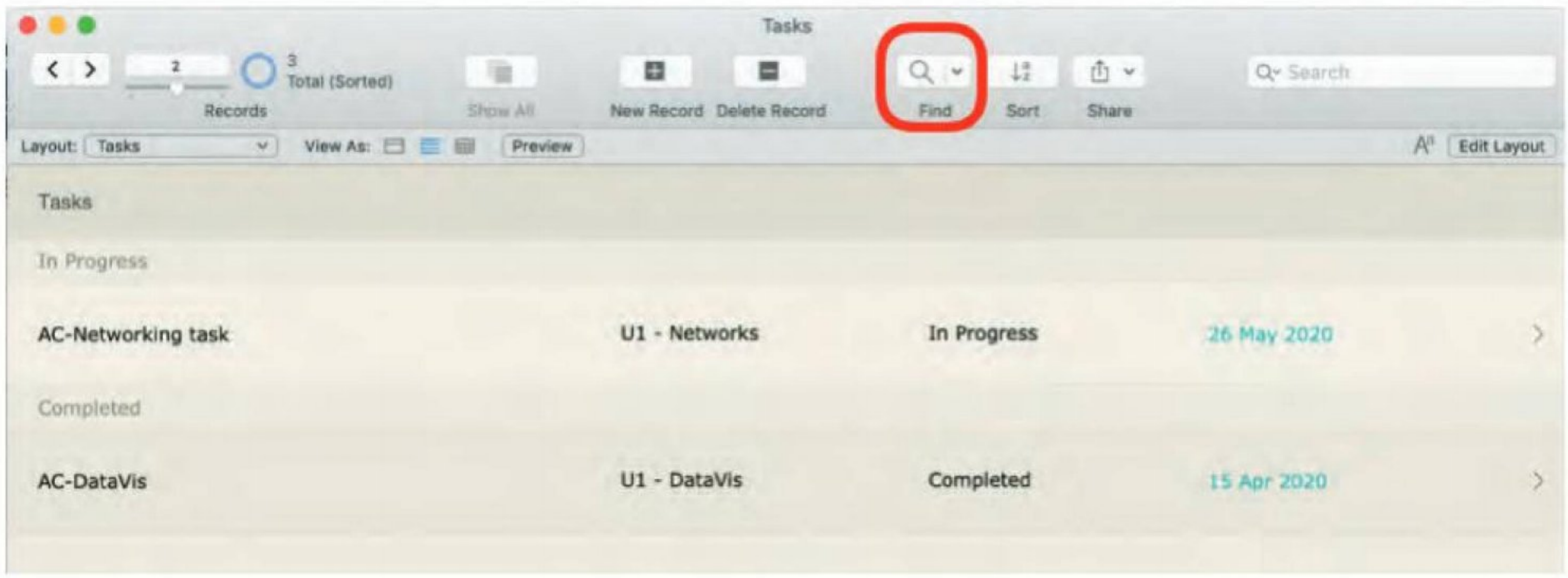


FIGURE 2.57 List view indicates fields that could be used by a FIND filter.

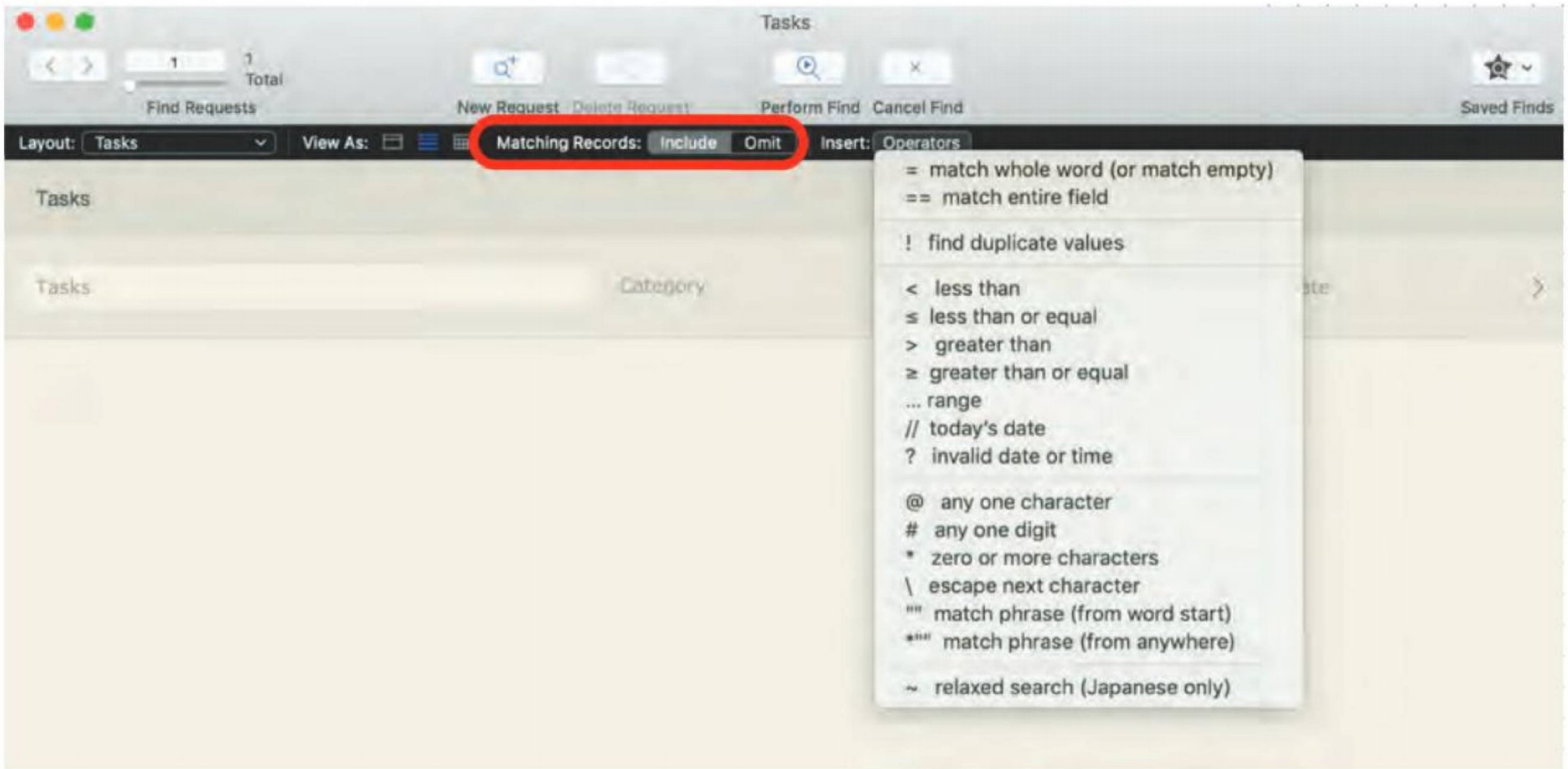


FIGURE 2.58 The FileMaker FIND filter chooses whether to include or omit the matches in the view/report.

Data visualisation

There are several options for data visualisation software. Excel, Tableau, Google Studio and Plot.ly all provide adequate charting opportunities to develop data visualisations and reporting dashboards.

Validation in this context is being done after the data has been recorded, captured and stored as a table of records or in database records. For visualisation software to deal with data that is unacceptable for display due to being out of range, and empty (or null) field, the wrong data type or in a different or incompatible unit, a filter will be applied.

A filter will cause empty or null fields to not be represented, rather than be plotted as a zero value. It is not acceptable to assign a zero value. Zero is still a value, when the data had no value. These null values must be found, rather than processed and permitted to create phantom data points, which appear to be valid.

Often the only way to discover such anomalous behaviour is to perform some trials to observe the patterns. The data sets are frequently so large that visual inspection is not possible or productive. One technique to locate anomalous data is to conduct a histogram sort and observe the distribution of data values. This may identify unusual outlier values, which may not be empty (null) or zero and just data entry errors.

When should you choose a spreadsheet and when a database?

The term 'big data' initially referred to data sets that could not fit into the RAM on a local computer. In our case, Excel is restricted by available RAM to 1 048 576 (rows) × 16 384 (columns = XFD) or about 10 GB file size.

With the development of online data streaming, 'big data' can also be described by the three 'V factors':

- volume – large amounts of unstructured data that could have a file size of tens of terabytes or hundreds of petabytes
- velocity – the rate at which the data is received. Some applications are processing data in near real time
- variety – unstructured data consisting of a variety of data types, including text, images, audio and video, which require processing to derive meaning from the metadata.

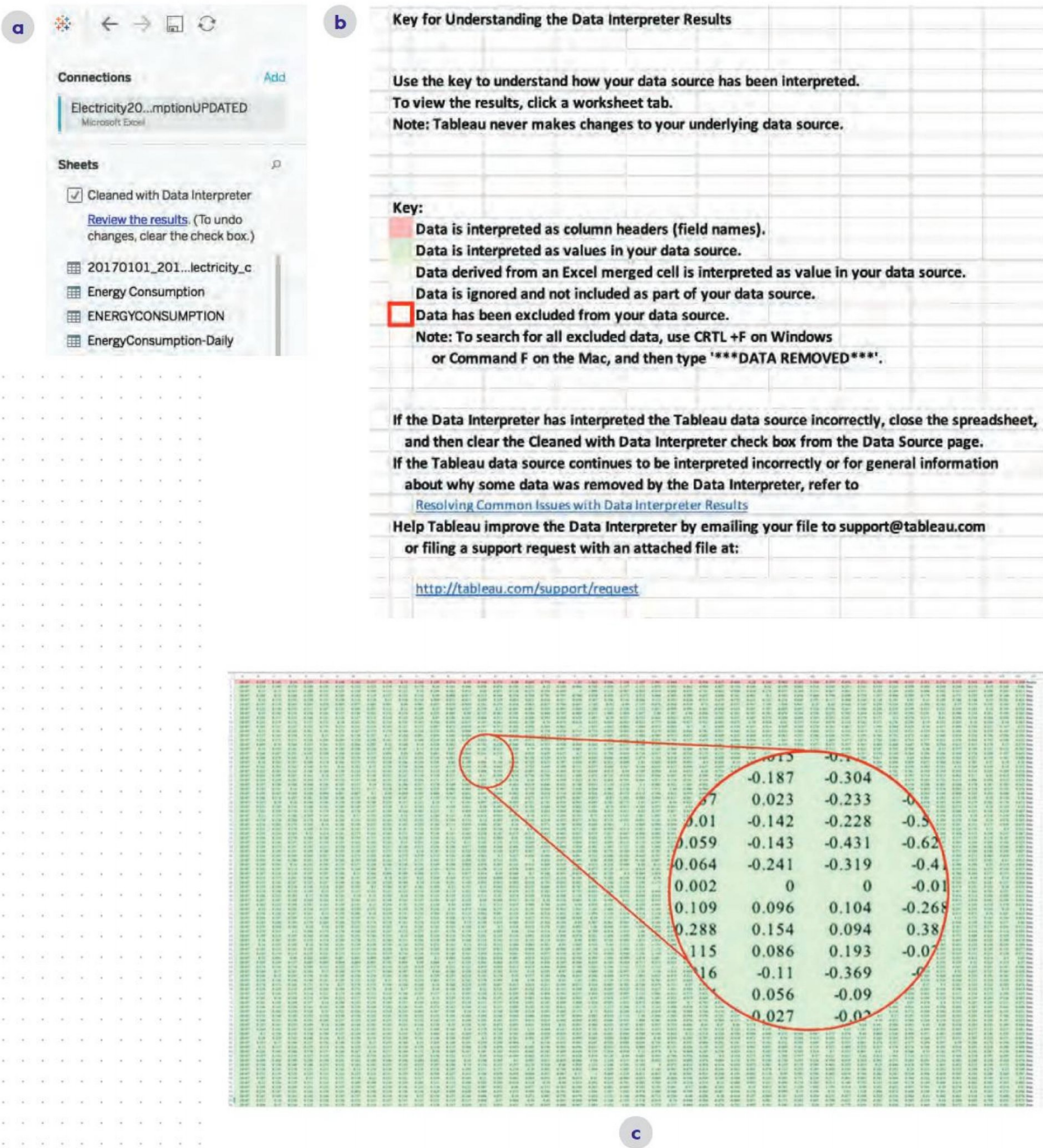
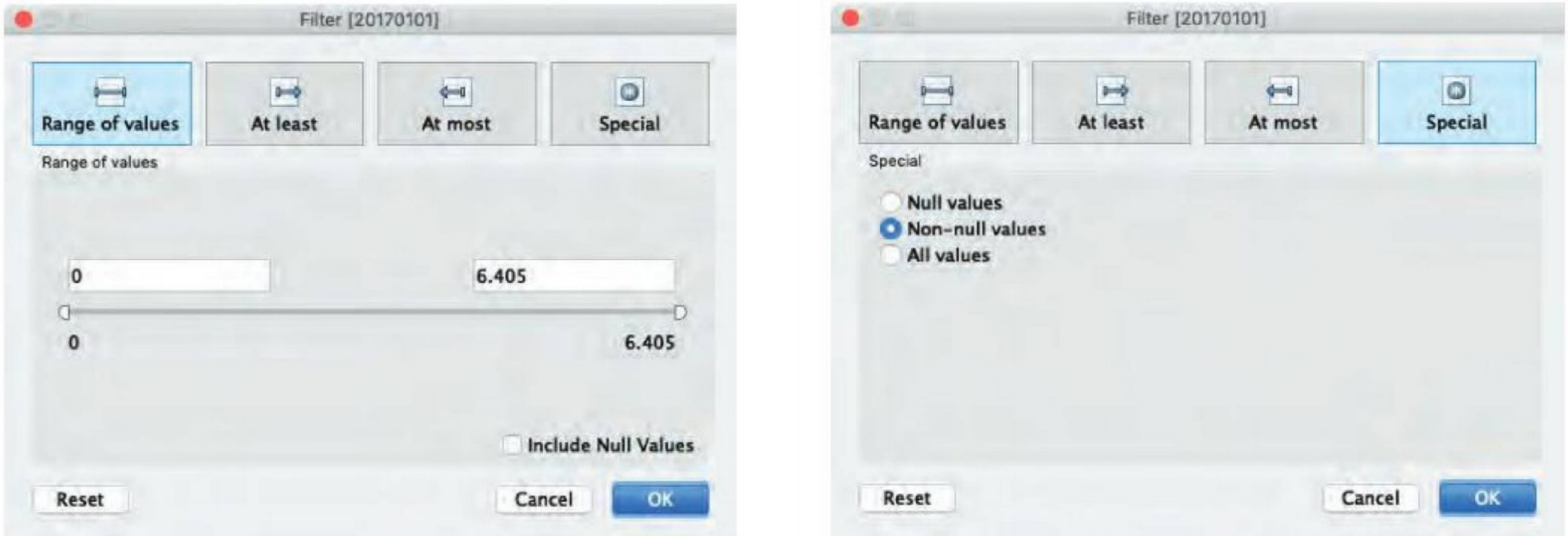


FIGURE 2.59 Tableau has a data interpreter function that removes empty/null values and 'cleans' the data in preparation for visualisation.

A further filter function allows any field to be inspected for both range and null values.



a b

FIGURE 2.60 Tableau data filter allows a nominated field to be checked for within range compliance and other value outlier behaviour.

Interactive charts

Excel and Tableau have controls that can alter settings so the chart display changes on the one screen. Data and calculations reside on separate sheets. Control settings allow choice of variables to show a chosen slice of the data set.

Excel has **Form Controls** that can be inserted into a sheet and will provide user choice of the chart and data displayed.

a

b

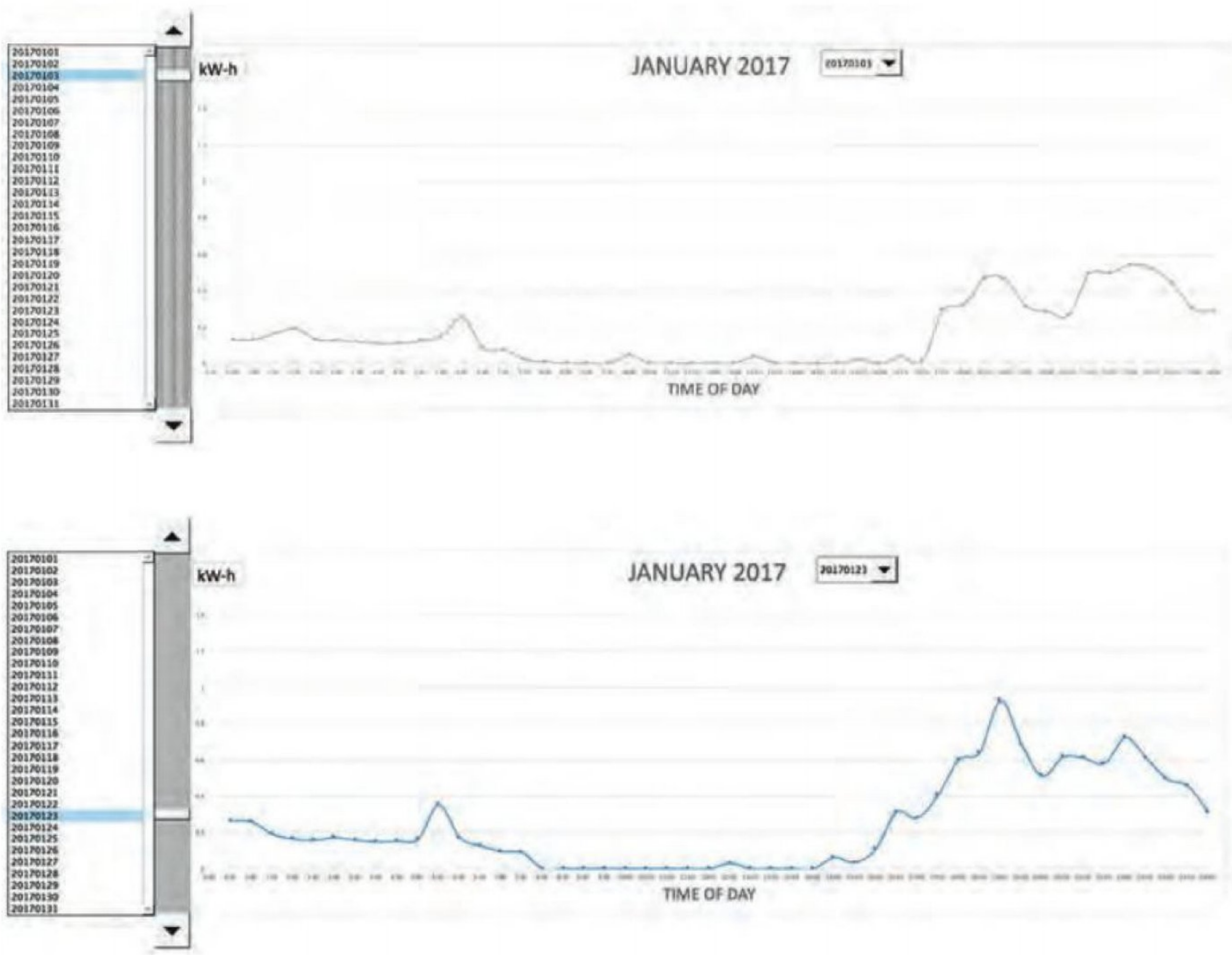


FIGURE 2.61 Excel charts controlled by Combo Box, Scroll Bar (slider) and List Box (drop-down menu)

There are several choices for Form Controls.

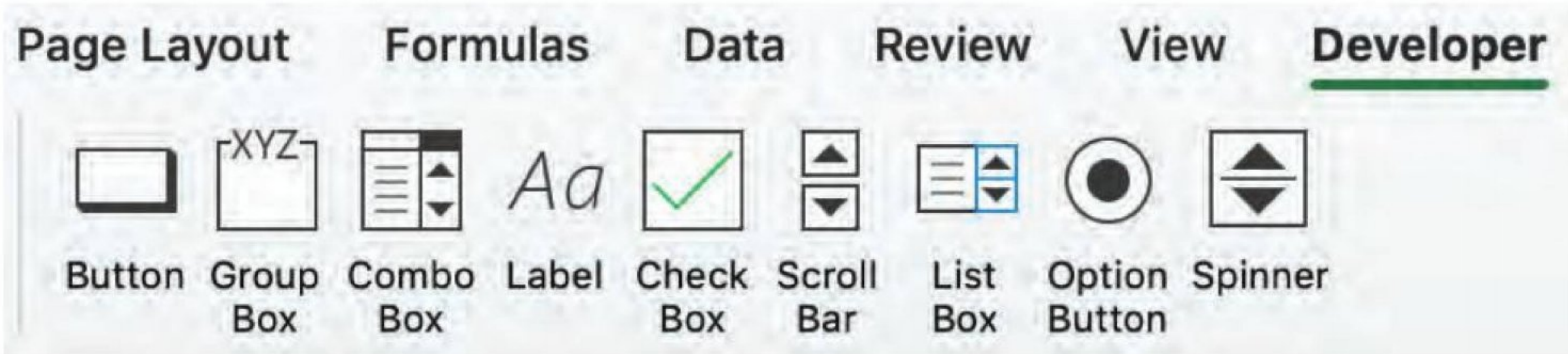


FIGURE 2.62 Excel has several Developer Tools Control Forms, which provide user control of the chart or sheet.

A clear benefit of using Form Controls to validate data entry is the reduction in typing errors. However, wrong choices from the displayed menu items may still occur.

CASE STUDY

Energy Superpowers

In 2018 Monash Tech School introduced a Year 8 Energy Superpowers program. This interactive program explores six energy usage profiles for the similarities and differences on electrical energy consumption and costs.

The entire program is written in the Wolfram Language using Mathematica as the application. The source data is held in a single Excel workbook with six worksheets with identical layout.

Privacy considerations

The data was provided to consumers at no cost by their energy providers. Monash Tech School acquired the data on the strict undertaking that the personal details should not be revealed except in the broadest sense to assist with matching energy use with user profile behaviour.

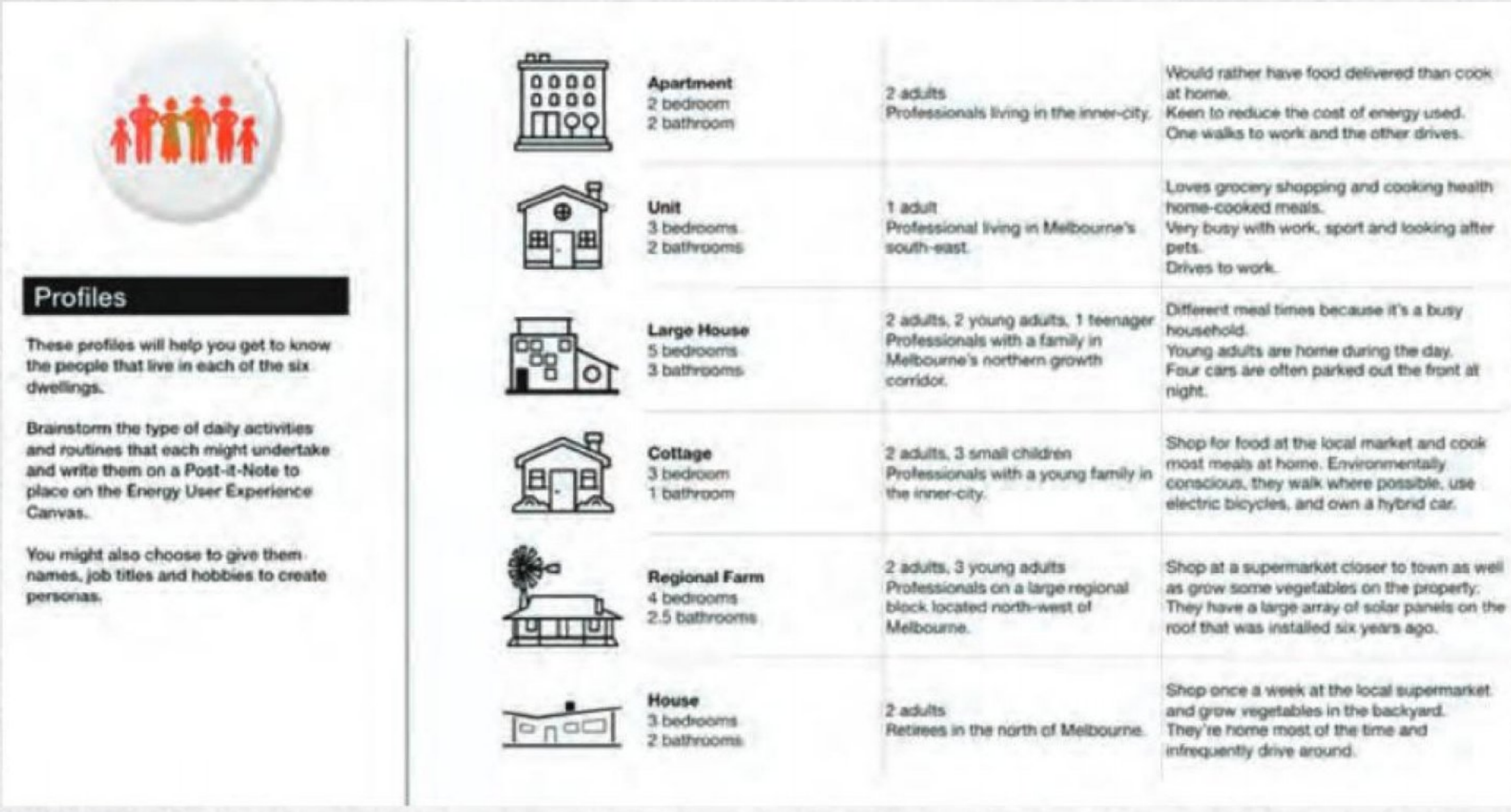
With thanks to Neil Carmona-Vickery, Director of Monash Tech School

a

Date	Time	user1	user2	user2	user4	user5	user6
Sun 1 Jan 2017	1	0.252	0.181	0.	0.068	0.356	0.328
Sun 1 Jan 2017	2	0.195	0.194	0.	0.094	0.055	0.087
Sun 1 Jan 2017	3	0.111	0.131	0.	0.048	0.048	0.096
Sun 1 Jan 2017	4	0.099	0.094	0.	0.112	0.045	0.054
Sun 1 Jan 2017	5	0.123	0.087	0.	0.051	0.044	0.09
Sun 1 Jan 2017	6	0.112	0.081	0.	0.084	0.048	0.05
Sun 1 Jan 2017	7	0.094	0.031	0.	0.077	0.037	0.086
Sun 1 Jan 2017	8	0.092	0.075	0.	0.046	0.047	0.048
Sun 1 Jan 2017	9	0.098	0.037	0.	0.106	0.035	0.082
Sun 1 Jan 2017	10	0.112	0.056	0.	0.046	0.046	0.051
Sun 1 Jan 2017	11	0.098	0.063	0.	0.103	0.033	0.077
Sun 1 Jan 2017	12	0.127	0.031	0.	0.146	0.046	0.052
Sun 1 Jan 2017	13	0.127	0.069	0.001	0.112	0.023	0.108
Sun 1 Jan 2017	14	0.141	0.05	0.062	0.084	0.153	0.285
Sun 1 Jan 2017	15	0.08	0.037	0.127	0.052	0.028	0.165
Sun 1 Jan 2017	16	0.08	0.069	0.232	0.095	0.02	0.141
Sun 1 Jan 2017	17	0.08	0.037	0.291	0.047	0.009	0.144
Sun 1 Jan 2017	18	0.213	0.044	0.709	0.111	0.	0.159
Sun 1 Jan 2017	19	0.143	0.069	1.118	0.508	0.	0.143
Sun 1 Jan 2017	20	0.133	0.037	1.301	0.118	0.17	0.153

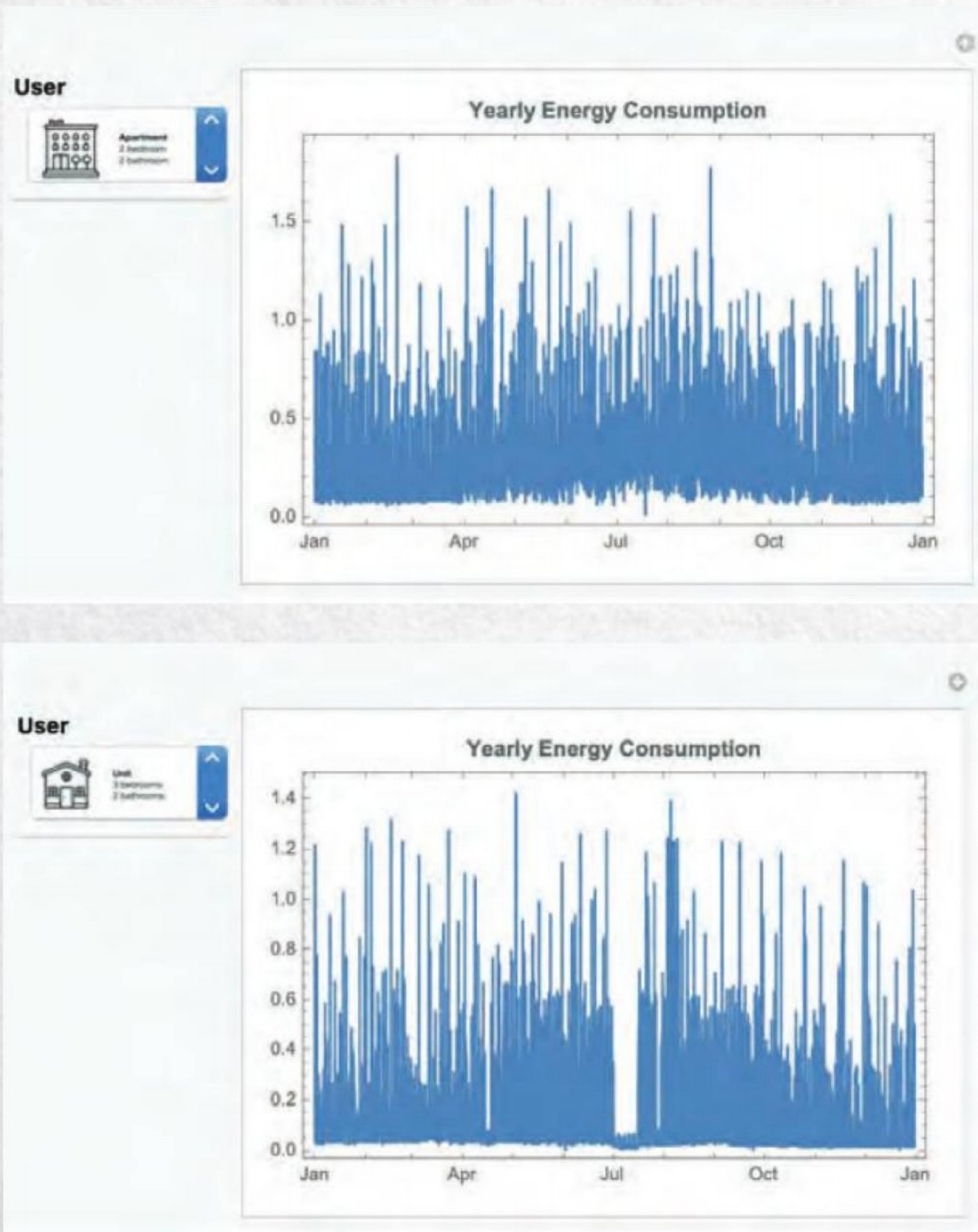
b

FIGURE 2.63 Data processed by the program has more than 105 000 data points.



With thanks to Neil Carmona-Vickery, Director of Monash Tech School

FIGURE 2.64 Six user profiles are provided with an indication of basic details.



With thanks to Neil Carmona-Vickery, Director of Monash Tech School

FIGURE 2.65 User energy consumption shows certain user behaviour and holiday periods.

When plotted in 3D, daily and monthly patterns across the year can be observed. The object can be rotated to allow identification of key points and daily, weekly and monthly activities.

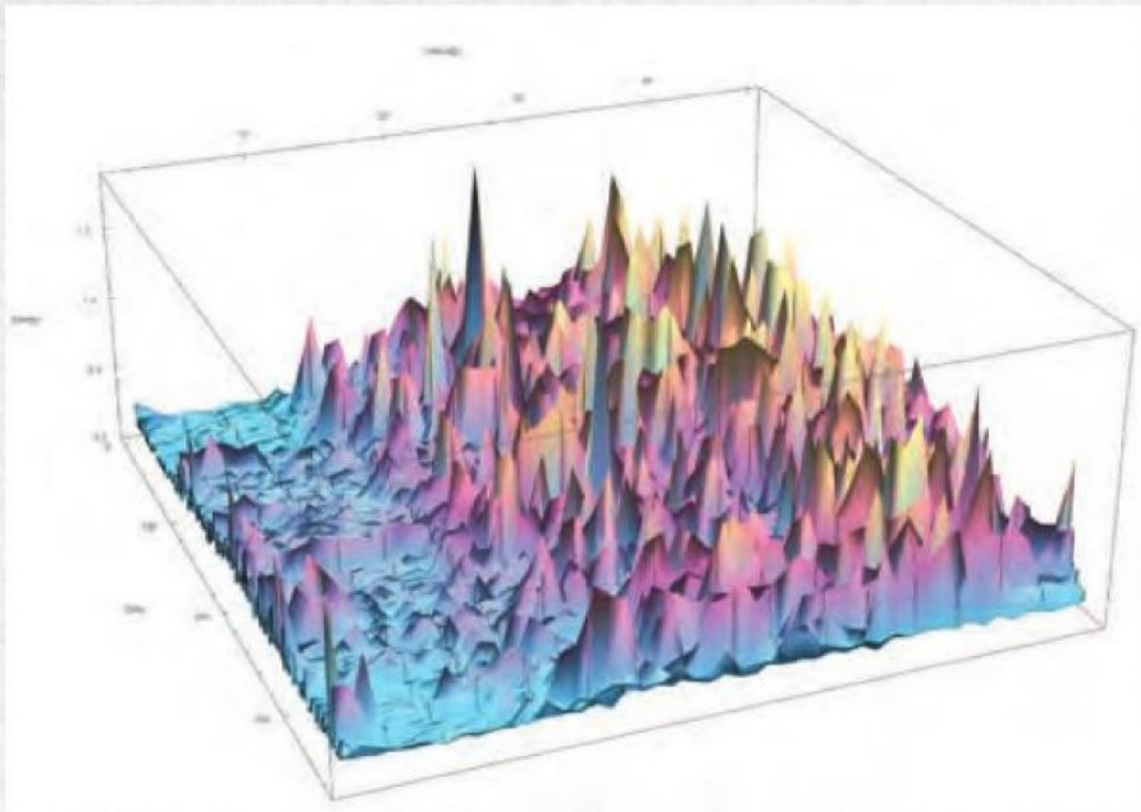


FIGURE 2.66 User 1, yearly energy consumption for each interval of 30 minutes for each day (17 520 data points)

So far, each chart is displaying raw data arranged by different time intervals. 17 520 data points are too many to consider individually. It is much better to consider summary data.

Monthly billing is a usual summary value of what retailers charge for their services. Consumers have the opportunity to modify their energy usage if they identify daily patterns of behaviour that consume large amounts of energy.

Manipulate the data

The energy supplier provides the half-hourly electricity usage readings in an Excel .CSV format file. The data is arranged with the first column having an 8-digit date in ISO8601 format followed by 48 columns for 30-minute intervals each 24-hour day. The file has two years of daily data, so there will be 730 or 731 rows. Manipulation requires each cell to be validated as holding numeric data from zero value to the daily maximum. The daily maximum could by determined by inspection of a scatter plot of all data or by using Excel functions MAX and MIN to determine the range of data. Other useful functions for indicating the behaviour of the data are: MEDIAN and AVERAGE.

723	20181224	0.133	0.171	0.164	0.161	0.174	0.151	0.145	0.15	0.127
724	20181225	0.178	0.182	0.18	0.167	0.163	0.158	0.142	0.142	0.13
725	20181226	0.127	0.122	0.134	0.137	0.141	0.147	0.136	0.12	0.12
726	20181227	0.351	0.281	0.208	0.17	0.165	0.16	0.154	0.147	0.152
727	20181228	0.197	0.178	0.163	0.149	0.128	0.123	0.138	0.13	0.122
728	20181229	0.181	0.153	0.131	0.13	0.162	0.124	0.117	0.125	0.121
729	20181230	0.179	0.157	0.272	0.19	0.161	0.161	0.134	0.123	0.138
730	20181231	0.148	0.156	0.178	0.151	0.135	0.13	0.135	0.13	0.123
731	max	1.020	0.976	0.923	1.191	0.716	0.769	0.674	0.639	0.692
732	min	0.085	0.083	0.074	0.074	0.069	0.070	0.073	0.066	0.066
733	average	0.258	0.224	0.201	0.188	0.167	0.156	0.145	0.141	0.138
734	median	0.219	0.192	0.173	0.154	0.145	0.139	0.135	0.132	0.130

FIGURE 2.67 Scope the range of the data by using Excel functions MAX and MIN

Alternatively, highlight the entire range of data, including dates to get a quick indication of the extent of the data set.

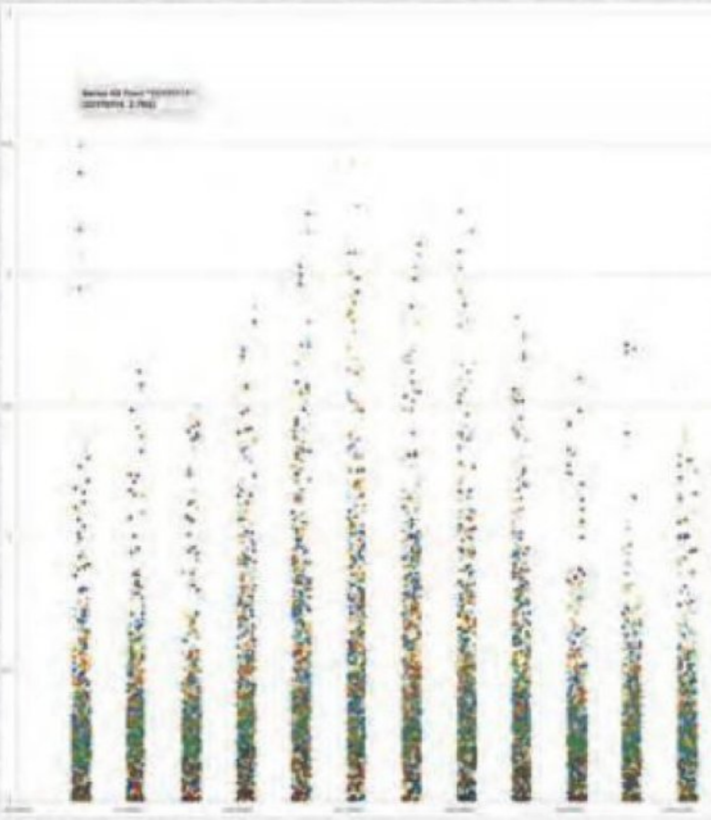


FIGURE 2.68 A quick scatter plot of all 2017 data. The highest value can be determined by inspection. The hover Tooltip reveals the greatest 30-minute energy consumption was on 20170114 in interval 40 with 2.765 – in other words, from 7.30 p.m. to 8.00 p.m. on 14 January 2017, when 2.765 kWh energy were used.

Taking a closer look

The default setting for Excel charts will plot each interval as a series, but it is the daily record that is of more interest. The table will need to be transposed for this view (Copy > PasteSpecial >Transpose). The time intervals also need to be added (Edit > Fill > Series ... > Columns > Linear > Begin0.5, StepValue0.5, StopValue24).

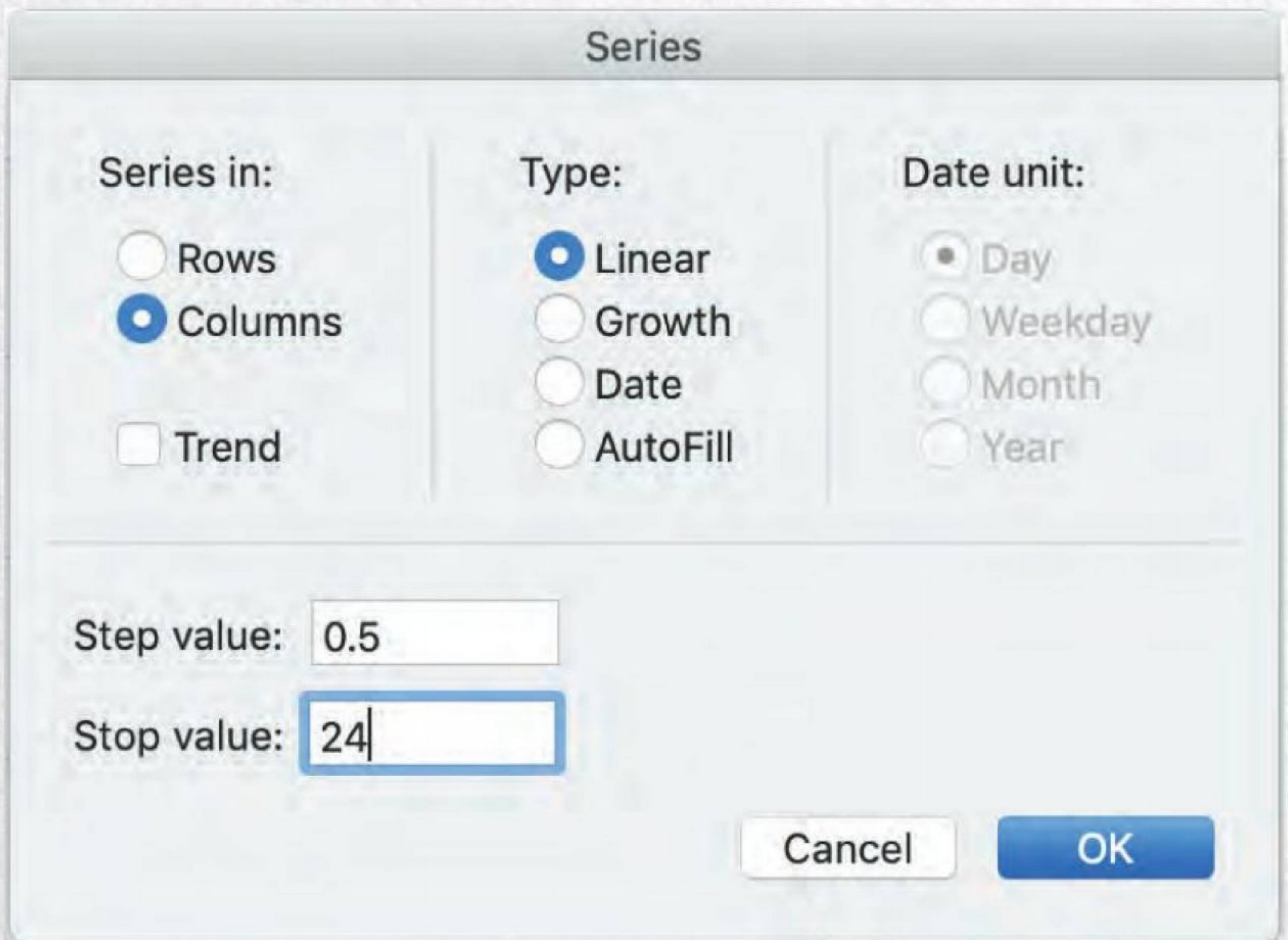


FIGURE 2.69 Insert Interval step values after transposing data.

> **2.10 THINK ABOUT APPLIED COMPUTING**
>
> Consider the 2017 energy usage scatter plot in Figure 2.68. When is the greatest energy consumption? There are 12 groups of data points, one for each month. Can you estimate how the energy might have been used? Why are the January figures unusual or unexpected?

> **2.11 THINK ABOUT APPLIED COMPUTING**
>
> Download the 2018 Energy Consumption file from NelsonNet, or source your own electricity usage data set.
>
> Determine:
>
> 1 The greatest 30-minute usage for the year.
> 2 The daily usage and identify when the greatest usage occurred and the value.
>
> There is a limit imposed by Excel on the number of series that can be displayed (255). We have 265. We will choose to consider the daily consumption for each month.

Ad 2018 Energy Consumption

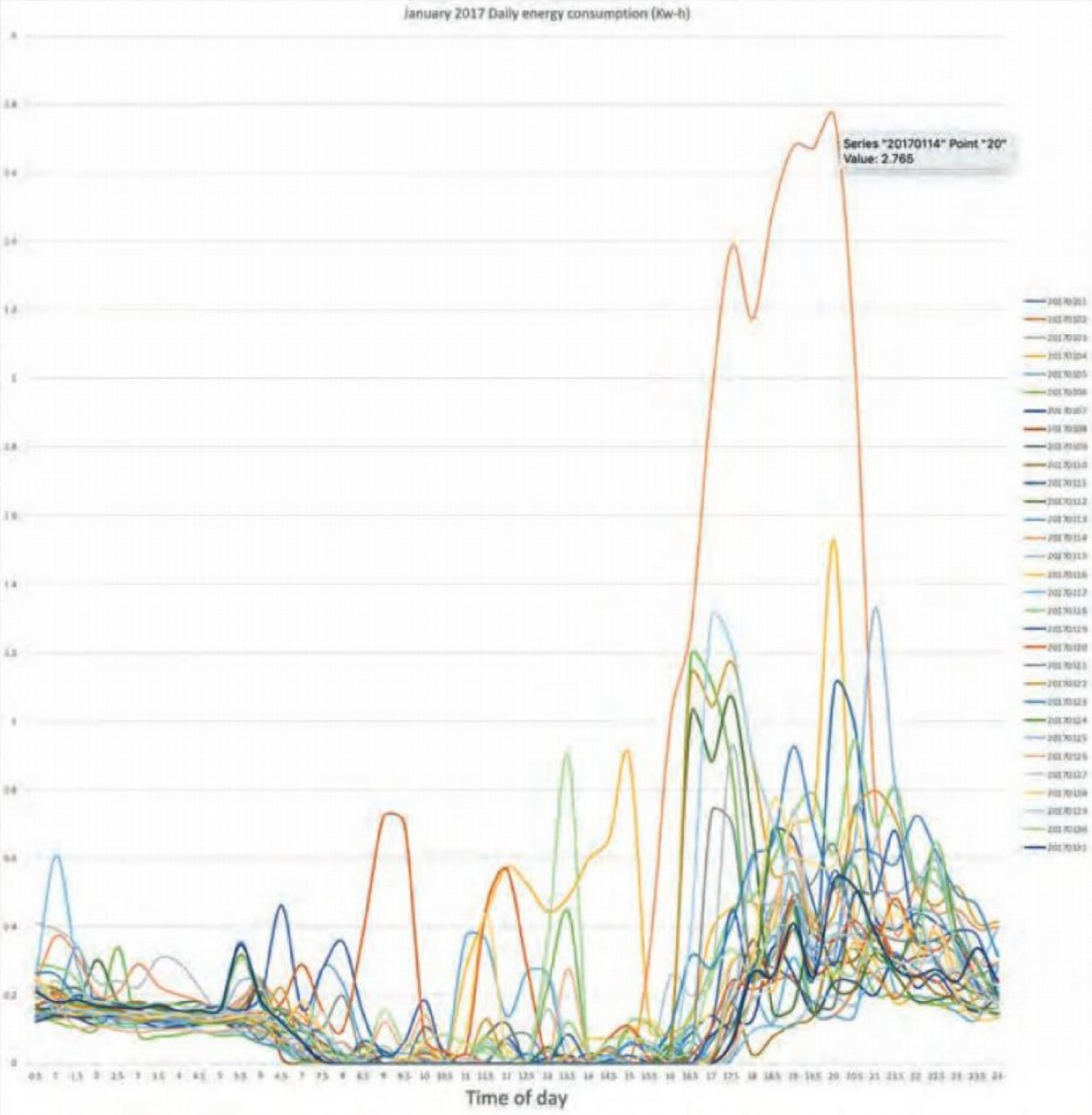


FIGURE 2.70 Energy usage for January 2017 indicates greatest consumption of 2.765 kWh was recorded on 20170114 at 20:00 (8 p.m.) for the preceding 30-minute interval.

Keep it simple

Just because you can graph 17 520 data points, should you? How can the 'story' of the data consumption be explained in a single simple chart? Consumers are probably not so worried about the time of day consumption; rather, the total cost each for month will help them to decide whether they need to change their usage behaviours.

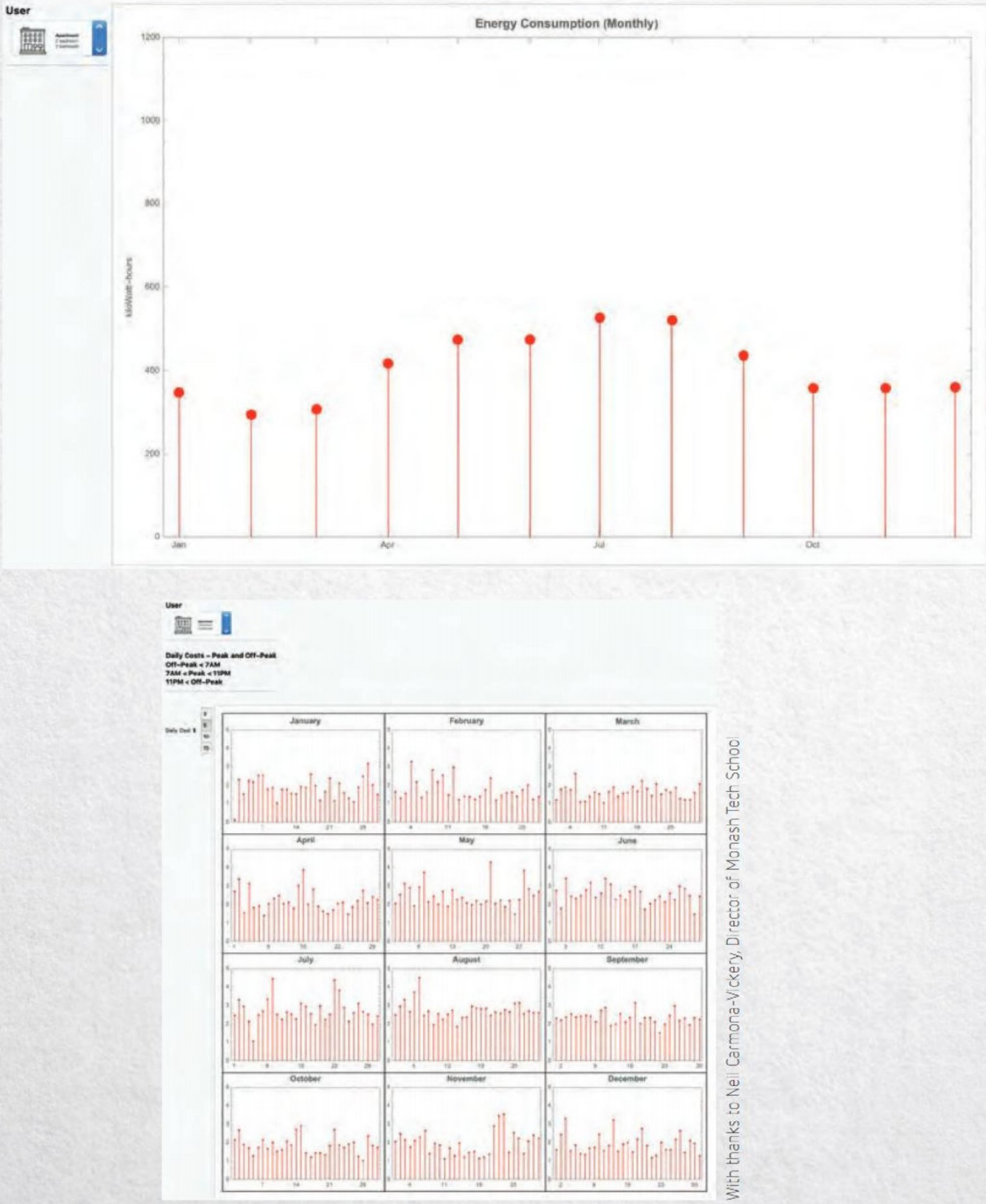
The steps for development of the visualisation will be first to calculate daily totals, then monthly. Bear in mind that the way retailers charge for electricity will depend on time of day, or peak and off-peak time zones. One common peak time zone is 3 p.m. until 9 p.m. inclusive, so after 3 p.m. is the 3.30 p.m. (or 15.5) recording until 9.00 p.m. or 2100 hours.

The data will be required to be processed into peak and off-peak costs. One way to do this is to create a new sheet that refers to the energy used and is multiplied by the corresponding tariff. A quick method of doing this is to create a duplicate worksheet, then to insert the calculated cells in the second sheet with relative references. These can then be copied down and then copied across to calculate daily energy costs. Break down the steps so that you first calculate interval (peak or off-peak), then daily total, then monthly total, and then annual energy cost. The main data of interest will be the twelve-monthly totals; however, an interactive chart that allows the user to choose the month to inspect the daily costs might also be worth considering.

Excel has a way of quickly summarising a collection of data by using a **pivot table**.

Manipulating data with Mathematica

The Monash Tech School Superpowers Program employed Wolfram Mathematica to manipulate the data sets. A function was applied to a 'list of lists' to convert energy usage to costs. Daily totals were then calculated. Interactive controls allow users to choose which consumer profile is shown. Each user has a distinctive colour assigned to assist with identification and consistency.



With thanks to Neil Carmona-Vickery, Director of Monash Tech School

FIGURE 2.71 Year with monthly consumption and Month with daily energy use for the apartment profile (in red)

Further development allowed application of different rates. The 'standing rate' is the default rate that retailers place consumers on if they do not opt into a peak/off-peak arrangement. The difference in costs is shown in a chart and in a table (Figures 2.72 and 2.73).

With thanks to Neil Carmona-Vickery, Director of Monash Tech School

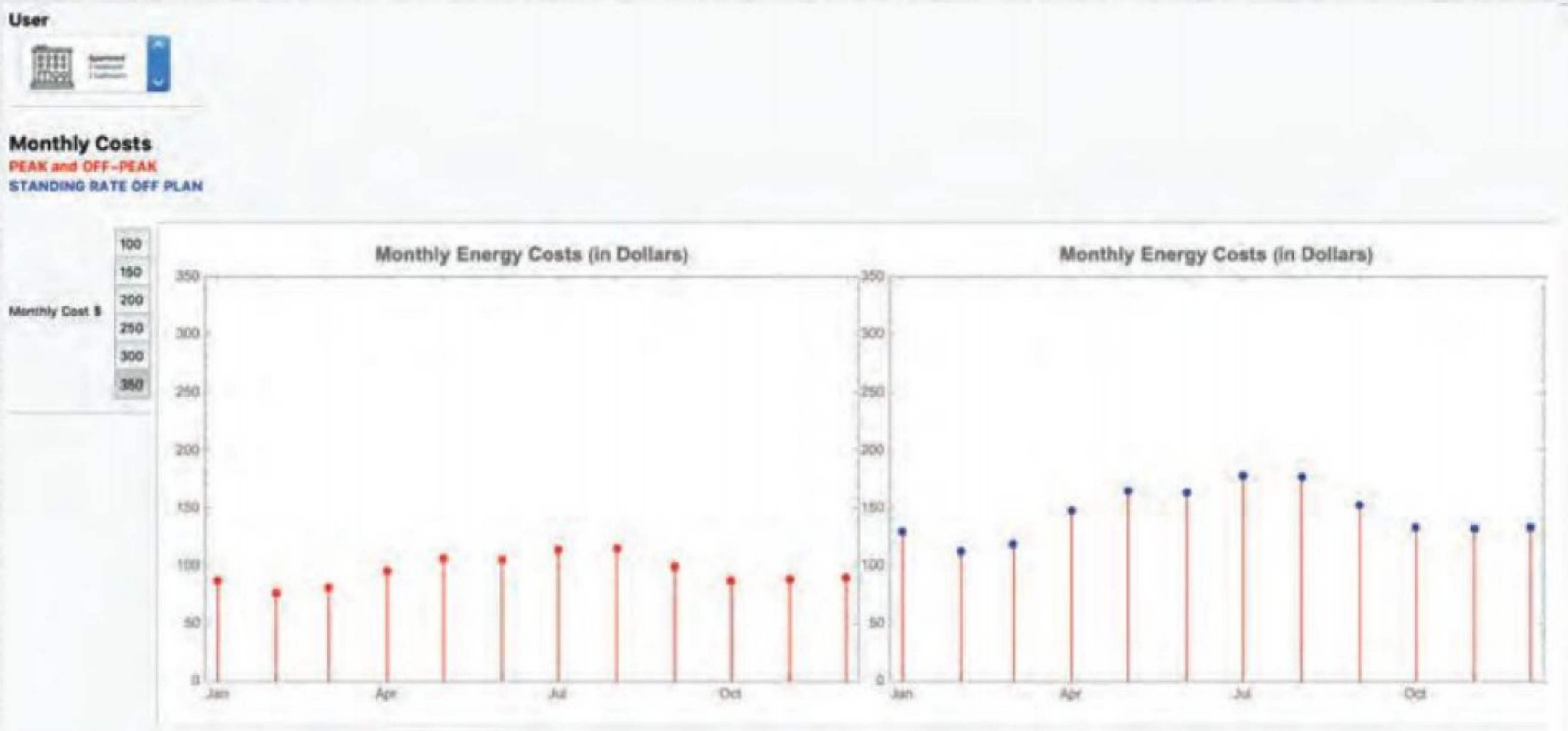
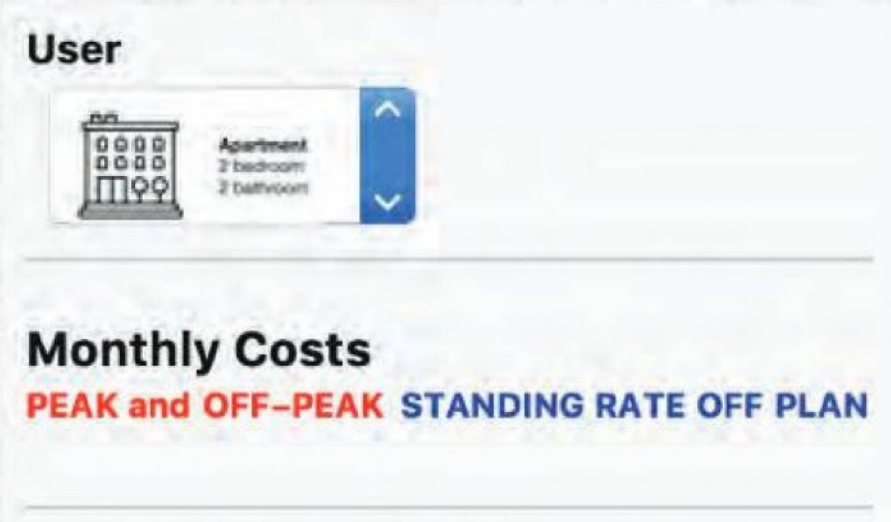


FIGURE 2.72 Comparison between peak/off-peak and the standing rate. Clearly there is a difference in the amount charged for the exact same consumption.



jan	87.48	jan	130.48
feb	77.07	feb	112.79
mar	81.14	mar	119.56
apr	96.37	apr	148.47
may	107.51	may	165.24
jun	105.69	jun	164.07
jul	114.64	jul	179.32
aug	115.38	aug	177.87
sep	99.36	sep	153.78
oct	87.93	oct	133.69
nov	88.81	nov	132.49
dec	89.97	dec	134.25
	1151.37		1752.04

FIGURE 2.73 A simple table of values, with an annual total, illustrates the differences as clearly as a chart.

Interactive Excel charts and dashboards

The interactive features of Tableau have been mentioned previously and are readily apparent in a Tableau workbook. Excel interactivity is not so obvious, and requires some preparation. The first step is to enable Developer Tools under Ribbon Options or Preferences. (Windows and Mac have slightly different processes.) Once enabled, the Developer tab will appear in the Ribbon above the Formula bar.

Several plots onto the same axes, one at a time

Open the worksheet with the intended data to be charted. Plot all the data onto the same axes, and adjust the 'look' according to previously discussed design principles for charts.

Ad 2017 energy consumption

FIGURE 2.74 January energy consumption with 31 days on the same axes

The task now is to plot only the selected day or day(s), also known as 'toggling' the chart on/off. We could use a check box for multiple charts or an Option button for a single chart to display this effect.

To achieve this goal, the raw data will need to be duplicated. The check box control cell will be cell A1. The chart will be read from the duplicate, not the raw data, which is never changed to preserve the authenticity and integrity of the data.

In this example, the simpler Option button will be used to show one day at a time.

Go to Developer > Option button.

Click onto the Worksheet to create the Option button. Control-click to open the Formatting dialog box.

9780170440806

FIGURE 2.75 Set the Cell link to A1, which will indicate the state of the Option button. Each subsequent Option button will increment the value, up to 31 days in January.

A shortcut to create the Option button is to Control-drag on the previous option button; the value is automatically incremented. Proceed to create 31 Option buttons.

Now for the formula. In the first duplicate data cell, enter the following formula. IF(A1=1, enter the corresponding raw data cell, in this case B2.

The duplicate cells have been chosen to be 40 cells below the originals for convenience. Then enter comma NA()). This will read, if the Option button is 1, or 1/01/2017, then read the first raw data cell, otherwise reads the cell as NULL. In other words, the cell will not plot a point on the chart.

Copy and paste the cell down, and go along cell by cell changing the 'trigger' value to the Option button value for that day. For example, the next cell below will read, IF(A1=2,B3,NA()) and the next IF(A1=3,B4,NA()) and so on until IF(A1=31,B32,NA()).

Now click, drag and highlight B41:B72, and Copy and Paste across the remainder of the duplicate data. All rows except one will switch to #N/A. Verify the row changes to correspond with the chosen option button.

FIGURE 2.76 For display, create an empty text box to place the Option buttons. Control-drag to highlight the Option buttons you want to move.

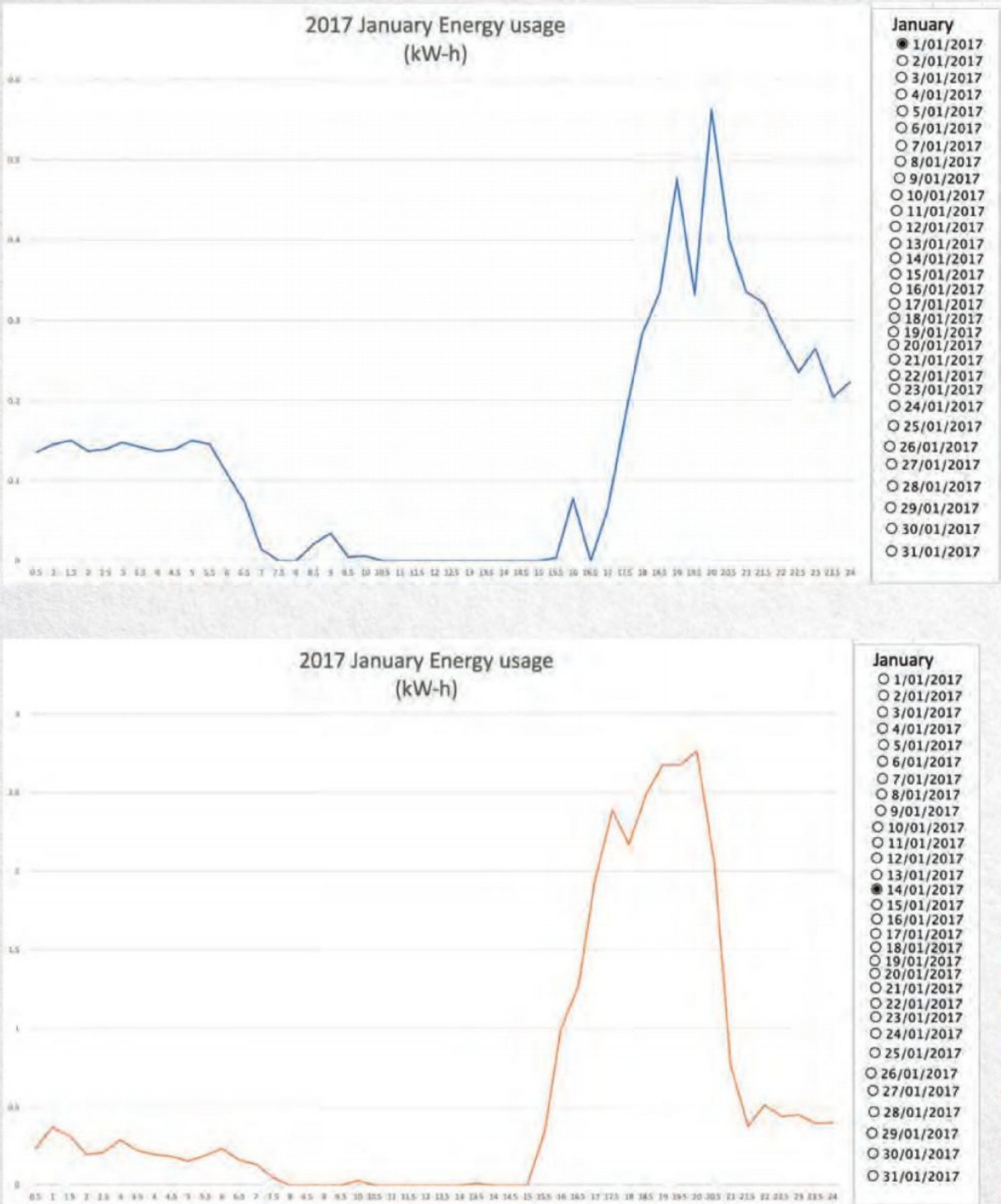


FIGURE 2.77 As the Option button is chosen, the corresponding chart is displayed.

Different charts onto the same worksheet

This is known as a dashboard. A dashboard provides a quick summation of several worksheets. The displays can update dynamically in real-time, or some other predetermined time interval. The sources of data can be locally held or streamed. Excel has several streaming functions available under Data > Workbook Connections.

Why not R and Python? A similar treatment could be attempted using Python, but the display of the processed data will need to be carefully considered as Python's charting abilities are limited. In contrast, Mathematica has extensive charting functions and comprehensive documentation with worked examples. Although the 'R' language is widely used in industry, it is considered very advanced and beyond the reach of VCE Applied Computing at this time.

The Stephanie Evergreen website provides opinion and advice on what is considered to be a 'good' data visualisation. The online tools are useful assistance when considering how to improve on a chart. Five sections are examined to determine how good the dataVis might be and where it can be improved. See the weblink.

Data Visualization Checklist

Evaluate your data visualisation chart

There were several design principles discussed earlier in this chapter. Your assessment task for Unit 1, Outcome 1 will be to create an effective data visualisation. So the question becomes, 'How do I know if my dataVis is as effective as it could be?'

Fortunately, there is an online data visualisation tool. Earlier in this chapter, we directed you to a weblink for a matrix tool (refer to page 46) where the Stephanie Evergreen website provides online advice with regard to qualitative charts. The Data Visualization Checklist is hosted on the same website. It is a compilation of 24 guidelines for formatting graphs. The checklist is a research-based collaboration by Stephanie Evergreen, Sena Sanjines, Ann Emery and Jennifer Lyons.

The process requires you to upload an image of the visualisation to be assessed. The figures below and on the following page show various scorecards for visualisations that have been tested on the website.

Created using https://datavizchecklist.stephanieevergreen.com, by Stephanie Evergreen, Sena Sanjines, Ann Emery, and Jennifer Lyons

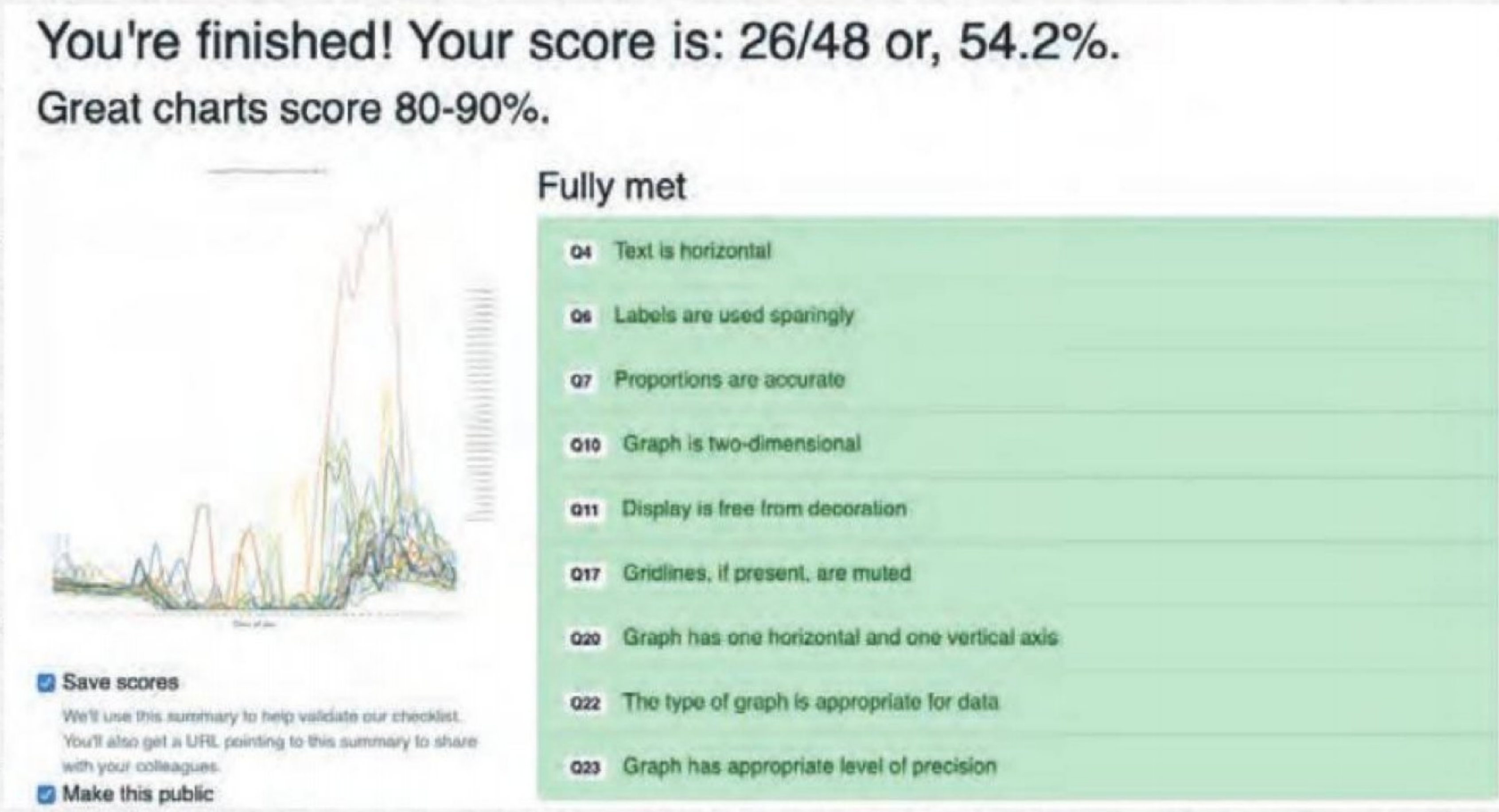


FIGURE 2.78 The chart had some highly scoring features

Created using https://datavizchecklist.stephanieevergreen.com, by Stephanie Evergreen, Sena Sanjines, Ann Emery, and Jennifer Lyons

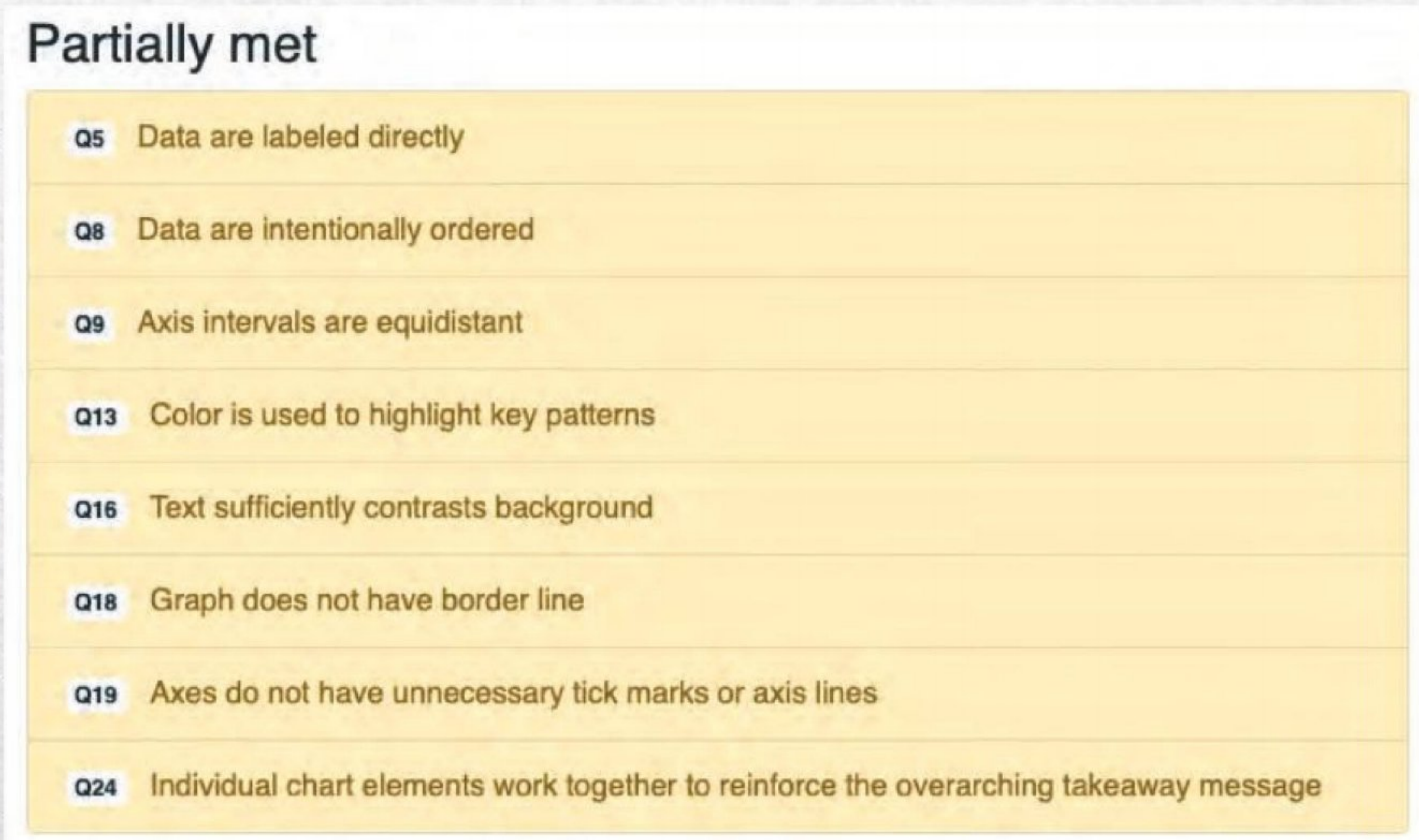
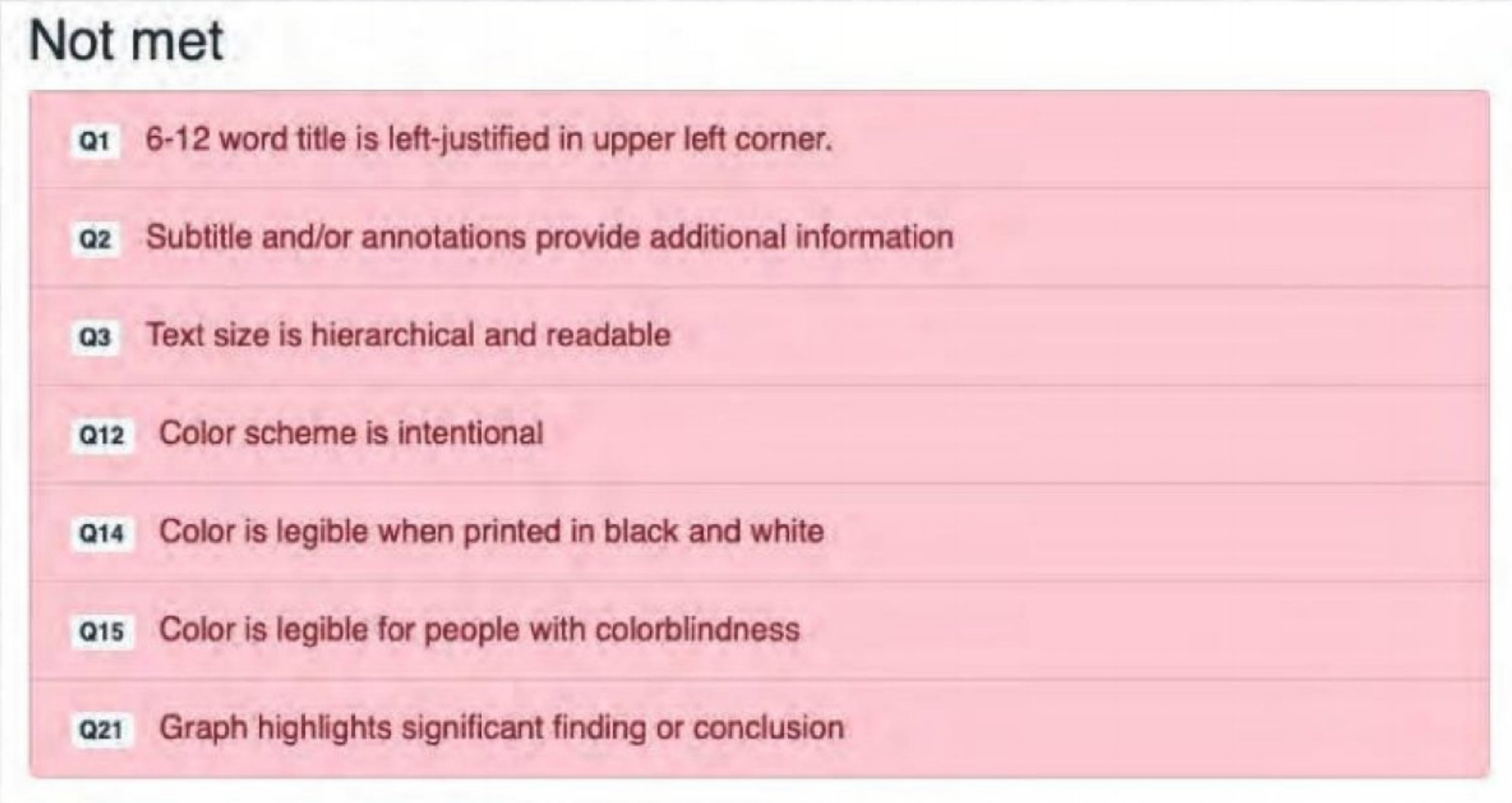


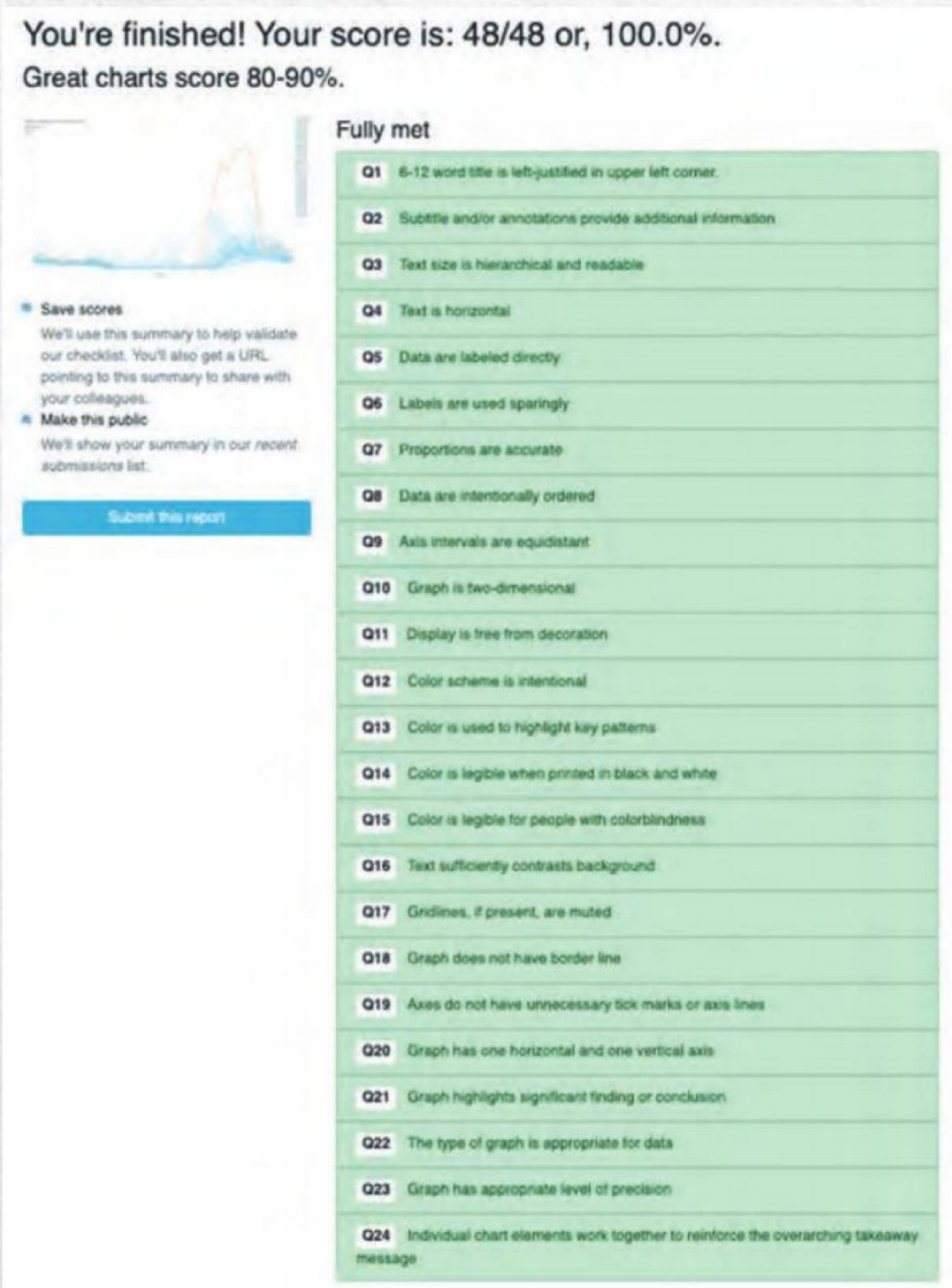
FIGURE 2.79 The chart had a few 'nearly there' moments and could be easily improved.



Created using https://datavizchecklist.stephanieevergreen.com, by Stephanie Evergreen, Sena Sanjines, Ann Emery, and Jennifer Lyons

FIGURE 2.80 There are many areas with room for improvement.

The alternate presentation was also assessed.



Created using https://datavizchecklist.stephanieevergreen.com, by Stephanie Evergreen, Sena Sanjines, Ann Emery, and Jennifer Lyons

FIGURE 2.81 A very highly rated data visualisation

> **2.12**
> **THINK ABOUT APPLIED COMPUTING**
>
> Choose one of your charts, submit a JPEG image and rate the effectiveness against the 24 criteria. Then attempt to improve your chart to make it a great graph achieving a score of over 80%.

Evaluating data visualisation presentations

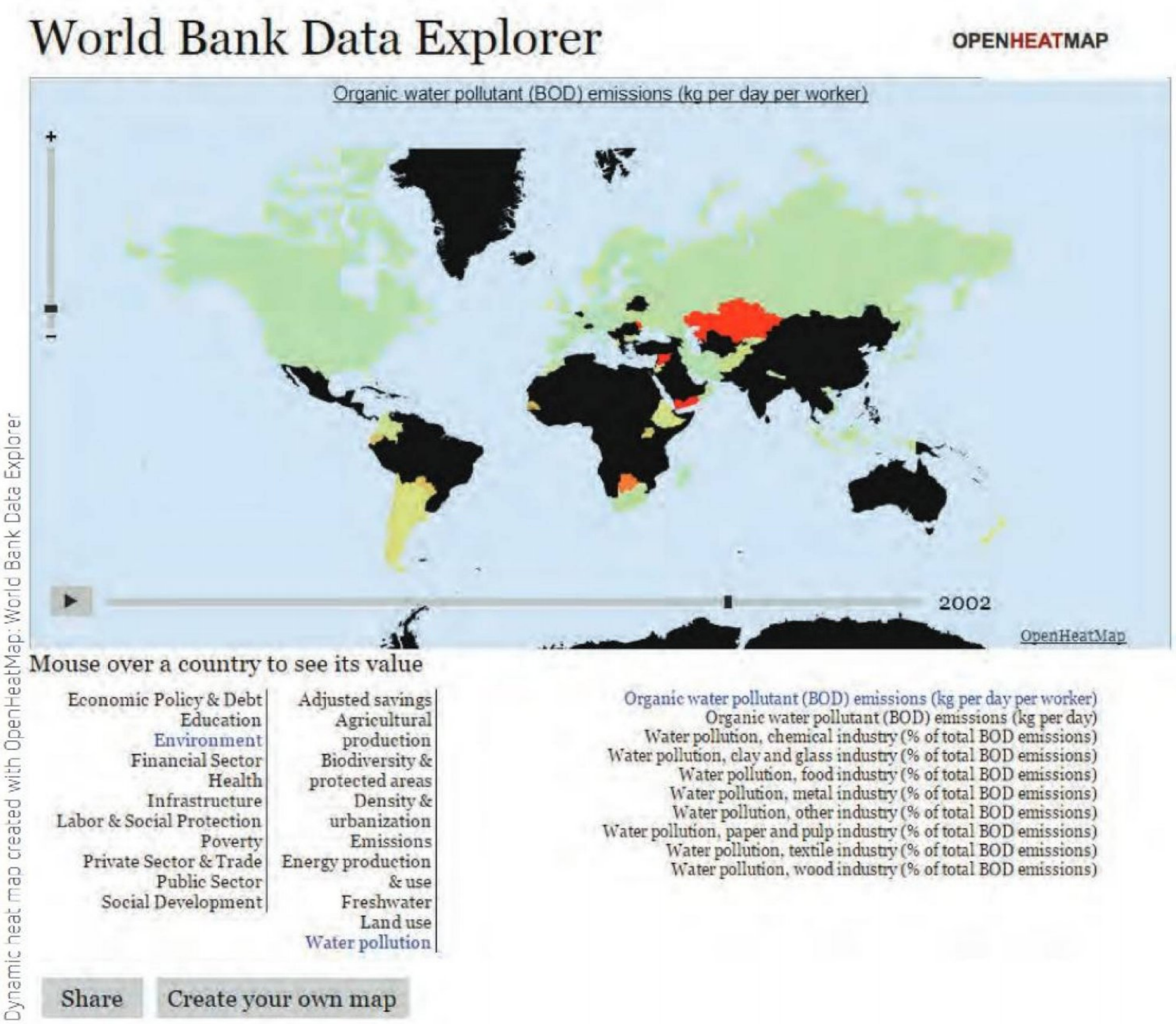
Evaluation involves measuring how well the solution meets the information need and the needs of a client.

Evaluation criteria

Evaluation criteria are measures that will be used to judge whether the solution meets the information needs of the client or user.

Criteria should relate to the initial solution requirements because if all the requirements are contained within the solution, then the information need should be met. As the criteria are going to be used to judge the solution, they should be framed or written so that they are quantifiable or measurable.

World Bank Data Explorer



Dynamic heat map created with OpenHeatMap: World Bank Data Explorer

FIGURE 2.82 Dynamic heat map created with OpenHeatMap

Quantifiable or measurable criteria are measures for which satisfaction is easily determined. An example of a measurable criterion is 'the solution displays the visualisation within five seconds of loading the website'. It is relatively easy to then judge whether the solution can do this, or not.

Evaluation criteria should also cover both effectiveness and efficiency. Effectiveness criteria should relate to how well the solution works and if it provides the information needed. There are a number of characteristics related to effectiveness including accuracy, accessibility, attractiveness, communication of message, completeness, timeliness and usability. For a solution to be considered effective, it needs to have these characteristics. Evaluation criteria for effectiveness might include:

- population sizes of towns are represented accurately
- visualisation contains appropriate contrast, space and balance
- data used is timely and no less than six months old
- data for all towns in Victoria is included
- solution is easy to use.

Efficiency criteria relate to saving time, cost or effort when retrieving the information from the solution. Examples of evaluation criteria for efficiency include:

- town information can be accessed within three seconds
- zooming in and out can be completed without using the keyboard
- solution will reduce the amount of money spent on printing the data.

Evaluation strategies

An evaluation strategy involves deciding how each evaluation criteria can be measured. It includes creating a timeline for evaluation to take place, deciding on the data required to help judge each criterion, and looking at the way the data required will be collected and how the data can be used to evaluate each measure.

One method to display an evaluation strategy, using one criterion for effectiveness and one for efficiency from the example above, is to use a table format, as shown in Table 2.7.

TABLE 2.7 The evaluation strategy

Evaluation criterion	Data used must be less than 6 months old.	Town information can be accessed within 3 seconds.
Data required	Date that the data was used to create the visualisation.	Time taken for a user to zoom in to a particular town and access the town information.
Data collection method	Interview the staff involved in collecting the primary data about population to establish the date it was collected.	Observe a user use the solution and, using a stopwatch, time how long it takes to access the information of a particular town.
How the data can be used	Compare the date that the data was collected to the current date to determine how old the data is.	Compare the average time taken, over 10 attempts, to access the information to the benchmark (3 seconds).

A range of methods can be used to collect the data required for evaluation. We have previously discussed using interviews, surveys and observations to collect data. In addition, checking download speeds, counting website hits, inspecting the solution output, reviewing error logs or timing how long it takes users to complete tasks are just some additional methods that enable evaluation criteria to be completed.

2.13 THINK ABOUT APPLIED COMPUTING

Create a strategy table by selecting two other of the sample criteria listed above. Complete the evaluation strategy for each.

Evaluation report

Finally, after the evaluation strategy has been completed, the evaluation report can be written. The evaluation report involves stating whether the solution is meeting the information need and the needs of the user. To provide evidence for the final conclusion, each evaluation criterion must be assessed to identify whether it has been achieved. If each of the evaluation criteria has been met, then the solution can be considered a success.

TABLE 2.8 Assessing each criterion to assist in preparing an evaluation report

Evaluation criterion	Data used must be less than 6 months old	Town information can be accessed within 3 seconds
Data required	Date that the data used to create the visualisation	Time taken for a user to zoom in to a particular town and access the town information
Data collection method	Interview the staff involved in collecting the primary data about population to establish the date it was collected.	Observe a user use the solution and, using a stopwatch, time how long it takes to access the information of a particular town.
How the data can be used	Compare the date that the data was collected to the current date to determine how old the data is.	Compare the average time taken, over 10 attempts, to access the information to the benchmark (3 seconds).
Data	4 months	3.6 seconds
Achieved	Yes	No

2

CHAPTER SUMMARY

Essential terms

accessibility the extent to which a solution meets users' needs in relation to additional needs, disabilities or language requirements

alignment refers to text: can be left, right, centre or full. Full alignment, which is also known as full justification, refers to text being aligned on the left margin of a column and spaced appropriately so that the last letter in a word on the same line is aligned with the right margin

annotated diagram a visual depiction of how a graphic representation should look; design indicates features such as differences in font size, colour and positioning of objects; the placement of objects must be planned so that a balanced, visually appealing and clear effect is attained

appearance a non-functional principle requirement relating to alignment, repetition, contrast and space

balance symmetrical arrangement of items left to right

camel case a typographic convention whereby each name begins with a lower-case letter and subsequent words are upper-case

chart (also known as a graph) a method of displaying data visually, where the data set is represented as symbols in the chart

constraint a restriction on what can and cannot occur in the creation of a software solution, external to the solution itself

contrast the visual difference in colour or tone between elements on a screen; there should be sufficient contrast between background and text or other page elements to make the information plainly readable

convention standard that has been developed to determine the presentation of documents and other output produced using information systems; rarely static, change over time as changes in technology and business occur; a convention is first decided on and adopted

coxcomb chart (or **rose chart**) a chart that is a variation on a radar and bar chart or pie chart

data set a collection of data with a common factor or purpose

data visualisation the presentation of data in a pictorial or graphical format

date standard a convention that states that all dates should be specified in unambiguous format: yyyy-mm-dd

dynamic data visualisation either an interactive with some user control of the view; or one with real-time or regular automatic updates of the data to provide the latest possible data available

economic constraint a constraint on the creation of a software solution that relates either to time or budget, or to both

effectiveness how well a solution meets the needs of its users, measured in terms of completeness, readability, attractiveness, clarity, functionality, accuracy, accessibility, timeliness, report formats, relevance, usability and communication of message

efficiency a measure of how much time, cost and effort has been applied to achieve the intended results

electronic validation machine-validation techniques to ensure the accuracy of data, which can be built into software, such as spell checkers. Types of electronic validation include range checking, existence checking and data type checking

evaluation measuring how well the solution meets the information needs and needs of the client

9780170440806

evaluation criteria measures that will be used to judge how well the solution meets the information needs and needs of the client

filtering inclusion of certain data in, or its exclusion from, a data set. Logic functions can be applied in choosing matching data that satisfies the conditions

flow visualisation a visualisation that involves representing data that illustrates the flow pattern of a data item or items

Form Controls Excel developer tools that change the settings chosen by the sheet or chart

format factors that govern the appearance of a digital solution, such as the typeface and background colour

functional requirements requirements of a digital solution that are directly related to what the solution will do

geospatial visualisations include data with geo-coordinates that are then overlaid onto familiar maps to indicate location and other values; also, **map-based visualisations**

Hungarian case a typographic convention that uses the object type to begin the variable name. Its use is no longer recommended

IPO chart (input–process–output chart) a defining diagram that shows how data is processed into meaningful information; it identifies what data is required for the solution (input), what information the solution needs to produce (output), and the processing steps required to transform the data into information, or the function of the solution

macro script that can automate certain actions; often used in applications to perform repetitive, predictable actions

manual validation the process of checking data entered for accuracy when done by a person, rather than by a machine

map-based visualisations visualisations that plot data onto geographic mappings; also, **geospatial visualisations**

matrix visualisation a visualisation that can be used to show the composition of individual items in the sample size. In this regard, it can be considered similar to pie charts

network visualisation a visualisation that shows relationships between different data items and relationships between different data sets

non-functional requirements characteristics that the user or client would like the digital solution to have; often tied to solution constraints

non-technical constraints factors that may limit or restrict solution development, specifically related to legal, social, usability or economic factors

outlier a data point that is either much bigger or much smaller than other data points in a set of data (e.g. on a graph)

Pascal case a typographic convention that is similar to camel case, with an upper-case letter at the start

pie chart circular chart that represents 100% of a whole. Segments represent the proportion of the whole.

pivot table a feature of spreadsheets that allows quick summarising of data

qualitative data categorical or non-numerical data that has been counted rather than measured; collected through interviews, focus groups or observation

repetition reuse of same or similar visual elements within a solution

rose chart (or **coxcomb chart**) a chart that is a variation on a radar and bar chart or pie chart

Sankey diagram a style of diagram used to illustrate flows of quantities by widths of arrows, typically energy flows or production losses

space area around and between objects

Spreadsheet Standards Review Board (SSRB) a body that recommends best-practice methods for spreadsheet model development

static data visualisation a visualisation that has a fixed view with fixed data; it does not change.

technical constraints constraints related to the hardware and software available for a project

testing process of checking that the data visualisation solution functions as expected

time visualisation a visualisation that represents a data item or data set over a period of time

9780170440806

usability a functional requirement principle relating to user capabilities and the ease of use of the solution; all spreadsheet solutions, including graphic representations of numerical data, need to be user-friendly; the user should be able to access the required information with ease and efficiency

validation the process of inspecting data, while it is being input into a solution, to check if the data is reasonable

Important facts

1 Databases and spreadsheets must be **organised** to be effective.

2 Choose **types of charts** with care so they visualise data clearly and accurately.

3 Data visualisations make **data patterns and relationships** clearer than lists of numbers.

4 **Dynamic interactive data visualisation** manually or automatically updates data and has user interaction.

5 **Design principles** affect the functionality and appearance of solutions.

6 There are **functional** and **non-functional** design principles.

7 Functional design principles are **usability** and **accessibility**. The non-functional design principle is **appearance**.

8 Appearance includes consideration of **alignment**, **repetition**, **contrast** and **space**.

9 A **constraint** is any factor that limits the data visualisation solution.

10 Typical constraints include software limitations and data set collection conditions.

11 **Design tools** are used to represent the functionality and appearance of your solution.

12 **Storyboards** can be used to show how the data visualisation animation might work.

13 **Flowcharts** can be used to show the procedure that users need to complete to create a data visualisation.

14 An **IPO chart** can be used to show how data is processed into meaningful information.

15 An IPO chart identifies what data is required for the solution (input), what information the solution needs to produce (output), and the processing manipulation activities required to transform the data into information.

16 **Wire frames** are annotated diagrams that provide a visual depiction of how your visualisation solution should look. Wire frame details are usually in black and white and show location, content and function.

17 A **mock-up** provides format information including: colour, size, font and 'look and feel' appearance.

18 ISO 8601 **Date standard** specifies that dates should be represented in the form YYYY-MM-DD. The hyphens can be omitted to use space more efficiently.

19 **Conventions** are general rules that are followed when using a particular format.

20 Common conventions include clear title, axis labelled, key or legend used, name of the author and the source of the data, unit of measurements shown and matching colours.

21 There are a range of software tools and functions that can be used to create data visualisations.

22 Many spreadsheet applications have a chart or graphing feature that helps manipulate the data into a visual form.

23 Google has a range of online cloud-based software tools that can manipulate data into a visual format.

24 Tableau Public is free software that can allow anyone to connect to a spreadsheet or file to create interactive data visualisations for the web. Academic licence is available for VCE students when arranged by their teacher.

9780170440806

TEST YOUR KNOWLEDGE

Review quiz

Types and purposes of data visualisation

1 Define 'data visualisation'.

2 List three types of data visualisations.

3 Explain the concept of geospatial visualisations.

4 Discuss one advantage of geospatial visualisations.

5 Describe a situation in which you would use a network visualisation.

6 Explain how timeline data differs from time series data.

7 Explain the purpose of a flow visualisation. Give an example to support your answer.

Design principles

8 What are the two main categories of design principles for data visualisation?

9 Explain the difference between functional requirements and non-functional requirements.

10 Define 'functional constraint' and 'non-functional constraint'.

11 Outline the purpose of a design tool.

12 Explain the difference between appearance design tools and functionality design tools.

13 List two design tools for appearance.

14 Discuss the advantage of creating a number of layout diagrams representing the data in a different format.

15 Explain the purpose of an IPO chart.

16 Outline a situation in which using a flowchart would be appropriate.

Formats and conventions

17 Define 'format'.

18 Define 'convention'.

19 Explain the difference between formats and conventions.

20 List four conventions usually followed in data visualisations.

Software tools and functions

21 List three software tools used to create data visualisations.

22 Describe the process of validation.

23 Explain the differences between data visualisations that are static, dynamic and interactive.

9780170440806

APPLY YOUR KNOWLEDGE

1 There are two main categories for data visualisations: explanation and exploration. What is the difference from the viewer point of view?

2 Which type of chart would you choose if the data were:

   a {temperature and time}

   b {stocktake data, numbers of each product}

   c {temperature recorded every 5 minutes, rainfall every half hour}

   d {contribution share of hours worked in a week}

EnergySampleFile

3 Download the *EnergySampleFile* in .xlxs or .csv format from NelsonNet. Use spreadsheet software, or an alternative, to do the following.

   a Create a monthly report on energy usage.

   b Estimate the cost of electricity for the chosen month where the rate for Peak is 40.2 cents per kWh and Off-peak is 17.5 cents per kWh. In addition to the usage charge, there is a service charge of $1.02 per day. Peak is the period between 3 p.m. and 9 p.m. each weekday. Weekends are charged at the Off-peak tariff.

   c List some of the assumptions you have made in calculating the monthly account.

   d Create a flowchart of steps for the calculation of the monthly energy cost.

   e Create an IPO chart for the spreadsheet calculations.

   f Create a monthly chart showing the daily energy usage and a chart of the daily energy cost using a bar chart and a line chart with two vertical axes.

   g Comment on some observations about similarities and differences of the two entity values (energy usage and cost).SampleElectricity Consumption.csv

9780170440806

PREPARING FOR Unit 1

OUTCOME 1

Your teacher will provide you with solution requirements and designs for which you need to gather and organise appropriate data, analyse it and present the findings as data visualisations. As part of the solution development you will validate your data and apply appropriate formats and conventions to the data visualisations.

Assessment

All assessments at Units 1 and 2 are school-based. Procedures for assessment of levels of achievement in Units 1 and 2 are a matter for school decision.

For this unit students are required to demonstrate two outcomes. As a set these outcomes encompass the areas of study in the unit.

Suitable tasks for assessment in this unit may be selected from the following:

- a folio of exercises or software solutions and a written report
- a presentation (oral, multimedia, visual) to present findings or software solutions.

Where teachers allow students to choose between tasks, they must ensure that the tasks they set are of comparable scope and demand.

Reproduced from the VCE Applied Computing Study Design (2020–2023) © VCAA; used with permission.

SAC Create a data story

A possible culmination task for school-assessed coursework would be the creation of a data story using data visualisation software. Students may choose from a range of teacher-provided data sets. Alternatively, students may identify a data set and cite full details of the source.

The process of validating data, manipulating the data set and presenting a description of the data behaviour would be the focus of the task. Students will document the development of the data visualisation from initial analysis through to an evaluation of the effectiveness of the final visualisation. A simple journal or blog space would be sufficient to record thoughts, steps and decisions.

Students would be expected to use suitable software (for example, Excel and Tableau) to first validate, and then manipulate, the data set before the creation of appropriate charts to convey a story about the data. While the data set will be static, the presentation may provide users with choices to select views of the data. This may be achieved through interactive controls or by having several charts accessed by a menu.

Clarity of design and ease of operation will be key performance criteria.

The format for submission will be discussed with your teacher. A range of possible formats are feasible, and local considerations will determine the final requirements.

Your teacher will also provide assessment criteria, a range of appropriate data sets or a source of data sets, journal or blog space details and a timeline for completion of the Unit 1, Area of Study 1, Data visualisation SAC.

9780170440806

CHAPTER

3 Designing software

KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

Digital systems

- functions and capabilities of key hardware and software
- components of digital systems required for processing, storing and communicating data and information

Data and information

- characteristics of data types
- types of data structures

Approaches to problem solving

- features of functional and non-functional solution requirements, constraints and scope
- design tools for representing the functionality and appearance of solution designs such as data dictionaries, mock-ups and pseudocode
- naming conventions for solution elements such as files, functions, methods and variables
- project plans to coordinate and monitor the tasks, including sequencing and time allocation to create software solutions

Reproduced from the VCE Applied Computing Study Design (2020–2023) © VCAA; used with permission.

FOR THE STUDENT

This chapter relates to VCE Applied Computing Unit 1, Area of Study 2: Programming. It introduces basic programming concepts such as software development tools, functions and capabilities of software and hardware, design tools, types of programming languages, and universal programming ideas such as pseudocode, data types and data structures. You will also begin learning a specific programming language that you will use to develop a response to a design brief for Unit 1, Outcome 2. This will involve the creation of a project plan as well as responding to a design brief in relation to functional and non-functional requirements, constraints and scope. You will also need to design and apply suitable testing and debugging techniques as well as evaluate the effectiveness of your solution and project plan.

FOR THE TEACHER

This chapter introduces students to some of the universal theoretical concepts behind programming that are required for Unit 1, Outcome 2. It does not assume knowledge of, or use source code from, any actual language. Little previous programming experience can be expected from many students, so it is important to introduce the chosen language to them early and give them time to build on their skills. Unit 1, Outcome 2 requires that students apply all stages of the problem-solving methodology to design, develop and evaluate a software solution in response to a design brief provided by their teacher. This design brief should include functional and non-functional requirements, constraints and factors that determine scope. It should be written to allow students to draw on knowledge of basic programming concepts, such as storage, logic, loops and calculations. Programming concepts such as logic, loops and calculations are covered in detail in Chapter 4. Students should complete a project plan as part of this outcome.

Shutterstock.com/Nirad

9780170440806

Information systems in programming

Information systems comprise people, data, processes and digital systems. In the context of programming, the key parts are:

1. people, who interact with systems according to their needs, such as programmers, data entry operators, system managers, technicians and end users
2. data, which is composed of raw, unprocessed facts and figures, such as someone's date of birth, that is used as input to be processed into meaningful information as output, such as someone's age
3. processes, which are the manual and automated ways of achieving a result, such as a manual data backup or an automated hard disk error scan
4. **digital systems**, which are made up of the hardware and software needed to support programming and software use.

Digital systems are made up of the following components:

- networks, which exchange data between devices
- protocols, or rules used to coordinate and standardise communication between devices
- application architecture patterns, which are sets of principles used to provide a framework for structuring solutions to recurring problems. For example, a **thin client** is a lightweight computer that has been optimised to work with a server-based computing environment; the server does the majority of the processing and computing. This application architecture was born out of the belief that, rather than use powerful computers, it is better to use 'dumb' workstations connected to a powerful central computer that does all the processing work for them
- software, which comes in three types: systems, applications and utilities
- hardware, or physical equipment for input, output, storage, processing and communication.

The following section will discuss in more detail the hardware and software components of digital systems.

Functions and capabilities of hardware

The physical components of digital systems are known as hardware. They include familiar items such as the monitor, mouse, hard disk drive (HDD), motherboard, graphics card and so on.

Hardware requires software instructions to control it; software requires hardware to carry out its instructions. They work together to form a usable digital system.

Hardware falls under a number of categories, including:

- **input devices**, which are peripherals, such as keyboards, mouses, styluses, scanners and data loggers, that enable users to send data and commands to software and the operating system

- **output devices**, which are peripherals, such as printers and monitors, that display information from a computer in human-readable form
- processing hardware
- storage hardware
- communication hardware.

The following sections cover processing, storage and communication hardware in greater depth.

Processing hardware

The key element of programming hardware is the **processing hardware** – the digital processor that converts data into information and controls all the other hardware in the system.

3.1 THINK ABOUT APPLIED COMPUTING

Research the CPUs of a mobile phone, a laptop and a desktop gaming machine. How do they differ, and how is their performance measured?

CPU

The **central processing unit (CPU)** is often thought of as the 'brain' of a digital system and it handles most of a system's data manipulation. The CPU is helped by other processors, such as those in the video card, hard disk drives and audio controller chips. Major CPU designers include Intel, AMD, ARM and IBM.

Reduced instruction set computing (RISC) CPUs, such as ARM, have smaller instruction sets than **complex instruction set computing (CISC)** CPUs, such as Intel's i7. Being cheaper and smaller, and therefore drawing less power and producing less heat, makes RISC CPUs ideal for use in smartphones and tablets.

GPU

The **graphics processor unit (GPU)** is a very fast and expensive processor specifically designed for high-speed image processing in graphics cards. Application software, such as Adobe Photoshop, video editors and 3D games, exploits GPU power to accelerate processor-intensive calculations.

Storage hardware

Storage hardware retains data and software for both immediate and later use. It comes in two main types: primary storage and secondary storage.

Primary storage

Primary storage is a computer's **random-access memory (RAM)**. It has billions of storage locations in silicon chips. RAM stores instructions and values including variables, arrays and other storage structures when programs are running or being created. RAM chips are volatile because they lose their data when electricity is turned off. Dynamic RAM (DRAM) is used as the main memory in computers; high-speed (and expensive) static RAM (SRAM) is used in graphics cards and CPUs.

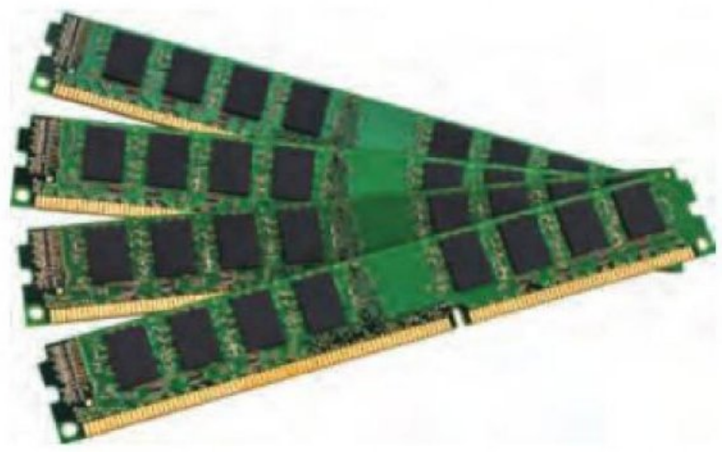


FIGURE 3.1 Random-access memory (RAM) modules

Secondary storage

Permanent **secondary storage** stores data, information and applications when they are not actively used. Secondary storage includes hard disk drives (HDD), solid state drives (SSD) and network-attached storage devices (NAS).

Hard disk drives (HDD) are aluminium disks densely crammed with magnetically recorded bits of 1 and 0. Spinning at up to 10 000 RPM, they store and retrieve data at incredible speed, with breathtaking accuracy and reliability. They are very cheap per megabyte of capacity, and are still the biggest and most reliable long-term storage medium you can find. In 2019, an 8 TB (approximately 8000 GB) HDD cost approximately $325, which works out to about 25 GB of storage per dollar.

Solid state drives (SSD) store data in non-volatile **NAND RAM** (similar to that used in flash drives and SD cards). They have no motors to age and fail, they run silently, start up instantly, consume less electricity, generate less heat and may access data faster than a HDD. Unfortunately, NAND RAM eventually loses its ability to be written to, stores less data per square centimetre of storage space, and is expensive. In 2019, a 1 TB SSD cost approximately $330, or around 3 GB of storage per dollar.

A **network-attached storage (NAS)** device is a networked team of HDDs. Using a NAS offers more speed, capacity (e.g. 12 TB), data protection (e.g. hot-swap disks), convenience and reliability than a simple USB hard disk alone.

> **3.2 THINK ABOUT APPLIED COMPUTING**
>
> Find online Australian prices for various sizes of SSD and HDD. Graph their costs against their capacity.

Network hardware and software is described in Chapter 7.

Communication hardware

Communication hardware is used for sending and receiving data and information from one place to another, such as via networks.

Ports are physical sockets or connectors that carry data between a computer and external devices, often referred to as peripherals. Universal serial bus (USB) is a standardised high-speed way to connect many devices, including flash drives, printers, modems, keyboards, mouses, speakers and smartphones.

In the days before USB, many manufacturers invented their own type of port. Computers were jam-packed with ports to accommodate individual makes of modem, printer, mouse, keyboard, monitor, joystick, etc.

As a programmer of a high-level language you will not need to worry about directly controlling devices such as printers or disk drives. Your programming language will allow you to issue commands such as 'display this' or 'store this data in memory' and the **operating system (OS)** will negotiate with the hardware to fulfil your requests. The OS knows how to talk to hardware because each device comes with a software **driver**, which is like a dictionary that tells the OS the commands that the hardware understands. The OS gives a generic command, such as 'print this', and the driver translates the command into language that the specific piece of hardware understands. Hardware misbehaviour is often caused by using an incorrect or outdated driver.

Storing 1 YB would take 2 500 000 cubic metres of 64 GB microSD cards – the equivalent of the volume of the Great Pyramid of Giza.

TABLE 3.1 Storage units

Unit	Symbol	Equivalent to	
		RAM	***Data storage***
Byte	B	8 bits (1 or 0), the basic unit of storage	8 bits
Kilobyte	KB	1024 bytes	1000 bytes
Megabyte	MB	1024 KB (roughly 1 million bytes – the size of two average novels)	1000 KB
Gigabyte	GB	1024 MB (PCs have gigabytes of RAM)	1000 MB
Terabyte	TB	1024 GB (hard disks have terabytes of storage)	1000 GB
Petabyte	PB	1024 TB	1 000 000 GB (1×10^6 GB)
Exabyte	EB	1024 PB	1 000 000 000 GB (1×10^9 GB)
Zettabyte	ZB	1024 EB	1 000 000 000 000 GB (1×10^{12} GB)
Yottabyte	YB	1024 ZB	1 000 000 000 000 000 GB (1×10^{15} GB)

Functions and capabilities of software

Software is used to control computing devices to process data. There are many types of software programs used to:

- calculate, such as spreadsheets
- store and organise data, such as databases
- entertain, such as games
- communicate, such as web browsers, email and instant messaging
- control devices, such as embedded software in TVs, toasters and car engines.

Hundreds of other programs exist. These pieces of software are all created by programmers.

Types of software

A computer uses **system software tools** to manage hardware and run the user's programs; for example, the OS, device drives and communication protocols.

Applications are used to perform work or complete larger tasks. Microsoft Word and Excel, Steam client and Mozilla Firefox browser are examples of popular applications.

Utilities are usually small, single-purpose software tools that do a specific job or add functionality to an operating system. They include text editors, audio format converters and DVD-burning software.

Windows users can run services.msc to see the dozens of tasks the OS is managing in the background.

THINK ABOUT APPLIED COMPUTING 3.3

Linux is a free, open source OS. Where is Linux used, and what benefits and drawbacks does it have compared with Windows and macOS?

The operating system (OS)

Operating systems, such as Windows, macOS, Linux, iOS or Android, are system software that controls a computer's hardware and runs the user's application software. Operating systems all perform similar functions:

- loading and saving data and programs
- displaying output and printing
- processing sound and music
- allocating memory for user programs
- watching the user's keyboard and mouse activity
- controlling network and internet access
- encrypting, decrypting, compressing and decompressing data
- caching downloads
- controlling user logins and maintaining security over accounts, files and access to resources
- running background programs – for example, disk defragmenters and virus scanners – and checking for upgrades to keep the system working efficiently.

Programming and scripting languages

There are more than 700 programming languages, but you will not have to learn them all.

Programming languages are used to give instructions to computer processors so they can calculate useful information or carry out tasks for humans. Whether your phone is playing an MP3, your car is turning on its anti-skid braking, or McDonald's is calculating staff wages, programming languages are needed.

Scripting languages conveniently store sequences of instructions that, alternatively, could be entered one at a time. Like human languages, there are many programming languages, each with distinctive grammar, punctuation and vocabulary. Most programming languages have special abilities or strengths that make them more useful than other languages for a particular task.

Professional programmers typically know a handful of languages and choose the best one for each job based on its strengths and weaknesses. Choosing the languages to learn is a big decision, but remember that learning one language makes it easier to learn others. The most popular programming languages include C (C++ or C#), Python, Java, JavaScript, Perl and PHP, SQL, Visual Basic and Swift.

- C, C++ or C# is used for writing low-level utilities and fast applications. This could include operating systems, embedded microcontroller programs, web-based applications and games.
- Python is an interpreted object-oriented programming language used for web and app development.
- Java is used for web applications and web services and for building Android apps.

- JavaScript is a client-side scripting language for websites.
- Perl and PHP are used in website and network programming.
- SQL, or structured query language, is a scripting language for database programming.
- Visual Basic and Visual Basic.NET are used widely to create applications for Windows-based computers.
- Swift is a programming language for macOS, iOS, watchOS and tvOS.

While programming languages may differ, they all do basically the same job: they control a digital system such as a computer, tablet or smart device.

Programming languages differ in the amount of direct control they give over a computer's hardware and operating system. With a **high-level programming language** such as Visual Basic or Python, programmers avoid having to worry about complex details of the structure of actual disk files or where data is stored in memory. High-level languages are simpler to use, but lack the control of complex, but more difficult to learn, low-level languages. Conversely, a **low-level programming language** such as C or machine code requires more skill and knowledge from the programmer, but allows more direct control of the workings of a computer.

High- and low-level programming languages each have their uses. To write a simple alarm clock program, a high-level language is fine. To write a device driver to control a printer, only a low-level language will do.

3.4 THINK ABOUT APPLIED COMPUTING

Research three popular languages to discover their origins. What did existing languages lack that led to the need for the new languages?

C has both high-level features (for example, FOR loops and arrays) and low-level features (for example, memory pointers and byte-level operators), making it popular and suitable for many occasions.

3.5 THINK ABOUT APPLIED COMPUTING

Visual Programming Language (VPL) is another type of programming language that has become increasingly popular in modern computing, in particular for teaching purposes. Research VPLs to discover their origin and uses. Would you consider using a VPL? Why or why not?

Software development tools

To develop software, you need a number of basic, essential tools, including an **editor**, a **compiler** or **interpreter**, and a **debugger**. The following section discusses these in more detail.

An editor is a specialised application software that is used for creating human-readable **source code** or, rather, human-readable programming instructions. Code editors come with specialist features designed to make programming easier by, for example, highlighting programming keywords, using code completion to reduce typos and other mistakes, detecting unbalanced parentheses and adding line numbering. Programming editors, such as the one shown in Figure 3.2, show colour-coding, indenting, collapsible text and line numbering.

A compiler converts source code into executable programs that a computer can carry out; that is, that a particular CPU and operating system, such as Windows or macOS, can understand. An interpreter directly executes source code without needing to compile it beforehand.

Executable code compiled for one **platform** will typically not work on another without being ported (re-compiled for another platform). Porting can be hard and expensive work; this is one reason why many apps are available for Windows but not Mac or vice versa.

3.6 THINK ABOUT APPLIED COMPUTING

Research online to choose a popular code editor. Examples are Notepad++ and Programmer's Notepad. What features does it possess that make it better than a plain text editor for creating source code?

```
function getState(  stsDOC, stsDSC, stsDJS, stsDBS, stsDWS )
function getStateReason( stsDWS, stsDOC, stsDSC )
{
        var stateReason = '';

        if( typeof stsDWS !== 'string' || stsDWS === '' ||
            typeof stsDOC !== 'string' || stsDOC === '' ||
        typeof stsDSC !== 'string' || stsDSC === ''
        ) {
                return '';
        }

        if (stsDSC !== 'NO' || stsDOC !== 'NO') {
            stateReason = 'AttentionRequired';
        } else if (stsDSC === 'NO' && stsDOC === 'NO' && stsDWS === '1900') {
            stateReason = 'Paused';
        } else if (stsDSC === 'NO' && stsDOC === 'NO' && stsDWS === 'NO') {
            stateReason = 'None';
        } else {
            stateReason = '';
        }

        return stateReason;
}
```

FIGURE 3.2 A programming editor

Part of the difficulty of porting is that programs must often be substantially changed to work under a different operating system.

A debugger helps programmers to find 'bugs', or programming errors. Sometimes debugging can take as long as the original programming time, or even longer if the program has been poorly designed. Debuggers may:

- highlight incorrect **syntax** (programming expression) and show how statements should be expressed
- allow programmers to set break points in code, where the compiler will stop and let the programmer inspect the current values of variables
- allow line-by-line stepping through code so developers can find exactly where a problem arises.

Few programmers today use separate editors, compilers and debuggers. Most use an **Integrated Development Environment (IDE)**, such as Microsoft's Visual Studio, that combines the development tools into a single package.

Figure 3.3 shows:

1. the toolbox of **graphical user interface (GUI)** objects the programmer can insert into the program
2. the code window, which the programmer uses to instruct the program how to act when an event takes place, such as the clicking of a button
3. the form, which will be the visible **interface** for the program's user
4. the **properties** of the selected **object**, which let the programmer modify an object's characteristics or behaviour
5. the project manager, with which the programmer can manage the various files and components related to the program
6. onscreen help to give brief reminders of what the currently selected object is like.

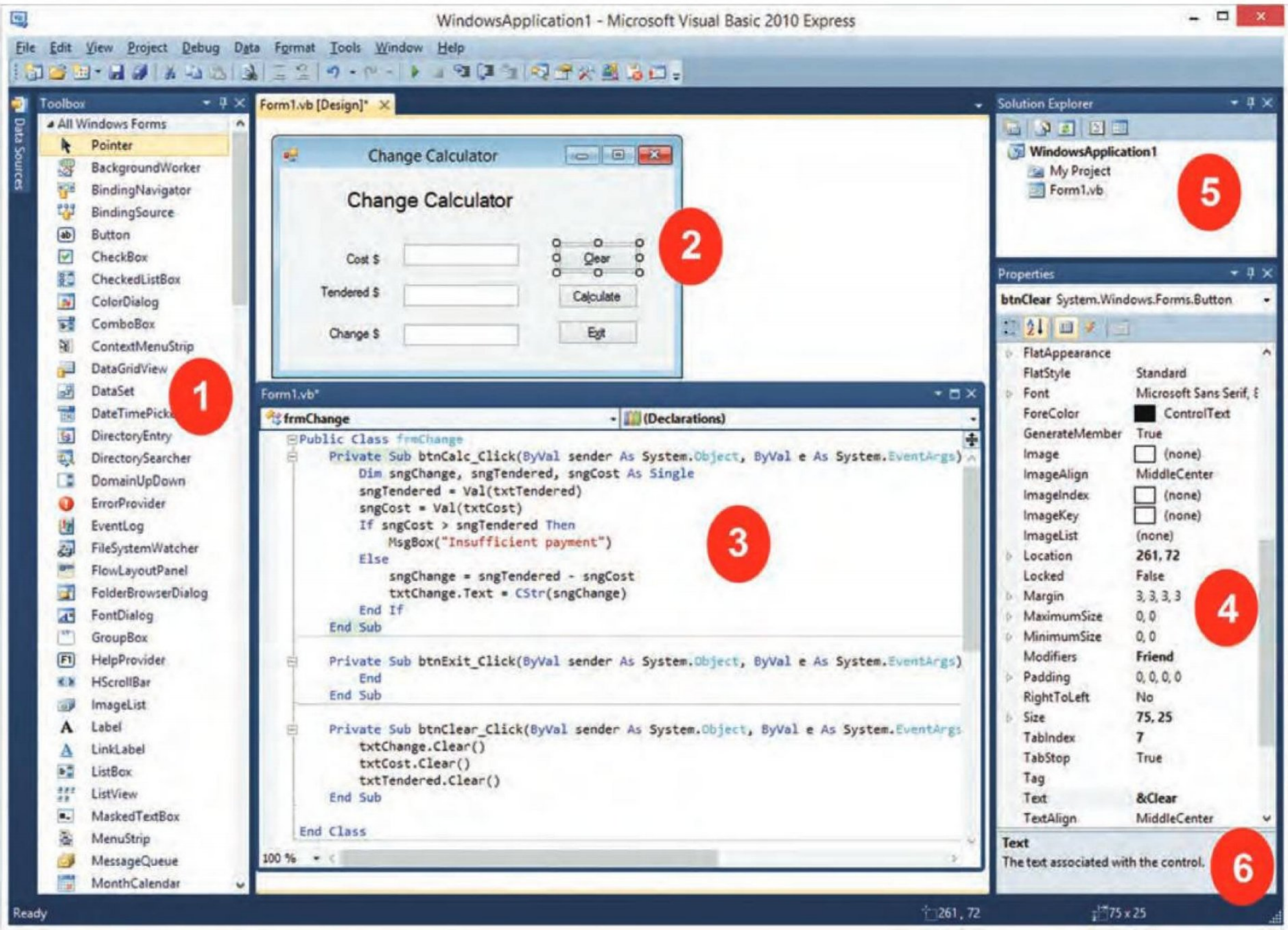


FIGURE 3.3 Example of an IDE

Storage structures

A **storage structure** is a location in RAM where data is stored during the execution of a program. The two main data storage structures are **variables** and **constants**. Variables are so-named because the value stored in a variable can *vary*, or be changed by a program. Constants, on the other hand, have fixed (unchanging) values during a program's execution, such as the value of pi, or the number of Australian states and territories.

Arrays can store many values in numbered 'slots'. For example, Figure 3.4 shows pseudocode that compares the rainfall of January and December using an array intRain that stores monthly rainfall figures. To address (refer to) an individual value, simply give the name of the array and the desired **index** (slot number).

```
IF intRain[1] < intRain[12] THEN
     DISPLAY "January was drier than December!"
END IF
```

FIGURE 3.4 Pseudocode addressing values in an array

Project plans

Features of project management typically involve the use of a **project plan** to coordinate and monitor tasks, including their sequencing and the amount of time required to complete each task. This might also involve tracking the milestones that exist within a project timeline.

You must use a Gantt chart as the project management tool to coordinate your project. A **Gantt chart** is a graphic timeline that:

- lists all tasks in a project
- organises the tasks in order
- shows which tasks must wait for other tasks to finish before they can begin
- allocates people and resources to tasks
- tracks the progress of tasks and of the entire project.

Shutterstock.com/Casper1774 Studio

FIGURE 3.5 Project plan in the form of a Gantt chart

You can create a Gantt chart using pen or paper, but it is faster and easier to create them using software. Suitable software includes spreadsheet software such as Microsoft Excel or Google Sheets, as well as specialised project planning software such as Microsoft Project. Many online project planning tools also exist, including OpenProject, GanttProject, TeamGantt and Redbooth, and most of them are free for small projects.

When using software to create your Gantt chart, you will not be assessed on your technical prowess with the software. Rather, you will be assessed on how well your Gantt chart demonstrates your understanding of the concepts and processes of project management.

Coordinating and monitoring tasks

It is critically important that all tasks within a project are coordinated and managed in order to ensure that a project has the best chance of succeeding. There is no guarantee that a project will be successful if managed well, but projects are more likely to be delivered on time and within budget if each task is tracked using information such as its expected and actual duration.

Tasks

A **task** is any activity within the project timeline that needs to be completed within a defined period of time. It is an essential component of the project as a whole and typically requires management of resources, such as people, software and hardware. All of the tasks that are required to complete the project should be shown in the project plan.

The names given to tasks should concisely describe the task with enough detail to be understood. For example, during the analysis stage, you might need to collect data by conducting observations of workers in a business in order to determine solution constraints. The collection of data should be included in the project plan with a set duration and a description such as 'conduct worker observations'. A task immediately after might be labelled 'determine solution constraints'.

Tasks can contain sub-tasks. Sub-tasks may be required if the core task, or activity, consists of more than one smaller task. It is particularly useful to show sub-tasks on a project plan when different sub-tasks involve different resources. This also allows for more flexibility in the allocation of time for tasks, such as allowing for tasks to be completed at the same time.

Task dependencies

In the management of a project, the project plan is very rarely made up of stand-alone tasks; more often, tasks rely on other tasks having been completed. For example, the task 'determine solution constraints' above could only be started if the task 'conduct worker observations' had been completed. This type of task is referred to as a **dependent task** as it depends on one or more other tasks having been completed in order to begin. Tasks that must be completed before another task can begin are called predecessors. The dependent tasks are called successors.

Any delay in a task that is a predecessor to another task can therefore have an effect on subsequent tasks, as they cannot begin until the initial task is complete. If a predecessor runs over time, all of its successors will be delayed, causing problems for other tasks and deadlines. This is where a Gantt chart becomes very useful in helping to monitor tasks and meet deadlines on schedule. It also helps you to visualise problems that may occur down the line if a predecessor is late. If a task runs over time, the length of time that can elapse before it affects other tasks is called **slack time**. When workers have slack time, you can reassign them to other tasks.

Dependencies are shown in Gantt charts using arrows linking one task (the predecessor) to another task (the successor).

Milestones

A **milestone** represents the achievement of a significant stage in a project. The milestone itself has zero time duration. For example, the completion of the printing of a questionnaire so that it can be distributed to respondents would be a task of zero time and represents a milestone. It follows tasks in which the questionnaire has to be researched, written, proofread and finally printed, all of which do take time.

Milestones are shown in Gantt charts using a diamond shape.

Determining scope

The **scope** outlines the boundaries or parameters of the solution so all stakeholders are aware of exactly what the solution will contain. The scope of the solution consists of two elements: what the solution will do and what the solution will not do.

9780170440806

Many house and land package contracts state exactly what is included with the package and exactly what is not included in the package for the price. For example, kitchen floor tiles might be included, but the garden might not be landscaped.

What the solution will do

What the solution will do is set out in a list of all the solution requirements (both functional and non-functional) that will be included in the solution.

What the solution will not do

What the solution will not do is set out in a list of all the solution requirements that will not be included in the solution.

Usually these are solution requirements that the client initially sought but that, because of constraints, have been left out of the solution project.

At the start of the project, it helps to outline what will and will not be included in the solution to prevent arguments later in the project between the client and the developer.

THINK ABOUT APPLIED COMPUTING 3.7

What problems do you think a clear scope of solution could avoid later in the project?

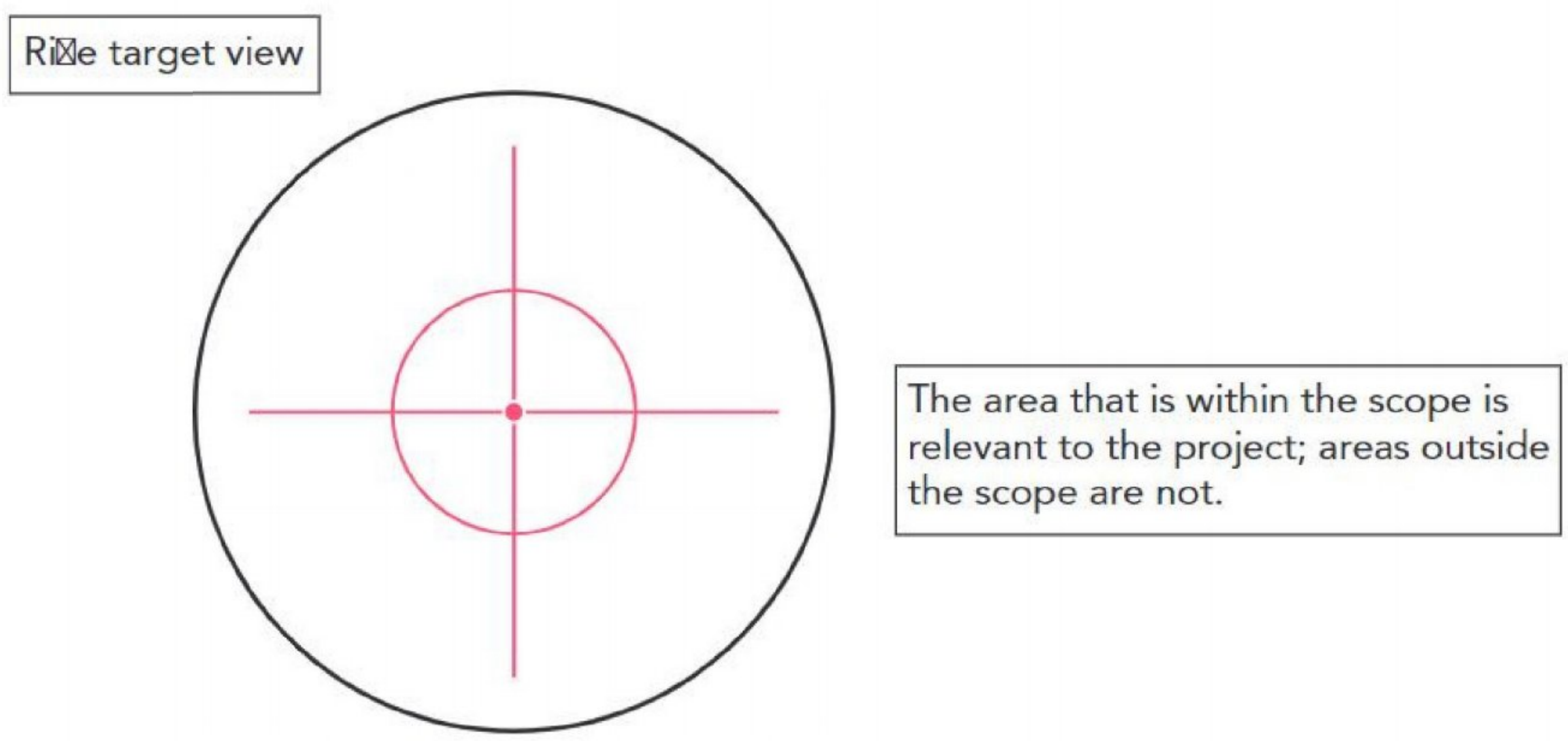


FIGURE 3.6 Scope of solution

Scope creep happens when a client changes the scope of a project (by increasing it) during the life cycle of the project. This can be very expensive to a developer in terms of time and money, as it means the project will take longer to complete, often without any extra money being provided by the client.

An statement of scope for a solution might be as follows: 'The solution will display population data of towns in Victoria in a visual format and graphically represent the distances between those towns. It will be created to be user friendly and, for privacy reasons, it will not display any personal details.

A statement for something that is out of scope might be: 'The requirement that allows the user to zoom in on a particular region or town will not be included in the project because of economic factors, but may be added at a later stage.'

In a design brief, aspects of the solution that are within scope are described without technical detail. There may be general descriptions of what the solution will contain in relation to functional and non-functional requirements. Determining what may be out of scope – for example, deciding what functionality may be delayed for later releases due to time constraints – normally occurs during the analysis stage.

Functional and non-functional requirements

Solution requirements are what the client needs from a solution; that is, what the system must do. These can be broken down into **functional requirements** and **non-functional requirements**.

Functional requirements

Functional requirements are directly related to what the solution will do. They typically involve: calculations; data processing; opening, reading and writing to files; data manipulation such as image editing; and other specific functionality required within the system.

Some examples of functional requirements for software solutions are that they:

- save customer data to a file
- calculate discount values on products
- set an alarm to go off at a particular time
- load a set of jobs into a timetable.

Functional requirements are usually described in terms of the inputs required, the sequence of operations that will be performed within the function, and the output(s) of the function after processing has occurred. These written descriptions often include dot-point descriptions for the purposes of clarity. An example of a functional requirement for a music performance system can be seen in Figure 3.7.

1. Add Performance

This screen will allow the entering of performance information. Information consists of the performance name, date and theme, as well as a unique performance ID (numeric). Also recorded is whether the performance is a concert, and, if so, the theme of that concert.

Inputs:

- Performance ID (unique)
- Performance Name, Performance Date
- Whether it is a concert or not
- Concert Theme

Sequence of Operations:

- Input will come in from the User Interface, all items except concert theme are required fields
- Themes should be retrieved from a file containing all available themes so a user can select one
- No two concerts should have the same ID
- A theme only needs to be selected if it is a concert
- No concert should be added if a concert with the same theme has occurred in the last six years
- Once data is entered and validated, it should be saved to a performance file

Outputs:

A pop-up box should be displayed showing the success or failure of saving the concert information. Once information is saved, the user should be returned to the main menu.

FIGURE 3.7 Functional requirement for a music performance system

Non-functional requirements

Non-functional requirements are those characteristics that the client would like the solution to have, but that do not affect what it does. Non-functional requirements are often tied to the constraints of the system. They can be categorised in terms of usability, reliability, portability, robustness and maintainability.

Non-functional requirements must be measurable – this means that it must be possible to test them to see if the requirement is met.

Determining non-functional requirements usually involves discussions with a client, such as asking if the software must work on different operating systems, or asking who the users of the system will be and the level of technical experience they have.

Usability

Usability relates to how easy a system is to learn and use. This is typically described in terms of efficiency and effectiveness. Common factors of usability include the clarity of the user interface and the intuitiveness of the functions within the system. The success of a system's usability is often measured in terms of user satisfaction. An example of a non-functional requirement related to usability is that 'users should be able to use all basic functions after one hour of training'. This non-functional requirement is highly reliant on the skills, expertise and needs of the intended users of the system.

Reliability

The **reliability** of a software solution relates to how much it can be depended upon to function as designed, and for how long. Typically, this requires that the software is deemed **fit for purpose** over time and that it is resistant to failure.

The reliability of a system is generally expressed as a probability measure, where *reliability* = 1 – *the probability of failure*. An example of a reliability measure is the prediction of the **uptime** of a system or solution, such as 'the system should have a 99.9% uptime over any 12-month period'; another might be 'the ability to add a new customer should be available to staff members during working hours on weekdays'.

Portability

The **portability** of a software solution relates to how easily it can be used in different operating environments. This can involve measuring how easy it is to move from one system to another system with the same architecture; how easy it is to reinstall a program on a new system; and the ability to use the same software on multiple operating systems and platforms.

The most time-consuming element of portability is the requirement to write software that will work on multiple operating systems and platforms. This typically means that a developer must separate the user interface from the core functionality and logic so that it is easy to create a new user interface for a new system; for example, creating an application that works on a mobile device as well as on a desktop computer.

Portability is increasingly important as mobile computing becomes the norm; users often expect that applications will work on mobile technology as well as on desktop technology.

Robustness

The **robustness** of a system relates to how well a software solution responds to errors that occur when the software is being used; that is, it should perform correctly in every situation a user encounters. Robustness is therefore an evaluation of the error-handling techniques built within a software solution. For example, the robustness of a piece of software would be evaluated by assessing how it responds to bad input from a user. Robustness is closely linked to the use of validation techniques as these can help prevent errors from occurring when users enter unexpected or invalid input. An example of a non-functional requirement related to robustness is that 'the system should reject invalid data entered by a user'.

Robustness is measured in terms of the number of failures, crashes and errors that occur while a system is running.

Maintainability

Maintainability is related to how easy it is to look after software once it is in use. This can involve fixing errors in the code, maximising efficiency and reliability, installing the software on new systems and, in some cases, expanding on the current functionality with new functionality. Simply put, maintainability is measured in terms of how easy it is to fix, modify or change the software once people are using it.

Often, maintainability is measured in terms of the number of hours a developer or administrator spends on the system to keep it running after it has been put in place. An example of a non-functional requirement related to maintainability is that 'fewer than 10 hours should be spent per quarter on maintaining the system'.

Constraints

Solution **constraints** are factors that may limit or restrict solution requirements, determined during analysis. At the stage when a design brief is created, these constraints are described only in general terms. Typically, constraints involve economic, technical, social, legal and usability factors.

Economic constraints

Economic constraints include time and cost.

The deadline by which the user or client needs to have the solution operational will define the time available to design and develop the solution. The longer this is, the more time there is to complete an in-depth analysis and detailed designs and to develop advanced features of the solution. The shorter the timeframe, the more quickly each stage in the problem-solving methodology needs to be completed.

Meanwhile, the budget (money) available to complete the project may affect the hardware and software (digital systems) available for use, the number and range of staff who are available to work on the solution and even the data used as input (if the data sets required need to be purchased).

A lack of either time or money may result in a re-evaluation of the user's requirements, or a re-evaluation of how the requirements can be achieved.

Technical constraints

Technical constraints are constraints related to the hardware and software available for the project. Available hardware and software, memory and storage capacity, processing and transmission speeds and security concerns are all examples of possible technical constraints.

For example, developers need to keep in mind that smartphone users may not always have access to a high-speed network connection, so they need to ensure that any animated data visualisation solution does not require a large amount of bandwidth to download and view.

> **3.8**
> **THINK ABOUT APPLIED COMPUTING**
> List three other technical constraints that developers of smartphone apps need to consider when developing a product.

Social, legal and usability constraints

Non-technical constraints relate to areas other than hardware and software. Usability and the user's level of expertise (which is a social factor) are examples. If a solution is being developed for users with little digital systems expertise, this may restrict some of the

Privacy laws were discussed in Chapter 1; copyright laws are discussed in Chapter 6.

requirements that would involve complex manoeuvres to complete. Creating a solution for a child audience may restrict the method used to input data into the solution.

Legal requirements are another type of non-technical constraint. Privacy laws may restrict features linked to displaying personal data in the solution, or to collecting data from the devices of someone using your solution. Copyright laws may restrict features that allow other users to upload content to the solution.

Characteristics of data types

In programming, classifying a variable by **data type** is a way of determining the data that variable can contain, as well as how that variable can be manipulated – that is, what it can do, and what can be done to it. While programming languages vary greatly from each other, data types do not; they are consistent across all programming languages. When programming, it is important not only to choose an appropriate data type when creating a variable, but also to select the most efficient data type. For example, it is not efficient to select a numeric data type that supports decimal places when creating a variable if that variable will only ever contain whole numbers. Similarly, storing a number as a string is not as efficient as storing it as a numeric data type, even if there exists a method of converting strings to numbers.

Numeric

The **numeric** data type consists of whole numbers, referred to as **integers**, and decimal numbers, referred to as **floating point numbers**. Integers can be referred to as *unsigned*, which means they can only store positive whole numbers, or *signed*, which means they can store both positive and negative whole numbers.

All numeric data types can have mathematical operations performed on them. The fundamental operations shown in Table 3.2 are the most common ones.

TABLE 3.2 Fundamental data type operations

Addition	+
Subtraction	–
Multiplication	*
Division	/
Whole number division (quotient)	//
Remainder after division (modulo)	%
Powers	**
Assign values	= or ←

When more than one operation appears within a line of code, the order of operations follows the same rules as in mathematics: brackets, orders, division and multiplication, addition and subtraction – or BODMAS. If two operators have the same precedence, they are evaluated from left to right.

Numeric data can also undergo comparisons, with those shown in Table 3.3 being the most common.

TABLE 3.3 Data type comparison operations

Less than	<
Less than or equal to	<= or ≤
Greater than	>
Greater than or equal to	>= or ≥
Equal to	== or =
Not equal to	!= or <>

Integer

Integers are commonly represented internally in a computer system as a group of binary digits, called *bits*. A bit is the smallest unit of data in a computer. A bit has a single binary value, 0 or 1, and bits are stored in multiples of eight, referred to as *bytes*; therefore, there are eight bits to a byte.

The maximum and minimum values of an integer depend on the computer architecture used to run the program and whether the integer is signed or unsigned.

In **32-bit computer systems**, integers that are signed have a minimum value of -2^{31} and a maximum value of $2^{31}-1$; from −2 147 483 647 to 2 147 483 647. Unsigned integers have a minimum value of 0 and a maximum of $2^{32}-1$ (4 294 967 295).

In **64-bit computer systems**, integers that are signed have a minimum value of -2^{63} and a maximum value of $2^{63}-1$. Unsigned integers have a minimum value of 0 and a maximum of $2^{64}-1$.

It is important to know the computer architecture on which a program will run before designing and developing a software solution, as going beyond the maximum and minimum values of integers can result in an **integer overflow**. This may result in a program crashing or producing inconsistent or invalid output. Integer overflows compromise a software solution's reliability and security.

The power of 31 for signed integers represents 32 bits, minus 1 bit that is needed to determine if the signed integer is positive or negative. The 1 that is subtracted from the total is due to computer systems counting from 0, rather than 1.

Floating point

Floating point numbers, also referred to as 'floats', are the computer representation of real numbers; that is, numbers that allow for decimal places.

Floating point numbers consist of two main parts:

- a *mantissa*, which contains the digits of the number that is represented. These can be either positive or negative
- an *exponent*, which helps determine where the decimal point is placed within the significand.

Two basic formats used for floating point numbers in computer systems are single precision and double precision. Single precision is used in 32-bit systems and double precision in 64-bit systems.

Character

The **character** data type is a symbol that has meaning. It can consist of any single meaningful unit, such as a letter, a number, a punctuation mark, a symbol, or even a space. For example, the word 'example' has seven characters.

What is determined to be 'meaningful' relies on something referred to as **character encoding**. Character encoding is one way a computer program can translate binary data into meaningful characters. There are many character-encoding schemes that handle different character sets. For example, ASCII is a character-encoding scheme that represents English characters, punctuation and numbers. UTF-8 encoding is a character-encoding scheme that can represent characters from other languages such as Japanese Kanji and Korean Hanja, as well as punctuation symbols such as those representing euro (€) and yen (¥).

A set, or sequence, of characters is referred to as a **string**. For example, the string 'I like pie' consists of eight characters that are letters, and two characters that are spaces. Strings are often implemented in programming languages as an array of characters.

Characters and strings can undergo the same comparisons as the numeric data type, as listed in Table 3.3. Depending on the programming language used, they can also undergo some of the fundamental data type operations as listed in Table 3.2, such as addition, multiplication and assignment. For example, when working with strings, addition acts to concatenate (add) strings together, which is useful when constructing strings that must incorporate variables. An example of this can be seen in Figure 3.8.

> **THINK ABOUT APPLIED COMPUTING 3.9**
>
> How many character encoding sets are there? Why do you think so many separate encoding sets have been created?

```
Input firstName
Print "Hello " + firstName + "!"
```

FIGURE 3.8 Pseudocode example of string addition

Boolean

> Boolean values take up only a small amount of space in memory, so it is tempting to use them to store any type of data that seems to have only two values. It is important that you consider future expansions to programs before making a decision that will limit a data type. For example, many old systems that had gender stored as a Boolean value are now having to be rewritten to change gender to a character data type.

Boolean data types have only two possible values: 0 and 1. These are often referenced with the words 'True' and 'False' in a programming language. This data type is named after George Boole, a 19th-century mathematician who was the first to define an algebraic system of logic. Boolean data types are very useful for systems that require decisions to be made or conditions to be met.

Much like integers, Boolean values can be treated mathematically, allowing for comparison operators such as those listed in Table 3.3. This allows for Boolean logic operations to occur in any programming language.

Boolean values can also be used with the fundamental operators *and*, *or* and *not* in statements where a condition must be met. For example, if a program is required to turn on a light in a room if it is dark and the light is not already on, it could use two Boolean values to test for this condition (Figure 3.9).

```
If isDark = True and lightOn = False Then
    turnOnLight()
EndIf
```

FIGURE 3.9 Pseudocode example of Boolean test conditions

Characteristics of data structures

A **data structure** is a method of organising data to allow particular operations to be performed on them efficiently; in this way, data structures are more complex than data types. The types of data structures used in Applied Computing are: arrays and associative arrays such as hash tables and dictionaries; stacks; queues; linked lists; fields; records; and classes.

Array

An **array** is a data structure that contains groupings of data. These elements are all of the same data type, such as character, numeric or Boolean. Arrays can also store groupings of other data structures, such as fields or records, or even other arrays. Arrays are very useful in programming, as they allow for related sets of data to be organised and ordered efficiently.

For example, a teacher might collect data related to the height of each student they teach and store this data as a one-dimensional array of floating point numbers. This is a much more efficient method of storing data than creating a variable to store each student's height, as it allows for faster sorting and searching.

The contents of an array are referenced using an index value – often an integer starting at 0. The way an array is stored means that each element has a set position within it; this allows for quick access to a particular element of the array, without the need to necessarily check every element.

Arrays can typically use the operations shown in Table 3.4.

Some programming languages, such as Python, allow for the use of lists as well as arrays. Lists behave like arrays but can contain more than one data type.

TABLE 3.4 Array data structure operations

Add or Append	+
Remove or Delete	−

```
arrayHeights ← [ 1.23, 1.35, 1.21, 1.61 ]
firstStudent ← arrayHeights[0]
secondStudent ← arrayHeights[1]
fourthStudent ← arrayHeights[3]
```

FIGURE 3.10 Pseudocode example of an array

Consider the array in Figure 3.10. In this example, firstStudent would contain the floating point number 1.23, secondStudent would contain 1.35 and fourthStudent 1.61.

In **pseudocode** (covered on pages 135–6), arrays are sometimes indexed starting at 1, but it should always be made clear in the pseudocode comments if this is the case.

Associative array

An **associative array** is a special type of array data structure, which consists of a collection of key and value pairs, where the key is unique and can be of any data type or structure. This makes it more flexible than an array with an integer index.

Associative arrays can typically use the operations shown in Table 3.5.

TABLE 3.5 Associative array data structure operations

Add or Append	assocArray[key].*add(value)* or assocArray[key] ← *value*
Remove or Delete	assocArray[key].*remove(value)*
Modify or change	assocArray[key].*change(value)* or assocArray[key] ← *value*
Lookup	assocArray[key]

For example, if a teacher were to collect student heights in a one-dimensional array, they would not be able to go back and find a particular student's height, as that data was not stored in the array. If the teacher used an associative array, they could store the data they collected as (key, value) pairs, where the key is the student's name, and the value is the height of that student. This would allow the teacher to look up any particular student to find out their height.

```
assocArrayHeights ← {}
assocArrayHeights["Paolo"] ← 1.23
assocArrayHeights["Shehara"] ← 1.35
assocArrayHeights["Phoebe"] ← 1.21
assocArrayHeights["Tuan"] ← 1.61

Print assocArrayHeights["Phoebe"]
```

FIGURE 3.11 Pseudocode example of an associative array

In the pseudocode example shown in Figure 3.11, Phoebe's height would be printed once the code was executed.

One limitation of an associative array that is important to know is that the key values are not organised or sorted in a consistent way. If the teacher wanted to print the heights of every student in sorted order, this would not be possible using just an associative array.

Dictionary

A **dictionary** is a synonym for an associative array. Associative arrays can also be referred to as 'maps' or 'symbol tables'.

Hash table

A **hash table** is a particular type of associative array: instead of (key, value) pairs, it uses (key, bucket) pairs, where the *bucket* (or *slot*) is a one-dimensional array. The *key* in the (key, bucket) index is computed using a **hash function** on the *value* that is to be inserted. Once this key has been computed, the value is then inserted into the bucket at the correct position.

The benefit of using a hash table over a regular associative array is that it allows for efficient searching. When dealing with very large amounts of data, it can take a very long time to find a particular element if every element has to be checked. Hash tables are faster as they move elements into smaller array 'buckets', meaning there are fewer items to look through when searching. The better the hash function, the faster the search, with a **perfect hash** function resulting in a hash table that has only one element in each bucket. This is quite rare, however, and **imperfect hash** functions are far more likely. An imperfect hash function is a function that possibly computes the same *key* index for more than one value. This results in a **collision**, which must be handled within a software program. Collisions are often handled by inserting all matching values from the hash into an array attached to the key index.

Consider Figure 3.12, a simple hash table that contains words from a book, where the hash function looks at the first character of the value to obtain the key.

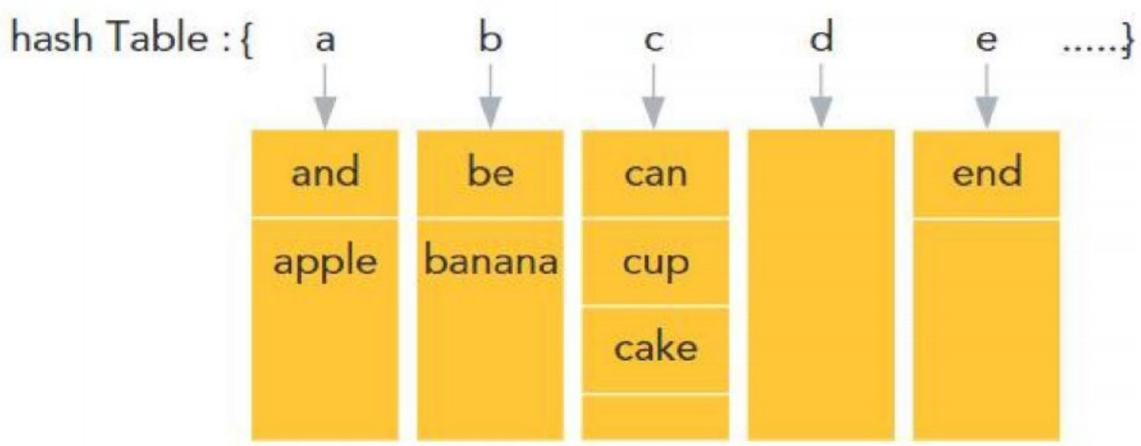


FIGURE 3.12 Hash table containing words from a book

If the next word to be inserted into the hash table is 'durian', the hash function would return the key as 'd' and the value of 'durian' would be inserted into the bucket with the 'd' key, as shown in Figure 3.13.

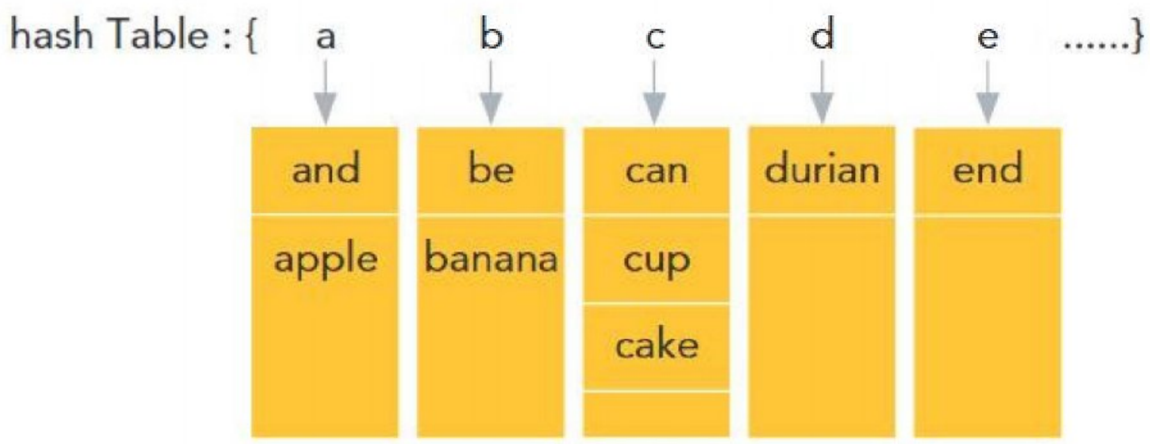


FIGURE 3.13 Updated hash table, adding 'durian'

If the next word after that is 'egg', the hash function would return the key as 'e'. A collision will occur, as the 'e' bucket already has 'end' in it, so the value 'egg' would need to be added to the end of the 'e' bucket. This is shown in Figure 3.14.

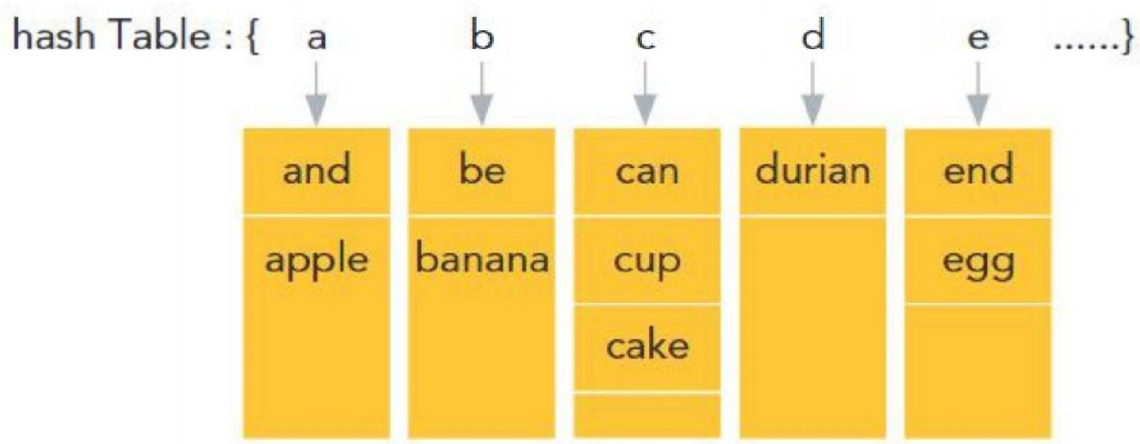


FIGURE 3.14 Updated hash table, adding 'egg'

Each time a collision occurs, the new value is added to the end of the bucket – notice that the arrays inside each bucket are not sorted.

Hash tables are most useful when there is a lot of data to store and it does not matter if they are unsorted. They are particularly useful for searching, as long as the hashing function does not create too many collisions.

Queue

A **queue** is a data structure that is best described using the analogy of a line at a cafeteria. The person at the start of the line is the next person who will be served, and any new person joining the line adds themselves to the end of the line.

Shutterstock.com/paul prescott

FIGURE 3.15 People in a queue

9780170440806

Much like people queueing for food, elements of data in a queue are inserted at the end of the queue (**enqueued**) and each element can only be accessed by taking it from the start of the queue (**dequeued**). This is referred to as **first in first out (FIFO)** access.

TABLE 3.6 Queue data structure operations

enqueue	queue.*enqueue(value)* Inserts a value at the end of a queue
dequeue	queue.*dequeue(value)* Removes and returns a value from the start of the queue
front	queue.*front()* Returns a value from the start of the queue, without removing it
empty	queue.*empty* Returns a Boolean value of True if the queue is empty, False if not
size	queue.*size()* Returns the number of elements in the queue

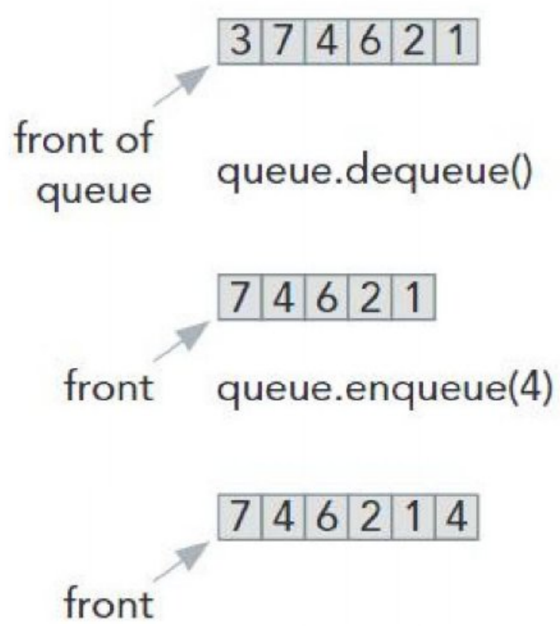


FIGURE 3.16 Queue operations on data using an array

THINK ABOUT APPLIED COMPUTING 3.10

Think of three situations in which you would need to use a queue in a software program.

Queues can contain any data type or structure, including other queues. In many programming languages, queues are implemented as arrays or linked lists. They are useful for implementing functionality to manage wait lists or access to shared resources (e.g. print queues) and handling multiprocessing software with parallel processing needs.

Stack

A **stack** is a data structure that is most often described using the analogy of dirty dinner plates stacked up waiting to be washed. As each plate is washed, it is removed from the top of the stack of dirty plates. The next dirty plate that was underneath it is now at the top of the stack. Any new dirty plate is placed on top of the current stack of dirty plates.

iStock/Getty Images Plus/Searsie

FIGURE 3.17 Stack of dirty plates

Just like in the plate analogy, elements of data in a stack are inserted at the top of the stack (**pushed**) and each element can only be accessed by taking it from the top of the stack (**popped**). This is referred to as **last in first out (LIFO)** access.

THINK ABOUT APPLIED COMPUTING 3.11

Think of three situations in which you would need to use a stack in a software program.

TABLE 3.7 Stack data structure operations

push	stack.*push(value)* Inserts an element on top of the stack, pushing all other elements down one place in the stack
pop	stack.*pop(value)* Removes and returns the last inserted element
top	stack.*top()* Returns the last inserted element, without removing it
empty	stack.*empty()* Returns a Boolean value of True if the stack is empty, False if not
size	stack.*size()* Returns the number of elements in the stack

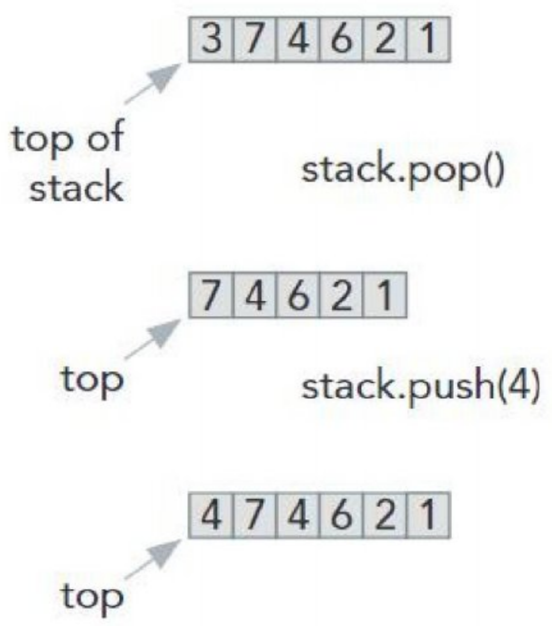


FIGURE 3.18 Stack operations on data using an array

Stacks can contain any data type or structure, including other stacks. In many programming languages, queues are implemented as arrays or linked lists. They are useful for implementing functionality, such as an undo operation in word processing software, or storing the history of visited web pages.

Linked list

A **linked list** is an ordered set of elements in which each element is connected to the next element in the list. This data structure allows data elements to be ordered into a sequence and allows for efficient insertion and removal of elements from any position in the sequence. For this reason, linked lists are particularly useful in sorting algorithms.

In a linked list, each element is referred to as a **node**. Each node contains a data element as well as the memory address of the next node in the linked list. This is typically referred to as a **pointer**.

Linked lists are often used to implement stacks, queues and associative arrays. The simplest type of linked list is referred to as a *singly linked list*; it only has the ability to traverse the list in one direction. Another common type of linked list is a *doubly linked list*, which allows for two-directional traversal, as each element in the linked list keeps track of the next element as well as the previous element in the list.

9780170440806

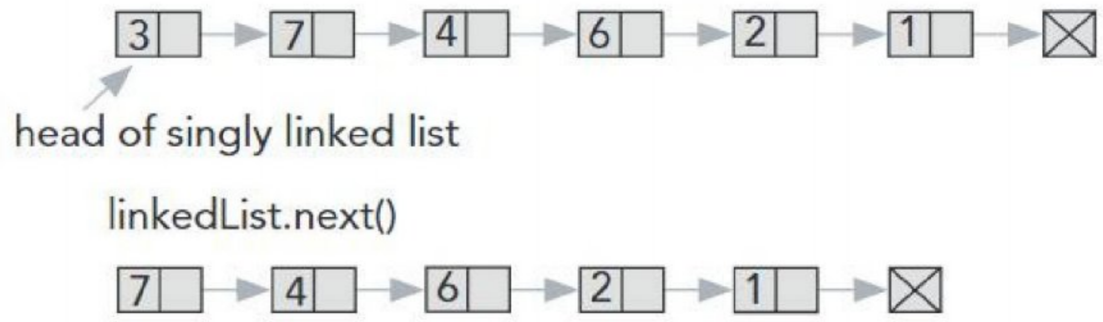


FIGURE 3.19 Example of a singly linked list of integers

TABLE 3.8 Data structure operations on a doubly linked list

Head	linkedList.*head()* Returns the first element in the linked list
Tail	linkedList.*tail()* Returns the last element in the linked list
Next	linkedList.*next()* Returns the next element in the linked list based on the current element
Previous	linkedList.*prev()* Returns the previous element in the linked list based on the current element

Linked lists make it very easy to add and remove elements at the start, middle and end of the list. Unlike arrays, whose index values must be shifted (increased) to make room for a new element being inserted, linked lists only need to change the pointer to the next element in the list, and the pointer to the previous element if it is a doubly linked list.

Record and field

A **record** is a basic data structure for collections of related elements. These elements may or may not be of the same data type. Most frequently, records are used in database systems but they are also commonly used in programming languages where they are referred to as *structs*.

A record consists of a number of **fields** that are typically fixed – that is, the fields do not tend to change once the record is defined and used. Each field has a name and each has its own data type.

For example, a customer record may contain fields such as firstName, familyName and dateOfBirth.

Records are most useful when variables in a collection are related to each other. It provides a logical method of ordering data within a program so that data can be accessed quickly.

In object-oriented programming languages, a record is essentially an object that has no object-oriented features, containing only collections of fields and values. Records and fields can also exist in some types of structured plain text files.

As records contain programmer-defined fields, there are no set operations that can be listed for the record data structure. Rather, there are common operations that can be performed on the record and the fields within it, such as assignment and comparison as well as adding or removing fields.

Class

Imagine a developer is writing a game in which players play golf.

In the game, the golf clubs that a player uses influence the outcome of the game. Players can upgrade them as they gain experience in playing courses. Each club carries information (properties) unique to that club, such as the quality of the club (its condition), the material of the grip, and the material of the club itself.

A golf club also has actions associated with it, such as the action of swinging the golf club, or the action of breaking the golf club because a player is frustrated. For those actions, a programmer would need to know the quality, grip material and club material of each and any golf club used by a player.

What happens if, each time a player uses the golf club, the quality decreases, the grip on the handle wears away a little, a new scratch is formed on the club and the player gets a little more frustrated?

Shutterstock.com/Yeongsik Im

FIGURE 3.20 Every swing of a club inluences the outcome in a golf game.

Writing a function in the source code of the game would not easily help a programmer keep track of all those things for each golf club being used, in particular if there are many players playing the game, all of whom who have many golf clubs. The source code would need to have a different variable for each club a player has, as well as a different variable for each player that plays the game. This becomes quite tedious to maintain within code as, to avoid hard-coding variables containing players and their clubs, the code would need to use multiple associative arrays that were synchronised with each other so that each index value matched a player.

Before object-oriented programming introduced classes and objects, there was no straightforward way to write the golf clubs in the golf game with a programming language that just used functions, subroutines and variables, unless the program saved information constantly to files and then read them back in for each golf club and each player. Reading and writing to files is very slow, and the game created would not be playable by more than a couple of players at a time.

Using a data structure called a **class** allows a programmer to solve the 'golf club' problem. A class is a programmer-defined data structure that exists in object-oriented programming languages. Classes group together conceptually similar functions and variables in one place. Classes work as templates for creating objects, which are instantiations of classes that exist in memory on the computer where the program is run. A useful way to describe a class by using an analogy is to think of it as a blueprint or an architectural design, such as one used to build a house. It describes everything that needs to be built to make a house, but is not a house itself. The actual house that is built using the design is an instance of the house blueprint. Classes work in the same way, as they describe all of the elements and components that are required by the object created from the class blueprint – this object is referred to as an instance of the class. The number of objects that can be created from a class is limited only by the amount of memory (RAM) in the computer system on which the program is running.

The purpose of a class is to create a template for objects with predetermined properties and behaviour. These templates can then be instantiated as objects or be used by another class in order to extend upon or change their behaviour. This allows for code re-use in programs where objects are similar to each other. In the golf club example, a programmer could extend on a base golf club class using inheritance to make woods, irons, wedges and putters.

Classes contain relevant variables, data types, data structures, methods and events.

Representing designs

Once a software requirements specification has been completed, it is important that considerable time is spent designing the software that is going to be written. This helps reduce the time and effort that goes into writing the software, as problems are normally resolved before any code has been written; there is nothing worse than needing to re-write code due to an issue that could have been resolved in the design stage!

Some common methods of representing designs are to use data dictionaries, mock-ups and pseudocode. Each of these methods has a different purpose in the design stage.

Data dictionaries

A data dictionary is used to plan the storage of software elements including variables, data structures, and objects such as GUI textboxes or radio buttons. A data dictionary should list every variable's name and data type or structure. It may also include the data's purpose, size, description, formatting and validation.

TABLE 3.9 Data dictionary

Name	Type	Format	Size	Purpose	Example
strCustomerID	String	XXX99	5	Customer ID	SMO40
dateDOB	Date	YYYY-MM-DD	Fixed	Date of birth	1992-12-28
sngSales	Single precision	$##,###.##	Fixed	Total amount spent	$12,456.78
boolClubMember	Boolean	Yes/No	Fixed	Is a member of the buyer's club?	Yes
strFamilyName	String	Xxxxxxxxxxxx	25	Customer family name	De Silva
strFirstName	String	Xxxxxxxxxxxx	15	Customer given name	Horatio
intAge	Integer	999	Fixed	Age in years	34
intMemYears	Integer	99	Fixed	Years a member	12

Data dictionaries are valuable when code needs to be modified later by other programmers and the purpose of a variable is unclear.

Mock-ups

If software will be used directly by people (rather than running hidden deep in the OS), it needs an interface – a place where people can control the program, enter data and receive output. A successful interface must be carefully designed to make it usable and clear.

To design an interface, use a **mock-up**, which is a sketch showing how a screen or printout will look. A mock-up should typically include the following features:

- the positions and sizes of controls such as buttons and scrollbars
- the positions, sizes, colours and styles of text such as headings and labels
- menus, status bars and scrollbars
- borders, frames, lines, shapes, images, decoration and colour schemes
- vertical and horizontal object alignments
- the contents of headers and footers.

In VCE Applied Computing it is not mandated that you use software to create your mock-ups. You may use software if you wish, but you may also create them by hand using pen and paper.

A mock-up can be considered successful if you can give it to another person and they can create the interface without needing to ask you questions.

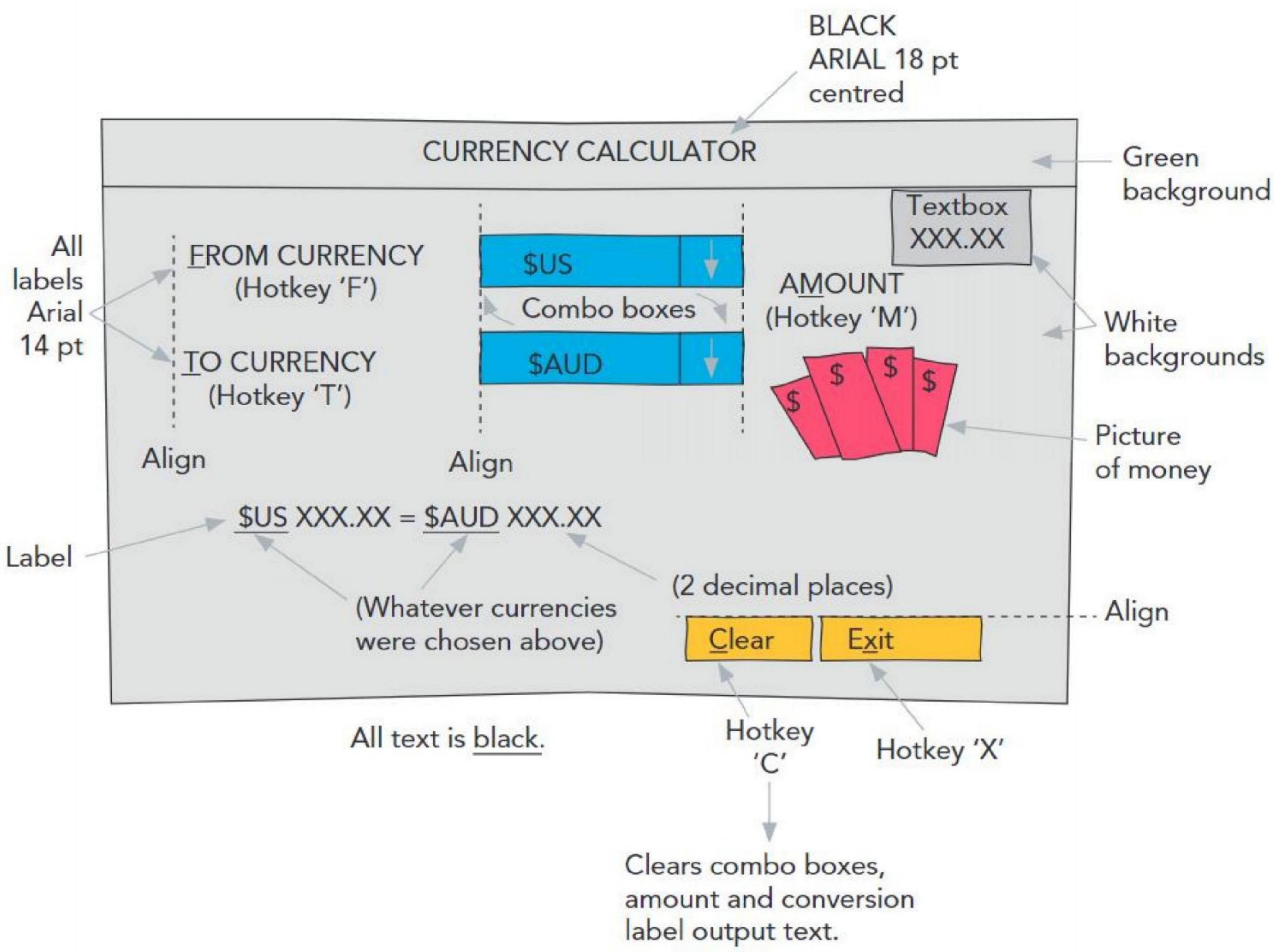


FIGURE 3.21 Mock-up of a screen interface

Pseudocode

Writing an algorithm in source code is slow. An algorithm written in source code also limits itself to use in only one compiler. Pseudocode, also known as Structured English, is a quick, flexible and language-independent way of describing a calculation strategy – halfway between English and source code. Once the algorithm is described in pseudocode, it can be converted into source code for any desired programming language.

A good algorithm can be extremely valuable and bring forth great change. A clever strategy can make software run twice as quickly or use half the amount of RAM. An ingenious idea can lead to the development of a program that was once considered impossible. For example, Google's PageRank completely changed the way the world searched the internet, and made billions of dollars for its inventors in the process. The invention of public key encryption finally cracked the age-old problem of how to encode and transmit secrets without having to also send an unlocking key that could be intercepted.

The pseudocode shown in Figure 3.22 determines if a year is a leap year.

THINK ABOUT APPLIED COMPUTING 3.12

What are the rules of pseudocode? Easy: there are none. As long as the intention of the calculation is clear, it is good pseudocode. If not, it is bad.

```
IF (year is divisible by 4 AND not divisible by 100)
OR (year is divisible by 4 AND 100 AND 400) THEN
     it's a leap year
ELSE
    it's not a leap year
ENDIF
```

FIGURE 3.22 Pseudocode to determine a leap year

In pseudocode, make sure that you specify assignment (the storage of a value) using the ← symbol rather than the equals sign (=) that is used in algebra and in most real programming languages; for example:

```
isLeapYear ~ True
```

The equals sign is reserved for logical comparisons, such as:

```
IF B=0 THEN SoundAlarm()
```

Common features found in pseudocode include:

- iterations/loops, such as **WHILE/ENDWHILE** and **FOR/NEXT**
- condition control structures, especially **IF/ELSE/ENDIF** blocks
- logical operators – **AND**, **OR**, **NOT**, **TRUE** and **FALSE**
- arrays, such as Expenses[3]
- associative arrays, such as Expenses["Gary"]
- records and fields, such as Customer.firstName, where Customer is the record and firstName is the field
- arithmetic operators (+ – * /) and the familiar order of operations, BODMAS.

Pseudocode punctuation and the names of key words are largely up to you provided it is clear what you mean; for example, it does not really matter if you prefer **WHILE/WEND** or **WHILE/ENDWHILE**.

To 'get data from keyboard', you could use **INPUT**, **GET**, **FETCH** or another keyword. To read data from a disk file, you could choose **INPUT**, **GET**, **READ** or something else. To avoid ambiguity, you could explain your pseudocode's conventions using comments within the pseudocode. Comments can be prefixed with a hash (#) or two forward slashes (//), or included in braces/curly brackets ({}).

```
# GET reads the keyboard.
# READ loads data from a disk file.
# DISPLAY shows output on screen.
# WRITE saves output to a file.
DISPLAY "What is your name?"
GET UserName
OPEN FILE "Users.txt"
READ data for UserName
IF new data exists THEN
    WRITE new data to file
ENDIF
```

FIGURE 3.23 Pseudocode to write usernames to a file

Programming naming conventions

A **naming convention** is a set of rules that is used when creating variables, subroutines, functions, methods, objects, classes and so on. in programming source code as well as in internal documentation. It is a consistent and meaningful way of labelling each of these elements so that they are easily read and understood. The most useful naming conventions allow a programmer to know the purpose of an element and, if relevant, its data type or structure.

Each programming language tends to have a set of language-specific conventions to follow when naming elements. For example, Microsoft.NET, Python and Swift all use a convention called 'camel case' within their code. Two other common naming conventions are 'snake case' and 'Hungarian notation'.

Camel case

Camel case, also known as camel caps, uses compound words and phrases as a naming convention, where each word after the first begins with a capital letter – for example, camelCase. No spaces or punctuation are included when naming variables and other elements using camel case. While multi-word variable and function names are useful, it is important that these are as short as possible while remaining meaningful. Often, this is achieved through abbreviating some of the words in the compound phrase. A variable named firstNameOfEmployeeWhoIsPartTime is not as effective as one named firstNameEmplPT as it is too long; writing code using the longer version of this variable would be very tedious.

Snake case

Snake case is very similar to camel case but, instead of compounding phrases into a single word without spaces, it joins each word in the phrase using an underscore – for example, snake_case. All words in snake case are lower case. Many programmers prefer snake case over camel case as it is easier to read the variable and function names with the underscores separating each word.

Hungarian notation

Hungarian notation, in particular a variety referred to as Systems Hungarian, is similar in style to camel case, in that it compounds words and phrases, without spaces, and each word after the first begins with a capital letter. Hungarian notation additionally adds a *prefix*; that is, an initial letter sequence, before the name of the variable. This letter sequence represents the data type or structure of the variable – for example, iNumEmployees to represent the number of employees as a whole integer. This can be useful when programming using programming languages that are not dynamically typed, as it immediately tells a programmer what data type or structure they are handling when they read the variable name.

TABLE 3.10 Some Hungarian notation prefixes

iNumEmployees	Integer
arrEmployees	Array
strEmployee	String
chEmployeeGender	Character
fpEmployeeHeight	Floating point
fnEmployeeFunction	Function
clCustomer	Class
objCustomer	Object
mGetAllCustomers	Method

Hungarian notation is a flexible naming convention, and many programmers create user-defined prefixes that are meaningful for the programming language they are using, such as some of those used in Table 3.10. For example, while classes and objects were not considered in the initial Hungarian notation prefixes, many programmers use 'cl' as the prefix to represent a class and 'obj' for an object. With the increasing prevalence of dynamically typed programming languages, however, most programmers prefer to use either the camel case or the snake case naming convention over a modified Hungarian notation.

File naming conventions

In much the same way as having consistent naming conventions in source code is important, it is also important that projects use a consistent naming convention for all files and folders related to the project. This includes logically named folders, subfolders, files and backups of files. This allows you to stay organised and makes it easy to identify files when you are looking for them.

Having effective file naming conventions takes time and effort, as well as careful pre-planning. While there are no set standards for naming files, you should take into account the following:

- Include the right amount of information in the name of the file. You should not have filenames longer than 30–40 characters, and some file systems have a maximum length for filenames (including folder names), so shorter is better than longer. At the same time, if a filename is too short, it will not provide you enough information to determine the purpose and use of the file.
- Use meaningful abbreviations when reducing the number of characters in a filename or folder name. For example, 'custOrderHistory' is more meaningful than 'cOrderHistory'.
- For portability reasons, filenames should contain little, if any, punctuation. Often, an operating system will disallow punctuation in filenames. For example, Windows systems do not allow asterisks (*) or question marks (?) in filenames, macOS systems do not allow the colon (:) and Unix systems do not allow slash (/), semicolon (;) or pipe (|). Also avoid using spaces, as many operating systems and programming languages require special handling to read and write to files with spaces in them. Typically, the underscore (_) and hyphen (-) are safe to use as punctuation, but any other punctuation should be used with caution.

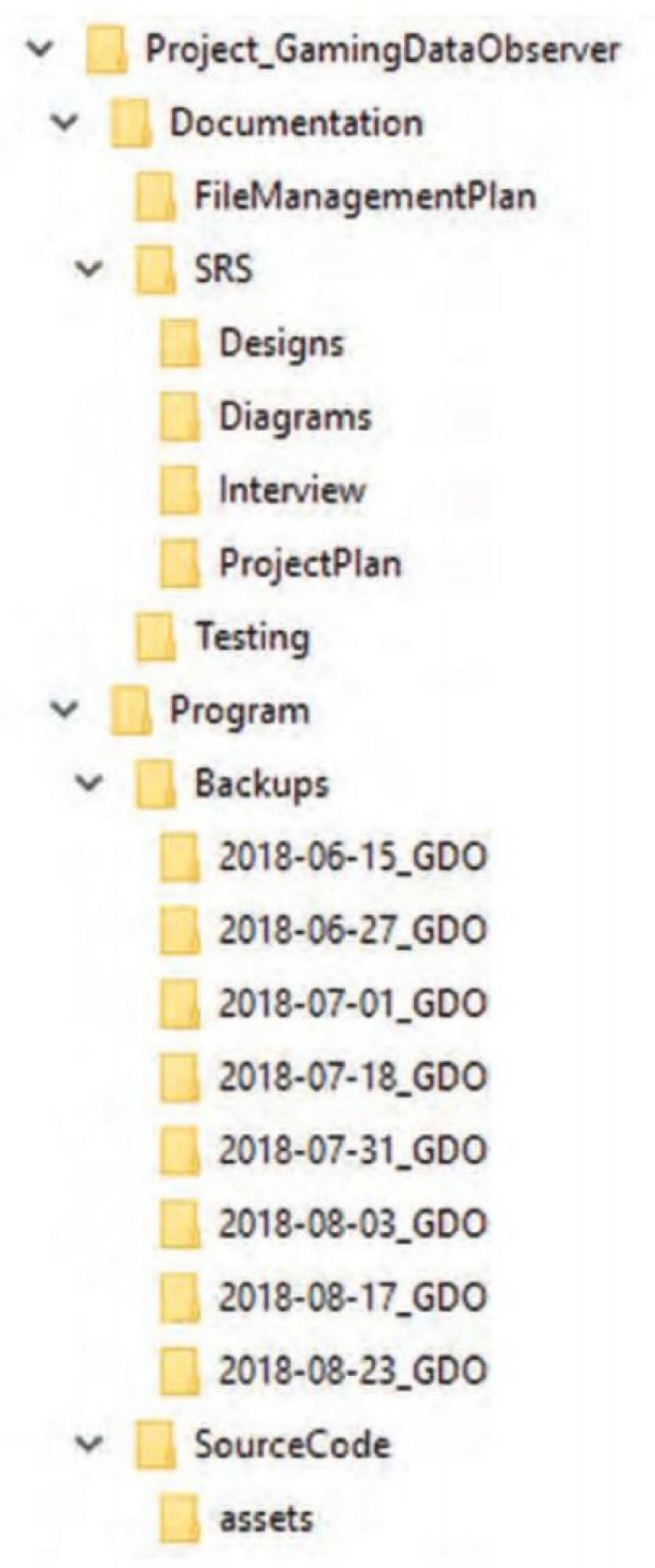


FIGURE 3.24 Example of file and folder naming conventions being used in a project

The default maximum length of a path (filename and folder names) on the Windows operating system running is 260 characters. In Windows 10, users can increase this value by changing the NTFS Group Policy.

9780170440806

- Begin your filenames with more general components, such as a date marker, and then include a more specific reference to the file.
- Dates on files should be in yyyy-mm-dd format, as this allows the files to be sorted in chronological order on all file systems.
- Separate your source code from your design documentation. You should keep your source code and your design documentation in subfolders so they are kept separate from each other.
- Use subfolders when relevant. For example, it is useful to separate images from source code, as this not only keeps files and folders orderly, it also makes it easier to reference images in the source code if it is required.
- If appropriate, include version numbers.
- It is useful to document the file naming conventions you have chosen to use for your files and folders. This is particularly helpful when maintaining software as it can be easy to forget the meaning of abbreviations and acronyms.
- Ensure that file naming conventions are followed consistently; there is no point having a convention if it isn't consistently applied.

3 CHAPTER SUMMARY

Essential terms

32-bit computer system a computer system with 32 bits of memory addresses

64-bit computer system a computer system with 64 bits of memory addresses

application software used to perform work or complete larger tasks; examples of popular applications include Microsoft Word and Excel, Adobe Photoshop and Mozilla Firefox browser

array an indexed set of related elements; all elements within an array have a numeric index and are of the same data type

associative array a collection of (key, value) pairs where each key is unique in the collection

Boolean a logical data type; Boolean data can hold one of only two possible values, true or false

camel case a naming convention used in programming whereby each word or abbreviation in a phrase after the first word begins with a capital letter; there are no spaces or punctuation using camel case

central processing unit (CPU) the 'brain' of a digital system; the handler of most of a system's data manipulation

character a data type representing a single letter, number, symbol, punctuation mark or space

character encoding code that allows a computer program to interpret binary digits (0s and 1s) into meaningful units representing characters and numbers. ASCII, UTF and Unicode are types of character encoding.

class a program code template for creating objects in object-oriented programming languages

collision when two different input values to a hashing function output the same hash value

communication hardware hardware that transmits data between computers and networked devices

compiler a program that turns source code into machine language that can be executed by a computer processor

complex instruction set computing (CISC) a type of microprocessor design that contains large sets of computer instructions; intended to compute complex instructions in an efficient way

constants fixed (unchanging) values during a program's execution, such as the value of pi, or the number of Australian states and territories

constraint a restriction on what can and cannot occur in the creation of a software solution, external to the solution itself

data structure a method of organising data to allow particular operations to be performed on them efficiently

data type different forms that variables and data may take, which determine the data that a variable may contain, and how the data or variable may be manipulated

debugger a tool used to identify errors in computer software

dependent task a task that requires one or more other tasks to have been completed in order to begin

dequeue to remove and return a value from the start of a queue

dictionary an associative array, also referred to as a map or symbol table

digital systems a elements such as hardware and software that, when connected, form a network

9780170440806

driver a computer program that controls a particular device that is attached to, or part of, a computer

economic constraint a constraint on the creation of a software solution that relates either to time or budget, or to both

editor specialised application software for creating human-readable programming instructions

enqueue to insert a value at the end of a queue

field a single data item in a record; e.g. FamilyName

first in first out (FIFO) a type of access to queued data, whereby the first element in a queue is the first element out of the queue

fit for purpose well suited for a role or purpose

floating point number computer representation of real numbers, with decimal places

functional requirements requirements of a digital solution that are directly related to what the solution will do

Gantt chart a project management tool for displaying project schedule information

graphical user interface (GUI) a form of user interface that allows users to interact with an application or piece of software

graphics processor unit (GPU) a very fast and expensive processor specifically designed for high-speed image processing in graphics cards

hash function a function that takes a key value and returns another, related, value that is normally smaller than the original value

hash table a data structure that uses a hash function to map keys to values by computing an index that is related to, but smaller than, the initial key

high-level programming language a programming language that is more or less computer-independent, typically using natural language elements

Hungarian notation a naming convention used in computer programming, in which the name of the variable or function determines its purpose and its data type or structure

imperfect hash a hash function where two or more keys can be computed to have the same hash index

index position within an ordered list, such as an array

input devices instruments and peripherals, such as keyboards, that allow users to give data and commands to software and the operating system

integer a data type representing whole positive and negative numbers; a number without a fractional or decimal component

integer overflow a condition occurring as a result of a mathematical operation where the output exceeds the maximum or minimum integer value that can be stored on that computer system

Integrated Development Environment (IDE) software that provides tools to programmers to aid in programming, such as source code editing, syntax highlighting, code completion, debugging aids, or tools to help construct a user interface

interface within software, the place where people control the program, enter data and receive output

interpreter a computer program that directly executes source code without needing to have it compiled beforehand

last in first out (LIFO) a type of access to stacked data, whereby the last element in a stack is the first element out of the stack

linked list a data structure containing an ordered set of elements where each element is connected to the next element in the list

low-level programming language a programming language that is heavily reliant on the architecture of a particular computer, with commands closely linked to processor instructions

maintainability the extent to which a solution is easy to look after once it has been put in place

milestone the achievement of a significant stage in a project; a milestone has zero time duration

mock-up a sketch showing how a screen or printout will look, which is used to aid in the design of an interface

naming convention an agreed set of rules by which to name files, folders and source code elements such as variables, functions, classes, methods and objects

NAND RAM a type of non-volatile storage technology that does not require any power to store data

9780170440806

network-attached storage (NAS) dedicated file storage that allows multiple users and clients to access data on a centralised disk

node an element in a linked list; contains a data element and a pointer to the next node in the list

non-functional requirements characteristics that the user or client would like the solution to have; often tied to solution constraints

non-technical constraints factors that may limit or restrict solution development, specifically related to legal, social, usability or economic factors

numeric a data type consisting of whole numbers, referred to as integers, and decimal numbers, referred to as floating point numbers

object any instantiated class that a program can inspect and/or change, in terms of appearance, behaviour or data

operating system (OS) the software program that manages a computer's hardware and runs programs

output devices instruments and peripherals, such as printers and monitors, that display information from a computer in human-readable form

perfect hash a hash function where no two keys can be computed to have the same hash index

platform a combination of operating system (OS) and central processing unit (CPU)

pointer a variable that stores the memory position of another variable's value

pop to remove an element of data from the top of a stack, moving all stack elements left over up one place

portability the extent to which a solution can easily be used in different operating environments

ports physical connectors (sockets) for cables

primary storage random-access memory (RAM), which provides storage for data, information and software during program execution

processing hardware (CPU, GPU) hardware that runs the operating system, utilities and applications

project plan a formal document that is used to guide a project's timeline, allowing for the coordination and monitoring of tasks, including their sequencing and the amount of time required to complete each task

properties characteristics such as width, colour, visibility

pseudocode code that designs algorithms in a clear, human-readable, language-independent format

push an element of data inserted at the top of a stack, moving all current stack elements down one place

queue a 'First In First Out' data structure, storing elements to be processed in order

random-access memory (RAM) a type of computer memory that can be accessed randomly; it is most often volatile memory that is lost if power is removed

record an indexed set of related elements; all elements within a record have a field index and may be of different data types

reduced instruction set computing (RISC) a central processing unit (such as ARM) that has smaller instruction sets than complex instruction set computing (CISC) CPUs

reliability how much a solution can be depended on to function as designed, and for how long

robustness how well a software solution responds to errors that occur when the software is being used

scope boundaries or parameters of the solution – what it will do and what it will not do

scope creep when a client changes the scope of a project (by increasing it) during the life cycle of the project

secondary storage (HDD, SSD) permanent storage

slack time the length of time that a task can run over time before it affects other tasks

snake case a naming convention used in programming in which each word or abbreviation in the middle of a phrase is joined to the one before it using an underscore

source code the human-written and human-readable version of a program

stack a 'Last In First Out' data structure

9780170440806

storage structure places in memory holding data that is being used by a program; includes variables, arrays, dictionaries and objects such as GUI objects

string a data type representing a set or sequence (collection) of characters

syntax the structure of statements in a programming language

system software tools software tools used by a computer to manage hardware and run the user's programs; for example, the operating system (OS), device drives and communication protocols

task any activity within a project timeline that needs to be completed within a defined period of time

technical constraints constraints related to the hardware and software available for a project

thin client a lightweight computer that has been optimised to work with a server-based computing environment; the server does the majority of the processing and computing

uptime time during which a machine, solution or application is operational

usability the extent to which a system is easy to learn and use

utilities usually small, single-purpose software tools that do a specific job or add functionality to an operating system

variable a method of storing and labelling data to be referenced and manipulated in a computer program

Important facts

1. **Information systems** comprise people, data, processes and digital systems.
2. **Digital systems** are made up of networks, protocols, application architecture patterns, software and hardware.
3. **Hardware** requires software instructions to control it.
4. **Software** requires hardware to carry out instructions.
5. **Hardware** includes input devices, output devices, processing hardware, storage hardware and communication hardware.
6. The **central processing unit (CPU)** is the core processing hardware of a computer system.
7. **Reduced instruction set computing (RISC)** CPUs have smaller instruction sets than **complex instruction set computing (CISC)** CPUs.
8. **Graphics processor units (GPUs)** are very fast processors designed for high-speed image processing in graphics cards.
9. **Storage hardware** comes in two main types: primary storage and secondary storage.
10. **Primary storage** is a computer's random-access memory (RAM).
11. **Secondary storage** is a computer's hard disk drives (HDDs), solid state drives (SSDs) and/or network-attached storage devices (NASs).
12. **HDDs** are a cheap and reliable long-term storage medium with large capacity.
13. **SSDs** are a fast and efficient storage medium that is more expensive than HDDs.
14. **NAS devices** are networked sets of HDDs or SSDs that provide for faster speed, higher capacity and data protection via redundancy.
15. **Communication** is used for sending and receiving data and information.
16. **Ports** are physical sockets or connectors that carry data between a computer and external devices such as peripherals.
17. **Software drivers** are computer programs that control internal and external computer devices.
18. **Software** controls computing devices to process data.
19. **System software tools** are used by a computer to manage hardware and run programs.
20. **Applications** are used to perform work or complete larger tasks.
21. **Utilities** are single-purpose software tools that complete specific jobs or add functionality to an operating system.
22. An **operating system** is system software that controls a computer's hardware and runs application software.
23. **Programming languages** are used to give instructions to computer processors.

9780170440806

24 **High-level programming languages** are languages that are computer-independent and use natural language elements.

25 **Low-level programming languages** are reliant on the architecture of a particular computer and have comments tightly linked to processor instructions.

26 **Source code editors** come with specialist features such as syntax highlighting, code completion and line numbering.

27 **Compilers** convert source code into executable programs.

28 **Interpreters** directly execute source code at run-time without needing pre-compilation.

29 Code that has been compiled for one platform will not work on another platform unless it has been re-compiled for that platform.

30 An **Integrated Development Environment (IDE)** is used by programmers as it provides tools to aid in programming.

31 A storage structure is a location in RAM where data is stored during the execution of a program.

32 Two main **data storage structures** are variables and constants.

33 **Variables** store values that can be changed by a program.

34 **Constants** store values that are fixed and unchanging during a program's execution.

35 A **project plan** helps track and monitor the progress of a project.

36 **Tasks** are activities within a project that take time and resources.

37 **Task dependencies** often exist within a project, where one task cannot begin until one or more other tasks have been completed.

38 A **milestone** is a task that takes no actual time but is a significant event within the project timeline.

39 The **scope** of a solution outlines its boundaries or parameters.

40 **Scope creep** is when a client changes the scope of a project during its life cycle.

41 **Solution requirements** consist of functional and non-functional requirements.

42 **Functional requirements** are directly related to what a solution will do and are usually described in terms of the required inputs, the sequence of operations that will be performed and the expected output.

43 **Non-functional requirements** are other requirements that the user or client would like the solution to have but that do not affect what the solution does.

44 Non-functional requirements are typically referred to as **quality requirements** as they involve measurable criteria that can be used to judge a system.

45 Non-functional requirements are often tied to the **constraints** of a system and can be categorised in terms of usability, reliability, portability, robustness and maintainability.

46 **Usability** relates to how easy a system is to learn and use, typically described in terms of efficiency and effectiveness.

47 **Reliability** relates to how much a piece of software can be depended upon to function as designed.

48 **Portability** relates to how easily software can be used in different operating environments.

49 **Robustness** relates to how well a solution responds to errors that occur when it is being used; does it perform correctly in every situation encountered by a user?

50 **Maintainability** relates to how easy it is to look after software once it is being used.

51 **Constraints** are factors that may limit or restrict solution requirements.

52 Typically, **constraints** involve economic, technical, social, legal and usability factors.

53 **Economic constraints** include time and cost.

54 **Technical constraints** relate to the hardware and software that is available for a project.

55 **Non-technical constraints** relate to areas other than hardware or software, such as usability, social and legal constraints.

56 **Usability constraints** involve the user's level of expertise.

57 **Legal requirements** may restrict features in software.

58 **Social constraints** are restrictions that are imposed by society but are not necessarily related to legality.

9780170440806

59 A **data type** is a method of classifying a variable to determine the data the variable can contain as well as how that variable can be manipulated.

60 **Data types** are consistent across all programming languages.

61 **Numeric data types** consist of whole numbers, referred to as integers, and decimal numbers, referred to as floating point numbers.

62 All **numeric data types** can have mathematical operations performed on them, following BODMAS order of operations rules.

63 **Numeric data types** can undergo comparisons, such as less than, greater than, and equal to.

64 **Integers** are represented in computer systems as a group of binary digits.

65 The **maximum and minimum values** of an integer depend on the computer architecture used to run the program.

66 **Floating point numbers** are the computer representation of real numbers; that is, numbers that allow for decimal places.

67 A **character data type** consists of any single meaningful unit such as a letter, number, punctuation mark or symbol.

68 **Character encoding** is a way in which a computer program can translate binary data into meaningful characters.

69 A set or sequence of characters is a **string**.

70 **Strings** are often implemented in programming languages as an array of characters.

71 **Boolean data types** have only two possible values: 1 and 0; these are often referred to as True and False, respectively.

72 A **data structure** is a method of organising data to allow particular operations to be performed on them efficiently.

73 An **array** is a data structure that contains groupings of data, traditionally of the same data type or structure.

74 An **associative array** is a special type of array data structure.

75 A **dictionary** is a type of associative array.

76 A **hash table** is an associative array that computes indexes using a hash function based on the value to be inserted.

77 A **queue** is a data structure where elements are processed on a first in first out basis.

78 A **stack** is a data structure where elements are processed on a last in first out basis.

79 A **linked list** is an ordered set of elements where each element is connected to the next element in the list using pointers.

80 A **record** is a data structure for collections of related elements that may or may not be of the same data type or structure.

81 Records consist of a number of **fields**; each field has a name and its own data type.

82 **Classes** group together conceptually similar functions and variables in one place to work as templates for creating objects; a class is like a blueprint or an architectural design.

83 **Objects** are instantiations of classes that exist in memory on the computer where the program is run; the number of objects that can be created from a class is limited only by the amount of RAM the computer system has.

84 Some common methods of representing designs are to use data dictionaries, mock-ups and pseudocode.

85 A **data dictionary** is used to plan the storage of software elements including variables, data structures and objects.

86 A **data dictionary** should list every variable's name and data type or structure, it may also include that data's purpose, source, size, description, formatting and validation.

87 A **mock-up** is a sketch showing how a screen or printout will look.

88 **Pseudocode** is a quick, flexible, language-independent way of describing a calculation strategy or algorithm.

89 A **programming naming convention** is a set of consistent and meaningful labelling rules for variables, functions, methods and other aspects of programming source code.

90 A **file naming convention** is a set of consistent and meaningful labelling rules for files and folders.

9780170440806

TEST YOUR KNOWLEDGE

Review quiz

Functions and capabilities of hardware

1 What is an input device? Provide an example.

2 What is an output device? Provide an example.

3 Explain how a CPU differs from a GPU.

4 What is the difference between primary storage and secondary storage?

5 Outline the differences between a HDD and a SDD.

6 What is communication hardware used for?

7 How many bytes are in the following?

  a 128 GB RAM

  b 2 TB HDD

  c 256 MB RAM

  d 7 GB SSD

Functions and capabilities of software

8 What are system software tools? Provide an example.

9 Explain how applications and utilities differ.

Programming and scripting languages

10 Name three programming languages that were not mentioned in this chapter.

11 What is the difference between a high-level programming language and a low-level programming language?

Software development tools

12 What are the essential tools used to develop software?

13 Provide an example of source code in a programming language of your choice.

14 What is the difference between a platform and an operating system?

15 Why would a programmer choose to use an IDE?

Storage structures

16 What is a storage structure? Provide examples as part of your explanation.

17 What is the difference between a variable and a constant? Provide examples as part of your explanation.

9780170440806

TEST YOUR KNOWLEDGE

Project plans

18 Outline some of the reasons why project plans are important when creating software solutions.

19 What is the difference between a task and a milestone?

Determining scope

20 Define scope.

21 What is scope creep?

Functional and non-functional requirements

22 What is the difference between a functional requirement and a non-functional requirement?

23 Categorise the following as functional or non-functional requirements. If the requirement is non-functional, provide its type.

- **a** Customer data should be saved to a file.
- **b** Users should only need an hour to be trained fully on the system.
- **c** Uptime should be 95% during summer months.
- **d** The system should reject invalid dates, such as 30 February.
- **e** Jobs should be able to be loaded into a timetable.
- **f** The system should stay running, with fewer than 5 hours per month spent maintaining the system.
- **g** All orders should be printed in a report at the end of a work day.

Constraints

24 When are solution constraints determined?

25 What do solution constraints typically involve?

26 Categorise the following solution constraints according to their type.

- **a** The system must be completed by the end of the year.
- **b** Data entry can occur via touch screen and joystick controller.
- **c** The user interface must be usable by colourblind users.
- **d** All data covered under copyright laws must by encrypted.
- **e** New system equipment must cost no more than $10 000 in total.

9780170440806

TEST YOUR KNOWLEDGE

Characteristics of data types and data structures

27 Select the most appropriate data types and structures for the following data.

- **a** Inventory: { “socks”: 30, “tables”: 34, “swing sets”: 12, “curtains”: 4 }
- **b** 137
- **c** False
- **d** 5.99
- **e**]
- **f** potato
- **g** 0400999888
- **h** 1
- **i** Customer: { Dillon, Corp, 25/07/1993, 123 Fourth Street, Fifthsville, VIC, 3000 }
- **j** +
- **k** Players: { Wajee, Sharon, Penny, Lola, Darius, Shehara, Ling, Rajkumar }

28 What is the difference between a dictionary and an associative array?

29 When would you use a record rather than an array to store a collection of related values?

30 A restaurant would like a new ordering system to process orders that are sent to the kitchen so that orders are cooked in the order in which they are received. Which data structure is the most appropriate to store kitchen orders?

31 An application to store library books on a hold list has been written so that when a new library book is added to the hold list, it is added so that it is borrowed next, in front of any other book on the hold list. Which data structure is the most appropriate to use for the hold list?

Representing designs

32 What is a data dictionary and what is its purpose?

33 What is pseudocode and what is its purpose?

34 What is the difference between ← and = in pseudocode?

9780170440806

35 State the values that will be returned or displayed in the following examples of pseudocode.

a
```
stkPokemon ← { "pikachu", "ditto", "abra" }
{ the first element is the top }

stkPokemon.pop()
stkPokemon.push("eevee")
stkPokemon.pop()
stkPokemon.pop()

DISPLAY stkPokemon.top()
```

b
```
qPokemon ← { "pikachu", "ditto", "abra" }
{ the first element is the front }

qPokemon.dequeue()
qPokemon.enqueue("eevee")
qPokemon.dequeue()
qPokemon.dequeue()

DISPLAY qPokemon.front()
```

Naming conventions

36 Name two types of programming naming conventions.

37 Why are programming naming conventions important in source code?

38 Why are file naming conventions important for projects?

9780170440806

APPLY YOUR KNOWLEDGE

Programming practice

1 Write a program that asks a user for their name. Once they have entered their name, the program should say hello to them.

2 a Write a program that asks a user for their date of birth. From their input, calculate their current age in years.

  b As an extension, calculate their exact age in years, months and days.

Programming task: Guessing game

Design and develop a program that will play a game with a human. The human thinks of an integer between 1 and 100. Your program will use the most efficient strategy possible to guess the number. After each guess, the human must confirm whether the guess was correct, or higher or lower than the secret number.

a Write pseudocode to describe the processing strategies involved in developing the program.

b Describe the functional requirements of the program.

c Design the program's interface with a mock-up.

d Develop the solution, using good naming conventions and internal documentation.

e Test the solution and fix all bugs.

9780170440806

CHAPTER

4 Developing software

KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

Approaches to problem solving

- processing features of a programming language
- characteristics of internal documentation
- formatting and structural characteristics of input and output such as file formats
- testing and debugging techniques to ensure software solutions meet requirements such as test tables and test data
- techniques for evaluating the efficiency and effectiveness of software solutions

Reproduced from the VCE Applied Computing Study Design (2020–2023) © VCAA; used with permission.

FOR THE STUDENT

This chapter relates to VCE Applied Computing Unit 1, Area of Study 2: Programming. It introduces basic programming concepts such as variables, instructions and control structures. You will learn about functions, classes and methods as well as appropriate methods of internal documentation. Later in the chapter, you will begin to understand testing and debugging techniques and the tools programmers use to help debug software, such as creating test data and constructing trace tables. You will continue to learn about the specific programming language that you will use to develop a response to a design brief for Unit 1, Outcome 2. This will involve the creation of a project plan as well as responding to a design brief in relation to functional and non-functional requirements, constraints and scope. You will also need to design and apply suitable testing and debugging techniques, and evaluate the effectiveness of your solution and project plan.

FOR THE TEACHER

This chapter continues from Chapter 3, introducing students to the core features of programming that are required for Unit 1, Outcome 2. It does not assume knowledge of, or use source code from, any actual language. Instead, pseudocode is used where relevant. Unit 1, Outcome 2, requires that students apply all stages of the problem-solving methodology to design, develop and evaluate a software solution in response to a design brief provided by their teacher. This design brief should include functional and non-functional requirements, constraints and factors that determine scope. It should be written to allow students to draw on knowledge of basic programming concepts, such as storage, logic, loops and calculations. Students should complete a project plan as part of this outcome. Project plans are covered in Chapter 3.

Shutterstock.com/metamorworks

9780170440806

Processing features of a programming language

All modern programming languages share the same types of features, such as the ability to store data within variables and execute instructions within control structures. While the syntax of individual programming languages may vary, the fundamental logic behind these processing features remains the same.

Variables

Variables are methods of storing data so that they can be retrieved later within a program. Without variables, it would be impossible to reference data once it has been stored in memory. A variable is typically used to store a data type or structure, but it can also be used to store a **pointer** to a function or method. Variables should be named appropriately, following consistent naming conventions.

Instructions

An **instruction** is a line of code that can be executed by a **compiler** or **interpreter**. There are two types of instructions in programming: definitions and statements.

A **definition** is an instruction that assigns a value to a variable. The first line in Figure 4.1 is an example of a definition.

A **statement** is a single line of code that, when executed, performs a single action. The last line Figure 4.1 is an example of a statement.

```
a ˜ 7

PRINT a
```

FIGURE 4.1 Pseudocode example of two types of instructions

Control structures

There are three fundamental control structures in programming: sequences, conditional statements and iterations.

Sequence

A **sequence** is a set of instructions that executes line by line, a little bit like a recipe; every line of code in the sequence is run in the order that it is written.

In the algorithm shown in Figure 4.2, a sequence of code runs that asks a user for their name, reads the name as input and then greets the user by name. Each line of code in the algorithm is run, in order, only once.

Conditional statements

Selective control structures (**selections**), such as **conditional statements**, allow the application to make decisions during runtime, dependent on given conditions. The code within a conditional statement can contain instructions, nested conditional statements and iterative structures.

```
ALGORITHM askName()
BEGIN
     PRINT "What is your name?"
     INPUT name
     PRINT "Hello, " + name + ". Nice to meet you."
END
```

FIGURE 4.2 Pseudocode example of a sequence of instructions

Conditional statements are **Boolean**, in that they are run based on the result of a condition being evaluated as either True or False. If the condition evaluates as True, the code within that condition is executed. If it evaluates as False, it is not.

The simplest type of conditional statement is one that tests against a single condition, such as seen in Figure 4.3.

```
ALGORITHM printPositive()
BEGIN
     INPUT intNumber

     IF intNumber > 0 THEN
         PRINT "The number is positive."
  °  ENDIF
END
```

FIGURE 4.3 Single condition selection control structure

Alternative execution

Another form of conditional statement involves an **alternative execution**. This means that if the condition is not met, an alternate block of code will be run. For example, in Figure 4.4, a user is asked if they like pie. If they respond with 'yes', they receive a happy comment. Alternatively, if the user does not input 'yes', a sad comment will be printed instead. Note that the user does not need to input 'no' for the sad comment to be printed – any input aside from 'yes' will execute the alternative code.

```
ALGORITHM likePie()
BEGIN
     PRINT "Do you like pie?"
     INPUT likePie

     IF likePie = "yes" THEN
          PRINT "Hooray!"
     ELSE
          PRINT "That makes me sad. :("
     ENDIF
END
```

FIGURE 4.4 Conditional/selection control structure with alternative execution

As you can see in comparing Figure 4.3 with Figure 4.4, the use of ELSE in a conditional statement is optional.

Conditionals with more than one logical expression

A conditional statement is not limited to testing only one logical expression; in fact, the number of logical expressions a single conditional statement can test is theoretically unlimited. Figure 4.5 contains pseudocode that uses the **logical operator** AND within a single conditional statement to check for two conditions to be simultaneously true.

9780170440806

```
ALGORITHM likePieCake()
BEGIN
    PRINT "Do you like pie?"
    INPUT likePie
    PRINT "Do you like cake?"
    INPUT likeCake

    IF likePie = "yes" AND likeCake = "yes" THEN
        PRINT "Hooray!"
    ELSE
        PRINT "That makes me sad. :("
    ENDIF
END
```

FIGURE 4.5 Simple conditional with more than one logical expression

When writing conditions with more than one logical expression, it can be useful to construct a **truth table** to make sure that no logic errors have been included. Truth tables uses Boolean algebra to test each combination of values in a condition. For example, Table 4.1 contains a truth table to check the logical expressions in the pseudocode from Figure 4.4. Because the logical operator connecting the two conditions is 'AND', both conditions need to be true for the conditional statement as a whole to evaluate as true, which is shown when only a single case in the truth table evaluates the whole condition as True.

TABLE 4.1 Truth table to evaluate AND

likePie	likeCake	likePie AND likeCake
True	True	True
True	False	False
False	True	False
False	False	False

If the logical operator connecting the two conditions was 'OR', the resulting truth table is what can be seen in Table 4.2. In this instance, the use of 'OR' expands the number of cases where the condition would evaluate to True to three: the only time it evaluates to False is when both conditions are False.

TABLE 4.2 Truth table to evaluate OR

likePie	likeCake	likePie OR likeCake
True	True	True
True	False	True
False	True	True
False	False	False

Truth tables are a systematic method of testing the logic of a conditional statement. They are particularly useful when there are more than two conditions within a single statement. For example, Table 4.3 contains a truth table for the conditional statement in Figure 4.6. The truth table shows there are three cases where the entire conditional statement would evaluate to True, and five cases where it would evaluate to False. This would not be easily apparent without constructing the truth table.

```
ALGORITHM multiConditions()
BEGIN
     INPUT a, b, c, d
     IF a < b AND ( b < c OR c < d ) THEN
          PRINT "Condition met."
     ELSE
          PRINT "Condition not met."
     ENDIF
END
```

FIGURE 4.6 Complex conditional with more than one logical expression

TABLE 4.3 Truth table for Figure 4.6

a < b	b < c	c < d	b < c OR c < d	a < b AND (b < c OR c < d)
True	True	True	True	True
True	True	False	True	True
True	False	True	True	True
True	False	False	False	False
False	True	True	True	False
False	True	False	True	False
False	False	True	True	False
False	False	False	False	False

A truth table can be used in conjunction with a **trace table** (see pages 174–6) to determine if an algorithm is without logical errors. They can also be used to help select **test data** for testing an algorithm, as discussed later in this chapter.

Chained conditional

The algorithm shown in Figure 4.7 uses a more complex set of conditional control structures (IF/ELSEIF) in order to react to user input when a condition needs to be tested more than once. This is referred to as a **chained conditional**. In this example, the user can select four operations: addition, subtraction, multiplication and division. The program must therefore test the user input four times to see if it matches against the four conditions given.

As the algorithm uses ELSEIF, it will check each conditional statement only if the condition before it evaluates as False. Without the use of ELSEIF, each conditional statement would be run in sequence regardless of whether the condition before it evaluated as True or False. This is an important characteristic of the conditional control structure that programmers often forget, resulting in logical errors in code.

```
ALGORITHM computeOperation()
BEGIN
     PRINT "What is the first number?"
     INPUT firstNumber
     PRINT "What is the second number?"
     INPUT secondNumber
     PRINT "What operation would you like to perform?"
     INPUT operationChosen

     Total ˜ 0
     IF operationChosen = "add" THEN
          Total ˜ firstNumber + secondNumber
     ELSEIF operationChosen = "subtract" THEN
          Total ˜ firstNumber - secondNumber
     ELSEIF operationChosen = "multiply" THEN
          Total ˜ firstNumber * secondNumber
     ELSEIF operationChosen = "divide" THEN
          Total ˜ firstNumber / secondNumber
     ELSE
          PRINT "Invalid operation chosen."
     ENDIF

          PRINT "The result is: " + Total
END
```

FIGURE 4.7 Pseudocode example of a chained conditional control structure

```
ALGORITHM ifElseExample()
BEGIN
     PRINT "What is the current temperature?"
     INPUT currentTemp

     IF currentTemp < 10 THEN
          PRINT "It is very cold."
     ENDIF
     IF currentTemp < 20 THEN
          PRINT "It is a little cool."
     ELSE
          PRINT "It is very warm."
     ENDIF
End
```

FIGURE 4.8 Pseudocode example of conditions with logical errors

Consider the pseudocode in Figure 4.8. Assume the current temperature is input as 9 degrees. The first IF condition tests to see if the temperature is less than 10 degrees – as 9 degrees is less than 10 degrees, the algorithm will print 'It is very cold' and then continue in sequence to the next line of code in the program, which is the second IF condition. The second IF condition will check if the temperature is less than 20 degrees: as 9 degrees is less than 20 degrees, it will print 'It is a little cool' and then continue in sequence to the next line of code in the program, which is the ELSE condition. This portion of the code is not run: as the IF statement it is attached to evaluated to True, the ELSE condition has not been met (9 degrees is not greater than 20 degrees). After the conditional statements are run, it is clear that a logical error has occurred, as two print statements have been produced instead of one.

There is no limit to the number of conditions that can be contained in a chained conditional statement. Note also that chained conditions do not need to contain an ELSE statement.

Nested conditional

Conditional statements can also be placed inside each other. This type of condition is referred to as a **nested conditional statement**. Nested conditional statements are useful when multiple conditions must be handled within the code. For example, Figure 4.9 contains pseudocode in which a conditional statement is used to check if it is raining. If it is, another check is made to see if the person has an umbrella. If they do, they use the umbrella; otherwise they get wet. The check for 'hasUmbrella' is nested within the check for 'isRaining'.

It is important to use consistent formatting in your source code. This not only makes the code easier to read, it also reduces the chance of logical errors occurring due to nested conditionals. The pseudocode in Figure 4.9 shows the hierarchy of the conditional statements because of the use of tabs and spacing; it is much easier to read than if no tabs or spacing was used.

```
ALGORITHM checkUmbrellaUsage()
BEGIN
     INPUT isRaining
     INPUT hasUmbrella

     IF isRaining = True THEN
          IF hasUmbrella = True THEN
               useUmbrella()
          ELSE
               getWet()
          ENDIF
     ENDIF
END
```

FIGURE 4.9 Pseudocode example of a nested conditional statement

Switch/Case

A **switch/case** statement is very similar to a chained conditional, in that it allows for multiple conditions to be tested. Figure 4.10 includes an example of the use of switch/case.

```
ALGORITHM computeOperation()
BEGIN
     PRINT "What is the first number?"
     INPUT firstNumber
     PRINT "What is the second number?"
     INPUT secondNumber
     PRINT "What operation would you like to perform?"
     INPUT operationChosen

     Total ˜ 0
     SWITCH operationChosen
         CASE "add"
               Total ˜ firstNumber + secondNumber
         CASE "subtract"
               Total ˜ firstNumber - secondNumber
         CASE "multiply"
               Total ˜ firstNumber * secondNumber
         CASE "divide"
               Total ˜ firstNumber / secondNumber
         DEFAULT
               Print "Invalid operation selected."
     ENDSWITCH

     PRINT "The result is: " + Total
END
```

FIGURE 4.10 Pseudocode example of a switch/case control structure

9780170440806

Not all programming languages have switch/case functionality implemented. In those programming languages that do, it is typically more efficient to use switch/case than it is to use chained conditionals.

Most programming languages include a method for exiting an iteration early. These are often referred to as 'breaks'.

Iteration

An **iteration**, also known as a loop, is used to repeat sections of code multiple times until a condition is met. There are four main types of iterations: WHILE loops, DO/WHILE loops, FOR loops and REPEAT/UNTIL loops.

WHILE loops

A **WHILE loop** is a section of code that is run when, and for as long as, a condition is met. These types of loops are useful when the programmer does not know when the condition might be met – for example, when running sections of code based on user input that will only cease when a user inputs a particular key sequence.

An example of a WHILE loop can be seen in Figure 4.11. This type of WHILE loop is very common as, when opening a file for reading, it is not possible to tell how many lines there are in the file until all those lines are actually read.

```
ALGORITHM readFromFile()
BEGIN
     INPUT fileName

     fileObject ˜ open filename for reading
     WHILE end of file is not reached DO
          nextLine ˜ read one line from fileObject
          PRINT nextLine
     ENDWHILE
END
```

FIGURE 4.11 Pseudocode example of a WHILE loop to read from a file

THINK ABOUT APPLIED COMPUTING 4.1

Research infinite loops and find out the different reasons they can occur. Learn the key commands your IDE uses to break out of infinite loops, then find some examples in the programming language of your choice and run them.

Key elements of WHILE loops are:

- They are used when it is unknown how many times the loop will execute, or if it will execute at all.
- The condition being tested within the WHILE loop must be met at least once for the code within it to be executed.
- If the condition being tested within the WHILE loop is always True, the loop will never terminate; this creates an **infinite loop**.

DO/WHILE loops

A **DO/WHILE loop** is similar to a WHILE loop in that it executes code within the loop for as long as a condition is met.

Figure 4.12 shows an example of pseudocode to read the contents of a file using a DO/WHILE loop.

The difference between a WHILE loop and a DO/WHILE loop is that a WHILE loop may not necessarily run if the condition being tested is never True, whereas a DO/WHILE loop always runs at least once. For example, in the pseudocode shown in Figure 4.11, it is assumed that there is at least one line to read in the file that is opened. If this is not the case and this code were to be implemented in a programming language, it would produce a runtime error.

```
ALGORITHM readFromFile()
BEGIN
    INPUT fileName

    fileObject ˜ open filename for reading
    DO
        nextLine ˜ read one line from fileObject
        PRINT nextLine
    WHILE end of file is not reached
END
```

FIGURE 4.12 Pseudocode example of a DO/WHILE loop to read from a file

Key elements of DO/WHILE loops are:

- They are used when it is unknown how many times the loop will execute.
- The code within the DO/WHILE loop will always execute at least once.
- If the condition being tested within the DO/WHILE loop is always True, the loop will never terminate; this creates an infinite loop.

FOR loops

A **FOR loop** is a section of code that is run a predefined number of times. Such loops are particularly useful to perform an action on every element of an array, or to perform a basic search through a set of elements in a data structure.

FOR loops need three pieces of information to execute. The first is a starting point or counter, the second is the end condition, and the third is a statement called an increment that increases (or decreases) the starting point (counter) so that it approaches the end condition.

Most programming languages have a special format for writing FOR loops. For example, in C and C++ a FOR loop is written as shown in Figure 4.13, and in Visual Basic (VB) it is written as shown in Figure 4.14.

```
for ( starting_point; end_condition; increment )
{
  ° statement(s);
}
```

FIGURE 4.13 FOR loop in the style of the C, C++ and C# programming languages

```
For counter = start_condition to end_condition
  ° Statement(s)
Next
```

FIGURE 4.14 FOR loop in the style of the Visual Basic programming language

An example of a FOR loop in pseudocode is shown in Figure 4.15. This FOR loop checks every element of an array to see if a word being searched for is found in the array. Each time the loop is executed, the end condition (i < iNumNames) is tested; the loop will continue to run for as long as this returns True.

Key elements of FOR loops are:

- They run for a set number of times and this number must be known beforehand.
- They will only execute the code inside them if the end condition is still being met, and may not execute the code at all.

- As with WHILE and DO/WHILE loops, it is possible for a FOR loop not to terminate; this occurs only if the increment and end condition are unrelated. This type of occurrence would be considered a logic error.

```
ALGORITHM checkArray()
BEGIN
     INPUT arrayNames
     INPUT searchTerm

     iNumNames ˜ length of arrayNames
     FOR i ˜ 0, i < iNumNames, i ˜ i + 1 DO
          If arrayNames[i] = searchTerm THEN
               PRINT "Found " + searchTerm
          ENDIF
     ENDFOR
END
```

FIGURE 4.15 Example of a FOR loop checking every element of an array

In some programming languages, FOR loops can also be written using an iterator loop, rather than a counter with an incrementor. An example of this type of FOR loop can be seen in Figure 4.16. Notice that a temporary variable, 'eachName', has been created. This is used throughout the FOR loop to refer to the current element in the array. Once the FOR loop has finished executing, the temporary variable is 'deleted' (freed from memory).

```
ALGORITHM checkArray()
BEGIN
     INPUT arrayNames
     INPUT searchTerm

     FOR eachName in arrayNames DO
          IF eachName = searchTerm THEN
                 Print "Found " + searchTerm
          ENDIF
     ENDFOR
END
```

FIGURE 4.16 Pseudocode example of a FOR loop using an iterator loop and temporary variable

REPEAT/UNTIL loops

Much like WHILE loops, **REPEAT/UNTIL loops** repeatedly run a source code within the loop; they differ, however, in the treatment of the condition that terminates the loop. A WHILE loop will run for as long as a condition returns True, whereas a REPEAT/UNTIL loop will run for as long as a condition returns False. Figure 4.17 demonstrates pseudocode that uses a REPEAT/UNTIL loop to read lines from a file.

```
ALGORITHM readFromFile()
BEGIN
     INPUT fileName

     fileObject ˜ open filename for reading
     REPEAT
          nextLine ˜ read one line from fileObject
     UNTIL end of file is reached
END
```

FIGURE 4.17 Example of a REPEAT/UNTIL loop to read from a file

Key elements of REPEAT/UNTIL loops are:

- They are used when it is unknown how many times the loop will execute.
- The code within the REPEAT/UNTIL loop will always execute at least once.
- If the condition being tested within the REPEAT/UNTIL loop is always False, the loop will never terminate; this creates an infinite loop.

Functions

A **function** is a block of code that is intended to be used repeatedly within an application. The code within a function executes an algorithm, and typically provides a **return value** as a result. To use a function within source code is to 'call' it. An example of a **function call** can be seen in Figure 4.18; both useUmbrella() and getWet() are function calls.

A function can, but does not need to, provide a return value back to where the function was called. When it does not return a value, it is referred to as a procedure.

```
ALGORITHM checkUmbrellaUsage()
BEGIN
     INPUT isRaining
     INPUT hasUmbrella

     IF isRaining = True THEN
          IF hasUmbrella = True THEN
               RETURN useUmbrella()
          ELSE
               RETURN getWet()
          ENDIF
     ENDIF
     RETURN False
END
```

FIGURE 4.18 Example of the use of function calls

Notice that functions in pseudocode use parentheses after the name of the function. This helps distinguish them from variables.

TABLE 4.4 Function declarations in some popular programming languages

C, C++, C#	`int max(iNumOne, iNumTwo);`
VB, VB.Net	`Function max(byVal iNumOne as Single, byVal iNumTwo as Single)`
Python	`def max(iNumOne, iNumTwo):`
PHP	`function max($iNumOne, $iNumTwo);`
Swift	`func max(iNumOne: Int, iNumTwo: Int) -> Int`

Functions require a **function declaration** that names the function and its **arguments**. In many languages, function declarations must include the return value data type. They can also include an optional reference to the **function visibility**.

Functions must then have a **function definition** written, which simply means that the function must be written. Some languages, such as C and C++, require that the function declaration is written separately, before the function definition, whereas other languages, such as Python and VB.Net, include the function declaration as part of the function definition. Once a function is defined and written, it can be used throughout the source code.

Many programming languages require functions to be declared before they can be defined. Similarly, functions must be defined before they can be used within the source code. For this reason it is a good idea to place all function declarations at the top of your source code, and to place function definitions near the top as well.

It is always advisable to use a built-in function whenever possible as they have been tested, and are less likely to contain bugs compared to a user-defined function.

Almost all programming languages have **built-in functions** that can be used without needing to provide a function declaration or function definition. These functions have been written by the creators of the programming language to execute common sequences of code, such as drawing a widget on a user interface, printing text to a screen, computing mathematical equations such as square roots and powers, or accepting input from a keyboard.

As well as having built-in functions, almost all programming languages allow you to include additional functions, such as those contained within third party packages, in your source code. These can be very useful to include so that you are not re-creating code that has already been written and tested.

Arguments/Parameters

4.2 THINK ABOUT APPLIED COMPUTING

Research some of the third party packages that are available to you in your chosen programming language. Find out how to include these packages in your own code. Most languages require that you use a special instruction such as 'include', 'using' or 'import'.

Functions can optionally require arguments, also known as **parameters**, which act as specific inputs that are 'passed' to the function. The data within the arguments passed to a function are assigned temporary variable names as part of the function declaration. This allows the use of **local variables** within a function and thus avoids the need to use **global variables** to access data that exists outside the function.

Many programming languages have two categories of arguments that can be passed to a function: those that are **pass by reference** and those that are **pass by value**. 'Pass by reference' means that the original data being passed into the function can be modified without needing to be 'returned'. 'Pass by value' means that the original data is left unchanged, even if the data in the temporary variable is modified within the function. In those languages that only use pass by value in functions, modifying the data stored in the original variables that have been passed to the function requires that the modified data must be returned back to the source code that called the function.

Figure 4.19 demonstrates how to declare a function using pseudocode. Note that the data types of the arguments, and the function's return value, are defined within a pseudocode comment. The arguments are included as part of the function definition to distinguish them from other types of input.

```
FUNCTION max( iNumOne, iNumTwo )
{ Purpose: return max value as an integer }
{ Arguments: iNumOne and iNumTwo are integers }
{ Output: integer, the maximum integer }
BEGIN
     IF iNumOne > iNumTwo THEN
          RETURN iNumOne
     ELSE
          RETURN iNumTwo
     ENDIF
END
```

FIGURE 4.19 Representation of a function in pseudocode

Function visibility

Visibility and scope are elements of programming that can be applied, not just to functions, but also to variables, methods, classes and events.

Security is an important aspect to consider when writing source code for modern applications. Aside from encryption (see Chapter 1, page 25), there are other ways a software developer can protect access to data within variables. The most common method is to use function visibility to restrict access to functions within applications.

Public

Public visibility of a function means that it is visible both inside the source code or class in which the function exists, and also via external source files, classes and applications.

While a programmer can explicitly refer to a function as public, there is no need to do this in most programming languages as public visibility is the default visibility of all functions.

Protected

Protected visibility of a function means that it is visible only to a class, or extensions of that class. This means that the functions and methods defined as protected within a class can be used by that class, as well as by any classes that inherit the class that contains the protected function.

Private

Private visibility of a function means that it is visible only to a particular class. Unlike protected functions, a private function cannot be used by a class that inherits the class that contains the private function.

Classes

As mentioned in Chapter 3, a class is a feature of object-oriented programming that allows a programmer to group together related functions and variables in one place. This acts as a template for creating user-defined objects.

In a business application, a programmer could write a 'user' class that contains typical user variables and methods, such as username, password, the ability to log in and the ability to change user details. They could then use this base 'user' class to create an 'administrator' class that adds other methods, such as the ability to change other users' passwords and user details. This is demonstrated in Figure 4.20, where the Administrator class inherits all of the variables and methods from the User class and then adds three more. These two classes, Administrator and User, could then be instantiated into objects within the program to allow for two different types of users to exist. This saves programming time, as the basic user methods do not need to be rewritten when creating a different type of user.

4.3 THINK ABOUT APPLIED COMPUTING

What type of visibility should the functions inside the user and administrator classes have? Why?

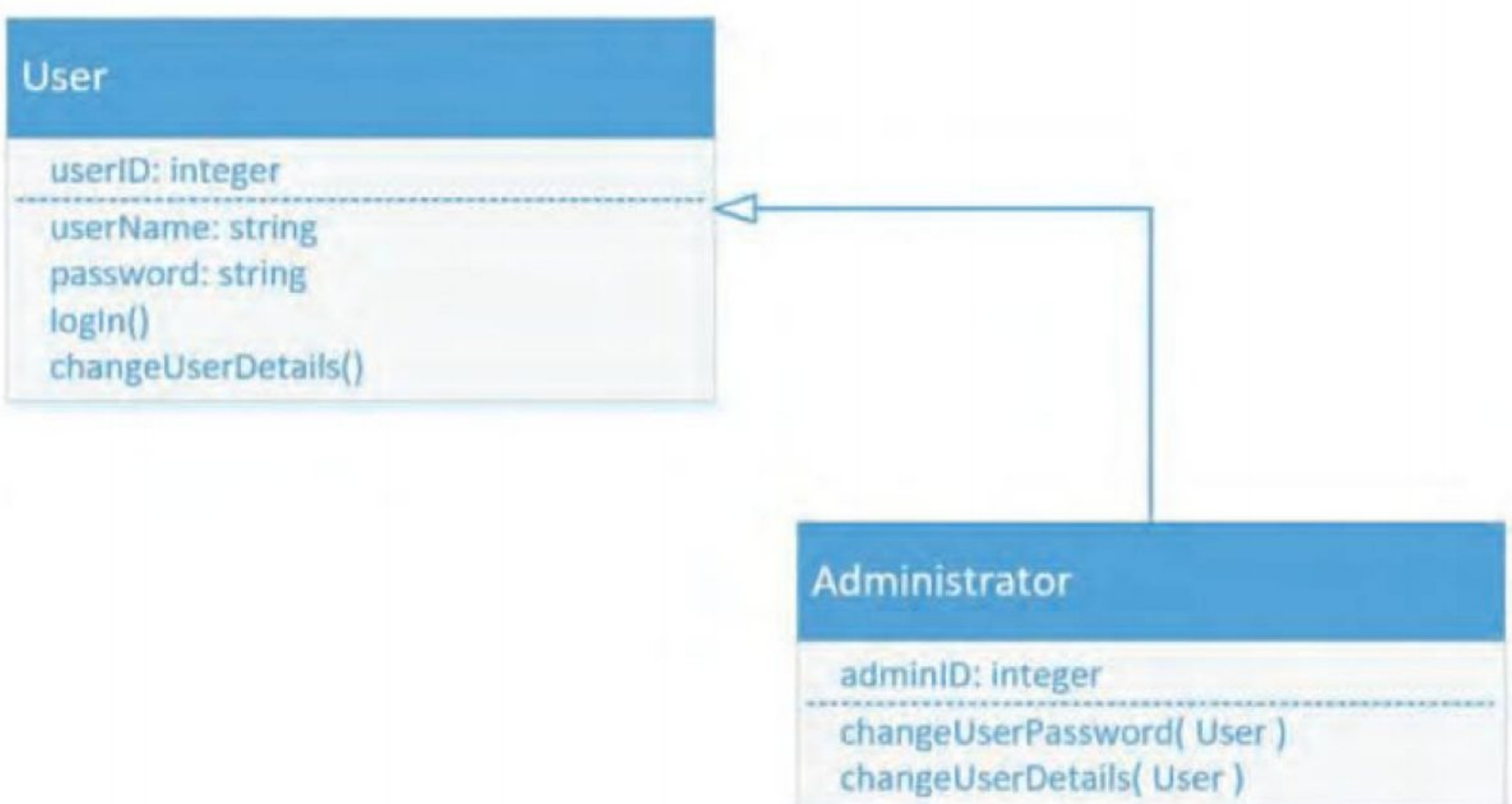


FIGURE 4.20 Example of inheritance

9780170440806

THINK ABOUT APPLIED COMPUTING 4.4

Take a look at some of the classes that exist in the programming language you are using. What type of methods are contained within them? As a start, you may want to look at user interface classes (widgets) to see what events they support.

Classes are also useful when security of data is paramount. Much like function visibility, variables inside a class can have **class visibility** and be 'hidden' from other parts of the code so that the only way to access the data in those variables is to use a method that has been written with security in mind. These methods may, for example, encrypt or decrypt data, or check user permissions before allowing a variable's data to be changed.

Methods

A **method** is a function or procedure that exists within a class. A special type of method is an **event**, which is a method that is called when an object's state changes; this means that something has occurred to trigger the event. For example, pressing a button in a user interface object can trigger an event to submit that button.

```
ALGORITHM useMethod()
BEGIN
     mathObj = create new object from math class
     mathObj.addTwoNumbers(4, 1)
END
```

FIGURE 4.21 Pseudocode of an object and a method

Events can be user-defined or they can be built-in events.

Internal documentation

One of the most important, yet most easily forgotten, aspects of writing good code is including meaningful **internal documentation**. Internal documentation consists of notes and comments written by a programmer within the code itself. It includes information about the program as a whole, as well as about each of the classes, functions, methods, objects and algorithms within it. It is often combined with meaningful, well-named variables to create manageable and effective code.

Internal documentation has no impact on the compilation or running of the code itself. It exists only to provide context and important information about the code itself. Often it is most useful when a programmer is reading through code that they did not write themselves, or that was written a long time ago. In this situation, internal documentation means that the programmer does not need to rely on remembering what they wrote or on interpreting complex algorithms to understand how the program works. Well-written internal documentation saves time as it reduces effort on the part of the developer, making it a core feature of efficiency in relation to the creation and maintenance of a software solution.

Internal documentation conventions

While there is no single set of conventions for internal documentation, there are many common elements that should be included in internal documentation:

- a **header comment**, containing the name of the file, a brief description of the program, the author's name, and the date when the program was first created
- documentation of classes and methods, describing their behaviour and how they are used, including any expected inputs and outputs and their respective data types
- function and subroutine comments, describing their purpose, as well as describing all inputs and outputs and their respective data types

- single-line comments, providing brief summaries of portions of code
- multi-line comments, explaining a complex algorithm within the code itself
- descriptions of how to test aspects of the software
- extra information on upgrades, changes or enhancements made to the program.

While internal documentation can include comments related to revisions and new versions, this does not replace the need to use an effective **version control system**.

Version control is the management of changes to source code files and other project-related documents. This is important throughout the duration of a software project. Version control systems typically run as stand-alone programs or web-based systems that not only help track changes to documents, but also allow more than one developer to work on source code at the same time. They also give a developer the ability to revert back to previous versions of source code.

One popular version control system is Git/GitHub.

When writing internal documentation, it is important that the comments made within the source code are well formatted to allow them to be easily read. Comments should be separated from code by a blank line before the comment. For multi-line comments, a blank line should be included both before and after the comment. A comment should be vertically aligned with the indent level of the current code to make it clear which code it is associated with. Depending on the programming language used, comments must be enclosed using special characters. Table 4.5 includes a list of the characters used in some popular programming languages.

TABLE 4.5 Common programming language commenting conventions

C, C++, Java, JavaScript, Swift	// this is a single-line comment /* ◦ This is a multiple line comment */
Python	# this is a single-line comment
PHP	# this is a single-line comment // this is also a single-line comment /* ◦ This is a multiple-line comment */
XML, HTML	<!-- this is a single-line comment --> <!-- ◦ This is a multiple-line comment -->
VB, VB.Net	' this is a single-line comment

Many programming languages have tools that make creating internal documentation easier. There are also tools that extract internal documentation to create reference manuals and online documentation for users (see weblink).

One tool that extracts internal documentation is *doxygen*, which supports multiple programming languages.

While it is important to include internal documentation in all software modules, the comments that are included should be meaningful and non-trivial. Unless it is being used for teaching purposes, internal documentation should not simply state or restate what is occurring in the code, in particular if this is clearly apparent in the code itself.

9780170440806

For example, the commented Python code in Figure 4.22 would be considered trivial as the code comments do little more than restate what the next line of code does.

```
def readFromCSV( csvFileName ):

    # open the CSV file for reading
    csvFile = open(csvFileName, "r")

    # read all of the lines in the csv file
    csvContents = csvFile.readlines()
    return csvContents
```

FIGURE 4.22 Poorly commented code

Better comments would be those seen in Figure 4.23. Notice that the code comments provide information about the function, its inputs and its outputs, including reference to data types.

```
# Function: Reads a CSV file and returns its contents
# Input: String, filename
# Output: Array of Strings, contents of the file

def readFromCSV( csvFileName ):

    csvFile = open(csvFileName, "r")
    csvContents = csvFile.readlines()
    return csvContents
```

FIGURE 4.23 Appropriately commented code

Characteristics of input and output

Inputs and outputs are typically saved into particular file formats, such as plain text files or structured text files. Each format has its own uses, with advantages and disadvantages depending on the context in which it is used.

There are two types of files: **text files** and **binary files**. Text files store data as easily readable plain text, or delimited text, while binary files store data in binary form, as is the case with images and sound. Binary files are not easily readable and are therefore more secure than text files.

Files can be opened using different modes, such as 'read', 'write', 'append' and 'binary'.

TABLE 4.6 File operations

open	file.*open*(*filename*, *mode*)
close	file.*close*()
read	file.*read*()
write	file.*write*(*character*)

Plain text files

A **plain text file** is a structured file that contains characters of readable data. This data can be structured with spacing, new lines and tabs, but can only be read as character and string data types.

```
namesAndHeights.txt - Notepad
File Edit Format View Help
Paulo   1.23
Shehara 1.35
Phoebe  1.21
Tuan    1.61
```

FIGURE 4.24 Example of a plain text file of names and heights

Plain text files are commonly used for configuration settings or for storing small amounts of data in simple software programs.

```
desktop.ini - Notepad
File Edit Format View Help
[.ShellClassInfo]
LocalizedResourceName=@%SystemRoot%\system32\shell32.dll,-21770
IconResource=%SystemRoot%\system32\imageres.dll,-112
IconFile=%SystemRoot%\system32\shell32.dll
IconIndex=-235
```

FIGURE 4.25 Example of a plain text configuration file from a Windows 10 system

While plain text files that are stored in a computer system can be opened and read by a human, they are not typically designed for human readability. Instead, they are designed for fast processing and reading by computer programs. This means that a plain text file often lacks comments, headings and subheadings that would make it more coherent for a human reader.

Delimited files

A delimiter-separated value (DSV) text file, or **delimited file**, is a particular type of text file, in which data values are separated by a programmer-selected character. This character is referred to as the **delimiter**. The most common delimiters used in DSV files are commas, tabs and colons. Delimited files allow for the storage of two-dimensional arrays in a structured, readable format.

When a comma is used as a delimiter in a delimited file, the file is referred to as a comma-separated value file or **CSV file**.

```
sampleCSV.csv - Notepad
File Edit Format View Help
PALADINO,Nathan,11,M,PAL0011,11A,MA071 G,IT011 B,AC011 A,PE011 A,EN011 G,BM011 B
CARNUCCIO,Lorenzo,11,M,CAR0022,11A,AC011 B,EN011 D,ME011 A,IT011 A,MA111 E,MA071 B
BRETHERTON,Jessica,11,M,BRE0033,11C,EN011 F,ME011 B,SA011 A,IT011 A,PY011 D,BM011 B
VEAL,Carena,11,M,VEA0044,11C,IT011 B,AC011 A,EN011 C,HI031 A,LS011 B,BM011 B
KHA,Ric,11,M,KHA0055,11B,MA071 G,IT011 B,DT011 B,PH011 A,EN011 G,MA111 A
```

FIGURE 4.26 Example of a CSV file containing student and subject data

In programming, delimited files are very useful when storing small amounts of data. When there is a lot of data, loading, reading and writing to a DSV file can be inefficient as it is very slow in comparison to reading from a database or an XML file. In addition, DSV files are not secure, as anyone opening the file can read its contents. For this reason, DSV files are not suitable for storing sensitive data, such as usernames and passwords, financial details or medical details. This risk to security can be reduced if sensitive data is encrypted.

XML files

Comprehensive documentation outlining XML can be found at W3Schools.

An **Extensible Markup Language (XML)** file is a type of text file that has been created using a set of rules for encoding the file into a format that can be read both by a human and by a computer program. XML makes it easier to store and transport data within a system, and between systems, as it is based on a set of standards and conforms to published conventions. XML was designed to be as self-descriptive as possible, which increases its 'human readability'.

XML files contain a **prolog**, or declaration, which is information that appears before the start of any data in the XML file. The prolog includes information that applies to the XML file as a whole, such as the version of XML it uses and the **character encoding** of the data within it.

While XML and HTML are both tag-based languages, the key difference is that XML does not have any predefined tags. Instead, XML tags are determined by the person who creates the XML file.

An XML file contains an XML **tree**, which is the set of elements contained within the file. The tree begins with a **root element** that is a **parent element** to **child elements**. These child elements are sub-elements of the root, but any element can contain sub-elements. This makes the structure of an XML file hierarchical, using the analogy of a family tree.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
- <library>
  - <book title="Finnikin of the Rock">
      <author>Melina Marchetta</author>
      <year>2009</year>
      <genre>young adult</genre>
      <price>19.99</price>
    </book>
  - <book title="Algorithms in a Nutshell">
      <author>George T. Heineman</author>
      <author>Gary Pollice</author>
      <author>Stanley Selkow</author>
      <year edition="1st">2012</year>
      <year edition="2nd">2016</year>
      <genre>programming</genre>
      <price>94.95</price>
    </book>
  </library>
```

FIGURE 4.27 XML file containing library books

An XML element can contain attributes, text or any other content.

TABLE 4.7 XML element types and characteristics

Root	The first element in an XML tree, and parent to all other elements. There can only be one root element. In Figure 4.27, this is represented by the <library> element.
Parent	Any element that contains sub-elements. An example in Figure 4.27 is the <book> element.
Child	Any sub-element to another element. An example in Figure 4.27 is the <author> element.
Sibling	Any sub-element on the same level as another sub-element is a *sibling* to it. The <author> and <genre> elements in Figure 4.27 are siblings to each other.
Attribute	An element can contain one or more attributes. An example of an attribute in Figure 4.27 is the 'title' attribute inside the <book> element.
Text	An element can contain text content. An example of text content in Figure 4.27 is '94.95' inside the <price> element of the book 'Algorithms in a Nutshell'.

While elements and attributes are user-defined, some naming rules still apply. Elements are **case-sensitive**, must start with a letter or an underscore, cannot start with the letters 'xml' and cannot contain spaces. They can contain letters, numbers, hyphens, underscores and full stops.

The advantages of using an XML file over a plain text file is that XML is an industry standard that is widely used and is portable across platforms. It allows rules to be set and used on data in a way that text files cannot. XML also allows storage of data in a way that does not rely on a user interface – the same data can be displayed in different formats and interfaces.

XML files are used for lots of different purposes, including:

- storing data – for internal and/or external systems
- storing configuration information
- storing user interface details – this is particularly useful for cross-platform and cross-system applications
- moving and sharing data between systems.

Using an XML file ensures that data received from a source is in an 'as expected' format. This ensures data integrity across systems.

Testing and debugging techniques

One of the most important steps when building an application is checking to ensure that each module that has been built meets the design specifications that were set out for it.

When checking to see if a module meets design specifications, it is important to make sure it performs as expected with appropriate inputs, that it is usable and efficient, and that it achieves what the design specification outlines. This often involves internal testing, where the programmer tests the program themselves; external testing, where a quality assurance tester tests the program using **test cases** based on the design specifications; and client testing, where the client participates in walk-throughs and reviews of the software to confirm that it is as they specified in the design stage.

In VCE Applied Computing, checking of modules to ensure they meet design specifications is mostly done through internal testing methods. An important aspect of this is to make sure that the modules that have been built are as bug-free as possible. This is achieved by completing appropriate testing activities, such as establishing test cases that determine test data and **expected results**, conducting tests, recording results, and then correcting any errors.

One method used in the testing process is **debugging**, which is a testing method that includes finding errors through the construction of trace tables and testing the system using test data.

Debugging

Different types of errors can occur throughout the development of a software solution. Each error can be categorised as a **syntax error**, a **runtime error** or a **logic error**.

Syntax errors

Each programming language has a defined syntax, which is a set of rules that define the arrangement and structure of source code within the programming language. These rules also define which symbols and characters can and cannot be used within variable names. Syntax errors occur while writing code and are typically fixed immediately, as they prevent code from compiling. These are errors such as missing brackets or semicolons, not passing the correct number of arguments into a function, or not putting quotes around characters and strings. Most IDEs can identify syntax errors during development, while the programmer is writing the code, and most compilers will indicate on which line a syntax error occurs, which allows a programmer to find the error easily.

Runtime errors

Unlike a syntax error, a software module with a runtime error will compile without any error notifications. It is not until the program is run and used that a runtime error can appear. These types of errors often result in the program crashing or printing error messages. For example, one type of runtime error is a **memory leak**, where a program continually uses more and more RAM while the program is running, such as when an infinite loop occurs. Other common runtime errors include **divide-by-zero errors**, opening non-existent files for reading, calling functions that do not exist, or validation errors that have not been handled within the source code.

Runtime errors are often found during the testing stage of producing a software solution, but software can, and does, get released with the possibility of runtime errors still occurring. This is because these errors tend to occur only when certain environmental or technical conditions are met, or when unexpected inputs are entered into the software. In many cases, once a runtime error is found in a distributed piece of software, the software company releases **patches** and software updates after correcting the error.

Logic/semantic errors

Put simply, logic errors, also known as semantic errors, occur when the logic of a software program fails. This means that the source code is syntactically correct but the software solution, once compiled or interpreted, does not produce the expected output when run. In this case, the output is often unintended, undesirable or incorrect. For example, a function written to return the square root of a number may instead return the square of a number. Similarly, a function in an air-conditioning unit that checks to see if the temperature in a room is greater than 30 degrees before turning itself on may activate the air conditioning at 29 degrees instead.

Logic errors can be very difficult to find, as there is nothing within a compiler or interpreter that will tell a programmer on which line a logic error appears. It is often up to the programmer as a human to construct test data and trace tables in order to track down the error manually.

Test data

In order to systematically test that a module works, appropriate test cases must be written. A test case is a set of steps that a tester uses to determine if the element being tested works correctly. It involves selecting test data, writing testing procedures and determining expected results. It is particularly important that appropriate test data is selected so that test cases can be run. At a minimum, the test data selected must ensure full coverage of the algorithm when test cases are performed. This means that all paths of all control structures are tested fully.

There are four main types of test data. The first involves validation test data that tests the validation techniques that have been included in the module. Data for this purpose must be selected to test any instances of existence checking, type checking and range checking that have been included in the source code. At a minimum, this should involve selecting test data to check for both valid inputs and invalid inputs. For example, if checking for a valid age integer being input by a user, a valid input would be a positive integer, likely in the range of 0 to 120. An invalid input would be a negative integer, such as –1.

The second type of test data will test all conditional statements, including the operators within those conditions. Selecting appropriate data to do this may involve constructing truth tables. It may also involve selecting appropriate **boundary values** to ensure full coverage of the operators included in the conditionals.

The third type of test data is designed to test all iterations, including any operators within those iterations. This may involve selecting or creating data that will ensure all conditions of those iterations are met. For example, if a WHILE loop is designed to iterate over each line of a file, test data would involve testing the following:

- when the file does not exist
- when the file exists but is empty
- when the file exists and has one line to read
- when the file exists and has more than one line to read.

The fourth type of test data involves creating test cases that will attempt to cover all of the functionality within the system from the perspective of a user. This typically means testing the graphical user interface to ensure that each screen within the application can be accessed appropriately and correctly. Typically, test cases are written based on design documents such as mock-ups, storyboards and/or site maps to ensure that the order and sequence of user interface elements are correct. This type of testing can be the most time-consuming, particularly if the application allows a user to take multiple paths to access particular screens. As a result, developers often write formal test cases that allow for automated software testing programs to be used to run these tests so they do not need to be performed manually. These test cases can be written to simulate the actions of a user navigating through the software solution. Within VCE Applied Computing, there is no formal requirement for students to develop, utilise or carry out automated testing of their modules.

Boundary values

Testing for boundary values involves selecting test data that will test the 'boundaries' of any condition or iteration within the code; that is, the maximum and minimum values available for a given input. Boundary values are particularly relevant for any algorithm that uses a range check. The general rule for boundary testing is that, at each boundary, test data should be selected to test both outside the boundary and inside the boundary.

For example, imagine a software solution designed for a Scout group that would only allow users to join if they are in the 'Venturer' age group (15–17 years). The boundary values for testing this age group can be determined by considering the type of data that is being collected. As this example involves testing for age, valid integers begin at 0 and are unlikely to go beyond 120. To test for the age of a particular age group would therefore require testing an age that is below the age group, within the age group, and above the age group.

The ages that are in range can be represented on a continuum:

0	1	2	3	...	14	15	16	17	18	19	...
						↑	↑	↑			

There are two boundaries for this particular test. One boundary exists at age 15 and the second boundary exists at age 17. The pseudocode that demonstrates these can be seen in Figure 4.28.

9780170440806

```
ALGORITHM checkVenturer( age )
{ Purpose: to check that a user is a Venturer based on age }
{ Inputs: age, as an integer }
{ Output: Boolean, True if age is valid, False if not }
BEGIN
  ° IF age > 14 AND age < 18 THEN
  °   ° RETURN True
  ° ELSE
  °   ° RETURN False
  ° ENDIF
END
```

FIGURE 4.28 Pseudocode to check an age range

In order to fully test this algorithm, the test data that must be selected are:

Age	Reason	Expected result
14	One year below the lower boundary of the valid range	False
15	The lower boundary of the valid range	True
17	The upper boundary of the valid range	True
18	One year above the upper boundary of the valid range	False

When selecting above and below a boundary, test data should be in the smallest increment possible given the context. In this case it is 1 year but, for example, if an algorithm were to test for a price range, the smallest increment would be 0.01. Similarly, if testing an algorithm that uses boundaries involving hours or minutes in a day, the smallest increment would typically be 1 minute, thus boundaries at the hour would be 59, 0 and 1.

```
ALGORITHM abstractBoundaries()
BEGIN
     INPUT a, b
     IF a < b THEN
          PRINT "Condition met."
     ELSE
          PRINT "Condition not met."
     ENDIF
END
```

FIGURE 4.29 Abstract set of boundary conditions

For a more abstract example, the pseudocode in Figure 4.29 would require the following boundary values to be tested:

a	b	Reason	Expected result
a	a + 1	a is guaranteed to be less than b	Condition met
a	a	a is guaranteed to not be less than b, as it is equal	Condition not met

For more complex algorithms, more test data needs to be selected, and this is when it can be very useful to construct a truth table to determine what that test data should be.

Consider the pseudocode example testing for multiple conditions in Figure 4.30.

```
ALGORITHM multiConditions()
BEGIN
  ◦ INPUT a, b, c, d
  ◦ If a < b AND ( b < c OR c < d ) THEN
  ◦   ◦ PRINT "Condition met."
  ◦ ELSE
  ◦   ◦ PRINT "Condition not met."
  ◦ ENDIF
END
```

FIGURE 4.30 Complex conditional with more than one logical expression

Consider also the truth table constructed for this algorithm, shown in Table 4.8.

TABLE 4.8 Truth table for Figure 4.30

a < b	b < c	c < d	b < c OR c < d	a < b AND (b < c OR c < d)
True	True	True	True	True
True	True	False	True	True
True	False	True	True	True
True	False	False	False	False
False	True	True	True	False
False	True	False	True	False
False	False	True	True	False
False	False	False	False	False

The test data required to fully test the conditions in Figure 4.30 must involve testing the conditions shown in each line of the truth table. For example, the second line of the truth table requires that the condition $c < d$ is not met, so a boundary test must be performed where c is not less than d (i.e. $c = d$).

For this pseudocode, at least eight test data elements must be written to test the conditional statement fully.

It can be useful to use an algebraic expression to map out which values are needed for a test condition to be met. For example, using the example again from Figure 4.30 and Table 4.8, the first line of the truth table can be interpreted algebraically in this way:

a < b	b < c	c < d	b < c OR c < d	a < b AND (b < c OR c < d)
a = b	b = c	c = d	True	True

This is because each condition can be guaranteed to be True if the algebraic conditions are met.

9780170440806

The algebraic expressions can be further simplified:

a < b	b < c	c < d	b < c OR c < d	a < b AND (b < c OR c < d)
a = b	b = d – 1	c = d	True	True

And finally:

a < b	b < c	c < d	b < c OR c < d	a < b AND(b < c OR c < d)
a = d – 1	b = d – 1	c = d	True	True

Therefore, only a value for *d* needs to be chosen, as the values for *a*, *b* and *c* can all be determined from the value for *d*.

It is important that test data is selected systematically, rather than in an ad hoc manner, as otherwise it cannot be guaranteed that the source code will be logically correct throughout all of its algorithms. As a result, a significantly large set of test data may be required to test the system, and this is why many large software development companies use automated tools to construct and run tests on source code.

Trace tables

To prevent logic errors occurring, programmers often construct trace tables to validate the logic of the algorithms used in their source code. Trace tables simulate the execution of a program, referred to as the **flow of execution**. Given test data, each processing feature within an algorithm is executed, step by step, and, based on the test data, the values of the variables that change within that algorithm are tracked to ensure that the logic within the algorithm is correct. This systematic method of tracking the execution of code allows for the thorough testing of even the most complex of algorithms.

As an example, consider the pseudocode in Figure 4.31. This simplistic algorithm uses a WHILE loop control structure to print values until a condition is met. The trace table to represent the flow of execution of this pseudocode is in Table 4.9.

```
ALGORITHM sampleWhileLoop()
BEGIN
     x ˜ 0
     y ˜ 0
     WHILE x < 32 DO
          x ˜ x + 8
          y ˜ y + 4
     ENDWHILE
     PRINT x, y
END
```

FIGURE 4.31 Pseudocode example of a WHILE loop

TABLE 4.9 Trace table for the pseudocode in Figure 4.30

Step	Statement	x	y	output
1	x ← 0	0		
2	y ← 0	0	0	
3	**While** x < 32 **Do**	0	0	
4	x ← x + 8	8	0	
5	y ← y + 4	8	4	
6	**While** x < 32 **Do**	8	4	
7	x ← x + 8	16	4	
8	y ← y + 4	16	8	
9	**While** x < 32 **Do**	16	8	
10	x ← x + 8	24	8	
11	y ← y + 4	24	12	
12	**While** x < 32 **Do**	24	12	
13	x ← x + 8	32	12	
14	y ← y + 4	32	16	
15	**While** x < 32 **Do**	32	16	
16	**EndWhile**	32	16	
17	Print x, y	32	16	32, 16

Another example, where a logic error exists within an algorithm, can be seen with the algorithm in Figure 4.32.

As this algorithm requires input from a user, the trace table can only be constructed using test data. For the purposes of this example, the test data is as follows (Table 4.10) and the flow of execution is tested twice, with two sets of inputs and the expected discounted price being listed with the test data.

```
ALGORITHM applyDiscount()
BEGIN
  ° INPUT fullPrice
  ° INPUT discPerc

  ° discPrice ˜ fullPrice – discPerc
  ° RETURN discPrice
END
```

FIGURE 4.32 Pseudocode containing a logic error

TABLE 4.10 Test data for the pseudocode in Figure 4.32

fullPrice	discPerc	Expected result
20.00	5	19.00
50.00	50	25.00

TABLE 4.11 Trace tables for the pseudocode in Figure 4.32

Step	Statement	fullPrice	discPerc	discPrice	output
1	**Input** fullPrice	20			
2	**Input** discPerc	20	5		
3	discPrice ← fullPrice – discPerc	20	5	15.00	
4	**Return** discPrice	20	5	15.00	15.00

Step	Statement	fullPrice	discPerc	discPrice	output
1	**Input** fullPrice	50			
2	**Input** discPerc	50	50		
3	discPrice ← fullPrice – discPerc	50	50	0.00	
4	**Return** discPrice	50	50	0.00	0.00

After completing the trace tables, the test data can be completed and the results are as shown in Table 4.12.

TABLE 4.12 Completed test data for the pseudocode in Figure 4.32

fullPrice	discPerc	Expected result	Actual result
20.00	5	19.00	15.00
50.00	50	25.00	0.00

From viewing the actual results from the trace table and the test data, it is clear that the algorithm is not calculating a correct discounted price given the inputs it is receiving; it is merely subtracting the discount percentage as if it were a dollar value to be discounted. The corrected algorithm can be seen in Figure 4.33.

```
ALGORITHM applyDiscount()
BEGIN
     INPUT fullPrice
     INPUT discPerc

     discAmount ˜ fullPrice * (discPerc / 100)
     discPrice ˜ fullPrice - discAmount
     RETURN discPrice
END
```

FIGURE 4.33 Discount pseudocode after the logic error is fixed

THINK ABOUT APPLIED COMPUTING 4.5

Look up the documentation for the IDE that you use with your selected programming language. If it supports tracing the flow of execution, run the tracing function over some of your code to see how it works.

Traditionally, a programmer would produce trace tables manually to test code and check for logic errors. Many programming languages today, however, have **integrated development environments (IDEs)** that allow a programmer to trace the flow of execution through a debugger built into the IDE.

Evaluating the efficiency and effectiveness of solutions

Evaluation typically involves checking to see how well a software solution has met its stated requirements. This evaluation is normally performed a set time period after the solution has been put into place, where the timeframe selected is relevant to the context in which the software operates. Generally, however, an evaluation would not occur in the first six months after software is first put in place and normally occurs between six and 12 months after the software solution has been put in place. This ensures that those using the new software still have some memory of the previous software but have had enough time to get used to the new software to make an accurate evaluation.

In order to conduct an evaluation, an evaluation strategy needs to be created. This normally occurs at the end of the design stage of the problem-solving methodology. The evaluation strategy specifies the timeframe in which an evaluation will occur and outlines the data that will need to be collected to complete the evaluation, including a description of the methods and techniques that will be used to collect that data. The strategy also makes clear how the data collected relates to the criteria that were written in the design stage.

The software solution is then evaluated in terms of efficiency and effectiveness.

Efficiency

The **efficiency** of a solution concerns how much time, cost and effort has been applied to achieve the intended results. This could include measurements against the speed of processing, the functionality of the software, or the cost of file manipulation.

'Cost', in relation to the efficiency of a solution, does not necessarily mean monetary cost. It can also refer to time, as manipulating files can be quite slow depending on the amount of data being manipulated and the selected algorithms that have been implemented to handle that data.

Speed of processing

Speed of processing is the time that it takes a solution's algorithms to take data, perform manipulations based on that data, and produce output – for example, the amount of time it takes to search for a customer in a customer database, or the amount of time it takes to load a previously saved order.

Functionality of the software

The efficiency measures related to the functionality of the software focus on the amount of resources a software solution requires to perform its functions. This does not necessarily involve only time, and can take into account memory (RAM usage) as well as file storage (HDD usage).

Cost of file manipulation

The cost of file manipulation is an efficiency measure that measures the time and the resources a software solution requires to search and sort through data stored within files. This is not a monetary cost; instead, it is the cost in time and resource usage.

Effectiveness

The **effectiveness** of a solution relates to how well a solution achieves its intended results. This typically requires measurements of the quality of the solution in relation to its completeness, readability, attractiveness, clarity, functionality, accuracy, accessibility, timeliness, report formats, relevance, usability and **communication of message**.

Completeness

Completeness is a measure of the extent to which a solution has put in place what was required in the design brief or software requirements specification. All elements of the functional and non-functional requirements must be implemented for a solution to be complete.

Readability

A solution's **readability** is the ease with which a user can understand the text displayed to them by the software. This depends on its content as well as on how it is presented. Readability related to content involves the use of vocabulary within the software solution – how easy each word is to understand – as well as how easy sentences as a whole are to read. Readability related to presentation involves font faces, sizes and colours, spacing between lines and width of the text across the screen.

Readability is often measured scientifically, testing speed of perception, distance required to perceive the content, visibility, blink techniques, rate of reading speed, eye movements required and fatigue.

Attractiveness

The degree to which a user finds a solution to be subjectively attractive is referred to as its **attractiveness**. This is often measured with data-collection techniques that use a scale to gauge the level of attractiveness perceived by a user. A survey is an appropriate technique to gather this data.

Objective measurements of attractiveness exist, such as the use of complementary or analogous colour combinations to select harmonious colours.

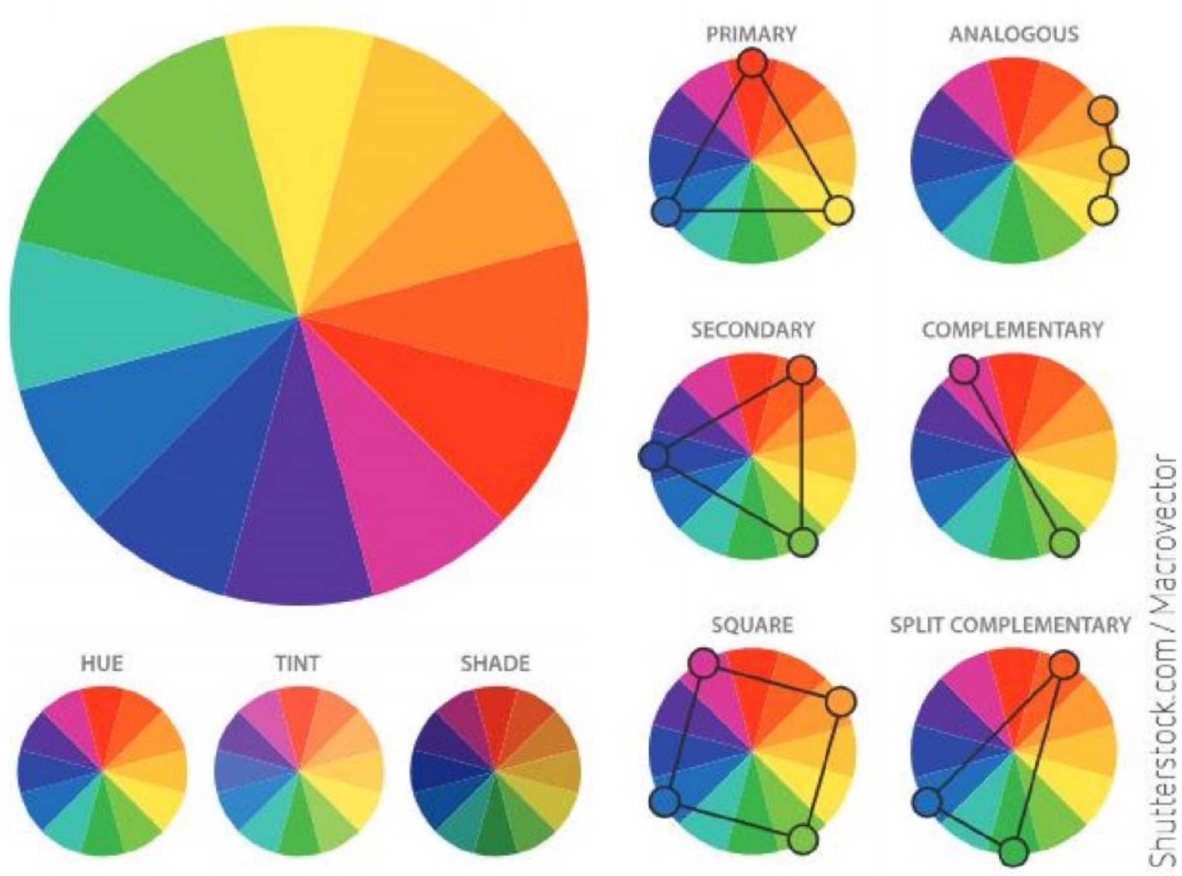


FIGURE 4.34 Different colour combinations

Clarity

Clarity measures the extent to which a solution is coherent and intelligible. This effectiveness factor is similar to readability; however, it focuses solely on the understanding the user has of the contents of the software system. This typically involves ensuring that all text is age-appropriate as well as suited to the level of skill and background knowledge of the expected user. Similarly, clarity involves ensuring that all presentation features have been used consistently throughout the software. Inconsistent use of font faces, colours and sizes reduces the effectiveness of headings and subheadings, for example, as it diminishes a user's ability to clearly distinguish between sections.

FIGURE 4.35 Example of a user interface that violates clarity measures in its presentation features

Functionality

The extent to which a software solution meets its purpose, given its ability to execute operations, is known as its **functionality**. Functionality is a way of measuring how useful the solution is to its users.

Measuring the functionality of a system typically involves collecting data over time. This can be done by tracking errors using log files, or by having users complete questionnaires or surveys in relation to the functional components of the system. It can also involve systematically checking, at the completion of a project, that all stated functional requirements of the software have been implemented.

Accuracy

A solution's **accuracy** is the degree to which it provides error-free outputs. With an accurate solution, all calculations and algorithms produce results that are expected given the inputs. Measuring effectiveness is often tied to the testing of a system. Once software has been installed and is being used, accuracy is measured by comparing actual results to expected results over longer periods of time. This may involve collecting data such as the number of times errors are found in calculations.

Unit tests are useful tools to measure accuracy across time, as they can be performed using automated software testing systems that don't require human interaction. If errors occur, such systems send an alert to a developer, or a set of developers, so problems can be fixed quickly.

Accessibility

Accessibility measures how well a solution functions for people with particular needs, disabilities or language requirements. This should not be confused with usability, as it is not a measurement of how easy a program is to use. Rather, it is a measure of a user's ability to access the software product.

For example, a software solution may need to be delivered in multiple languages if it is intended for an international audience. It may also need to have text captioning or subtitle capabilities for users who are deaf or hard of hearing, or use only certain colour combinations for users who are colourblind.

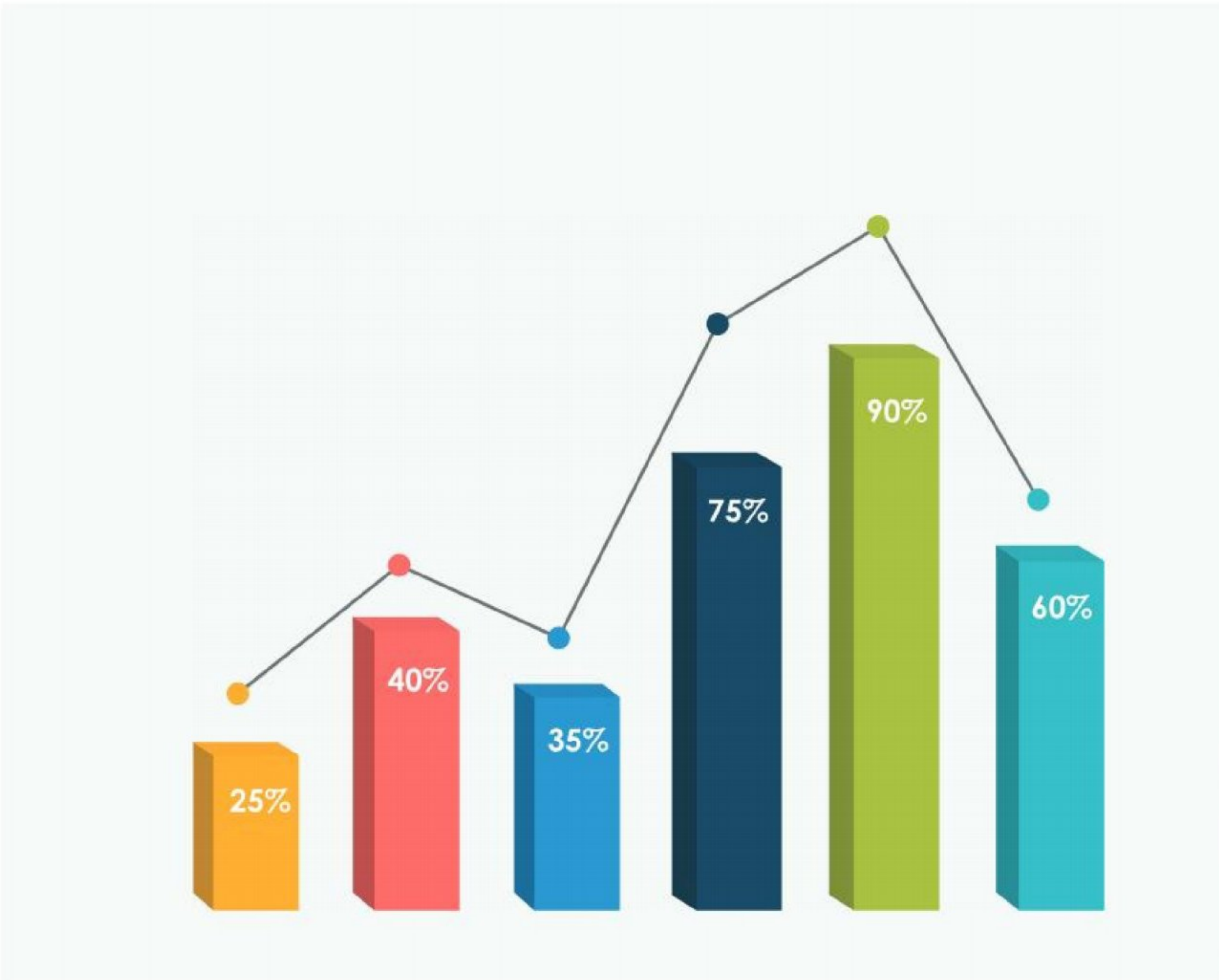
Timeliness

Timeliness is how well a solution meets expectations around the accessibility and availability of data; that is, how quickly data is read, written or displayed to a user, as well as how appropriate that data is with respect to when it was accessed.

For example, a piece of software should respond in a reasonable amount of time to a request from a user to retrieve a set of data from a file. What is considered reasonable may vary widely, depending on the context of the software solution and how much data is involved. A real-time traffic system, for example, would not be very useful if the data it outputs is days or even hours old, as it does not provide the data in a timely manner.

Report formats

The effectiveness of different **report formats** will vary according to the context in which they are delivered. For example, a report that uses infographics may be more appropriate, and therefore more effective, if it is intended to be delivered online to a wide range of users, whereas a report using histograms or stem-and-leaf plots may be more appropriate for an audience of statisticians.



Shutterstock.com/dianpurdi41

FIGURE 4.36 Infographics can be effective report formats.

Relevance

The **relevance** of a software solution is the extent to which all its elements are related and logically connected to each other. Measuring relevance could, for example, involve checking that the pathways that users can take through the solution are intuitive. This is typically done through user experience testing, also known as usability testing.

Usability

Usability is a measure of how easy and 'learnable' a piece of software is. This is generally expressed in terms of the time it takes for users to learn how to use the functionality of a system. Much like the measure of relevance, usability is typically gauged through user experience testing.

FIGURE 4.37 Some of the elements that are considered as part of user experience testing

Communication of message

Measuring the extent to which a solution effectively communicates its intended message may consider the use of region-specific formatting and conventions, such as currency or date conventions. It may also involve some of the same elements as readability, clarity, accessibility and relevance.

For example, the currency format in Germany and Greece uses a comma to mark the separation between euros and cents, and a full stop to mark thousands: €1.234,56. The currency format in Australia and the United States uses the opposite convention: $1,234.56. As well, different symbols are used to indicate euros and dollars. It is important that the selected format matches the region where a software solution will be used.

Effectiveness criteria

Examples of the criteria that could be used to evaluate the effectiveness of a solution are included in Table 4.13.

TABLE 4.13 Criteria for evaluating the effectiveness of a solution

Completeness	Were all of the client's functional and non-functional requirements implemented in the software system?
Readability	Can users easily read every part of the software program? Are the fonts chosen appropriate in size and face to the system on which the software is installed?
Attractiveness	All colours on each individual user interface should be complementary colours. Were the colour choices appropriate to the context?
Clarity	Is the language used in the software age appropriate? Are headings and subheadings used consistently throughout the software?
Functionality	Does the system respond appropriately to user input errors? The software must have 99% uptime over any 12 month period.
Accuracy	Is all of the data stored accurate in relation to how it was entered? Are all calculations accurate 100% of the time? Are all reports produced within the correct date ranges, including boundary values?
Accessibility	The software must deliver all content in English, Spanish and Greek. How well can someone who is hearing impaired or vision impaired access the system?
Timeliness	Does the software respond to requests within an acceptable timeframe?
Report formats	Are all of the reports the system produces appropriate to their contexts?
Relevance	Is all of the information produced and shown by the software system relevant to its intended use? Are there any parts of the system that are not used often (or at all) by users?
Usability	Are all of the elements within the software easy to use? Are there sections of the program where users are more likely to make errors?
Communication of message	Have appropriate and region-specific conventions been used for all displayed data (e.g. currency conventions, date/time format)?

4

CHAPTER SUMMARY

Essential terms

accessibility the extent to which a solution meets users' needs in relation to additional needs, disabilities or language requirements

accuracy the extent to which a product correctly and precisely produces outputs

alternative execution code that is run if a condition is not met

arguments specific inputs passed into a function that act as local, temporary variables

attractiveness the extent to which a product is deemed visually pleasing

binary file a computer-readable file, such as an executable program, image or sound file

Boolean a logical data type; Boolean data can hold one of only two possible values, true or false

boundary values maximum and minimum edge values possible for a given input

built-in functions functions that the creators of the programming language have written to execute common sequences of code

case-sensitive able to distinguish between upper case and lower case letters

chained conditional a conditional statement that handles more than one possible conditional outcome

character encoding code that allows a computer program to interpret binary digits (0s and 1s) into meaningful units representing characters and numbers; for example, ASCII, UTF and Unicode are types of character encoding

child element any sub-element of a parent element in an XML file

clarity the extent to which a product is coherent and intelligible

class visibility accessibility of a class from source code – public, private or protected

communication of message the extent to which a product succeeds in communicating its intended purpose

compiler a program that turns source code into machine language that can be executed by a computer processor

completeness the extent to which a software product has all the required elements

conditional statement a control structure that allows a programmer to write lines of code that are only run when a particular requirement is met; also referred to as a selection

CSV file a file type that stores data in a tabular format in plain text, with table columns separated by commas (delimiters)

debugging the process of identifying and removing errors from computer software

definition an instruction that assigns a value to a variable

delimited file a text file where data values are separated by a programmer-selected character

delimiter a character used to separate data values in a delimited file

divide-by-zero error an error that occurs when an arithmetic equation is attempting to divide by 0

DO/WHILE loop an iteration over a set of instructions, conditions and/or iterations that is repeated for as long as a condition is met; it is always run at least once

effectiveness how well a solution meets the needs of its users, measured in terms of completeness, readability, attractiveness, clarity, functionality, accuracy, accessibility, timeliness, report formats, relevance, usability and communication of message

efficiency a measure of how much time, cost and effort has been applied to achieve the intended results

event a special type of method that is called when an object's state changes

9780170440806

expected results output expected from an algorithm, assuming it is logically correct

Extensible Markup Language (XML) metalanguage that allows for user-defined tags and rules for encoding documents in a format that is readable by humans and machines

flow of execution the order in which instructions, conditions and iterations are executed or evaluated

FOR loop an iteration over a set of instructions that is repeated a set number of times

function a sequence of related code that has been given a name that can be called from other points in the source code

function call the use of code within an algorithm to execute the contents of a function

function declaration a sequence of code that names a function and its arguments

function definition to define (write) the contents of a function

function visibility the accessibility of a function from source code – public, private or protected

functionality the extent to which a solution meets its purpose

global variables variables that are defined outside any function and can be accessed by all functions throughout the source code

header comment a set of meaningful comments at the top of a source code file, outlining information such as the name of the file, its purpose, the author's name and the date of creation

infinite loop an iteration that will never reach the condition upon which it can terminate

instruction a unit of code that can be executed by a compiler or an interpreter

integrated development environment (IDE) software that provides tools to programmers to aid in programming, such as source code editing, syntax highlighting, code completion, debugging aids, or tools to help construct a user interface

internal documentation notes and code comments contained within source code that describes the code

interpreter a computer program that directly executes source code without needing to have it compiled beforehand

iteration also known as a loop; the repetition of a section of code until a condition is met

local variables variables that are defined inside a function and that can only be accessed by that function

logic error when source code is syntactically correct but contains an error resulting in unintended, undesirable or incorrect output

logical operator a Boolean operator used to combine expressions, such as AND, OR

memory leak the failure of a program to release memory that is no longer needed, causing impaired performance, application failure and/or system failure

method an action an object can carry out; e.g. window.refresh, golfClub.swing

nested conditional statement when a condition contains one or more additional conditions within its structure

parameters see **arguments**

parent element any element in an XML file that contains at least one sub-element

pass by reference to pass data into a function as an argument so that it can be modified without needing to be returned

pass by value to pass data into a function as an argument so that it cannot be modified without needing to be returned

patches sets of changes to a software application designed to update or fix it

plain text file a structured file that contains characters of readable data

pointer a programming language element that stores the memory address of another data value located in memory; the pointer 'points' to that memory space

prolog information in an XML file that appears before the start of the document's contents, including information such as the XML version and character encoding that is being used

readability the extent to which a solution is able to be read, in relation to understandability, as well as font sizes, colours and type faces

9780170440806

relevance the extent to which a solution is appropriate and closely connected to the needs of the intended users

REPEAT/UNTIL loop an iteration over a set of instructions that is repeated for as long as a condition is not met; it will always execute at least once

report formats a measure of the extent to which a solution's reports are delivered in appropriate formats

return value a value or set of values that is passed back to the origin of a calling function, often to be assigned to a variable, used in an equation, or tested within a conditional statement

root element the parent element to all other elements in an XML file

runtime error an error that occurs while a program is running

selection see **conditional statement**

sequence a set of instructions that executes line by line in the order in which it is written

statement a single line of code that, when executed, performs a single action

switch/case a conditional statement that handles more than one possible conditional outcome

syntax error an error, often typographical, in source code that violates the set of rules that define a programming language

test case a set of steps that a tester uses to determine if the element being tested works correctly, often outlining test data, testing procedures and expected results

test data data that has been specifically identified to be used in a test case

text file a structured file containing sequences of characters that are not encrypted, such as a plain text file or CSV file

timeliness the extent to which a solution delivers information at an appropriate, relevant and useful time

trace table a table used to test an algorithm, typically by hand, to ensure that no logic errors occur

tree the structure of an XML file that contains a root element and all of its sub-elements

truth table a table used to represent all of the combinations of values for inputs and their outputs, typically used to test conditional statements

usability the extent to which a system is easy to learn and use

variable a method of storing and labelling data to be referenced and manipulated in a computer program

version control system a software product that manages the revisions, changes and parallel editing of source code and its related documentation

WHILE loop an iteration over a set of instructions that is repeated for as long as a condition is met

Important facts

1. **Variables** are used to store data so they can be retrieved and changed in a program.
2. **Instructions** are a unit of code that can be executed by a compiler or interpreter.
3. **Definitions** are a type of instruction that assigns values to variables.
4. **Statements** are single lines of code that, when executed, perform a single action.
5. There are three **fundamental control structures** in programming: sequences, conditions and iterations.
6. **Sequences** are sets of instructions that execute line by line; every line of code is run in the order in which it is written.
7. **Conditional statements** are control structures that allow programmers to write lines of code that are only run when a particular conditional requirement is met.
8. **Code** within a conditional statement can contain instructions, sequences, other conditional statements or iterations.
9. **Conditional statements** are Boolean, as they are run based on the result of a condition being evaluated as either True or False.
10. **Alternative executions** are run when a condition is not met; they are typically run by using an 'else' statement.
11. **Conditional statements** can be written to include one or more logical expressions; the number of logical expressions in a conditional statement can in theory be infinite.

9780170440806

12 **Truth tables** are used to test each combination of values within a condition.

13 Truth tables can be used to help determine if **logical errors** exist.

14 **Chained conditional statements** are used when a condition needs to be tested more than once; switch/case conditionals also allow this.

15 **Nested conditional statements** occur when one condition is placed inside another condition; there can in theory be an infinite number of nested conditionals.

16 **Iterations** repeat sections of code multiple times until a condition is met.

17 There are four **types of iterations**: WHILE, DO/WHILE, FOR and REPEAT/UNTIL.

18 **WHILE loops** run for as long as a condition is met; they are used when it is unknown how many times the loop will execute.

19 **DO/WHILE loops** are like WHILE loops but will always run at least once; WHILE loops may not necessarily run if the condition is never met in the first place.

20 **FOR loops** are run a predefined number of times.

21 **REPEAT/UNTIL loops** will run for as long as a condition is not met; they will stop only when a condition is met.

22 A **function** is a sequence of related code to which a programmer has given a name.

23 When called, **code within a function** typically returns a value.

24 **Function declarations** name a function and its arguments.

25 **Built-in functions** can be used without needing to define the function yourself; they are provided by the creators of the programming language.

26 Functions can require **arguments**, also known as **parameters**, which are temporary variables that are part of the function declaration.

27 **Local variables** exist only within a function, whereas **global variables** exist throughout the whole program.

28 **Function visibility** refers to the ability of other parts of the source code to 'see' the function and use it.

29 **Public visibility** means that all parts of the source code can call the function.

30 **Protected visibility** means that only the current class or extension of that class can call the function.

31 **Private visibility** means that a function is only visible to a particular class, and cannot be seen outside that class, even in class extensions.

32 **Classes** allow programmers to group together related functions and variables in one place, creating a template for objects.

33 **Methods** are functions that exist within classes; one type of method is an event, which is a method that is called when an object's state changes.

34 **Internal documentation** exists within source code and includes the notes and comments a programmer makes about the program.

35 **Internal documentation** has no impact on the compilation or execution of software; it is ignored by the compiler or interpreter.

36 **Internal documentation** should include header comments, documentation of classes and methods, function comments, and comments describing complex portions of algorithms within the source code.

37 **Internal documentation** should be meaningful and non-trivial; it should not simply state or restate what is occurring in the code.

38 **Computer files** are resources that allow for the recording of data on any type of storage device.

39 **Text files** store data as easily readable plain text, whereas **binary files** store data in binary form, as in the case of images and sound.

40 **Plain text files** are structured files containing characters of readable data; they are commonly used for configuration settings or storing small amounts of data.

41 **Delimited files** are particular types of text files that are structured using delimiter-separated values; the most common type of delimited file is a **CSV file**, in which fields are separated by commas.

9780170440806

42 **Extensible Markup Language (XML)** files are a type of text file that can be read both by a human reader and by a computer program.

43 **XML** makes it easy to store and transport data within a system as well as between systems.

44 **XML** uses trees, roots, parents and children to separate data.

45 **Testing** is one of the most important steps in building a software program.

46 Appropriate **testing activities** include determining complete sets of test data, stating the expected results, conducting tests, recording results and then correcting any errors.

47 **Debugging** is a testing method that includes finding errors through creating trace tables and testing the system using test data.

48 Three main types of **errors** can occur in software solutions: syntax errors, runtime errors and logic errors.

49 **Syntax errors** occur when source code has not followed the predefined language rules of the chosen language; these errors prevent a solution from compiling.

50 **Runtime errors** occur when source code has been compiled and is being executed; the error causes the program to crash or print error messages.

51 **Logic, or semantic, errors** occur when the logic of a software program fails; in this case, while the code is syntactically correct, it does not produce the output that is expected.

52 **Logic errors** are best detected through the use of test data and trace tables.

53 There are four main types of **test data**: validation test data, condition testing data, iteration testing data and test cases covering all functionality from the perspective of the user.

54 **Condition and iteration testing data** involves testing for boundary values of any and all conditions and iterations within the code.

55 The general rule for **boundary testing** is that at each boundary, test data should be selected to test outside the boundary as well as inside the boundary.

56 **Truth tables** are useful to help determine test data as they show all of the potential outcomes of logical expressions used in conditions and iterations.

57 **Trace tables** simulate the flow of execution of source code.

58 **Evaluation** involves checking to see how well a software solution has met its stated requirements.

59 **Evaluation** typically takes place a certain amount of time after the solution has been put in place; the timeframe selected is relevant to the context in which the software operates.

60 Software solutions are evaluated in terms of **efficiency** and **effectiveness**.

61 **Efficiency of a solution** concerns how much time, cost and effort has been applied to achieve the intended results; this is measured against speed of processing, functionality of the software or cost of file manipulation.

62 **Effectiveness of a solution** relates to how well a solution achieves its intended results; this is measured against completeness, readability, attractiveness, clarity, functionality, accuracy, accessibility, timeliness, report formats, relevance, usability and communication of message.

9780170440806

TEST YOUR KNOWLEDGE

Review quiz

Processing features of a programming language

1 Explain how a variable differs from an instruction.

2 Which of the following are most likely to be variables?

  a seekAlternateRoute()

  b False

  c potato

  d user.firstName

  e 2019

  f isValid

  g “chocolate”

3 Categorise the following as definitions or statements.

  a intNumCustomers ← 7

  b executePrintJob()

  c strCustomerID ← getCustomerID()

4 Name three types of control structures.

5 How does a sequence differ from an iteration?

6 Pirru writes some code to loop over a set of data. Inside the loop she has a print statement that will print a number to the screen every time the source code within the loop is run. She runs the code, and it runs without errors, but nothing is printed to the screen. What are the possible iterations Pirru could have used? Why?

7 State the values that will be returned or displayed in the following examples of pseudocode.

  a

```
x ˜ 1
y ˜ 3
z ˜ 7
IF (x < z) OR (y > z) THEN
     RETURN True
ELSE
     RETURN False
ENDIF
```

  b

```
yearsService ˜ 10
IF yearsService < 10 THEN
     RETURN "Newbie"
ELSEIF yearsService < 20 THEN
     RETURN "Experienced"
ELSE
     RETURN "Expert"
ENDIF
```

8 How are functions represented in pseudocode?

9 Outline the difference between a function declaration and a function definition.

10 How is a class different from an object?

9780170440806

11 What is the relationship between a method and an event?

12 Show how methods are represented in pseudocode.

Internal documentation

13 Justify the use of internal documentation.

14 Outline the essential elements that should be included in a header comment.

15 What special characters are used to indicate comments in the programming language you are learning?

Characteristics of input and output

16 What is the difference between a text file and a binary file?

17 What are four operations that can be performed on files?

18 What is a plain text file?

19 What are delimited files?

20 What is a CSV file?

21 Discuss two advantages of using an XML file over a CSV file.

22 Describe three XML elements.

Testing and debugging techniques

23 Describe the difference between a test case and test data.

24 Explain the difference between a syntax error and a runtime error.

25 What is the purpose of a truth table in relation to testing?

26 Explain what a boundary value is.

27 Why is it important to use test data that focuses on boundary values?

28 Provide two reasons why trace tables are useful when debugging.

29 Complete a trace table for the following algorithm:

```
ALGORITHM computeResult()
BEGIN
     x ˜ 7
     y ˜ 1

     WHILE y < x DO
          y ˜ x * y
          x ˜ x + x
     ENDWHILE
     PRINT x, y
END
```


9780170440806

TEST YOUR KNOWLEDGE

Evaluating efficiency and effectiveness

30 Describe the steps involved in an evaluation.

30 What is an evaluation strategy?

31 Outline the elements that can be used to measure a solution's efficiency.

32 Outline the elements that can be used to measure a solution's effectiveness.

9780170440806

Major programming task: Hangman

Your task is to design, develop and evaluate a program that will allow a user to play the game 'Hangman'. The program will randomly select a word from a list that the user will then attempt to guess by suggesting letters from the alphabet. Each time a letter is guessed incorrectly, an element of a hanged man is drawn. If all elements of the hanged man are drawn, the player loses. If the player guesses all of the letters in the word, or guesses the word itself, before the hanged man is drawn, the player wins. A player can guess the word at any time, but if they guess incorrectly, an element of the hanged man is drawn.

1 Describe the scope, constraints and functional requirements of the program.
2 Design the program's interface with a mock-up.
3 Write pseudocode to describe the processing strategies involved in developing the program.
4 Develop the solution, using good naming conventions and internal documentation.
5 Construct a set of test data that will test all conditions – for example, when a letter is found, when more than one letter is found, and when no letter is found.
6 Test the solution and fix all bugs.
7 Write a report evaluating your solution in terms of efficiency and effectiveness.

If you haven't played Hangman before, you can find out the rules at the weblink. There are many online Hangman games you can search for and play as part of your research to create your own version of the game.

9780170440806

PREPARING FOR

Unit 1

OUTCOME 2

On completion of this unit the student should be able to design, develop and evaluate a software solution using a programming language in response to a teacher-provided design brief.

Steps to follow

To achieve this outcome, you will draw on key knowledge and key skills outlined in Area of Study 2. This Outcome requires that you use a programming language to create a working solution to a teacher-provided design brief, undertaking the problem-solving activities of design, coding, validating, testing, creating internal documentation and evaluation. You are required to prepare and use a project plan to monitor your progress of the solution that you are writing. Your project plan should record all identified tasks, their sequencing and the time allocation of each task, all milestones and all dependencies, and some indication of the critical path. This project plan does not need to be prepared using dedicated project management software; for example, using spreadsheet software would be sufficient, as long as all of the expected elements are demonstrated within the project plan created.

To encourage you to meet all of the requirements fully, you should follow your project plan, making annotations and corrections as you monitor your progress in completing the solution. Ensure you read the design brief your teacher gives you carefully, paying close attention to scope, constraints, functional requirements and non-functional requirements.

Documents required for assessment

- Project plan, with annotations and modifications
- Design documents: data dictionary, mock-ups, pseudocode
- Source code and all related files (e.g. images, data files)
- Internal documentation within the source code
- Evidence of testing using appropriate test data
- All documents used to construct your tests, including test data, any truth tables used to select your test data, and trace tables; these should be kept and submitted
- Evaluation report of the solution, including reference to efficiency and effectiveness criteria

Assessment

You will be assessed on the following measures:

- the accuracy and completeness of your project plan
- the accuracy and completeness of your design documents
- your choice of data types and data structures
- your choice of file types
- appropriateness of processing features selected, created and used
- use of programming naming conventions and file naming conventions
- validation techniques
- internal documentation
- debugging techniques
- thoroughness of testing
- the accuracy and completeness of your evaluation report.

9780170440806

Unit 2

INTRODUCTION

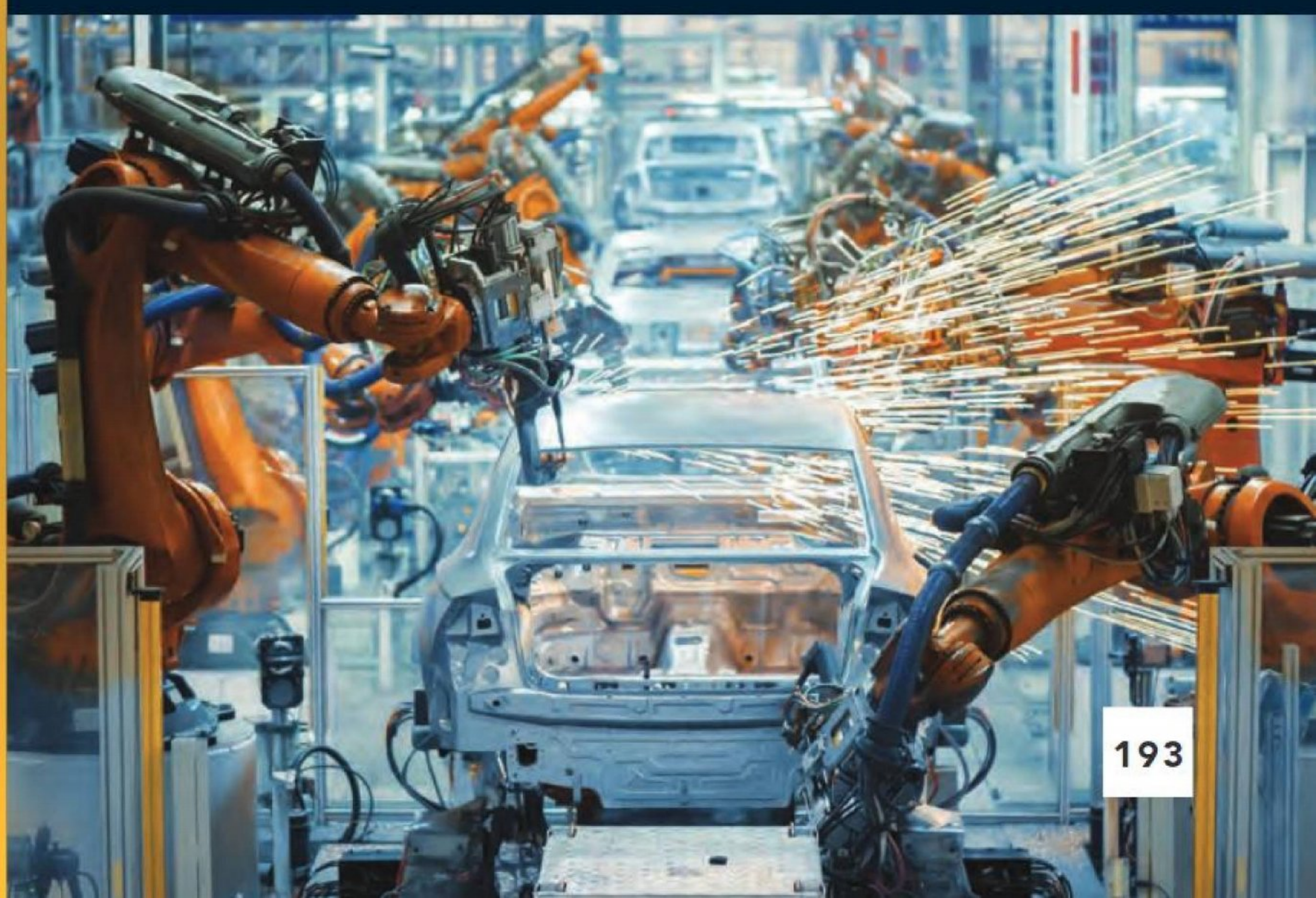
During Unit 2 of Applied Computing you will look at a range of emerging and innovative technologies, such as augmented and virtual realities, blockchain, drone and wearable technologies and the Internet of Things (IoT). You will gain an understanding of how networks are used to communicate data and information, identify threats to data and information in a networked environment and investigate a range of security measures that can be used to protect against these threats.

Area of Study 1: Innovative solutions

OUTCOME 1 You will work collaboratively in groups to develop an innovative solution to an identified need or opportunity. Each group will follow the problem-solving methodology to create a solution to the need or opportunity using innovative or emerging technology. Each group will also prepare a progress plan to assist in the management of the task.

Area of Study 2: Network security

OUTCOME 2 Your teacher will provide you with a case study. You will be required to examine the capabilities and vulnerabilities of a network, design a network solution, discuss threats to data and information and propose security strategies to protect the data and information.



Shutterstock.com/Jenson

9780170440808

CHAPTER

5 Approaches to problem solving

KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

Digital systems

- components of digital systems
- types of digital devices used for a range of current and emerging applications such as smartphones, smart refrigerators and virtual assistants
- emerging trends in digital systems and the importance of innovation to organisations, such as improving efficiency and effectiveness of customer service and maintaining competitiveness
- functions and capabilities of digital systems used by individuals and organisations, such as assistive technologies, financial services, global positioning system (GPS) devices, robotics and traffic management

Interactions and impact

- goals and objectives of digital systems
- economic issues involving emerging technologies, such as access, deskilling, job loss, misuse and sustainability
- the impact of current and emerging technologies, such as automation, cyberbullying and the decline of physical human interactions and interpersonal skills

Reproduced from the VCE Applied Computing Study Design (2020<en dash>2023) © VCAA; used with permission.

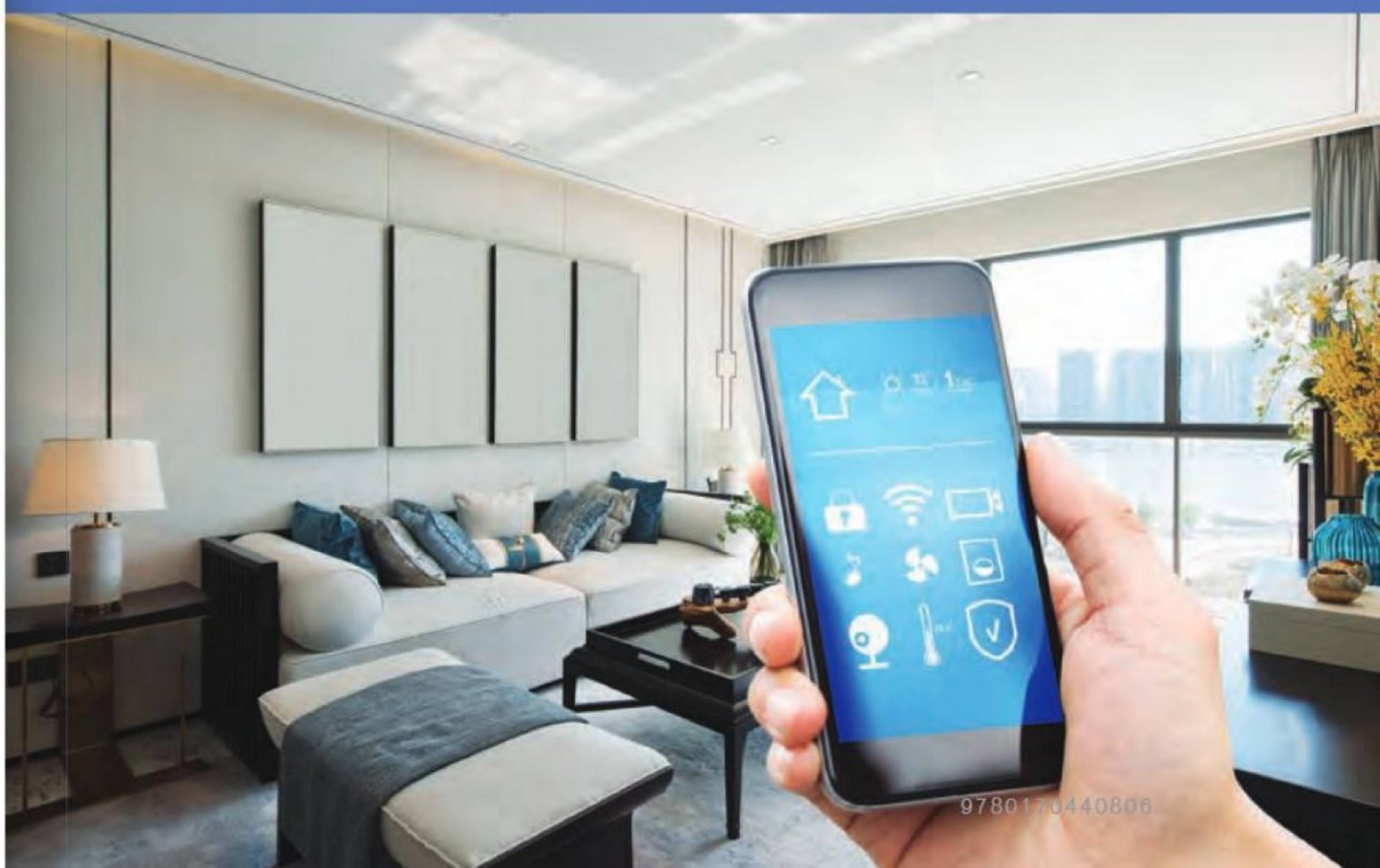
FOR THE STUDENT

This chapter, along with Chapter 6, covers the content for Unit 2, Area of Study 1, Innovative solutions. For this unit you are required to work in groups to analyse, design, develop and evaluate an innovative solution to an identified need or opportunity involving a digital system. In this chapter the components used in digital systems, the types of digital devices used and emerging trends in technology are introduced. The goals and objectives of digital systems and a range of issues related to the use of technology are discussed.

FOR THE TEACHER

In this chapter a range of technology used in current and emerging technology is introduced.

For Unit 2, Outcome 1, students are expected to work in groups to create a solution to an identified need or opportunity. This may be achieved using a programming language, spreadsheet software, web-authoring software, presentation software and a tool for planning a project. Chapter 6 deals with approaches to problem solving, and also covers knowledge required to complete Unit 2, Outcome 1.



Shutterstock.com/zhudifeng

9780170440806

Components and emerging trends in digital systems

You were introduced to digital systems in Chapter 3. A **digital system** can consist of a single device, a collection of devices or a network. A digital system is used to transform data into information, providing output that meets the needs of the user. Digital systems are found in a wide range of applications, including smartphones, drones, microcontrollers and robotic devices. Desktop and laptop devices are common examples of a typical digital system.

A digital system consists of hardware and software. To complete a task using a device, both hardware and software are essential. Refer to Chapter 3 for more specific details of digital systems.

As technology evolves, and advances are made in the development of microchips and smaller and more powerful processors, the use of a range of devices beyond traditional desktop or laptop computers continues to grow. Digital technologies are now used in a range of devices that previously did not use such technologies, and new and innovative uses of technology are constantly evolving.

Emerging technologies are technologies that are perceived to be capable of making significant change within society. There are many ways in which an organisation can use emerging technologies to make improvements in the way they operate and the products and services they offer. The increasing **datafication** of society, the increasing number of smart devices in everyday life, the constant growth in both processing power and miniaturisation of technology, the increasing use of **automation** and the rise of artificial intelligence can, and do, have a significant impact on both individuals and organisations.

'Datafication' is a term that is used to describe the process of turning aspects of everyday life into quantifiable, measurable data. This can include the number of hours of sleep a person has each night, the kilojoules they consume, the number of steps they take or the hours they spend sitting or standing each day.

Organisations can use this data to find new customers, increase customer retention, improve customer service, improve marketing campaigns, track social media and predict sales trends. This data can be used to inform their long-term strategic and operational decisions, leading to an increase in their productivity and profitability.

The online streaming service Netflix has risen to prominence by using data. The Netflix algorithm collects data about what users are watching, what they are searching for and what they are skipping over. This data can be used to create a tailored list of movie recommendations or to commission a new series. Netflix is one of the most successful online companies.

In this Area of Study, you will investigate the use of digital devices in emerging technology. Then you will work collaboratively, following the problem-solving methodology, to develop an innovative solution. The various items of technology described below are examples of the types of technology you could consider for this task.

Smartphones

A **smartphone** is a mobile telephone integrated with a digital system allowing functionality not previously associated with telephones, such as an operating system, web browsing, email and the ability to run software applications. The first smartphone evolved out of personal digital assistant (PDA) devices. These offered a full keyboard and allowed users to input, store and retrieve data, including calendar, address book and note-taking functionalities, as well as the ability to install additional software applications.

Early PDAs were stand-alone devices whose users were required to synchronise their portal device to their desktop computer using a wired connection. In 1992, IBM released a device named Simon that offered wireless connectivity, allowing users to send emails and

Shutterstock.com/Lukmanazis

FIGURE 5.1 A smartphone is more than just a phone.

faxes. In 2007, LG and Apple released smartphones that featured a touchscreen, and in 2008, HTC released a smartphone that included the Android operating system.

An app store or app marketplace is a portal that is used to distribute software applications (apps). App stores distribute apps that are either operating-system based or phone-manufacturer based. Because smartphones use an operating system and software applications, they require regular software updates, as improvements and fixes to the software are developed. These updates are often managed through the app store.

Key features of a smartphone are an embedded motherboard that includes a processor, memory and storage space, internet connectivity, a physical or onscreen keyboard, camera and GPS capabilities. Smartphones often offer multiple communication methods, such as connectivity via mobile networks or through wi-fi or Bluetooth connections.

One of the biggest advancements in smartphones was the ability to connect to an app store. This allowed users to search for, and download, software applications to run on their phones.

Smart refrigerators

An example of a regular household appliance that has been integrated with technology is the refrigerator. **Smart refrigerators** offer the functionality of a regular refrigerator but also offer the ability to connect to the internet through a wi-fi connection. Smart refrigerators can include a touchscreen, internal cameras, a range of temperature-controlled zones, and connectivity through a smartphone app. Some smart refrigerators can also connect with other smart devices in the home, including dishwashers, microwaves, televisions and virtual assistants. Features included will vary by brand and model.

The functionality a smart refrigerator offers may include the ability to:

- control temperature by drawer or compartment
- view the contents of the refrigerator using an internal camera
- send alerts when the refrigerator encounters a problem
- control other settings using a smartphone app.

In some models a barcode reader is installed, allowing products to be scanned when they are placed in or taken out of the refrigerator. This creates a list of those items the refrigerator currently contains, which the owner can view through a built-in display or via the smartphone app. They can then use this information to create a shopping list.

The touchscreen on a smart refrigerator allows the user to add the use-by dates of particular items, create shopping lists and even broadcast an image from a smart television. Other smart refrigerators have sensors, and doors that open automatically when the sensors detect touch. The same sensors are used to detect if the door is left open, in which case it will be closed.

The technology used in smart refrigerators includes a microcontroller consisting of a motherboard, processor, memory and input/output controllers, which is connected to the

Getty Images/Bloomberg

FIGURE 5.2 Smart refrigerators can connect to other home appliances, such as dishwashers and microwaves.

various hardware of the refrigerator including sensors, camera, barcode scanner, thermostat, internal display and keyboard. Wi-fi connectivity also allows smart refrigerators to receive software updates, containing software fixes and advances in functionality.

Virtual assistants

A **virtual assistant** is a software agent that can perform tasks or services for an individual. Virtual assistants are also known as intelligent personal assistants. A software agent differs from regular software applications in that it is capable of acting with some degree of autonomy to complete tasks. Each software agent is designed to complete specific tasks or behaviours.

The first modern virtual assistant was Apple's Siri, which came pre-installed on the iPhone 4S in 2011. Other well-known virtual assistants include Amazon's Alexis, Microsoft's Cortana and Google's Now.

Virtual assistants can be used:

- in instant messaging applications, to analyse text input by the user and formulate a response
- in instant messaging and telephone applications, to analyse and respond to audio input
- with embedded smart speakers, such as Google Home, allowing the user to request information, then analysing the request and retrieving information.

If the virtual assistant is embedded within a smart speaker, a user can ask the assistant questions, control other digital devices within the home that are connected, and play media. In business, virtual assistants are used in completing online transactions – for example, booking travel and hotel arrangements or scheduling meetings.

Virtual assistants are also used in the health field to monitor a user's daily activity, energy intake and heart rate. The assistant analyses the data and provides the user with feedback on how to improve their health.

Some software developers are releasing third-party apps that can be installed in addition to the virtual assistant to provide additional functionality on digital systems.

Emerging trends

The use of virtual assistants in business is helping organisations to become more effective and efficient in the way that they operate. Many workers within organisations use online calendars to schedule appointments, commitments and meetings, and these online calendars

THINK ABOUT APPLIED COMPUTING 5.1

What other smart household devices are there? What extra functionality do these devices have apart from their primary function?

THINK ABOUT APPLIED COMPUTING 5.2

List at least three organisations that you have connected with that use an automated phone system. What are the advantages and disadvantages of these systems?

9780170440806

can be shared with other users. Virtual assistants have been created that, given the right permissions, can schedule appointments and meetings automatically, at a time when all participants are available. The software will search through all the shared calendars, select a time when everyone is available and then send notifications about the meeting, and any subsequent alterations, to the participants.

Virtual assistants can also be used to connect with an organisation's customers directly. An organisation can install virtual assistant software applications that allow customers to interact with the organisation – for example, to order food, listen to a particular song or book a hotel room. Actions can be created that allow the task to be completed. In addition, an organisation can collect data about each user's history and interests to be able to customise services that will best meet the user's needs.

Virtual assistants can also be used to allow organisations to monitor the status of networks and information systems. With software correctly configured, a user can ask for the current performance of a system and receive real-time feedback about the status of websites, networks or database management systems. It is also possible for a virtual assistant to send an alert to a user if any issues or security alerts occur.

Voice-based virtual assistants can control utilities such as electricity, lighting and heating. By collecting data via voice command, the digital system is able to recognise activity in the office space and turn appliances on or off. A history of usage of office space can be collated over time, which will also allow the assistant to make decisions based on past history.

Drones

A **drone**, or unmanned aerial vehicle (UAV), is an aircraft without a human pilot aboard. Some drones can be controlled by a user on the ground, who communicates with the drone via a remote control. Other drones can fly with a degree of autonomy, using instructions pre-programmed in the drone's onboard microcontroller.

Shutterstock.com/stevemart

FIGURE 5.3 A drone is an aircraft without a human pilot.

Initially, drones were used mainly for military purposes. An unmanned aircraft would be sent on a mission that was considered routine (e.g. flying over a pre-programmed path to take surveillance footage) or one that was too dangerous for a human pilot.

Drones are now used in a range of areas including agriculture, postal delivery services and scientific research. In particular, the use of drones for recreational use has become mainstream in recent years, due largely to the decreasing cost of purchasing a drone.

Drones come in a wide variety of sizes, with the largest being similar in size to a small aircraft and requiring a runway for take-off. Vertical take-off and landing (VLOT) drones can take off and land vertically, while some drones are small enough that they can easily be launched from the hand of the user.

Features used in drones include:

- a microcontroller (sometimes referred to as the flight controller)
- position and movement sensors
- actuators that control movement in the propellers and motor
- flight stack software (to allow the drone to fly on a set route via several waypoints)

- GPS technology
- obstacle and collision avoidance technology
- stabilisation technology
- a ground station controller (GSC) or a smartphone app
- internal compass and fail-safe function
- camera.

Manufacturers release software updates to fix bugs or to add extra functionality to the drone. In recent times, updates have included information related to no-fly zones, which, when used in conjunction with the drone's GPS technology, can alert users if the drone is flying too close to an area in which their use is prohibited.

5.3 THINK ABOUT APPLIED COMPUTING

What are some issues associated with the use of drones? Find out if there are any regulations that apply to flying drones in your local area.

Emerging trends

Drones are used in a wide variety of ways to improve efficiency and effectiveness and allow organisations to gain a competitive advantage.

Agriculture is an industry in which there are many uses for drones. The large land mass or size of farms and stations, the difficulty of access to the land and the advantages associated with an aerial view make drone use desirable.

Drones can save farmers money by reducing the time needed to travel across their property. Drones can be equipped with cameras to allow farmers to view their land from above and identify issues with soil quality, weed control, irrigation or livestock.

Drones can be equipped with hardware that allows the spraying of pesticides, fertilising or even watering to maintain crops in a time-efficient manner.

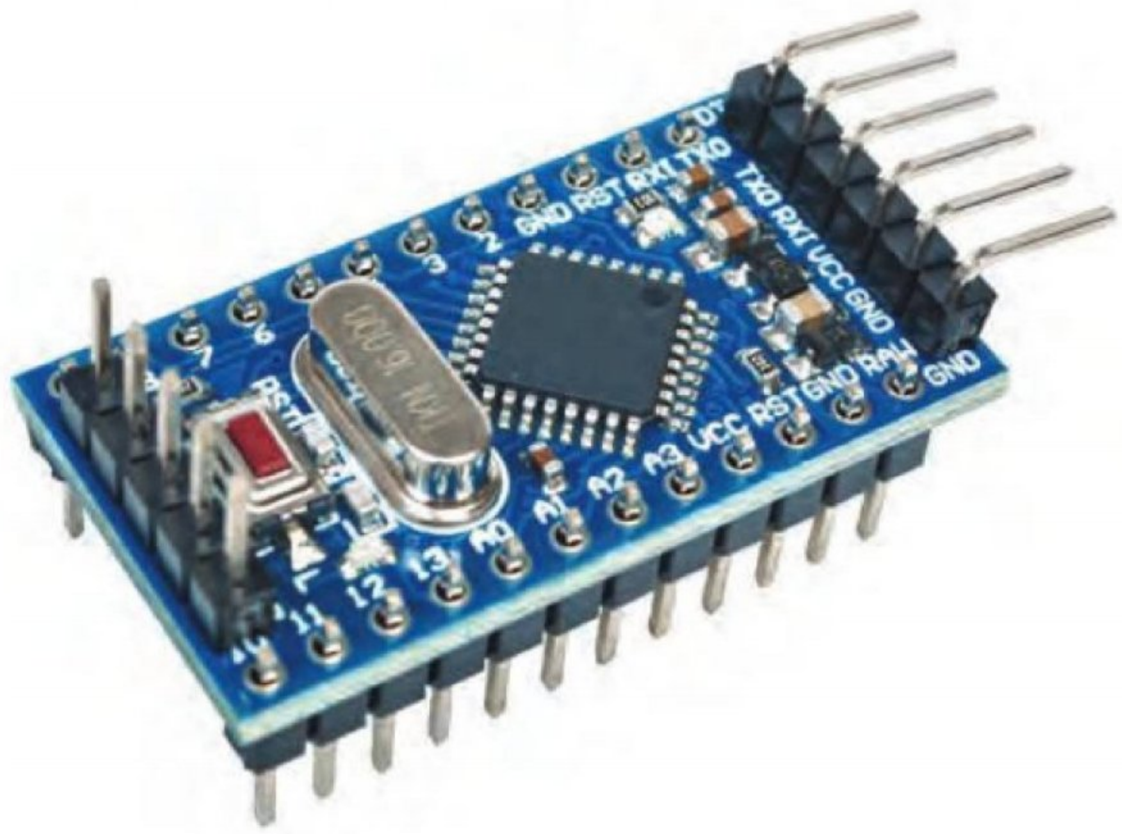
© 2019 Clare Farrelly, used under license from Austockphoto

FIGURE 5.4 Drones can aid in increasing crop production and monitoring crop growth.

Microcontrollers

A **microcontroller** is an integrated circuit. An integrated circuit is a small chip that holds anywhere between hundreds and millions of transistors, resistors and capacitors. A microcontroller incorporates a processor, memory and input/output (I/O) controllers within the one chip. A microcontroller's processor can vary from 4-bit to 64-bit.

Microcontrollers are used in embedded systems. An embedded system is designed to perform one or two specific processing tasks. A microcontroller is embedded within another device such as a refrigerator or a microwave oven, where it performs limited tasks.



iStock/Getty Images Plus/viabo

FIGURE 5.5 A microcontroller

This is different to a general-purpose system, as found in desktop and laptop devices, which is designed to perform a wide range of tasks. Microcontrollers can be found in a range of devices including televisions, smartphones, cars, printers, vending machines and drones.

A microcontroller takes input from the device it is controlling and then sends signals to different components in the device in which it is embedded. Some microcontrollers are connected to a small display to allow the output of data to the user, such as the clock/timer on an oven.

Microcontrollers are designed to be self-sufficient and to be readily usable without additional components because they incorporate the processor, memory and input/output controls used to connect with the components of the device. Microcontrollers differ from microprocessors in that the processor, memory and I/O controllers of a microcontroller are incorporated in the one chip, while a microprocessor contains only the processor within the integrated circuit while memory and I/O controllers are located in separate chips.

As a microcontroller is designed to perform a limited amount of tasks, they can be less expensive to develop compared to general-purpose controllers and are also very efficient at processing data.

Nanosatellites

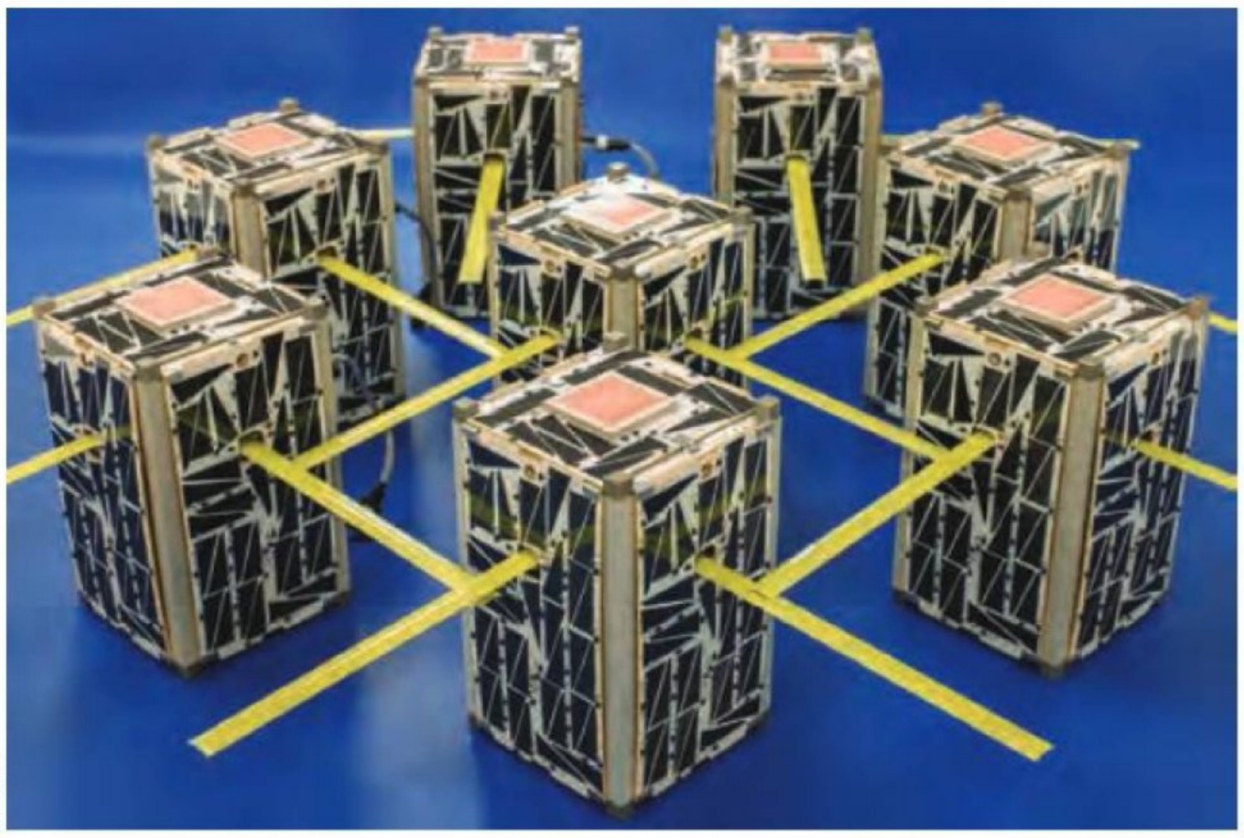
A **satellite** is an object in space that orbits around a bigger object. Satellites can be either natural objects – such as Earth, which tracks around the Sun, or the Moon, which circles Earth – or human-made objects. The great majority of human-made satellites are launched to perform a specific task. They include communications and navigation satellites. Communications satellites distribute data around Earth by receiving signals from a base station and transmitting them to another base station in a different country.

A **nanosatellite** is a small satellite that may weigh anywhere between 1 and 10 kilograms. As with drones, smaller and smaller satellites are being produced that offer the same functionality as traditional satellites at a fraction of the cost. Some nanosatellites are designed to operate independently, while others work as part of a satellite swarm, in which smaller satellites communicate with a mother satellite located nearby.

As the barriers to the use of satellites continue to drop, the number of private organisations – not just governments – that use nanosatellites is starting to increase.

One such implementation of a nanosatellite is in the form of CubeSat technology. A CubeSat is a miniature satellite made to a standard specification. Each cube has a height and width of 10 cm and a length of 35 cm and a weight of between 1 and 2 kilograms. A number of CubeSats can be joined together to create a larger nanosatellite. Within each CubeSat, a number of microcontrollers are used for processing information, along with a range of other circuit boards.

Find out more about nanosatellites by exploring the nanosats website.



NASA

FIGURE 5.6 Several CubeSats can be joined together to form a larger nanosatellite.

Features of nanosatellites include solar panels that are used to charge onboard batteries, altitude control to ensure that the correct orbit is maintained, technology to allow the nanosatellite to communicate, and thermal management to ensure the device continues to operate in extreme weather conditions.

Besides the cost of developing a satellite, the cost of launching a satellite is significant. The advances in technology that allow the development of smaller satellites has also decreased the cost involved in launching. Each nanosatellite needs a propulsion method to allow the device to be launched into space. Some satellites use electric-based propulsion, while others use solar-based propulsion.

Nanosatellites are often launched in swarms into what is known as low-Earth orbit, as opposed to traditional satellites, which are launched into deep orbit. This also assists in reducing the cost associated with the technology. Satellite swarms allow a larger coverage of Earth's surface and this can lead to greater and more detailed data collection about the planet. In satellite imagery of Earth, images of the same location can be captured more frequently, giving researchers access to more timely information. For example, in the field of natural disasters, the improved frequency of images may allow researchers to interpret patterns more accurately and make better forecasts of potential dangers.

Emerging trends

A swarm of nanosatellites is used to capture images of Earth on a daily basis. As each individual nanosatellite works in conjunction with the swarm, large areas can be photographed. These images are sent to a ground station that allows a complete profile of Earth to be compiled each day. This provides real-time information to assist in agriculture, climate monitoring, firefighting and other natural disaster responses, among other uses.

Nanosatellites are increasingly being used for the transmission of data. Traditional satellite communication has been an expensive form of data communication. There have been issues with reliability as it can be dependent on the location of the few larger satellites orbiting Earth, along with the availability of bandwidth. Using a swarm of smaller, less expensive nanosatellites, it is possible to provide satellite communication to a wide catchment area of Earth, allowing areas with little existing internet infrastructure to access fast and inexpensive connections.

Because of their expense, traditional satellites tended to focus on areas of Earth that were heavily populated, but nanosatellites are increasingly being used in areas that are less populated. The shipping industry benefits from the use of nanosatellites. The route and progress of ships can be tracked and documented, and potential dangers and suspicious activity identified.

Environmental events can be monitored using nanosatellite technology. Individual natural disasters, deforestation effects or ocean levels can be monitored more closely and with greater frequency, giving researchers detailed information with which to analyse the significance of these events.

Augmented reality

Augmented reality (AR) is used to enhance environments or situations and offers perceptually enriched experiences. It involves overlaying computer-generated images with real-life images to provide a composite view.

A popular example of AR is the mobile game Pokémon Go, which was released in 2016. The game allows users to view the world around them through their smartphone camera. Items including icons, scores and creatures were projected as overlays on the screen. The effect was as if those items were actually present in the real world.

To be able to operate AR, a device is needed that contains a processor, a display, sensors and input devices, including a camera. Eyeglasses, head-up display and contact lenses can also be used to view AR images. In addition, AR requires the use of an accelerometer, GPS, a gyroscope and a compass to assist with the location, movement and orientation of the device. AR software must be installed on the device. It will use the device's resources and create the overlay of visual images.

AR software can operate in a number of different ways. Marker-based software operates using a visual image or marker – for example, a QR code. Once the device recognises the

Shutterstock.com/Zapp2Photo

FIGURE 5.7 Augmented reality results when a computer-generated image overlays, or augments, the real world.

marker, an image will appear on the display of the device. This could be a three-dimensional (3D) version of the image used in the marker.

Other AR software is markerless. This is used in location-based situations, where the device's GPS, compass, gyroscope and accelerometer are used to determine its location. Depending on the location, relevant content will be provided – for example, restaurants or shopping centres in the nearby area (Figure 5.7). Often, markerless AR systems are used in conjunction with mapping software to offer the functionality of providing directions to particular points of interest.

Another type of AR software functions by projecting light or a hologram onto an object.

5.4
THINK ABOUT APPLIED COMPUTING

What is the difference between augmented reality and virtual reality?

Virtual reality

Virtual reality (VR) involves using a digital system to generate 3D images that the user can interact with in a simulated environment using specialised equipment, such as a helmet with a screen inside or gloves fitted with sensors.

Virtual reality provides the user with an experience completely different from the real-world environment around them. It differs in this way from augmented reality, which enhances the user's existing environment.

Many virtual reality set-ups involve the use of a headset incorporating a range or visual displays and sensors. The visual display within the headset broadcasts a virtual environment to the user, immersing them in a virtual world where they can interact with objects. To achieve this interaction, a range of controllers, gloves and external sensors are used to track the user's movements.

Virtual reality is used in a range of fields, including for military, sporting, healthcare and educational purposes.

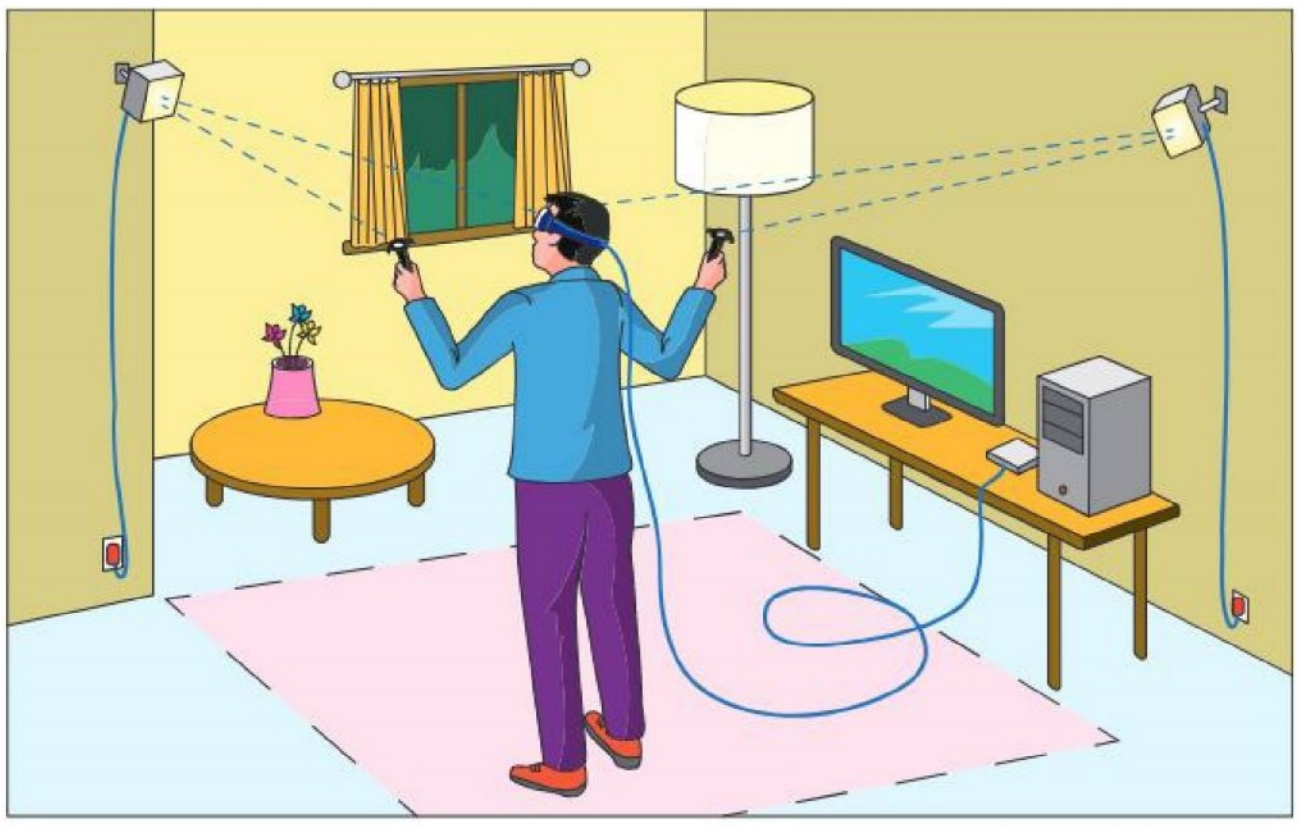
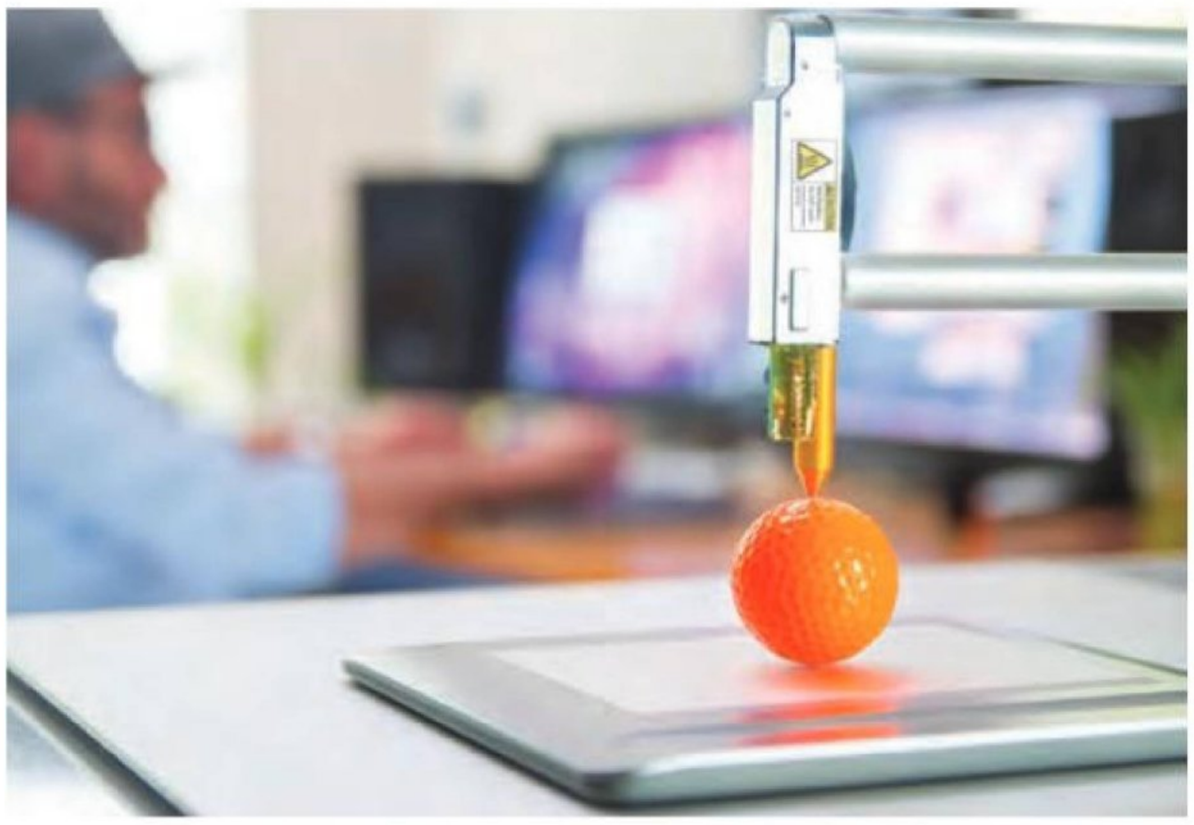


FIGURE 5.8 In virtual reality, sensors track the user's movement and a visual display simulates a different environment.

3D printers

Three-dimensional (3D) printing involves creating a physical object from digital data. First, a 3D model needs to be created, generally using CAD software. Once completed, the model can be created by printing many thin layers of a material together.



Shutterstock.com / Alexander Kirch

FIGURE 5.9 3D printer

There are many types of 3D printing technology available, each creating a physical object using slightly different methods. Regardless of the particular technology used, 3D printers usually contain similar components. These include:

- a frame that holds the printer together
- a bed onto which the object is printed
- a nozzle that is used to deposit the material from which the object is made
- head movement mechanics that move the nozzle across the print bed
- stepper motors used to position and control the speed of the nozzle
- motors to provide power to the printer and the cooling system
- firmware software that controls and manages the device
- application software used to convert the CAD file into a physical object.

The most commonly used 3D printing method is fused deposition modelling (FDM). FDM works by loading plastic or metal filament into the printer nozzle. The nozzle is heated and a motor pushes the filament through the nozzle, causing the filament to melt. The printer moves the head to the specified location, laying down the filament onto the bed, where it cools down and solidifies. Once one layer is complete, the next layer is applied. This process is repeated until the physical object is fully formed.

A stereolithographic (SLA) printer uses mirrors that aim a laser beam over a pot of resin, selectively curing and solidifying the physical object and building it up layer by layer. SLA can create high-resolution 3D objects, but can be considerably slower than other forms of 3D printing technology. Digital light processing (DLP) works in a similar way to SLA printing but, rather than using laser beams, it projects light onto resin using LED screens or a UV lamp that is directed to the build surface. Powder bed fusion is a 3D process in which thermal energy is used to bind powder particles inside a build area to create a physical object.

THINK ABOUT APPLIED COMPUTING 5.5

Research the different objects that can be made using 3D printing. What type of 'ink' is needed to make each?

Early uses of 3D printing included making jewellery, household gadgets, and model car and plane accessories. As the technology matures, it will be used to print body parts, houses, car parts and food, and in the restoration of relics.

Emerging trends

The use of 3D printing technology can save organisations time and money. In the development of new products, it is common after the design stage to create a mould or prototype of the

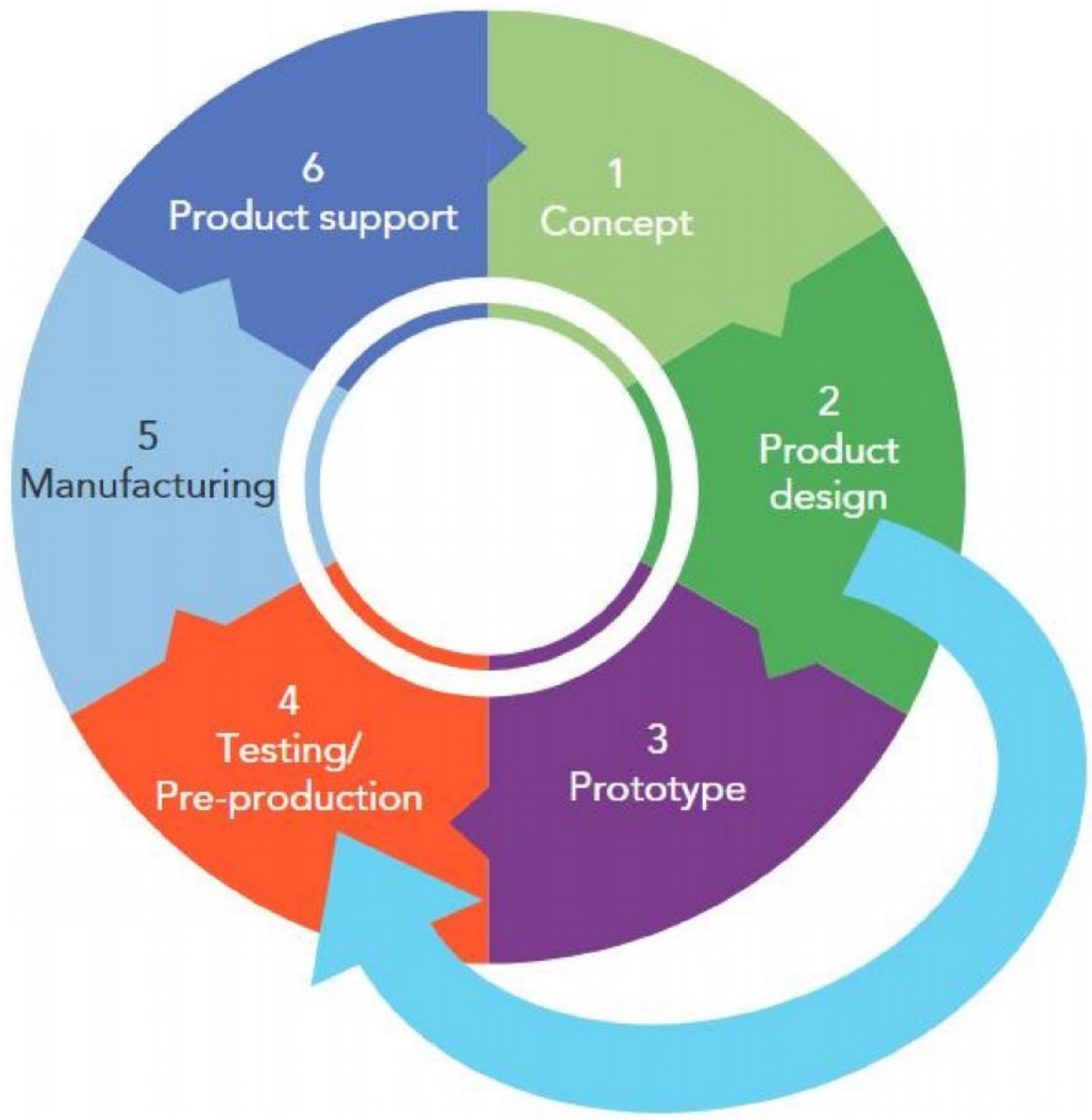


FIGURE 5.10 3D printing makes it possible to skip the prototype stage in the product development life cycle.

product before production begins. This is a time-consuming and costly step in a product's life cycle. With the use of 3D printing technology, it is possible to skip this step and begin to manufacture the product without the need for a mould or prototype.

3D printing technology allows organisations to make changes to a product design after production has commenced. Traditionally, once a product has been designed and production started, changes could only occur periodically. This would involve completing the design stage again and creating a new mould or prototype. It is now possible to make tweaks or changes to a product, based on feedback, to provide a product that best suits the customers' needs.

Wearable technology

Wearable technology is the name given to small smart devices that can be worn on, or implanted into, the body or woven into clothing. Examples of wearable technology include fitness trackers, smart jewellery and watches, and head-mounted displays.

Advances in technology have resulted in the miniaturisation of components, creating a whole range of new possibilities for the ways in which digital system devices can be used.

Wearable technologies not only perform general-purpose tasks similar to laptops or mobile devices; they also provide users with the functionality to perform specific tasks geared towards the user's particular needs. Wearable technology has been around for a number of years and has begun to achieve mainstream usage, and the range of products and functions offered is rapidly developing.

Implantable devices are either inserted onto, or implanted under, the skin. They include pacemakers and defibrillators, which collect biometric data from the user and transmit this data, using sensors, to a smartphone or laptop. Wearable tattoos, which are worn on the skin, can also be used to collect biometric data. These tattoos are often designed to change colour to alert the user of potential problems.

Smart watches are the most common type of wearable device. Smart watches offer a range of functions similar to those of a smartphone in addition to functions such as monitoring

Shutterstock.com/Alexey Boldin

FIGURE 5.11 Synchronising a smart watch with a smartphone

UV rays or pollution levels. Smart jewellery is used to inform a user of any activity such as calls, messages or emails they may have missed while they are unable to access their phone.

Activity or fitness trackers, normally worn on the user's wrist, often contain a pedometer to count steps, accelerometers to measure speed, and altimeters to assist in calculating physical activity. They can also include GPS with mapping software so location can be tracked. Some activity trackers also contain heart-rate monitors and have the ability to calculate energy or kilojoules expended.

5.6 THINK ABOUT APPLIED COMPUTING

Describe one situation in which a head-mounted display could be used.

Smart clothing is used to monitor a user's physiology. It provides biometric data, such as pulse rate, temperature, muscle stretch, heart rhythm and physical movement. Smart clothing involves microcontrollers, sensors and actuators that are integrated into the clothing using Bluetooth technology to transfer the data to a smartphone or laptop application.

A head-mounted display (HMD) is a display worn on the head as glasses or as part of a helmet. It has a small display in front of one or both eyes. Uses of HMDs include aviation, engineering, gaming and sport. The display may use a cathode ray tube (CRT), a liquid crystal display (LCD) or organic light-emitting diodes (OLEDs) to project the image to the user. Most HMDs display only a computer-generated image, while others can superimpose computer-generated images over real-life images (augmented reality).

Emerging trends

Workplace health and safety issues can be monitored through the use of wearable technology. Workers can be constantly monitored and alerts can be sent if factors that predict a possible event are detected – for example, workers working extended hours or driving for too long, or weather conditions that are becoming extreme.

Internet of Things (IoT)

The **Internet of Things (IoT)** refers to the billions of devices around the world that are connected to the internet, collecting, receiving or sending data. The term 'IoT' is mainly used for devices that traditionally would not be expected to have an internet connection, and that can communicate independently without the need for human interaction.

Thanks to advances in technology and the miniaturisation of components, along with a reduction in production costs, the range of devices that can now connect to the internet

iStock/Getty Images Plus/undefined

FIGURE 5.12 Representation of the Internet of Things (IoT)

has increased significantly. This has increased the functionality of everyday devices as they can be used to collect data as input, communicate the data to other devices and also receive instructions from other devices. Such devices may include virtual assistants, security cameras, wearable technology, headphones, energy meters, household appliances, electrical systems and smart cars.

Emerging trends

Businesses are able to harness Internet of Things (IoT) technology to collect valuable data, gain a competitive advantage and guide future innovation. IoT technology can help organisations reduce expenses and remain competitive. IoT devices can be used to monitor equipment and information systems. By reviewing past history and a range of variables, IoT technology can be used to predict potential issues and failures, and can create alerts that attempt to avoid issues occurring, therefore reducing downtime.

Organisations can also save money on power and utility costs by monitoring and controlling energy systems to ensure they run efficiently and avoid unnecessary usage. IoT devices can assist in managing and controlling the production of goods by monitoring production processes in real time, tracking the use of raw materials and re-ordering materials when necessary.

Data from smartphones, the internet, social media and wearable technology can be collected, monitored and analysed. This allows organisations to create a profile of users and to predict future behaviour and trends so they can offer personalised products and services to their customers. Organisations can also use this data to create and build a stronger relationship with their customers and foster a sense of trust and loyalty.

Robotic devices

In the manufacturing industry, robots are being installed to reduce or eliminate the need for human workers to complete repetitive and dangerous processes. Jobs that involve working in high-risk environments such as extreme weather conditions, handling dangerous materials or performing processes that are highly repetitive, and where there is a high chance of injury, are areas where robots are replacing humans.

FIGURE 5.13 There are pros and cons of using robots in the workplace.

Compared with humans, robots operate more efficiently. Robots operate at the same pace all day long, they don't get tired or require breaks, sick days or holidays, and they are not required to attend training or other events. Although the cost of installing robots can be extremely high, in workplaces with high turnover the long-term operating cost can be far less than the cost of employing human workers. Robots can potentially operate 24 hours a day, improving the efficiency of production.

In terms of effectiveness, robots, once programmed correctly, complete tasks with consistency. This results in the production of goods of a consistent standard, avoiding the issues that arise from mistakes or faulty workmanship. The demand for quality assurance programs within the organisation is lessened as the likelihood of poor-quality products is reduced.

When required to complete tasks that are dangerous, repetitive or undesirable, human workers may lose motivation and become disengaged or unhappy. These staff can be redeployed or retrained to positions that are more satisfying and rewarding, leading to a more productive workplace. New jobs requiring new skills are created as robots need to be constructed, programmed, supervised and maintained.

Functions and capabilities of digital systems

There are many existing technologies that are designed to make the lives of individuals and organisations easier. These include:

- assistive technologies
- financial services
- global positioning systems (GPS)
- robotics
- traffic management.

Assistive technologies

Assistive technology is any technology that allows an individual or organisation to complete a task that they would not normally be able to complete, or that makes it easier or safer to complete day-to-day activities.

Devices and situations in which assistive technology is used include:

- accessibility aids
- cognitive aids
- personal emergency response systems
- home automation
- mobility aids.

Accessibility aids

Accessibility aids are designed to make everyday tasks accessible to everyone, especially those people who have an impairment or disability. Such aids include software designed to make it easier for someone to access data and information while using a digital or information system. This can include text-to-speech software, which allows users to hear information rather than read it on a screen and is particularly useful for people with vision problems. Speech-recognition software allows a user to input data without the need for a keyboard. This can be useful for people who have limited use of their hands or arms.

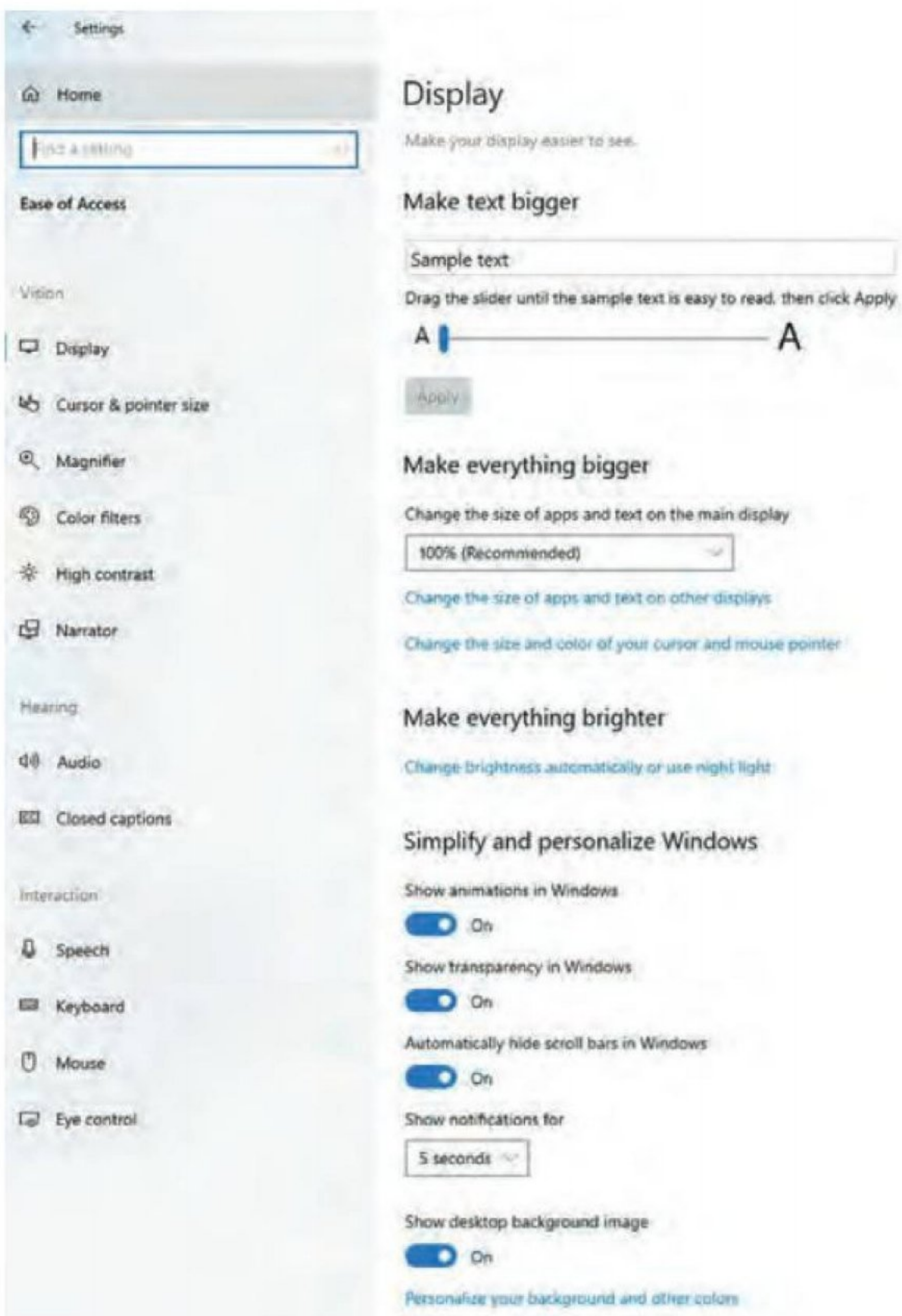


FIGURE 5.14 Access settings in Windows

Other accessibility aids include:

- closed caption software, which allows hearing-impaired users to read, rather than listen to, the spoken words in audio and video files
- keyboard shortcuts and sticky keys for those users who prefer not, or are unable, to use a mouse
- visual cues to alert a user to a particular action, rather than sound alerts
- contrast control, text and icon resizing
- contrast control or alternative text for images to make the data and information easier to read.

Cognitive aids

Cognitive aids assist people with a physical, cognitive or emotional impairment that affects processes such as attention, memory, navigation, planning or sequencing of events or tasks. Software can be used to provide alerts to the user that they have to complete a task or attend an appointment. Games, quizzes and 'brain training' applications can help to improve or maintain a person's memory.

Cognitive aids are often used by the elderly to help them to live an independent lifestyle for as long as possible.

Personal emergency response systems

Personal emergency response systems, when triggered, can send an alert to a nominated mobile phone number. Different sensors can detect a range of inputs, including environmental conditions such as smoke, water and gas levels, or biometric data including heart rate, temperature and evidence of falls. The style of sensors used can be tailored to the individual needs of the user. In addition, many such systems come with a help button installed that allows the user to call for assistance directly; they may also include cognitive aids such as alerts and reminders.

Home automation

Home automation systems employ sensors and actuators to complete day-to-day tasks in the home, such as opening and closing doors and cupboards, locking and unlocking doors and controlling energy systems. Smart home technology is also available to make your home more secure by allowing access to your CCTV stream via your smartphone, turn on all your lights if there is an intrusion, create a 'mockupancy' scene that gives the impression that your home is occupied, and automatically lock your external doors at nightfall.

FIGURE 5.15 Smart home technology can be controlled from your watch, tablet or phone.

Mobility aids

Mobility aids include devices such as wheelchairs, walking frames and electric scooters. Electric wheelchairs and scooters include electric motors and use sensors to assist with navigation. They are operated with a joystick, which is controlled either by hand or with the chin, depending on the level of disability.

Read more about assistive technology.

Alamy Stock Photo / Libby Welch

FIGURE 5.16 Computerised wheelchair

Financial services

There is a wide range of technology used within the financial services industry. Traditional uses of technology in this field have included online banking, automatic teller machines and point-of-sale payment systems. As technology advances, there are new ways to use it that enhance the services offered. Buy-now, pay-later services have harnessed technology to offer customers the chance to purchase goods immediately and pay for them over the following months. These services are becoming an alternative to credit cards. At the same time, banking and financial apps allow customers to analyse their spending and saving habits to take better control of their finances.

Customer service

Financial services organisations have been early adopters of the use of virtual assistants for customer service purposes. In particular, **chatbots** or automated voice and text systems are increasingly being used as the initial interface with customers. As the technology used in virtual assistants improves, the efficiency of these interactions is also improving and becoming more seamless, saving time for the customer. Flowing on from this, the organisation requires less staff for handling customer queries, so the technology is also saving the organisation money and increasing profits.

Multi-factor authentication

A growing number of organisations are using **multi-factor authentication** when users complete financial transactions online.

Previously, a user was required to provide one form of evidence, usually the combination of a username and a password, to be able to complete a transaction online. This introduces a serious threat to security. If an unauthorised person were to gain access to that one form of evidence, they would be able to complete transactions without the knowledge of the authorised user.

Multi-factor authentication requires the user to provide two or more pieces of evidence to be able to complete a transaction online. Usually, a username and password are required along with at least one other form of evidence. Different forms of secondary evidence are explained below.

Security token

A security token is a small hardware device that is used to generate an authentication code. Often, before a user can complete a specific transaction online, they are required to generate an authentication code using the security token.

Shutterstock.com/SVRSLYIMAGE

FIGURE 5.17 Security token

Personal identification number (PIN)

A personal identification number (PIN) is used as a primary authentication technique when using an automated teller machine (ATM), but PINs can also be used as a part of a multi-factor authentication when completing transactions online. Users are often asked to set up a PIN (one that is different from the PIN they use to access an ATM) for use on online financial systems. To complete particular transactions the PIN must be supplied along with the username and password. PINs can also be sent to the user via SMS, email or QR code verification.

SMS-based verification

Users are sometimes sent an SMS message while completing a transaction online. The SMS is sent to a pre-registered phone number. Once received, the SMS will contain details about how a transaction can be authorised. It may contain a one-off PIN or a hyperlink, or it may require the user to reply to the message.

Email verification

In a similar way to SMS-based verification, users wishing to complete a transaction online can be sent an email message containing details about how a transaction can be authorised. As the number of hoax and phishing emails is very significant, however, many financial institutions have started to avoid the use of email verification.

QR code verification

A QR code is a machine-readable code that consists of a collection of black and white squares. Each user can be given a unique QR code, which they store on their device. When completing a financial transaction they are required to scan the code to assist in verifying their credentials.

One-time passcode

A one-time passcode is similar to a PIN, but the user can only use the passcode for one transaction. To complete a second transaction, they will have to generate a new one-time passcode. If the user makes a mistake with the original passcode or the passcode expires, they will also need to generate a new passcode.

Secret questions

In some systems, when the user first completes their registration they are asked to set up one or more 'secret questions' and an accompanying answer to each question. When completing a transaction that requires multi-factor authorisation, they will be asked one or more of the secret questions, to which they must respond with the correct answer.

Location-based factors

The use of location-based factors involves the automated collection of data regarding the device a user is using to complete a transaction online, including the current location of the device. This data is used to identify strange patterns in the use of financial systems – in particular if a user completes a transaction in one location and a second transaction occurs soon afterwards in a different location.

Requiring multiple pieces of evidence during the authentication stage reduces the risk of a transaction being completed by an unauthorised person.

> **5.7**
> **THINK ABOUT APPLIED COMPUTING**
> Which combinations of evidence do you think provide the best protection for online users of financial systems?

Fraud detection

The use of technology is playing an increasing role in the detection of fraud in the financial services industry.

In the financial industry, **fraud detection** involves identifying unauthorised transactions. Many organisations rely on data analysis and artificial intelligence to assist them in identifying transactions that may be deemed suspicious and worthy of further investigation.

Fraud detection involves viewing a transaction and comparing it to previous transactions to see if it is reasonable or consistent with previous transactions. Criteria that can be used to compare transactions can involve the amount of the transaction, the location in which it takes place, the type of purchase, the frequency of transactions or even the time of day.

'Data-mining' software is used to identify if there are any factors that are outliers to the regular transactions conducted by the user. If an outlier is identified, the organisation will investigate the transaction further, sometimes contacting the user directly for confirmation.

Artificial intelligence (AI) technology is used to 'data mine' through large data sets of previous transactions, which contain a number of transactions that are known cases of fraud or are considered suspicious. The algorithm used by the AI technology is then refined until a high proportion of the fraudulent or suspicious transactions are identified. When the developers feel confident the system is highly accurate, the technology can be used in real time to monitor transactions as they occur.

Data mining involves searching through existing large data sets in an attempt to find new patterns or relationships within the data.

Electronic funds transfers

Electronic funds transfers (EFT) allow a payment to be made from one bank account to another using a digital system.

Examples of electronic funds transfers include:

- ATM transactions
- direct deposits

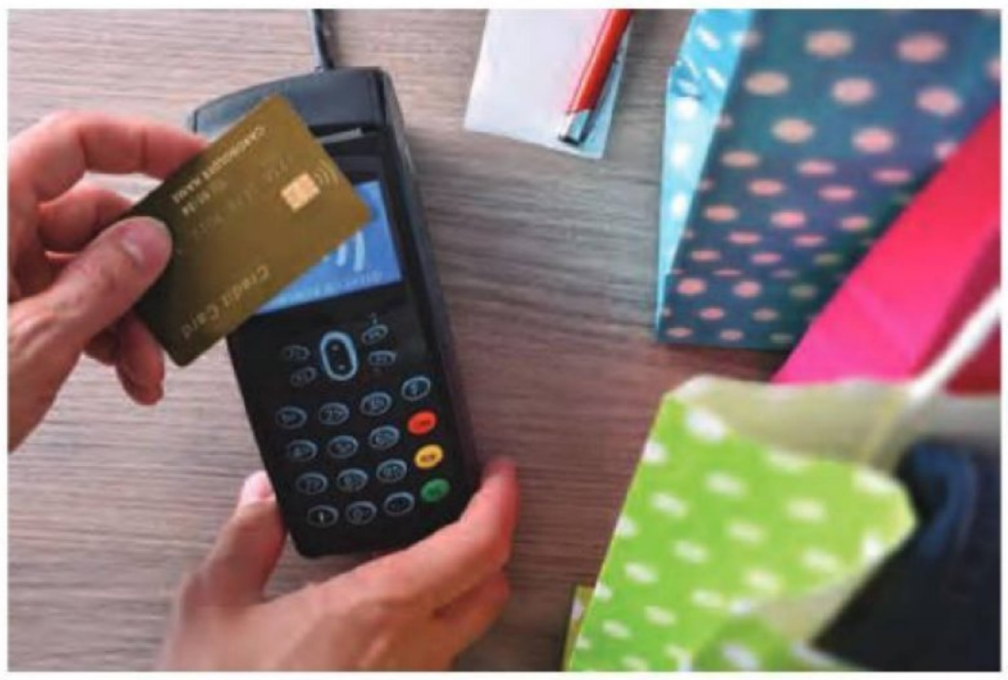
- direct debits
- credit card or debit card transactions
- online bill payment
- instant payments.

For credit card and debit card transactions at the point of sale, a **payment terminal** is used to facilitate the transaction. A payment terminal is a hardware device that captures credit card or debit card details and sends them to a bank for authorisation. Once authorised, funds are transferred from the customer's bank account into the organisation's bank account; sometimes these funds are transferred through an intermediate account known as a merchant account. Payment terminals can read card details using a card reader, or by using **near-field communication (NFC) technology**.

NFC technology

Payment applications, such as Apple Pay, Google Pay and WeChat Pay, which allow the storage of card details, can be downloaded onto smartphones, reducing the need for people to physically carry their cards with them (provided they carry their phone).

NFC technology allows contactless communication at a very close distance, without the need to configure any settings using radio waves. Many credit cards and debit cards have a chip installed that allows the transmission of data via near-field communication technology, such as Visa payWave or MasterCard's Tap and go. There has been a massive increase in the number of contactless transactions in recent years.



Shutterstock.com / Davizro Photography

FIGURE 5.18 Chips in credit cards allow contactless transactions.

Data communication between the payment terminal and the bank can be via Bluetooth, a wired network, wi-fi or a mobile network. Payment terminals used in aeroplanes may even use satellite communication to transfer the data. Encryption is used to protect the details of each transaction.

Many smartphones have near-field communication (NFC) technology built in. It is possible to set up NFC settings to complete a range of tasks, including sending messages to nearby devices, although to receive these messages the receiving device may need to install an NFC reader from an app store.

Virtual currencies

Currency (or money) used to pay for goods and services is issued and controlled by a country's government and is legal tender within that country. Currency plays a key role in the economy of the country in which it was issued.

Virtual currency is unregulated, digital money, which is issued and (usually) controlled by the virtual currency's creators, and is used and accepted among the members of an online virtual community.

One type of virtual currency is **cryptocurrency**, which uses encryption to secure transactions. Bitcoin is probably the most well-known cryptocurrency.

In virtual currencies, transactions occur between two users online over an internet peer-to-peer network. Each user of the currency has a virtual wallet, which stores data about the user's account. In cryptocurrencies, the wallet will contain the public and private keys used to encrypt and decrypt the transaction data. Details of each transaction are then sent to a **blockchain** to be verified and stored. A blockchain is a collection of devices connected using the internet, each device storing a copy of transactions in what is known as a block.

Users who make their devices available as part of the blockchain are rewarded in the virtual currency in which they are hosting the database.

Find out more about virtual currencies and cryptocurrencies.

Blockchain

Blockchain technology was first utilised with the introduction of Bitcoin virtual currency. A blockchain is a collection of data records (blocks), or a database, that is stored on multiple devices (chain). In some cases the data can be stored on millions of computers.

The purpose of a blockchain is to allow data to be input and stored without allowing the data to be manipulated after storage.

When a financial transaction occurs, the transaction first needs to be verified: all the computers that store the database or block must confirm the transaction and its details. Once a transaction is verified, it is given a 'hash value'. A hash value is a value that is calculated to act as a unique identifier for that transaction in the database. Once a hash value is calculated, the data is added to the block or database on each device, along with all the previous data stored.

As the database is stored on multiple devices, it is very difficult for an individual to change the contents of any transaction because to do so they would need to access every device on the blockchain to manipulate every copy of the database. Blockchains offer security in that each new transaction is added to the top of the block, in sequential order to the previous transaction, making it very difficult for anyone to delete a specific transaction. Also, if any transaction is edited, a new record and hash value are created, setting up an audit trail that can be used to alert the users of the blockchain that a modification has occurred.

Devices that are used to store the block or database also require verification. Users who wish to store the blockchain database on their device need to complete a process known as 'proof of work'. To do this, a device needs to complete certain complex mathematical calculations to prove that it has significant processing power and functionality to be able to be part of the blockchain.

Global positioning system (GPS)

Global positioning system (GPS) uses a network of satellites that provide location and time data to any connected device. GPS is operated and controlled by the United States Government.

Satellites orbiting Earth on a range of different paths allow connection to the system from any location on Earth at any time. Line of sight is required, meaning natural or man-made obstacles may affect the signal.

The user is not required to transmit any data when using GPS. Rather, each device with GPS software installed acts as a receiver, which only receives data from the system. Devices that receive GPS data are known as GPS receivers.

Each GPS satellite transmits data regarding its location, time and date using radio waves. As radio waves travel at a constant speed, the time taken for each signal to be received by a receiver indicates how far away it is from the satellite. To determine the precise location of a receiver, triangulation is required. The receiver receives data from multiple satellites and uses an algorithm to calculate its location. The accuracy of the location identified by the GPS receiver may vary depending on a number of factors, but usually it is accurate to within 10 metres.

GPS does not require the services of any other network. The user does not need to be connected to the internet or to a mobile network to receive GPS data. GPS software is now installed in most smartphones, allowing individuals and organisations to harness the power of the technology.

The most obvious use for GPS technology is to allow a user to determine their location. This could apply to an individual trying to orient themselves in a new city, a ship traversing the ocean or a pilot determining their location while mid-flight.

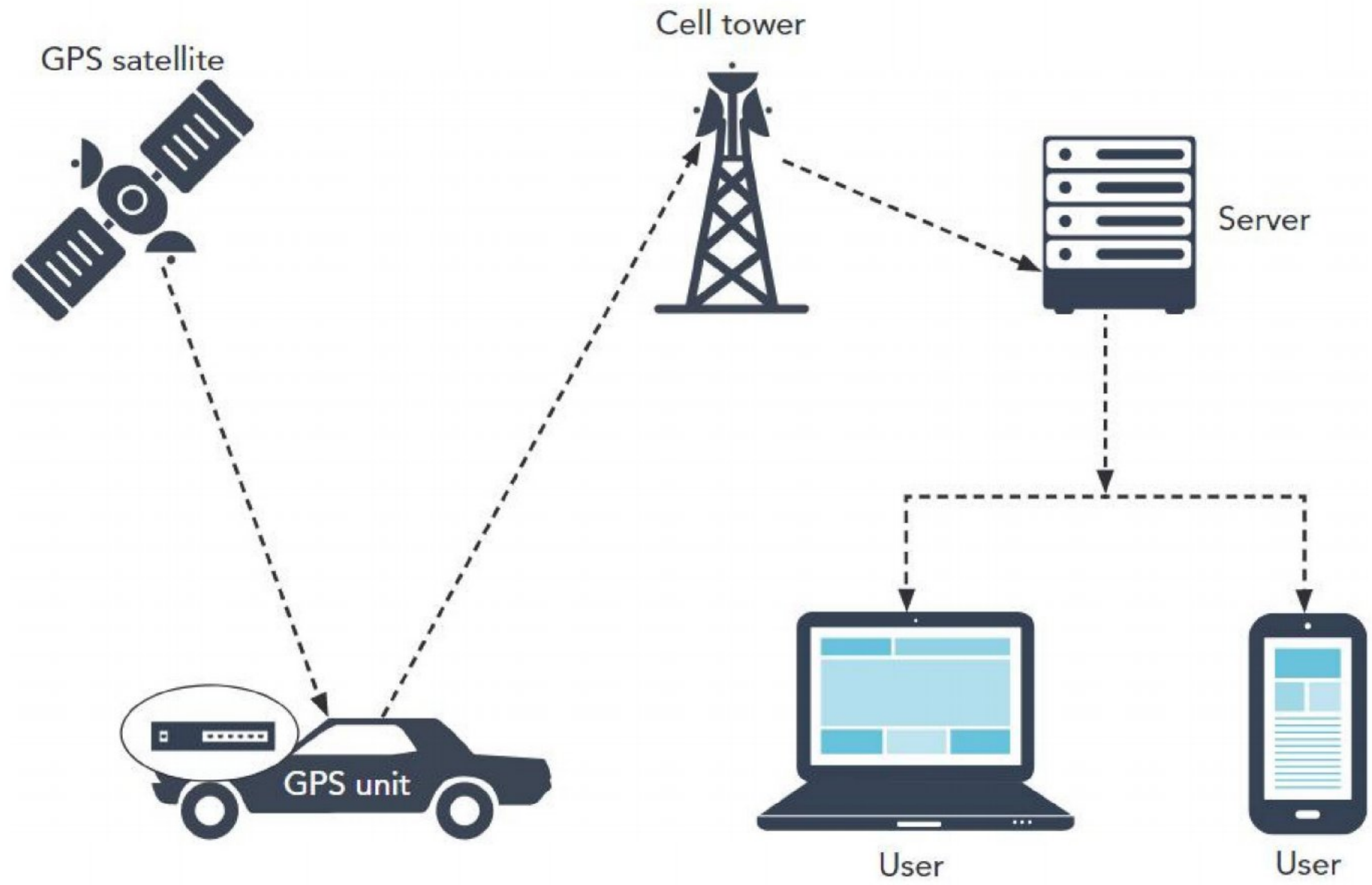


FIGURE 5.19 GPS in your car works by receiving information from satellites about location. This is sent to a phone tower and onto the internet and to your computer or phone.

Using location data, the navigation required to get from the current location to a specific destination can be determined. GPS data can be used in conjunction with mapping software to display the route a vehicle should take to get to a chosen destination. Often, multiple routes are identified and, by using a combination of live traffic conditions and historical data, the approximate time each route will take can be determined. This also applies to a range of other transport options.

GPS also makes it possible for organisations and individuals to track the progress of a person, a vehicle or an item. Many car hire organisations allow authorised users to view the tracking of a person's ride. Businesses can track the progress of a delivery van and use this information to inform customers of an estimated time when their delivery may arrive. Software applications are used to track the location of a user's smartphone in case it has been lost or stolen.

GPS plays an important role in the development of new maps. Location data in the form of latitude and longitude coordinates can be downloaded.

An often-overlooked functionality of GPS is to deliver the precise time. Each satellite has an atomic clock installed, and each time it transmits data, it includes the current time and date. Whenever a GPS receiver collects data and identifies its position, it also calculates the precise time and date of that location using the data from each satellite.

Robotics

A **robot** is a device that can complete a specific task with autonomy. Robotics incorporates intellectual property from a range of different fields, including engineering, which is related to the design and building of robots, and computing, which is related to creating the information systems used to control a robot. The range of situations in which robots are used is constantly increasing. Robots are now used in the agriculture, manufacturing, medical and military industries, to name just a few.

There are many types of robots, but they usually have the three following elements:

- a physical structure designed to achieve a particular task
- electrical components that power and control the mechanism
- programming code.

Components usually found in robots include a power source, which could range from mains electricity, batteries or compressed gas to hydraulics or solar power. Actuators are used to control and move the device. The role of an actuator is to convert the storage energy from the power source into motion. An actuator commonly found in robotic devices is an electric motor that causes a wheel to rotate, creating movement within the device.


Shutterstock.com / Nor Gal

FIGURE 5.20 A robot vacuum cleaner uses mains electricity to charge a storage battery. Touch sensors and infra-red sensors direct its route around the house.

Sensors are often used in robots to collect input about their working environment. This allows robots to perform their tasks and gives them the ability to make changes in response to any changes in their situation. Sensors may detect touch, sound, visual nformation or even temperature.

Some robots are used to manipulate objects, and these robots are often produced with arms and hands that allow them to perform the task for which they were designed. The hands of a robot are referred to as the end effectors, while the arm is referred to as the manipulator.

The actuators and sensors within a robot are connected to its microcontroller, which is used to control its actions. Software applications are written using a programming language; these provide the instructions needed by the robot. Some robots, such as a robot vacuum cleaner, are programmed to perform stand-alone tasks (Figure 5.20); other robots, such as a drone, can be controlled by user input via a device like a remote control.

Traffic management

Australia's population continues to increase, particularly in the capital cities. This is leading to an increase in the number of cars on the road. The procedures and processes used to efficiently and effectively control the flow of vehicles on the roads are known as **traffic management** strategies. State governments, which have responsibility for traffic management, are turning to technology to help them. Technologies such as speed cameras, red-light cameras and point-to-point and bus lane camera systems play an important role in the field of law enforcement.

Red-light cameras often use in-road sensors to detect if a car has passed through an intersection after the light has turned red. The sensors are installed on individual lanes on a road so as to be able to detect the exact lane in which a car crossed the stop line after the light had turned red. When an infringement is noticed, the sensor will trigger the camera, which is installed above ground and connected using wires, to take an image of the offending vehicle.

Speed cameras transmit a radar beam towards oncoming traffic. As the beam hits a car, signals are sent back to the camera, which then calculates the speed at which the vehicle is travelling.

In point-to-point camera systems, a number of cameras are set up at various set locations along a road. As a vehicle passes a camera, an image is taken and the time is recorded. As

FIGURE 5.21 Cameras are used in traffic management systems.

the vehicle passes the next camera, another image is taken and the time is again recorded. The average speed of the vehicle is then calculated by dividing the distance between the two cameras by the time taken.

Cameras are also used to read vehicle number plates. As a number plate is recorded, details are retrieved from a number of databases to check if there are any known concerns about either the driver or the vehicle. The data stored and accessed in this way could include the owner's current licence status, whether the car is insured and registered, and if the owner has any outstanding traffic offences.

Cameras can also be set up to capture images of the passenger compartment within a vehicle to check that drivers are not breaching any road rules, such as by using a mobile phone while driving.

Traffic management systems also monitor and control the flow of traffic on the roads. Sensors and cameras can detect how busy sections of roads and highways are. This data is transmitted to a central server, where decisions can be made to decrease or increase the traffic limit in some areas, or be used to alter or synchronise a collection of traffic lights that will improve the flow of traffic in that area. In some countries, once traffic incidents are detected, the system is able to broadcast traffic warnings over traditional AM/FM radio networks, or send messages and real-time traffic updates to in-car GPS systems, to alert drivers to avoid particular areas.

Goals and objectives

Each digital system has a collection of goals. A **goal** is a broad statement of the desired result that a digital system is expected to achieve. Goals are roughly similar to purposes or aims.

A Year 11 student might set the following goals for a school year:

- Receive an A grade in Applied Computing.
- Participate in extracurricular activities.
- Investigate university options.

An **objective** defines how a goal will be achieved. Objectives tend to be specific and measurable. They specify the 'who, what, when, where and how' of the way to achieve a goal. There may be one or more objectives that need to be met to ensure that the larger, more general, goal is achieved. Objectives need to be measurable, so they usually contain numbers or percentages. The numbers or percentages provide a quantitative measure to determine whether the goal was achieved.

The Year 11 student could define the following objectives in order to measure whether each goal is being achieved:

- Receive no less than 90% for each assessment task in Applied Computing.
- Attend at least 90% of classes.
- Create comprehensive notes in each class.
- Complete five hours homework in Applied Computing per week.
- Read the relevant textbook chapter at least once.
- Visit the teacher once per week outside class to ask questions.

To achieve the goal of receiving an A grade, the student has created a series of specific tasks that define what is needed to be done to achieve the final goal.

5.8 THINK ABOUT APPLIED COMPUTING

Identify the quantitative data in each of the Year 11 student's objectives.

Goals and objectives of digital systems

Each digital system has a list of goals and objectives that are used to measure whether the system is achieving the purpose for which it was designed. A swarm of nanosatellites used to take images of Earth's surface might have the following goals and objectives.

Goal

- To develop a composite image of Earth's entire surface daily.

Objectives

- 16 nanosatellites are used each day to take imagery of Earth's surface.
- Each satellite orbits at 500 km above Earth's surface.
- Each satellite orbits Earth on one of eight orbit paths.
- Each satellite completes one rotation of Earth every 48 hours.
- Each satellite takes an image every two seconds.
- Images are sent back to a base station each hour.
- A composite image of Earth is produced each day.

5.9 THINK ABOUT APPLIED COMPUTING

Create a list of goals and objectives for one other emerging technology.

For Unit 2, Outcome 1, working in groups, you are required to analyse, design, develop and evaluate an innovative digital solution in response to an identified need or opportunity. It would be useful to identify the goals and objectives of the innovative solution as part of your analysis of the need or opportunity. The goals will describe the purpose of the solution and how each purpose will be achieved.

Economic issues involving emerging technologies

Economics is the study of the production, distribution and consumption of goods and services. It involves the study of scarcity and choice, based on the idea that the demand for resources (e.g. money, supplies, equipment) is greater than the resources that are available.

Decisions need to be made about how these resources are to be allocated and used. These decisions sometimes lead to negative consequences for some stakeholders. These are referred to as economic issues.

Economic issues involving emerging technologies can include:

- access
- deskilling
- job loss
- misuse
- sustainability.

Access

The scheme One Laptop per Child is attempting to bridge the gap within the digital divide.

Technology has become more affordable. There is a large range of digital devices available, and an increasing variety of methods used to access high-speed internet connections. However, there still remains a **digital divide** in society. This term refers to the gap between those people who have access to technology and those who do not. This gulf in the availability of technology can be seen between different countries, between people within a country, and between organisations.

With the increased usage of smartphones around the world, the digital divide is not necessarily linked to a lack of internet connection, although this is still one factor. Other factors include the cost, reliability and speed of the internet connection, the cost of new technology, levels of existing infrastructure, the technology skills of users, training opportunities and the availability of technical support.

Economic factors play a role in creating a gap between individuals, organisations or countries in terms of accessing emerging technologies. The availability of resources required to purchase and maintain new technology can be prohibitive to some users, while reliable, high-speed bandwidth can also be expensive.

One of the features of many emerging technologies is that they offer additional functionality and connectivity to existing digital networks. For example, smart devices and other devices related to the Internet of Things (IoT) use existing network infrastructure to communicate data, whether to a home network, an organisation's local area network or the internet. The full power of these devices can only be achieved if the existing network infrastructure has the capability to handle this increase in functionality and data communication.

One initiative to improve infrastructure in Australia is the implementation of the National Broadband Network (NBN), where a fibre-optic cabling backbone is being installed to increase internet speeds. This is designed to increase the availability of reliable high-speed internet to businesses and households throughout Australia. This, in turn, increases accessibility and the opportunities for users to use existing and emerging technologies. However, there are still areas of the country where there is no NBN service available or it is yet to be installed. In areas where the NBN has been installed, there will be individuals and organisations that are unable to afford the cost of maintaining an NBN connection.

THINK ABOUT APPLIED COMPUTING 5.10

What other factors may restrict a user's ability to access emerging technology?

The existing knowledge and skills of a user can also act as a barrier in accessing emerging technology. As new technology is developed and released, those users with limited existing skills are less likely to be able to adapt to, or even understand, the new technology. For example, elderly users may be reluctant to embrace assistive technologies that might help them to complete day-to-day tasks more easily. Some farmers may be reluctant to embrace the use of technologies such as drones and satellite imagery – not through a lack of funds or resources, but rather through lack of knowledge, skills and existing understanding. A lack of training and technical support may be another barrier to the uptake of emerging technologies. This could be related to the geographic location of the user, the quality of the internet connection or the cost of accessing these services.

Deskilling

Deskilling occurs when a skilled worker is replaced, or their job specification is reduced, as a result of the implementation of technology. In any given industry, this can result in fewer workers having the skills to complete specific tasks, and can lead to existing jobs being eliminated altogether. In society more broadly, this can result in an overall reduction in knowledge and can have negative effects on culture and lifestyles.

In manufacturing, for example, a significantly reduced number of tasks may now require workers to complete them as robots are installed to complete other tasks. Trained tradespeople and craftspeople may no longer be required to apply the skills that they were trained in, and may focus instead on the limited number of tasks that robotic devices are unable to complete.

Technology is introduced in order to achieve an increase in productivity, maintain product consistency and reduce the overall cost of the production process. The downstream effect of this could be that workers become less engaged and enthusiastic about their roles, and the quality of the products manufactured becomes more uniform and less personalised.

There is a range of traditional jobs and roles in which job specifications have been altered or reduced following the introduction of automation. Roles have been replaced or redefined in the field of art and design as the use of software applications in design work becomes commonplace. Similarly, 3D printing can be used to produce a wide range of products that were previously designed and manufactured by highly skilled workers. This may eventually lead to age-old skills involved in manually designing and creating products being lost forever.

In the commercial aviation industry, the number of tasks that are conducted using a digital system has increased significantly, and the number completed by a human pilot has

Alamy Stock Photo/Alexander Tolstykh

FIGURE 5.22 Automated pilot systems are capable of completing many of the tasks once completed by human pilots.

been correspondingly reduced. Automated pilot systems can now conduct up to 90% of the tasks that are required in the time between when an aeroplane leaves the gate at one airport and when it arrives at the gate at the destination airport.

The use of AI, GPS, virtual assistants and IoT devices allows conditions to be monitored, decisions made and tasks completed without the need for human intervention. Human pilots are still required to monitor the automated system and complete tasks that cannot be automated, and are able to override decisions when they deem it appropriate. It is not uncommon for pilots of large commercial aircraft to spend some time each year flying small, less sophisticated aircraft that contain less automation. This ensures that they do not lose the skills of flying, such as taking off and landing, that are often performed by autopilots on regular commercial flights.

THINK ABOUT APPLIED COMPUTING 5.11

List 10 skills that you think might be lost due to the introduction of technology.

Job losses

An economic issue that overlaps with deskilling is that of job losses. A **job loss** occurs when a job formerly completed by a human in an organisation is no longer required. This results in the worker becoming redundant (excess to needs). Sometimes, a job loss occurs as a direct result of new technology being introduced in the organisation.

Examples of the use of technology leading to job losses include the use of automatic teller machines, electronic funds transfer and online banking. These have reduced the number of financial transactions that require the involvement of a human worker, and have thus reduced the demand for bank staff. One of the reasons for the increasing popularity of cryptocurrencies is that they avoid traditional banking systems; the growth in cryptocurrencies may also affect the demand for traditional banking products and banking staff.

Automatic checkout systems at supermarkets and department stores are reducing the number of situations in which a skilled human worker is needed. The growing use of virtual assistants and chatbots to interact with customers is replacing the need for customer service and call centre staff.

The use of robots in the manufacturing process is well documented. The widespread use of smart devices in businesses has reduced the number of tasks that require a human to complete them. Smart refrigerators have the ability to keep track of items added to or removed from them, and can use this information to create shopping lists and place online orders. The use of smart refrigerators in cafes and restaurants may not lead directly to job losses, but it has the effect of reducing the number of tasks staff must complete. This, combined with other efficiencies related to the use of technologies (e.g. the use of automated coffee machines), may lead to a reduction in the number of staff needed.

Drones are used in the agricultural industry for a range of applications, including crop spraying, monitoring of crops and irrigation systems and aerial photography. This is reducing the need for skilled agricultural workers. The use of assistive technology is allowing elderly and disabled people to live a more independent lifestyle, thus reducing the demand for skilled medical and nursing staff.

THINK ABOUT APPLIED COMPUTING 5.12

Research the types of tasks that businesses often send offshore. What technology do businesses use to allow offshore workers to complete these tasks?

As technology and communication methods improve, more organisations are completing jobs offshore. This allows them to employ workers in other countries, where wages may be lower, to complete tasks that once were completed by workers in their home country. A challenge for these organisations and countries is to reskill and retrain their existing workforce in areas of new employment; otherwise, these workers may become excess to needs.

Sustainability

Sustainability is a process of managing change in a balanced and controlled way so that other factors such as natural resources, climate, pollution levels and economic resources are not negatively affected.

As the amount of technology in the world increases, issues related to the production and use of technology need to be addressed. Technology involves the use of many raw materials, some of which are in limited supply, and some dangerous chemicals. Technology companies may decide to manufacture in countries where wage costs are low and the workers come from lower socio-economic backgrounds. Finally, once a technology has outlived its usefulness or been replaced by a newer model, what happens to the hardware and associated equipment?

CASE STUDY

Smartphones are as rare as Earth

Since 2014, more than one billion smartphones have been sold worldwide each year.

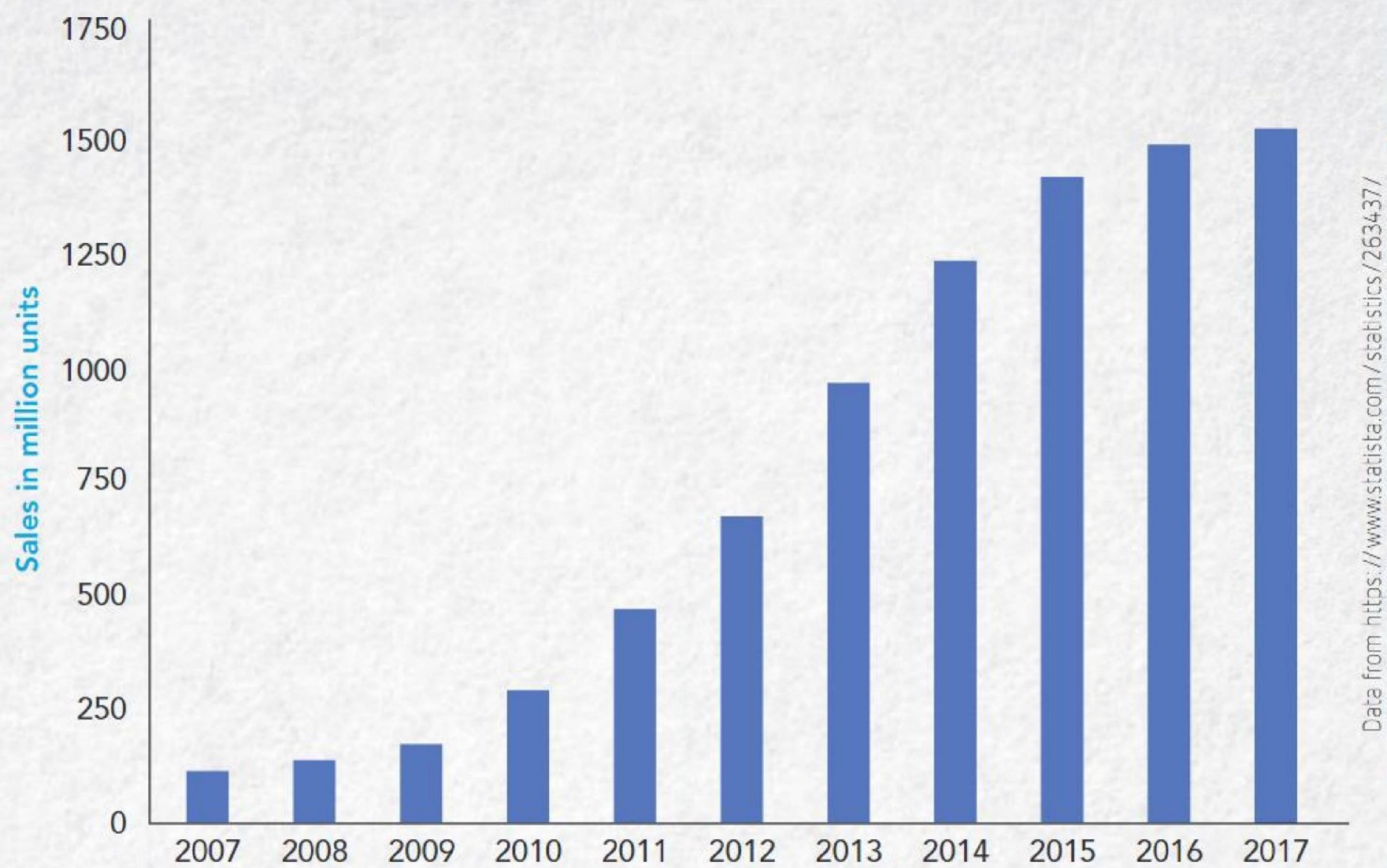


FIGURE 5.23 Number of smartphones sold each year (millions)

One smartphone can contain around 70 different metals. That is 84% of all the non-radioactive elements on the periodic table. Some of these metals are called 'rare-earth metals' because they are found scattered in the ground, not in high concentrations like gold or silver. Mining costs can therefore be high.

These rare-earth metals include scandium and yttrium. It is the properties of these rare-earth metals that make smartphones so smart. They are used in the smartphone's circuitry and display. Without neodymium and dysprosium, your smartphone would not be able to vibrate. Without indium tin oxide your screen would not work as a touchscreen.

Mines for these rare-earth metals are often located in developing countries such as Chile, Peru, Zambia, Democratic Republic of the Congo, Kazakhstan and Mexico, where mine workers receive very low rates of pay and work in very poor conditions. Often the method used to extract the metals from the ground is known as surface mining, which results in large

THINK ABOUT APPLIED COMPUTING 5.13

Give three reasons why smartphone developers have their products produced in factories owned by other organisations.

areas of the land being disrupted and causes long-lasting effects to the landscape. Industrial chemicals used in the process of surface mining drain into Earth's surface and waterways.

Manufacturing of many electronic and technological components is completed in developing countries. As with the mining of raw materials, the workers involved in manufacturing work under poor conditions, for incredibly long hours, and receive very low wages. This leads to a range of social and health issues for the workers, who often live close to the factory, rarely see their families, and may develop health problems as a result of the production process. The production of smartphones takes place in factories owned by third-party organisations that produce the phones on behalf of the smart phone developers.

THINK ABOUT APPLIED COMPUTING 5.14

In which country was your smartphone produced? How much profit did the organisation selling the smartphone make?

The distribution and lifespan of smartphones also lead to issues with sustainability. Smartphones are sold across the world and are often designed to be hard to repair. New apps requiring greater resources to function are constantly being released, and as a result many users purchase a new phone approximately every 18 months. This results in a large quantity of underutilised technology and wastage. There are also issues related to the disposal of discarded phones since they often cannot be recycled using the regular services provided by local councils.

An ethical issue related to the production process is that the organisations designing and selling smartphones are making significant profits as a result of the low costs of manufacturing devices in developing countries.

Smartphone manufacturers are, however, taking action. Some quirky models are coming to market with parts that can be swapped out and recycled when they are superseded. Companies are working towards conflict-free supply chains and setting up recycling depots. Companies that work with non-profits such as the Electronic Industry Citizenship Coalition (EICC) sign up to meet certain standards in worker rights, environmental impact and transparency.

Impact of current and emerging technologies

Emerging technologies are capable of making significant change within society. These changes can be positive and negative. In Unit 2, Outcome 1, you are required to create an innovative solution to a need or opportunity. As part of this outcome you are required to discuss potential issues that may arise due to the development of the innovative solution.

Issues and impacts can be related to:

- automation
- cyberbullying
- human interaction
- interpersonal skills.

Automation

The term 'automation' refers to processes that can be completed without the need for human intervention. Automation can be used to operate machinery, capture, communicate and store data and information, control and steer cars, ships, aircraft and other transport vehicles, or in the manufacture of goods and products.

Control of an automated system can be as simple as an on-off switch, or it can be as complicated as a complex algorithm containing a range of control and processing structures and multiple variables, and using a range of complex hardware devices.

Getty Images/Ian Waldie/Bloomberg

FIGURE 5.24 Automation is widely used in the mining industry.

Impact of automation

An automated system may be complex and expensive to install. It is often expected that, over the long term, once a process is automated, the system will be more efficient and effective and will produce consistent results and save the organisation money. In the short term, however, the cost of setting up the system can be costly and it may take time for the system to be fine-tuned to become fully effective.

Automation has an impact in deskilling the labour force (see pages 221–22) and creating job losses (see page 222). A range of new jobs may be created that require workers with new skills. Existing workers may have to be retrained. One question to ask is how the education system will cope with training students for a range of jobs that may not yet exist.

Automation needs a digital system to be designed and programmed to complete specific tasks. If the task changes – to a new task or to one it was not originally designed to do – it can be difficult, expensive and time-consuming to re-program the system. In this situation, human workers may still need to be used to complete these activities.

In the manufacturing industry, part of the reason for using automation is to be able to produce products of a consistent quality. This has, however, reduced the individuality of manufactured products. Consistency may be achieved, but the quality may not be the same as that achieved by a human craftsperson.

> **5.15**
> **THINK ABOUT APPLIED COMPUTING**
> List the impacts of automation of the workforce, and add two more that have not been discussed here.

Cyberbullying

Cyberbullying occurs when a digital system is used to threaten, harass or intimidate another individual. This could be in the form of lying, spreading rumours, playing cruel jokes or making threats. Its intent is to cause harm to another person or a group of people.

Bullying existed in various forms before the digital age. However, the anonymity and accessibility of the internet has been a major contributing factor in the increase in bullying. Although in many cases a cyberbully may know their target, it is not always the case that the individual being bullied knows who is responsible.

The increased use of smartphones has led to an increase in cyberbullying as online platforms and social media have become more accessible to users. Features of a smartphone such as social media apps, internet access and a camera have contributed to an increase in bullying using emerging technology.

People posting messages using electronic communication often feel more detached, and less responsible for their actions, than if they were communicating the same message face to face. Once messages are posted online, the potential audience who can view them is unlimited and they are very difficult to delete.

Cyberbullying can cause stress and anxiety for the targeted individual. This can result in an individual becoming less productive, which also affects their social life, work life and the organisation in which they function.

Cyberbullying

Human interactions

By nature, humans are social creatures, who require interaction with others, whether in social situations, forming relationships, creating networks or discussing emotions and feelings. **Human interactions** involve people communicating with other people. This can be through a face-to-face conversation, a text message or a smile. Research has shown that socialising and interacting with other people is a fundamental human need along with the need for food, shelter and water.

As technology has advanced, the increased use of automation and the ability to complete tasks online using an internet connection or to work from remote locations and while travelling have led to a significant decrease in human interactions, particularly face-to-face interactions. In many ways these advances in technology have helped people stay connected with others, whether through email, social networks, video conferencing or forums and blogs. The rise in popularity of smartphones has accelerated this connectivity but, as a result, actual face-to-face human interactions are no longer necessarily as frequent as in previous years.

Impact of loss of human interaction

Evidence suggests that regular human interactions offer health benefits for individuals. A lack of human interaction has been known to lead to a range of health issues, including loneliness, depression and associated mental health issues.

Human interactions are important within organisations. They help to achieve effective collaboration between members of the organisation that cannot always be achieved through online communication. By meeting face to face, people are able to have more in-depth conversations, build relationships, and pick up non-verbal signals and body language more easily.

In the classroom, human interaction plays a major role in learning. The teacher's role in the classroom is that of a facilitator, interacting with students. Students also interact with one another through discussion and group work. With the increasing number of courses being delivered online, many of the advantages of the traditional classroom may be lost to students.

THINK ABOUT APPLIED COMPUTING 5.16

List the impacts of the loss of human interaction due to technology, and add two more that have not been discussed here.

Interpersonal skills

A decrease in human interactions can lead to a reduction in interpersonal skills. **Interpersonal skills**, or so-called soft skills, are skills that are required to interact and communicate effectively with others. These skills are important in building relationships with family, friends and work colleagues. More and more organisations are team-based, so employees are expected to be able to build positive relationships with other people, such as being supportive, respectful and collaborative.

Interpersonal skills include the ability to:

- make a decision
- listen
- negotiate
- communicate verbally and non-verbally
- problem-solve.

Communication can be verbal (spoken and listened to, or written) or non-verbal, such as through eye contact, smiling, frowning, turning towards the speaker or nodding. Good interpersonal skills result from both types of communication. It is usually the non-verbal forms of communication that suffer when technology is used to communicate.

THINK ABOUT APPLIED COMPUTING 5.17

Observe the individuals within a group – it might be in a coffee shop, a library, at home, at school or another place. Write down three different types of non-verbal communication that you see being used within that group.

5

CHAPTER SUMMARY

Essential terms

accessibility aids technological solutions designed to make everyday tasks accessible to everyone, especially to those with an impairment or disability

assistive technology any technology that allows an individual or organisation to complete a task that they would not normally be able to complete, or makes it easier or safer to complete day-to-day activities

augmented reality (AR) overlaying computer-generated images onto real-life images to provide a composite view

automation when processes or procedures can be completed without the need for human intervention

blockchain a collection of data records (blocks), or a database, that is stored on multiple devices (chain)

chatbot software that uses voice or text prompts to mimic human conversation

cognitive aids documents or prompts designed to assist people with a physical, cognitive or emotional impairment that affects processes such as attention, memory, navigation, planning or sequencing of events or tasks

cryptocurrency a type of virtual currency that uses encryption to secure transactions

cyberbullying the use of a digital system to bully another individual; this could be in the form of intimidation, harassment, rumours or threats

datafication the process of turning one or more aspects of everyday life into quantifiable, measurable data

deskilling the replacement of a skilled worker, or a reduction in their job specification, occasioned by the implementation of technology

digital divide the gap between those people who have access to technology and those who do not

digital system a system designed to store, process and communicate information in digital form

drone an aircraft without a human pilot

economics the study of the production, distribution and consumption of goods and services

electronic funds transfer (EFT) a system of transactions that allows financial payments to be made from one bank account to another, using a digital system

emerging technologies new technologies that are perceived to be capable of making significant change within society

fraud detection identification of unauthorised financial transactions

global positioning system (GPS) a network of satellites that provide location and time data to any connected device

goal a desired result, such as one it is expected that a digital system can achieve

human interactions instances of people communicating with other people

Internet of Things (IoT) collectively, the billions of devices around the world that are connected to the internet, collecting, receiving and sending data

interpersonal skills skills required to interact and communicate with others

job loss occurs when a job, formerly completed by a human, is no longer needed in an organisation

microcontroller an integrated circuit that incorporates a processor, memory and input/output (I/O) controllers within one chip

9780170440806

multi-factor authentication a process that requires a user to provide two or more pieces of evidence of identity, such as a username and password combination and an SMS or email code, in order to be able to complete a transaction online

nanosatellite a small satellite that weighs between 1 and 10 kilograms

near-field communication (NFC) technology digital technology using radio waves that allows contactless communication between devices over a very small distance, without the need to configure settings; examples include payWave and Myki

objective one of a series of steps that define how a goal will be achieved

payment terminal a hardware device that captures credit card or debit card details and sends these details to a bank for authorisation

robot a device that can complete a specific task with autonomy

satellite an object in space that orbits around a bigger object

smartphone a telephone integrated with a digital system, allowing functionality not traditionally associated with telephones, such as an operating system, web browsing, email and the ability to run software applications

smart refrigerator a refrigerator with the functionality of a regular refrigerator but that also has the ability to connect to the internet through a wi-fi connection

sustainability the process of managing change in a balanced, controlled way so that other factors are not negatively affected

Three-dimensional (3D) printing the process of creating a three-dimensional physical object from digital data, usually by building up material in successive layers

traffic management procedures and processes used to efficiently and effectively control the flow of vehicles on roads

virtual assistant a software agent that can perform tasks or services for an individual

virtual currency unregulated, digital money, which is issued and (usually) controlled by its creator and is used and accepted among the members of an online virtual community

virtual reality (VR) the use of computer-generated three-dimensional images and specialised equipment to allow a user to interact with a simulated environment different from the real world

wearable technology small smart devices that can be worn on, or implanted into, the body or woven into clothing

Important facts

1. A **digital system** consists of hardware and software and can consist of a single device, a collection of devices or a network.
2. A **smartphone** is a telephone integrated with a digital system allowing functionality not normally associated with telephones, such as an operating system, web browsing, email and the ability to run software applications.
3. **Smart refrigerators** offer the functionality of a regular refrigerator but also offer the ability to connect to the internet through a wi-fi connection.
4. A **virtual assistant** is a software agent that can perform tasks or services for an individual.
5. **Emerging technologies** are technologies that are perceived as capable of making significant change within society. These include drones, microcontrollers, nanosatellites, augmented reality, 3D printing, wearable technology and the Internet of Things.
6. **Assistive technology** is any technology that allows an individual or organisation to complete a task that they would not normally be able to complete, or makes it easier and safer to complete day-to-day activities.
7. **Accessibility aids** are required to make everyday tasks accessible to everyone, especially those people who have an impairment or disability.
8. **Cognitive aids** assist people with a physical, cognitive or emotional impairment that affects processes such as attention, memory, navigation, planning or sequencing of events or tasks.
9. There is a wide range of technology used within the **financial services industry**, such as online banking, automatic teller machines and point-of-sale payment systems.

9780170440806

10 **Multi-factor authentication** requires the user to provide two or more pieces of evidence to verify their identity to be able to complete a transaction online.

11 **Fraud detection** involves identifying unauthorised financial transactions.

12 **Electronic funds transfers (EFT)** allow a financial payment to be made from one bank account to another using a digital system.

13 **Near-field communication (NFC) technology** uses radio waves to allow contactless communication over a very close distance, without the need to configure any settings.

14 **Virtual currency** is unregulated, digital money, which is issued and (usually) controlled by its creators and is used and accepted among the members of an online virtual community.

15 **Global positioning system (GPS)** uses a network of satellites that provide location and time data to any connected device.

16 A **robot** is a device that can complete a specific task with autonomy.

17 **Traffic management** is a group of procedures and processes that are used to efficiently and effectively control the flow of vehicles on the roads. This includes speed and red-light cameras and point-to-point and bus lane camera systems.

18 A **goal** is a broad statement of the desired result that a digital system is expected to achieve.

19 An **objective** defines how a goal will be achieved. Objectives tend to be specific and measurable.

20 **Economic issues** involving emerging technologies can include:

- access
- deskilling
- job loss
- misuse
- sustainability.

21 **Issues and impacts** related to emerging technologies include:

- automation
- cyberbullying
- human interaction
- interpersonal skills.

9780170440806

TEST YOUR KNOWLEDGE

Review quiz

Components of digital systems

1 What is meant by a digital system?

Digital devices used for a range of current and emerging applications

2 List the advantages of using a smart refrigerator rather than a normal refrigerator.

3 List and describe two tasks performed by a virtual assistant.

4 Describe the components included in a microcontroller.

5 Give one example of where augmented reality is used.

6 Discuss two benefits of wearable technology.

Emerging trends in digital systems

7 Explain what is meant by the Internet of Things (IoT). Describe one benefit of the IoT for organisations.

8 Give two reasons why a person might wear a wearable technology.

9 Explain how robotic devices ensure quality control.

10 Explain how drones are used to improve efficiency in the agricultural industry.

11 What is meant by the term 'a swarm of satellites'? How appropriate is this term?

Functions and capabilities of digital systems

12 Give two examples of assistive technology and how they are used to improve a person's quality of life.

13 List three uses of technology in the financial industry.

14 Describe how GPS identifies a car's location.

15 Describe the components that are used in a robotic device.

16 Discuss one use of technology in traffic management.

Goals and objectives of digital systems

17 Explain the relationship between goals and objectives.

9780170440806

Economic issues involving emerging technologies

18 List two factors that will affect a user's ability to access emerging technology.

19 Explain how deskilling could be a result of emerging technology.

20 Discuss the relationship between automation and possible job loss.

21 Explain how mobile phone production could be made more sustainable.

Impact of current and emerging technologies

22 Describe two ways that automation could affect a worker in the car industry.

23 What is cyberbullying? How does technology contribute to it?

24 Explain why human interactions are important.

25 Give three reasons why good interpersonal skills are important in a workplace.

9780170440806

APPLY YOUR KNOWLEDGE

Answer questions 1–3 below for each of the following emerging technologies:

- Internet of Things (IoT)
- drones
- wearable technology
- augmented reality
- nanosatellites.

1. Identify the digital system components used.
2. Discuss trends in the use of the technology by individuals or organisations.
3. Discuss economic, legal and social issues related to the use of that technology.

9780170440806

CHAPTER

6 Development and evaluation of an innovative solution

KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

Data and information

- techniques for collecting data to determine user needs and requirements, such as interviews and surveys

Approaches to problem solving

- techniques for documenting the development of solutions
- solution specifications such as functional and non-functional requirements, constraints and scope
- characteristics of creative and innovative solutions
- design tools and techniques for representing solution designs, such as mock-ups, pseudocode, sitemaps and storyboards
- functions and techniques for developing innovative solutions
- techniques for validating and testing solutions
- evaluation criteria and techniques for evaluating the efficiency and effectiveness of innovative solutions
- tools and techniques for coordinating and monitoring projects, such as Gantt charts

Interactions and impact

- key legislation and how emerging technologies are affected by: the *Copyright Act 1968*, the *Health Records Act 2001*, the *Privacy Act 1988* and the *Privacy and Protection Act 2014*
- ethical issues arising from the development of emerging technologies

Reproduced from the VCE Applied Computing Study Design (2020–2023) © VCAA; used with permission.

FOR THE STUDENT

This chapter, along with Chapter 5, contains information that is required for Unit 2, Area of Study 1. For this outcome you are required to work as a member of a team to develop an innovative solution to a need or opportunity related to the use of an emerging technology, following the problem-solving methodology. In this chapter you will find information about the laws and other issues related to the use of emerging technologies. The steps involved in the problem-solving methodology are discussed, starting with data collection and continuing through to evaluation. These steps need to be demonstrated as part of the outcome for Unit 2, Area of Study 1.

FOR THE TEACHER

This chapter discusses issues related to the use of emerging technologies, including legal, economic and ethical issues. Then there is discussion of the stages of the problem-solving methodology, which students need to demonstrate as part of the outcome. At the completion of Chapters 5 and 6, students should be able to demonstrate the key skills required for Unit 2, Outcome 1, where, working in groups, they create an innovative solution to a need or opportunity using an emerging technology. The solution can be in the form of a proof of concept, a prototype or a fully developed product.

Shutterstock.com/baranozdemir

9780170440806

Key legislation and emerging technologies

Legislation is the body of laws, created by governments, that individuals and organisations in each government's jurisdiction must follow. The introduction of new technologies presents significant challenges for governments as well as for developers and users of the emerging technology.

By their nature, emerging technologies have the potential to cause significant change in society. Each emerging technology – for example, artificial intelligence, recreational drones, nanosatellites and virtual assistants – can provide new and potentially unseen functionality and features. Governments must be able to respond quickly to create, modify or enforce existing legislation to ensure that these technologies uphold the values of the society in which they function. Equally, the developers and users of emerging technology need to ensure that they are not breaching any existing laws in their government jurisdiction.

In Australia, there are three levels of government. The **federal parliament** is responsible for creating legislation that applies to all individuals and organisations within Australia. **State parliaments** are responsible for creating legislation that applies to all individuals and organisations within a particular state, such as New South Wales or Victoria, while **local councils** create by-laws that apply within their individual local council area.

Both federal and state parliaments create new laws by introducing an Act of Parliament. An **Act of Parliament** is a document that contains a set of conditions and rules about a particular area or topic. Once parliament passes an Act, it then becomes law in that jurisdiction.

Acts of Parliament (or pieces of legislation) that have an effect on emerging technologies include the:

- *Privacy Act 1988*
- *Privacy and Data Protection Act 2014*
- *Health Records Act 2001*
- *Copyright Act 1968.*

Privacy Act 1988

The ***Privacy Act 1988*** is a federal law. It contains conditions and rules related to how and when personal data and information can be collected by Australian Government agencies and private organisations that turn over more than $3 million annually. The Privacy Act was designed to be technologically neutral, but a great deal has changed since 1988 and the vast advances in emerging technologies will seriously challenge the current Act in the future.

Refer to pages 28–30 in Chapter 1 to revise the *Privacy Act 1988* and the Australian Privacy Principles.

CASE STUDY

Emerging technologies and privacy

Technology is not the enemy of privacy. Technology can be privacy enhancing.

...

Developments in biometric technologies have been at the forefront of this change. Back when the Privacy Act was introduced in 1988, many biometric technologies were largely confined to science fiction movies. Of course, a few, such as the use of fingerprints in law enforcement, were well established. However, the concept that biometric technologies could become part of our everyday consumer transactions was almost unthinkable.

A person standing in line at a bank branch in 1988 would struggle to conceive a future where they could phone their bank, be identified by voice recognition technology, and transact from the comfort of their own home. Yet today, this is a reality.

A worker signing a time sheet as they arrived at work in 1988, would struggle to conceive a time when they would be required to have a fingerprint scanned to clock on. Yet for some people today, this is a reality.

A young adult entering a nightclub in 1988 would struggle to conceive a future where they would have to submit to a face scan before being allowed entry. This would have been the crazy plot of some futuristic television show. But today, this is also a reality.

We are likely to continue to see an increasing use of biometric technologies like those I have just mentioned, as well as iris scanning, palm scanning, and many others, in ways that we cannot predict. Assuming that these new technologies are developed in a way that is genuinely sensitive to privacy, this need not necessarily be a bad thing.

What is interesting about biometric technology is that we tend to hear both that it is good and bad for people's privacy.

... For example, voice recognition technology is being rolled out in some call centres to identify callers, leading to more effective protection of clients' personal information.

On the other hand, we hear that biometric technology has the potential to invade our privacy. For example, in the film *Minority Report*, individuals confront ubiquitous iris scanning infrastructure and technology which allows their every activity to be tracked.

How do such obviously divergent views on privacy and biometrics coexist?

The answer is: because biometric technology is what we make it. Biometric technologies are not inherently good or bad for privacy, and privacy is not a blocker to the use of biometric technologies. These technologies can become good or bad for privacy depending on how they are designed, developed and deployed.

Pilgrim, T. (27 May 2010). Privacy in Australia: Challenges and opportunities [Speech], Office of the Australian Information Commissioner website – www.oaic.gov.au CC BY 3.0 Au

THINK ABOUT APPLIED COMPUTING 6.1

Think of one other emerging technology. How does the *Privacy Act 1988* affect it?

Privacy and Data Protection Act 2014

The ***Privacy and Data Protection Act 2014*** is a Victorian state law. It contains conditions and rules related to how and when personal data and information can be collected by Victorian Government agencies, including local councils, and private organisations that conduct contract work on behalf of the Victorian Government and councils.

The *Privacy and Data Protection Act 2014* is based on 10 Information Privacy Principles (IPPs), which, although different, do have some similarities to the Australian Privacy Principles (APPs).

An abridged version of the 10 Information Privacy Principles (IPPs) is shown in Table 6.1.

TABLE 6.1 Information Privacy Principles (IPPs)

IPP 1 Collection An organisation can only collect personal information if it is necessary to fulfil one or more of its functions. It must collect information only by lawful and fair means, and not in an unreasonably intrusive way. It must provide notice of the collection, outlining matters such as the purpose of collection and how individuals can access the information.
IPP 2 Use and disclosure Personal information can only be used and disclosed for the primary purpose for which it was collected, or for a secondary purpose that would be reasonably expected.
IPP 3 Data quality Organisations must keep personal information accurate, complete and up to date. The accuracy of personal information should be verified at the time of collection, and periodically checked as long as it is used and disclosed by the organisation.
IPP 4 Data security Organisations need to protect the personal information they hold from misuse, loss, unauthorised access, modification or disclosure. An organisation must take reasonable steps to destroy or permanently de-identify personal information when it is no longer needed.
IPP 5 Openness Organisations must have clearly expressed policies on the way they manage personal information.
IPP 6 Access and correction Individuals have the right to seek access to their own personal information and to make corrections to it if necessary. An organisation may only refuse in limited circumstances that are detailed in the PDP Act, for example where disclosure might threaten the safety of an individual.
IPP 7 Unique identifiers A unique identifier is an identifier (usually a number) that is used for the purpose of identifying an individual. Use of unique identifiers is only allowed where an organisation can demonstrate that the assignment is necessary to carry out its functions efficiently.
IPP 8 Anonymity Where lawful and practicable, individuals should have the option of transacting with an organisation without identifying themselves.
IPP 9 Transborder data flows If an individual's personal information travels outside Victoria, the privacy protection should travel with it.
IPP 10 Sensitive information The PDP Act places special restrictions on the collection of sensitive information. This includes racial or ethnic origin, political opinions or membership of political associations, religious or philosophical beliefs, membership of professional or trade associations or trade unions, sexual preferences or practices, and criminal record. Organisations can only collect sensitive information under certain circumstances.

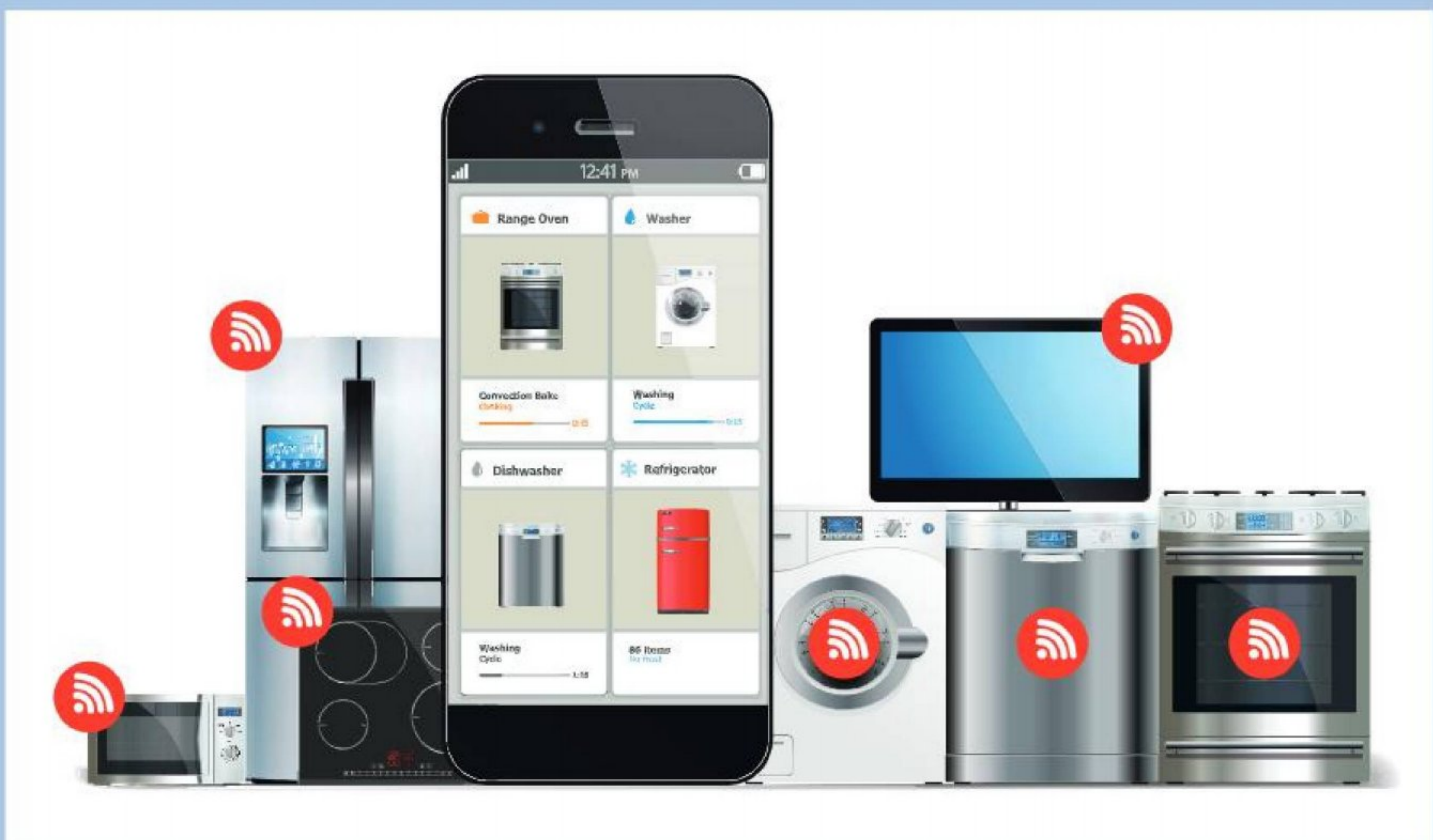
Extract from Information Privacy Principles, Office of the Australian Information Commissioner website – www.oaic.gov.au CC BY 3.0 Au

The Internet of Things (IoT) consists of billions of interconnected devices that collect, receive and send data. IoT devices collect enormous amounts of data, including personal data, and communicate this data, all without the user being actively involved. Data collected via IoT devices is often shared with a range of other devices. Application programming interfaces (APIs) can be used to capture data used in one information system so it can be used in another information system (which may also be independent of the producers of the IoT devices).

RESEARCH

Smart home devices

Smart home devices such as internet-connected speakers, televisions, fridges and microwaves are designed to make our busy lives easier. Who would not want to be able to control their home with their mobile phone from anywhere in the world? You could turn lights on and off, get your fridge to order the shopping or turn on the heating to make sure your house is warm when you arrive home.



Shutterstock.com / Black Jack

FIGURE 6.1 In a smart home, many appliances can be controlled using a smartphone.

Gaming consoles, for example, are found in millions of homes in Australia. These devices rely on internet connectivity to stream video and voice chat. They rely on apps to keep the system and applications up to date. The trouble is that cybercriminals are finding that these devices are a new source of data and information they can use to rob you of money or your identity.

1 Review the information presented in Table 6.1. How does this information relate to the use of always-on smart devices in the home?

2 In the light of these concerns, what steps can you take to ensure that your privacy is protected?

Health Records Act 2001

The ***Health Records Act 2001*** is a Victorian state law containing conditions and rules related to how and when an individual's personal health data and information can be collected by both Victorian government agencies and private organisations that either offer health services or handle health records.

This includes health service providers, private organisations, Victorian and local government departments, public hospitals, other public bodies such as Victoria Police and VicRoads, and sole practitioners and partnerships such as doctors, dentists and physiotherapists.

The Health Records Act is based on 11 Health Privacy Principles (HPP) that, although different, are very similar to the Information Privacy Principles (IPPs).

TABLE 6.2 A summary of the Health Privacy Principles. The principles in full can be found in the Act.

HPP 1 Collection	Only collect health information if necessary for the performance of a function or activity, and with consent (unless consent cannot be obtained and all other criteria under HPP1 are satisfied). Notify individuals about what you do with the information and that they can gain access to it.
HPP 2 Use and disclosure	Only use or disclose health information for the primary purpose for which it was collected or a directly related secondary purpose the person would reasonably expect. Otherwise, you generally need consent.
HPP 3 Data quality	Take reasonable steps to ensure health information you hold is accurate, complete, up-to-date and relevant to the functions you perform.
HPP 4 Data security and retention	Safeguard the health information you hold against misuse, loss, unauthorised access and modification. Only destroy or delete health information in accordance with criteria of HPP4.
HPP 5 Openness	Document clearly expressed policies on your management of health information and make this statement available to anyone who asks for it.
HPP 6 Access and correction	Individuals have a right to seek access to health information held about them in the private sector, and to correct it if it is inaccurate, incomplete, misleading or not up-to-date.
HPP 7 Identifiers	Only assign a number to identify a person if the assignment is reasonably necessary to carry out your functions efficiently.
HPP 8 Anonymity	Give individuals the option of not identifying themselves when entering transactions with organisations, where this is lawful and practicable.
HPP 9 Transborder data flows	Only transfer health information outside Victoria if the organisation receiving it is subject to laws substantially similar to the HPPs.
HPP 10 Transfer/closure of practice health service provider	If you are a health service provider, and your business or practice is being sold, transferred or closed down, without you continuing to provide services, you must give notice of the transfer or closure to past service users.
HPP 11 Making information available to another health service provider	If you are a health service provider, you must make health information relating to an individual available to another health service provider if requested by the individual.

MY HEALTH RECORD

ISSUE

My Health Record

The My Health Record system started in Australia in 2018. It provides a fast online summary of your key health issues so any doctor anywhere can access your patient information. For example, you might live in Hamilton, but could be on holiday at Lakes Entrance and be unfortunate enough to be nipped by a crab. When you are taken to the hospital, the treating doctor in Lakes Entrance can access your health record to see that you are allergic to a variety of antibiotics. This will inform the doctor on the best way to treat you.

This is the scenario for all people who chose not to opt out of the system. People who opted out did so mainly because of concerns about data security. Since the Cambridge Analytica scandal in 2018, in which it was revealed that the personal data of millions of people was harvested from their Facebook accounts and used for political purposes, people have been wary of large online personal data repositories.

The Australian Government has established a Digital Health and Security Centre to secure digital national health records across Australia. This centre will monitor and assess any cybersecurity threat and make regular reviews and improvements to their approach.

How does My Health Record align with the Health Privacy Principles summarised in Table 6.2?

Copyright Act 1968

The ***Copyright Act 1968*** is an Australian federal law containing conditions and rules related to copyright. **Copyright** gives the owner of an original work (such as a software application, a movie, a song or a photographic image) the right to control who can use the original work, and how.

Copyright applies to any original work created in Australia and is automatic and free. This means that the owner of the original work does not need to apply for copyright; rather, copyright is granted automatically as soon as the work is created. No fee needs to be paid in order for copyright to apply. The only requirement is that the work is in a tangible format, not just a thought in your head.

The IoT poses some problems for the concept of copyright. Internet of Things devices can be used in an individual's home, in their car, in their workplace or when they are shopping, to help them with a range of tasks. Devices on the IoT are interoperable. **Interoperability** is the ability of a device to connect and interact with other devices, particularly devices belonging to different information systems.

The functionality of interoperable devices can be seen to conflict with the purposes of the Copyright Act. For two devices to communicate, they must be able to share software instructions with each other. Software is an original work, and hence is governed by the Act. The owner of the original work has the right to control who can use that work, and in what way.

Emerging technologies and ethical issues

Ethics are beliefs, principles or standards that individuals, organisations and society have regarding acceptable behaviour. Ethics are sometimes referred to as knowing right from wrong. An **ethical issue** arises when current behaviours or practices could be considered to be the wrong thing to do. These behaviours or practices may not necessarily breach any existing laws, but they still can be considered unethical.

THINK ABOUT APPLIED COMPUTING 6.2

If your neighbour took photos of you and your friends, who would hold the copyright on those photos under existing laws? Do you consider this behaviour to be unethical?

Drones

Imagine you and your friends are in your backyard. A neighbour's drone invades the space above your head. The camera on the drone captures images of you and your friends and sends them to your neighbour's smartphone. The use of recreational drones (under 2kg in weight) is not yet well regulated. The *Privacy Act 1988* only applies to organisations with an annual turnover of $3 million. Your neighbour is unlikely to be in this category.

Shutterstock.com/ymgerman

FIGURE 6.2 At present, there are few laws overseeing the use of recreational drones.

The Civil Aviation Safety Authority (CASA) is currently reviewing the regulations for the recreational use of drones. At the moment, your only path of redress is to make a privacy complaint to the Office of the Australian Information Commissioner.

The use of drones in Australia – report

Nanosatellites

Nanosatellites were discussed in Chapter 5 (see pages 200–2). Nanosatellites were once the domain of government organisations; now, private organisations use them as well. These private organisations are launching nanosatellites in increasing numbers and are using them to capture images of large areas of Earth's surface. One private organisation that operates commercial nanosatellites is Planet (see weblink). It has more than 130 satellites in orbit, and an ability to monitor your area of interest anywhere on Earth.

Planet

Nanosatellites have the ability to cross international borders, which complicates the question of complying with privacy laws. Usually, a country's laws only have effect within that country's jurisdiction, and that normally means within the borders of that country.

Shutterstock.com/K303

FIGURE 6.3 Nanosatellites are flying over your neighbourhood.

International laws exist that stop individual countries claiming outer space within their jurisdiction. International law states that objects (including satellites) launched into space are to be registered to a particular country, and that the laws of that country will apply to the object. For example, if a satellite is launched from Australia, or if an Australian organisation owns a satellite, the satellite can be registered to Australia, and Australian laws will apply to the satellite.

Some nanosatellites carry extremely powerful cameras that allow amazingly detailed images to be taken. As the cost of constructing and launching satellites has become significantly less, and as the size of satellites decreases, private organisations are entering the industry in the hope of using satellite technology for commercial gain. Launching and flying of these satellites is subject to international law and there are regulatory requirements in national legislation, such as the *Space Activities Act 1998*. This Act states that, if an Australian citizen launches a satellite overseas, they must provide an overseas launch certificate. If any damage is caused by that satellite (such as in a collision with another satellite), then the Australian Government is responsible.

Nanosatellites tend to have a limited period of useful life. At the end of that period, a satellite is decommissioned and is sometimes moved to what is termed a 'graveyard orbit' further away from Earth. As the number of nanosatellites increases, the issue of space junk becomes increasingly important.

THINK ABOUT APPLIED COMPUTING 6.3

Who owns the copyright of the images taken by a nanosatellite?

THINK ABOUT APPLIED COMPUTING 6.4

What are the ethical issues raised by moving nanosatellites into graveyard orbits?

Augmented reality

Augmented reality was discussed in Chapter 5 (see pages 202–3). Pokémon Go, a popular augmented-reality game, was released in Australia in 2016. Players roam the physical environment in a treasure hunt for Pokémon characters, which they 'capture' using their phone camera. The phone uses its GPS capabilities to reward the players with Pokéballs and potions. The search for Pokémon has taken players to cemeteries, hospitals, private land (potentially breaching trespass laws) and public footpaths. All data collected during the playing of the game is owned and stored by Nintendo and Niantic Labs (as stated in their privacy policy). Their privacy policy goes on to say that this data can be shared with third parties for research and analysis.

THINK ABOUT APPLIED COMPUTING 6.5

Which of the IPPs out of the *Privacy and Data Protection Act 2014* (see Table 6.1) affect an augmented-reality game such as Pokémon Go?

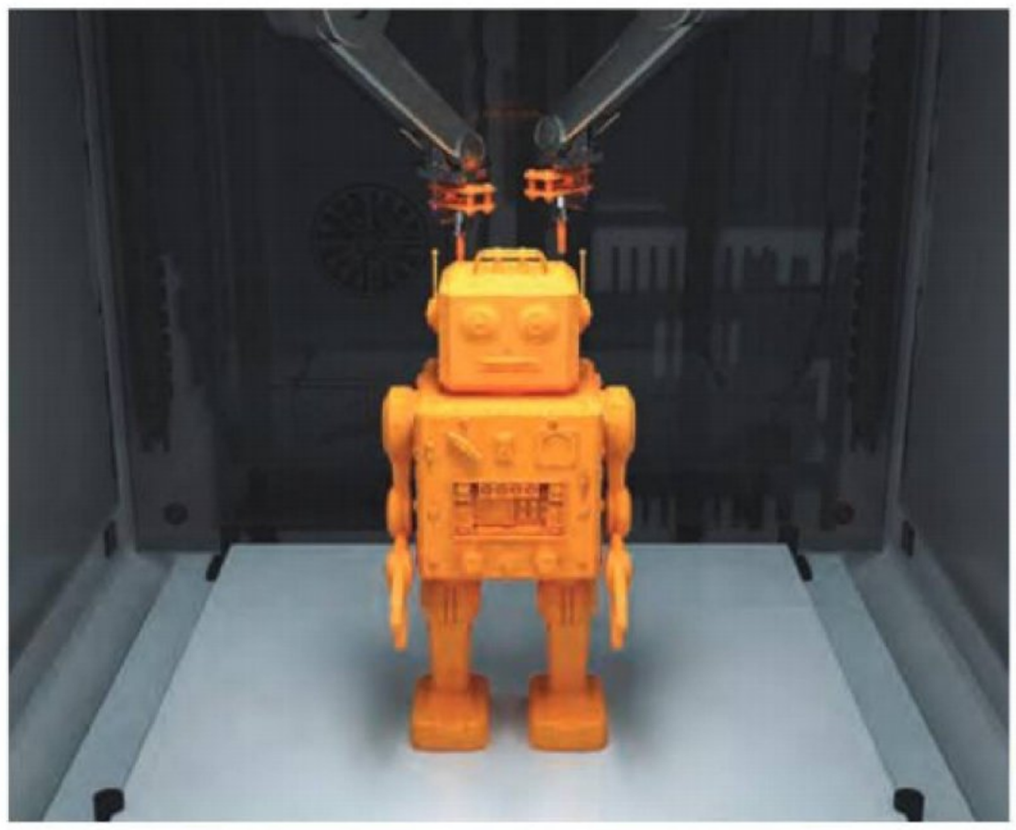
Shutterstock.com/Matthew Corley

FIGURE 6.4 Collecting Pokémon on private land could breach trespass laws.

3D printing

Artistic works are automatically protected by copyright law. These works can include 3D-printed objects and the design files that were used to print those objects. The copyright holder has the right to publish or communicate the files electronically.

If more than 50 copies of an object are 3D-printed, then it is considered that they are *industrially applied* and will be protected under design law. Objects cannot be protected under both copyright law and design law.



Shutterstock.com/Phonlamai Photo

FIGURE 6.5 A 3D-printed robot toy

The environmental cost of 3D printing is significant. The most common material used as filament in the process is plastic, which causes significant pollution during its production and is not recyclable. The amount of electricity used in the 3D printing process is significantly greater than in traditional printing methods. Some 3D printing processes release harmful particles into the air, potentially causing health problems for those people within close proximity of the printer.

THINK ABOUT APPLIED COMPUTING 6.6

Who owns the copyright on the orange robot shown in Figure 6.5? What is the cost to the environment in producing this toy?

Wearable technology

Wearable devices such as smart watches and activity trackers are a booming business. More than 125 million smart watches were produced in 2018, and it is predicted 190 million will be produced in 2022. These devices have become a vast store of health and medical data such as heart rates, blood pressure, the exercise people do and how many kilojoules they consume. Apps can be downloaded that extend a device's ability to track fertility cycles and blood glucose levels, provide medication reminders and produce electrocardiograms (ECG). Very soon, there will be little information that our smart device does not know about us. It is predicted that device makers will look to a new source of revenue: selling health data to insurance companies. The time may not be too far away when your insurance premium will be determined by the amount of daily exercise that you undertake.

THINK ABOUT APPLIED COMPUTING 6.7

How do the *Privacy and Data Protection Act 2014* and the *Health Records Act 2001* relate to wearable technology?

Coordinating and monitoring projects

A project consists of a series of interrelated tasks that need to be completed to be able to create a solution to a need or opportunity.

A project has the following characteristics:

- a clearly defined purpose
- a start time
- a limited timeline
- a number of tasks.

Examples of a project include:

- building a house
- writing an essay for school

- moving house
- installing a new computer network
- developing a software solution.

You can draw on a number of tools to assist you when planning projects. Most are designed to show different aspects of a project's status, such as the time taken or which tasks depend on one another. The project-management tools discussed below should be used to complement one another, not in isolation. Project-management software does not create tasks and assign resources. You must make informed decisions, and the software will help you with the management, documentation, presentation and communication of project information. You can update the documentation easily when changes are made during the course of the project.

For Unit 2, Outcome 1, you will, in collaboration with other students, analyse, design, develop and evaluate an innovative solution to an identified need or opportunity involving a digital system. The first part of that task will involve creating a project-management plan, then monitoring the project and updating the plan as required during the course of the project. The first part of the project-management plan will be the project table; this is followed by a Gantt chart. Finally, keep a record of the progress and any changes required to the timeline in the project log. These project-management tools are all explained below.

Project table

The project manager will brainstorm, on a separate sheet of paper or word-processing document, all the possible tasks involved in analysing a problem, and designing, developing and evaluating a solution (Table 6.3).

TABLE 6.3 The main headings of a project table for publishing a report, with sample data

Task number	Task name	Duration (days)	Resources required	Task milestone (Y/N)	Dependent tasks (predecessors)
1	Departmental reports	5		No	N/A
2	CEO's report	3		No	N/A
3	Proofread	2		Yes	1, 2
4	Obtain artwork	5		No	N/A
5	Cover design	3		No	4

Gantt chart

A **Gantt chart** provides a standard format for displaying project schedule information. It lists the project tasks worked out in the project table, and their corresponding start and finish dates, in a calendar format (Figure 6.6). Gantt charts not only show a timeline for completion of the project, but they can also highlight tasks that are critical to the timely completion of a project. Using a Gantt chart makes it easy for the project team members to see when tasks need to start and how long they should take. Many Gantt charts also show milestones and basic task dependencies. **Dependencies** show the relationships between tasks, when a task should begin or end in relation to another task. Although you can use dedicated project-management software to generate Gantt charts, simple ones can be constructed in spreadsheet programs.

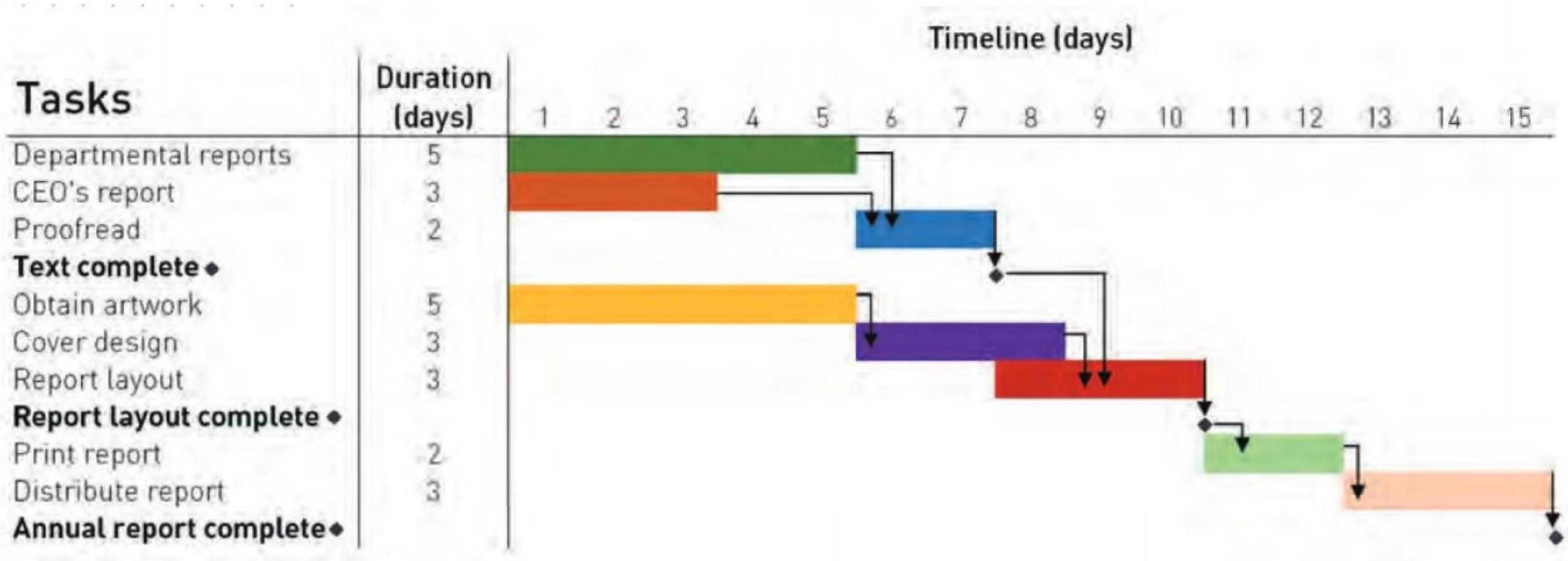


FIGURE 6.6 An example of a Gantt chart created in Microsoft Excel. It displays the information shown in the project table (Table 6.3). Simple dependencies have been shown with arrows. Different colours can be used to make it easier to identify individual tasks.

Annotations

Annotations are comments or notes that are placed on the Gantt chart to explain any changes that may be made. These provide useful information to project team members about problems that may be occurring and may identify any recurring issues with the project. These annotations may also become useful after the project is completed, providing lessons about things that should be avoided in future projects.

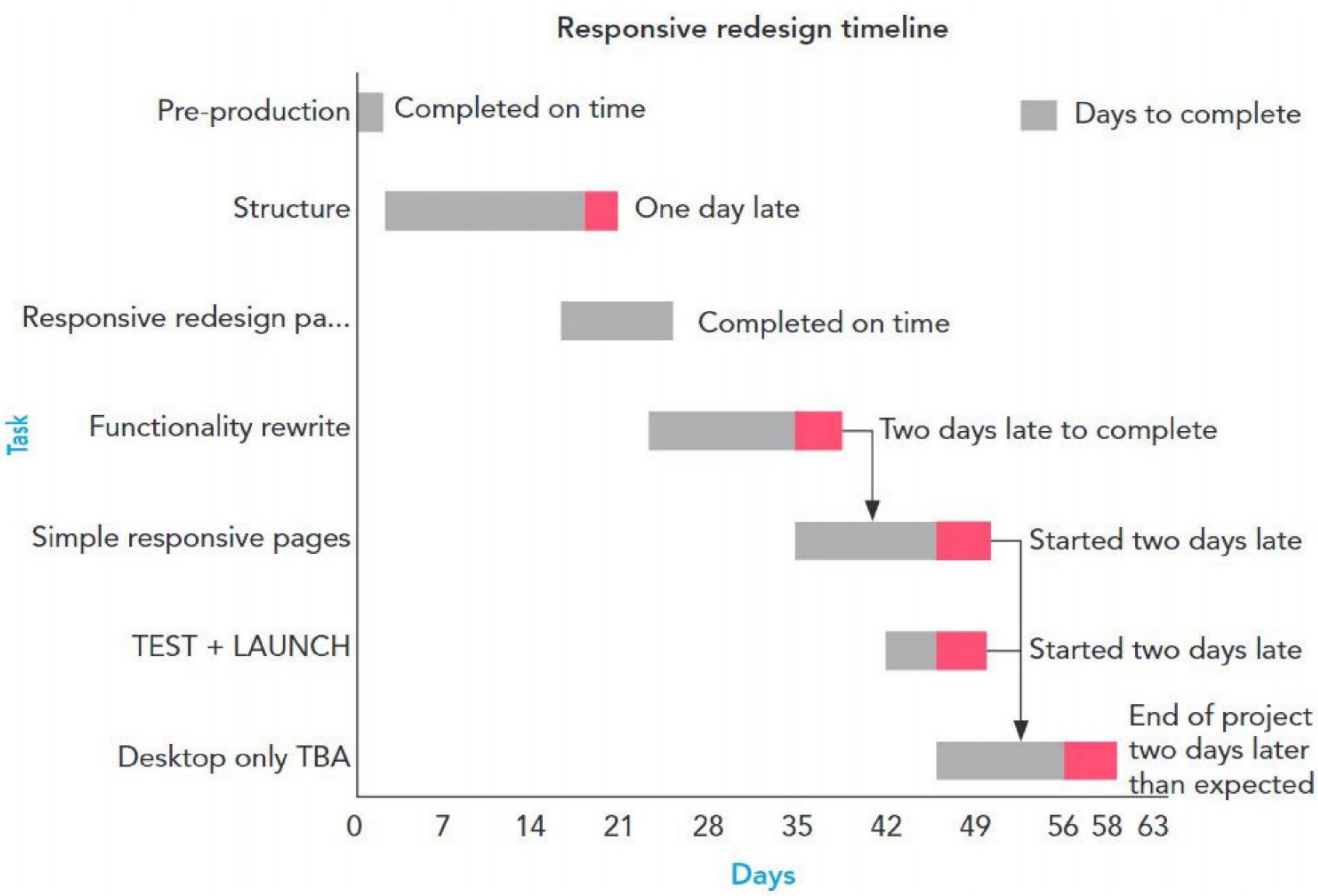


FIGURE 6.7 Gantt chart with annotations

Adjustments

Adjustments are any changes that may need to be made to the timeline as a result of events that have occurred as the project progressed. Sometimes, if a task takes longer to complete than expected, the delay may not have any effect on other tasks. If this situation occurs, adjustments to the project plan will be required.

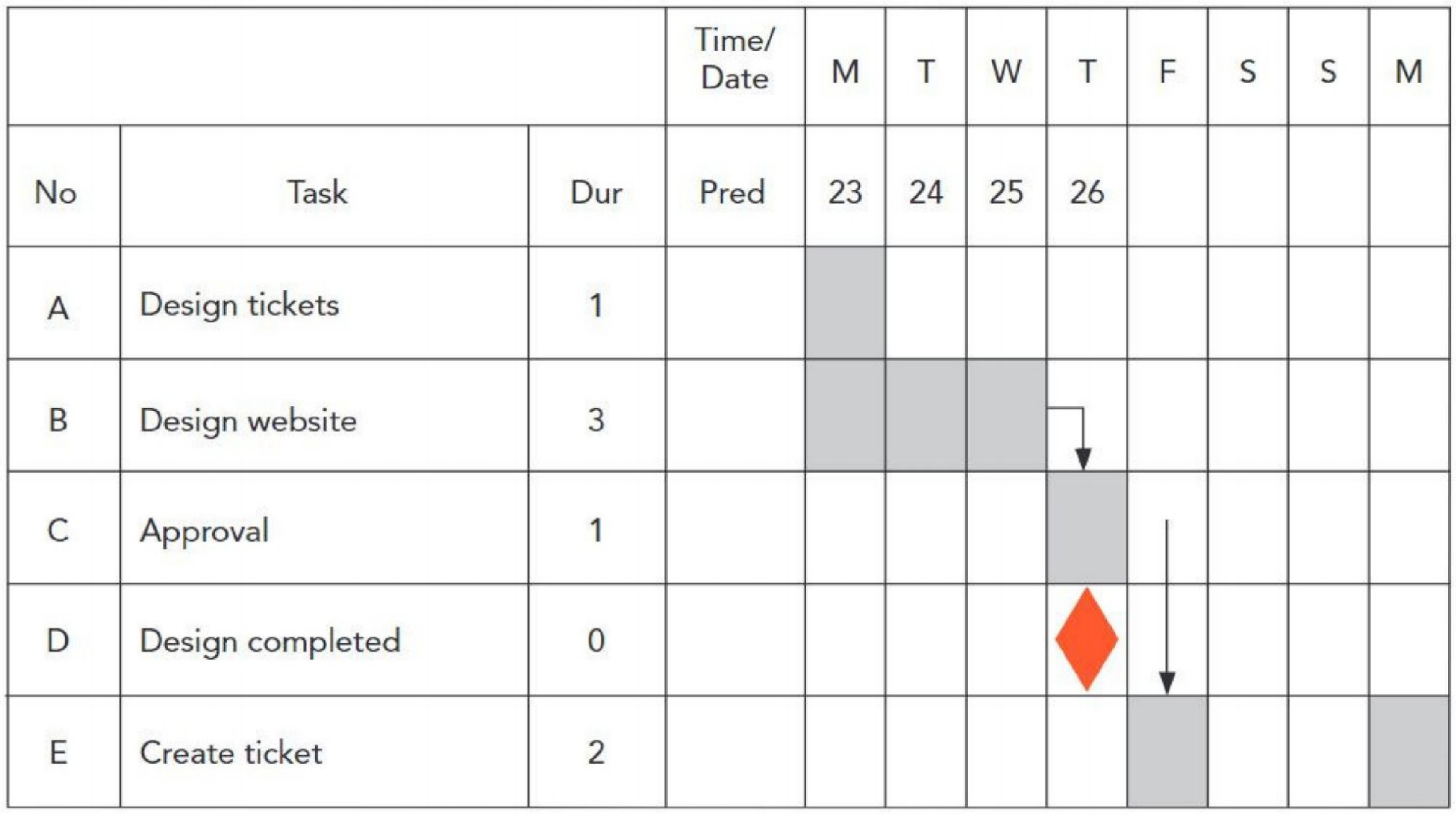


FIGURE 6.8 Gantt chart before adjustments

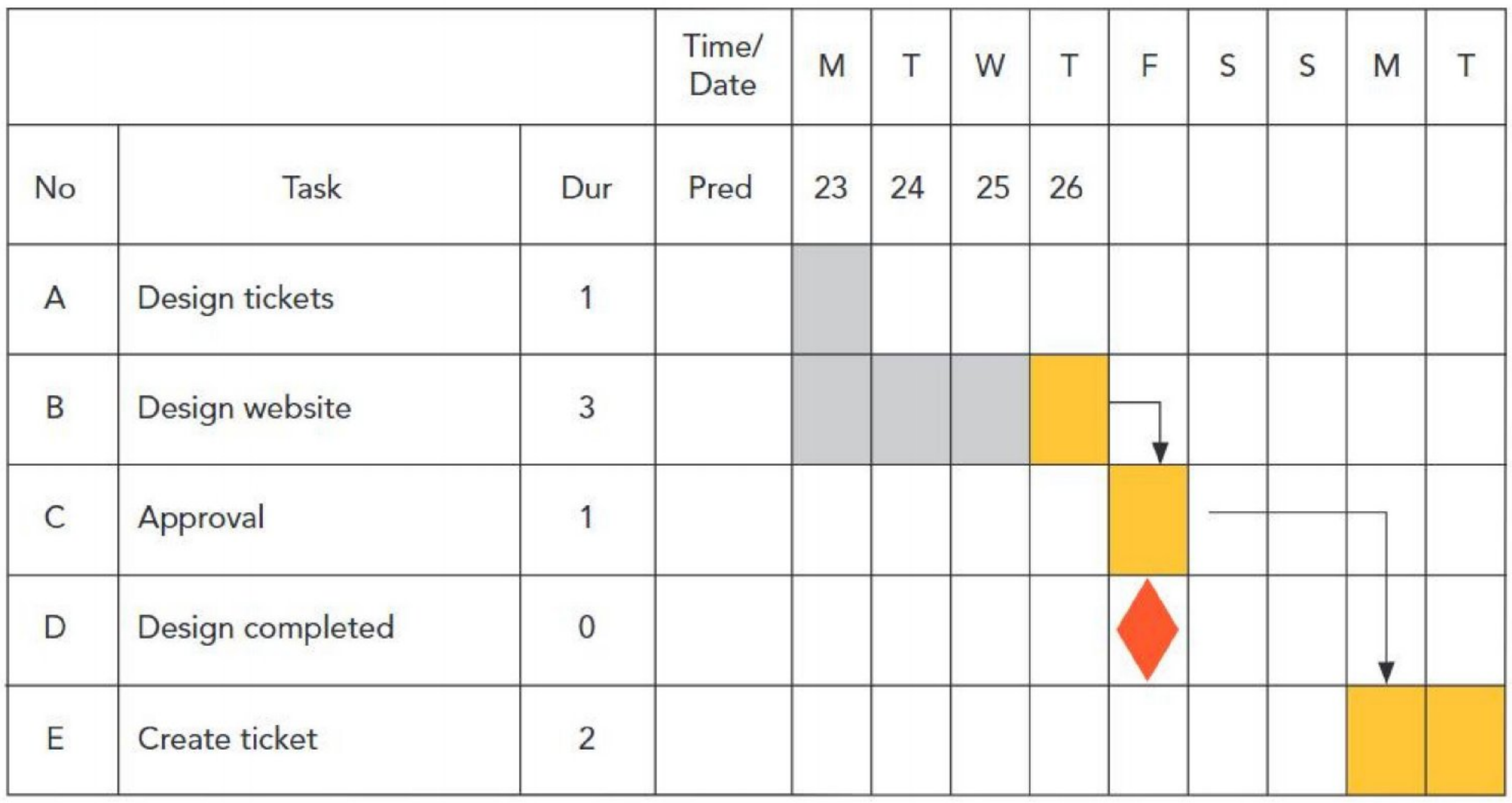


FIGURE 6.9 Gantt chart after adjustments

Project logs

Once a project is under way, a **project log** can be used to document the development of the solution. A project log is similar to a diary in which events that affect each task in the project are written down to create a history of the project. These logs may assist by providing details that can be used to complete tasks, or by indicating problems that occurred so they can be avoided in later stages of the project.

	A	B	C	D	E	F	G	H	I	J
1	Issue #	Date	Reported by	Item/Functionality	Description	Issue type	Priority	Impact	Suggested fix	Status
2	1	2020-05-09	Developer	Input box submit button	Spacing too close to text box	Usability	Medium	High	Increase spacing introduc colour and label with meaning	Resolved, adjusted size, colour and labelling
3	2	2020-05-09	Developer	Title	The font size of the main input screen is too small, hard to read.	Design	High	High	Adjust the size and colour contrast	Resolved, larger font, darker colour
4	3	2020-05-09	Developer	What to do is not clear	After reading the introduction screen, the search and sort buttons are not clearly marked.	Usability	Medium	Medium	Adjust the size and colour shading of the button. Add a label	Partly resolved, need to work out how to include a label inside
5	4	2020-05-09	Developer	Image quality of LOGO	Image is very poor quality	Data	Medium	Low	Locate better original and re-capture better quality screenshot. Completely redraw to create new original.	Still looking to locate image, yet to be attempted.
6	5	2020-05-10	Reviewer	Sources and Privacy statement	There is no acknowledgement of sources, destination of data or privacy statement	Data	Low	Low	Add Sources citation link and link to Privacy terms and conditions to lower left of opening screen	Yet to be completed

Project log template

FIGURE 6.10 Project log template created in Microsoft Excel. Columns C, F, G and H are data validation drop-down lists. Columns G, H and J also have conditional formatting applied to indicate priority and impact by colour: low, medium, high, critical and showstopper. This template is available on NelsonNet.

Collecting data

A *need* responds to an existing *problem* (reactive), whereas an *opportunity* is the development of a new *solution* for a problem that doesn't necessarily already exist (proactive).

As part of the analysis stage of the problem-solving methodology, you will identify the requirements, constraints and scope of the solution. To do this, you will need to collect data about the identified need or opportunity.

Techniques that can be used to collect data, such as observation, **interviews** and **surveys**, are discussed in Chapter 1. Refer to pages 6–8 to review these techniques.

CASE STUDY

An artistic endeavour at Keen College: Identifying the opportunity

Ms Curry is the Year 11 Art teacher at Keen College. Her students are currently working on a major piece of assessment, in which they are experimenting with different materials and techniques to produce a piece of artwork that is significant to them. Each student must keep a visual diary to document their artistic journey. Ms Curry is very proud of her students' achievements and wants to make their finished artworks available to the whole school community, including other students, teachers and all parents.

Ms Curry mentioned this idea to Mrs Mustard, the Year 11 Applied Computing teacher. Ms Curry said that she thought that perhaps a digital solution would work best. Mrs Mustard was delighted as she has been looking out for a possible project for her students. Ms Curry and Mrs Mustard have decided to combine the talents of their two classes to come up with a solution to this opportunity.

Shutterstock.com/Layland Masuda

FIGURE 6.11 Ms Curry working with one of her art students

Solution specifications

The problem-solving methodology is used to create a solution that will meet the needs of the users.

The first stage of the problem-solving methodology is the analysis stage. This is often considered the most crucial stage. Many organisations invest a lot of time, effort and money in this stage. They consider that getting the analysis stage correct and having a clear picture of what the user requires will save time and money in the latter stages.

The analysis stage is typically about 'What?' questions. What is the current information need? What is required to meet the current information need? What constraints may restrict the requirements? What is the scope of the solution?

The analysis stage consists of three activities:

- determining solution requirements
- determining solution constraints
- deciding on the scope of the solution.

6.8
THINK ABOUT APPLIED COMPUTING

Explain how completing a thorough analysis may save time and money in the latter stages of the methodology.

Solution requirements

Solution requirements state what the client needs from the solution. What features do they want in the solution? Solution requirements can be broken down into functional and non-functional requirements.

9780170440806

Functional requirements

Functional requirements are directly related to what the solution will do. These become the functions that the software incorporates. Examples of functional requirements for a smart refrigerator might include the ability to:

- detect items stored in the refrigerator
- track use by and expiry dates
- control temperature by drawer or compartment
- send alerts when issues are encountered with the refrigerator
- control settings using a smartphone app.

Non-functional requirements

Non-functional requirements are the characteristics users or clients would like the solution to have but that do not affect what the solution does. Examples include that it:

- is user-friendly
- has a clear user interface
- does not display any personal details
- is compatible with different operating systems
- has app portability so it can operate on devices of different sizes.

CASE STUDY

An artistic endeavour at Keen College: Project planning and solution requirements

Ms Curry and Mrs Mustard arranged to get their Art and Applied Computing (AC) students together one lunchtime. There were three students who took both classes so these students were designated as the 'go-betweens', taking on the role of keeping both classes up to date with what was happening.

The AC students want to apply the problem-solving methodology to finding a solution to this opportunity. The students sat down together and generated a project table of the key tasks that would have to be performed. Then they created a Gantt chart, which they shared with the students of both classes so that everyone was aware of the tasks, their dependencies and the timeline they needed to work within.

The students would then document all stages of the process in the project log (see pages 245–6) and update the Gantt chart if task requirements changed.

The first stage of the problem-solving methodology is *analysis*, starting with determining the solution requirements. The AC students decided to interview each art student individually to find out what they wanted. Half the AC students agreed on the interview questions and then conducted the interviews.

The other half of the AC class created a Survey Monkey survey about user requirements and sent this out to all teacher, student and parent email addresses stored on the school database.

As a result of the interviews and the survey, it was determined that the solution requirements would be:

Functional

- Display one piece of art from each art student.
- Present, in some format, each art student's visual diary.
- Make the gallery available to all students, teachers and parents.
- Have the ability to add more art pieces.
- Alert users when more art pieces have been added.

Non-functional

- Image files need to be downloaded in a reasonable time.
- The solution has to be accessible.
- Solution is available to use on all devices.
- The gallery is available 24/7.

Constraints

Solution constraints are factors that may limit or restrict the solution requirements, both functional and non-functional.

When buying a car, the amount of money available for the purchase may restrict the user's requirements. The user may not be able to purchase the car they really want. Instead, they may need to re-evaluate their needs and buy a car that is closer in purchase price to the money that they have available.

Like requirements, constraints can be broken down into groups: economic, technical, social, legal and usability.

Economic

Economic constraints include time and available budget.

The deadline by which the user or client needs to have the solution operational will define the time available to design and develop the solution. The longer the time available, the more opportunity there is to complete an in-depth analysis, do detailed designs and develop advanced functions of the solution. The shorter the timeframe, the faster each stage in the problem-solving methodology needs to be completed.

Meanwhile, the budget (money) available to complete the project may affect the hardware and software (digital systems) available for use, the number and range of staff who are available to work on the solution and even the data used as input.

A lack of either time or money may necessitate a re-evaluation of the user's requirements, or a re-evaluation of how the requirements can be achieved.

Technical

Technical constraints are constraints related to the hardware and software available for the project. Available hardware and software, memory and storage capacity, processing and transmission speeds, and security concerns are all possible technical constraints.

For example, developers need to keep in mind that smartphone users may not always have access to a high-speed network connection. They need to ensure that any animated data visualisation solution does not require a large amount of bandwidth to download and view.

6.9 THINK ABOUT APPLIED COMPUTING

List three technical constraints that developers of a drone may need to consider when developing the product.

Social, legal and usability

Non-technical constraints relate to areas other than hardware and software.

Usability and the user's level of expertise (social) are examples. If a solution is being developed for users with little expertise in digital systems, this may restrict some of the requirements that would involve complex manoeuvres to complete. Creating a solution for a younger audience may restrict the method used to input data into the solution.

Legal requirements such as privacy laws may restrict features linked to displaying personal data in the solution, or to collecting data from the devices of someone using the solution. Copyright laws may restrict features that use material for which permission has not been granted; for example, as a background in an augmented-reality application.

CASE STUDY

An artistic endeavour at Keen College: Constraints

On the basis of the solution requirements identified from the interviews with the art students, the AC students brainstormed all the constraints that they could think of and categorised these under headings, as follows.

Economic

- School Council has made $500 available to produce the solution.
- Ms Curry wants the solution available one week before the school Open Day at the start of Term 4.

Technical

- The school has a Bring Your Own Device program, but most students use a Windows laptop.
- Download speeds at the school can be very slow.

Legal

- Student privacy if student names are published
- Copyright of student work

Usability

- Accessibility for deaf and blind members of the school community

Scope of the solution

After the solution constraints are identified and the solution requirements are adjusted based on the constraints, the scope of the solution can be created.

The **scope of the solution** outlines the boundaries or parameters of the solution so all stakeholders are aware of exactly what the solution will contain and what it will not contain. It is best to outline at the start of the project what will and will not be included in the solution. This can prevent disagreements later in the project between the user and the developer.

The scope of the solution consists of two elements:

- *what the solution will do* – a list of all the solution requirements (both functional and non-functional) that will be included in the solution
- *what the solution will not do* – a list of all the solution requirements that will not be included in the solution. Usually, these are solution requirements initially sought by the client, but that have been left out of the solution project because of constraints.

THINK ABOUT APPLIED COMPUTING 6.10

List three problems that a clear scope of solution can avoid later in the project.

CASE STUDY

An artistic endeavour at Keen College: Scope of the solution

The AC students considered the solution requirements on the basis of the identified constraints. They decided that, as the time and funds available for the production of the solution were limited, they would remove the students' visual diaries from the functional requirements. This would then enable them to produce the solution within the time frame required and the budget available. They agreed that they would publish each artist's name along with a description of the artwork. A copyright notice would be visible on the same page as each of the artworks.

They agreed that the solution scope would be:

In scope

- Display one piece of art from each art student.
- Make the gallery available to all students, teachers and parents.
- Thumbnail image files will be viewed on screen without the need to download the full-size image.
- Each piece of artwork will have the artist's name and a short description.
- There will be a visible copyright notice on each page displaying artwork.

Out of scope

- Have the ability to add more art pieces.
- Alert users when more art pieces have been added.
- Solution is available to use on all devices.
- Have the gallery available 24/7.
- Present, in some format, each art student's visual diary.

Characteristics of solutions

A **solution** involves formulating a way to solve a problem. A **need** is essential to the solution so that the user may complete the task. For example, it might be finding a solution to an existing error or shortcoming in an information system and creating a new information system that allows a user to complete a task that is essential to them. An **opportunity** occurs when circumstances or factors are present that may allow for a new solution to be developed – one that does not already exist, or that people have not even thought of yet. An opportunity involves creating a new unique solution that will be something the users will need.

Once a need or opportunity is identified, existing or similar solutions need to be investigated to identify whether they can be used to meet the need or objective. If an existing solution cannot be used, then an innovative solution is required.

The design stage of the problem-solving methodology is about formulating an original way to solve a problem. It may be possible to create an original solution using existing digital system components, or it may be necessary to take a completely new approach. To qualify as an **innovative solution**, a solution must be unique and developed independently. To develop an innovative solution requires creative problem solving.

If an innovative solution becomes widely used and copied by others, it can be considered to be innovative. A temporary tattoo, for example, is being developed to monitor the glucose levels of diabetic patients. This replaces the painful finger-prick that had to be done several times a day. Electrical and computer engineers are developing the tattoo to have Bluetooth capabilities that will send information to the patient's doctor or to be stored in the cloud.

Steps often followed to devise an innovative solution include these:

- Identify a need or opportunity (see page 246).
- Analyse the problem (see pages 247–51).
- Come up with more than one solution.
- Pick the best solution.

More than one solution

When designing and developing an innovative solution, it is advisable to consider more than one way to meet the need or opportunity. This occurs during the design stage of the problem-solving methodology. Coming up with more than one solution can occur using a range of methods, such as:

- generating a solution idea
- creating alternative designs.

Although these two methods sound similar, they involve slightly different techniques.

Generating solution ideas

There are a variety of techniques available to generate solution ideas.

Brainstorming

A common technique is brainstorming. Often in a brainstorming session, the participants are encouraged to offer any possible idea they can think of that may solve the problem, regardless of how crazy or far-fetched the response may be considered by others.

When brainstorming, try not to hamper the imagination by rejecting ideas too soon. Brainstorming is a process in which ideas are presented in a non-judgemental, spontaneous, unstructured – and admittedly somewhat haphazard – process. The only rule is that no idea is criticised or rejected; every idea, no matter how outrageous or silly, goes onto a list of possible solutions.

Participants must have no fear of being ridiculed or judged, making mistakes or breaking rules. While some or many ideas *do* turn out to be ridiculous, sometimes a half-baked, half-comical concept may in fact turn out to be an unexpected work of creative genius, or it may stimulate a related idea that would be perfect.

There are certain rules that you need to adhere to when you are running a brainstorming session. The most important one is that no one judges any contribution. You must accept all contributions. An idea that may seem slightly crazy to begin with can sometimes be workshopped into a great idea. In the 1970s, a brainstorming session came up with the idea of a pet rock. The idea was workshopped, and before long you could buy not only a pet rock but a pet rock house and a training manual. Everyone had to have a pet rock in the 1970s and the idea made the company millions of dollars. It was the pre-technological version of a Tamagotchi.

Make sure that everyone listens to everyone else's ideas. Specify that only one person talks at a time and there is only one idea at a time. This not only ensures that even the shyest member of the group will contribute, but also makes it easier to record the ideas. Using these rules will assist you in getting a large number of ideas with which you can work.

Getty Images/Al Freni/The LIFE Images Collection

FIGURE 6.12 Pet rocks were all the rage in the 1970s.

One strategy behind the use of brainstorming is that allowing participants to be free-thinking means they are not constrained by traditional approaches to solving a problem. The hope is that they may come up with a creative way to meet the need or opportunity.

Mind mapping

Mind mapping is ideal for complementing the process of brainstorming. Mind mapping is a technique for quickly generating and linking ideas. It is a creative and flexible tool that enables you to add, connect, organise and reorganise ideas. Mind-mapping software is generally flexible enough that you will not need to stop very often to learn how it works; in other words, when you are mapping, your creative flow will not often be interrupted.

Unlike old-fashioned sheets of paper or whiteboards, electronic mind maps can stretch endlessly in any direction. You can easily add or remove links between items or allow entire branches of thought to be moved to new locations, and you will not face the laborious task of copying out all of the scribbled ideas at the end of the brainstorming session. The mind map can be saved for later development, printed, saved as an image, or transferred to a word processor.

6.11 THINK ABOUT APPLIED COMPUTING

Other techniques used to generate solution ideas include:

- experimenting with similar solutions
- role-playing
- visualisation
- reverse engineering
- questioning.

Choose one technique and find out what it involves.

Shutterstock.com/Andrey_Popov

FIGURE 6.13 Mind maps allow you to generate and link ideas easily.

Creating alternative solutions

When designing the solution to a problem, the first idea you have is rarely the best one. A different strategy might be cheaper, easier, faster, more effective, or may better meet the client's demands. Questions to be asked when thinking about alternative solutions are:

- What else can we …?
- What if …?

Substitute

Replace part of the problem with something else. For example, if you have always keyed contact details into a spreadsheet, you might find it more useful to use a database instead.

Combine

Join unconnected things together, such as making travel easier by combining a suitcase and a scooter to make a scootcase.

FIGURE 6.14 A scootcase – a suitcase combined with a scooter

Adapt

Use an existing component in a different way, such as using mind-mapping software to create a site map for a website. The first spreadsheet was created using the concept of paper-based accounting books.

Strip

Reduce the problem right back to its most basic parts and see what is left. For example, the tiny and cheap computer, the Raspberry Pi, is a stripped-down Linux PC with minimal components.

Inspecting the basics may reveal the nature of a problem more clearly.

Compare

Ask yourself, 'What other thing do I know that resembles *this* problem, and how does that other thing work?' For example, when sending a number of print jobs to a single printer, how can they be handled? Like a group of people waiting at a gate, you could organise them into a queue and process them in the order of their arrival.

Sleep on it

Creators often reach a point where they can make no further progress. Rather than dwelling on the same failed ideas, it is often better to let them go and think of something else. While the front of your brain is enjoying a walk with your friend, or an episode of *My Kitchen Rules*, the back of your brain will busily be pulling ideas together to create a solution.

Research

Thomas Edison said: 'Through all the years of experimenting and research, I never once made a discovery. I start where the last man left off.' It is important to learn from your predecessors so you do not waste time re-inventing the wheel.

How have other people solved problems similar to the one you face? You are unlikely to be the first person in history to have faced such a problem before. How have others coped? Their successes may lead you in the right direction, and their failures may prevent you wasting time.

Visualisation

Geniuses often make their thoughts visible because words cannot adequately convey the ideas they have. Albert Einstein was famous for his non-verbal thought experiments. He visualised travel at the speed of light as travelling on a train. He said that written words and numbers did not play a significant role in his thinking process.

Pick the best solution

Once alternative solutions have been workshopped, the next step is to select the one that will provide the best solution to meet the need or opportunity. While one idea may be attractive to the developer, the client may have non-technical constraints or priorities that will make one strategy more attractive than another.

Providing a range of solution ideas lets you choose the one that best suits your requirements. You may have used an idea successfully in the past, but it may not be appropriate in the current circumstances. Although previously proven strategies can be useful, you need to be willing to think outside the box. Old strategies will not work for you in every situation, and it is lazy and unimaginative to assume that they will.

A successful problem-solver will consider current functional and non-functional requirements and relevant constraints to develop an imaginative range of options, from which the best solution can be chosen and developed into a detailed design.

The criteria for choosing the best solution include:

- how easy it is to use
- accessibility
- how long it will take to implement
- scalability (how easily the product can be increased in capacity)
- its scope for future modification and enhancement
- the degree to which it satisfies all requirements
- the degree to which it copes with constraints
- ease of implementation
- development cost, future running and maintenance costs

- the amount of labour required to create it and keep it working
- the amount of disruption likely to be caused to the organisation
- compatibility with existing hardware, software, data and procedures
- the amount of training required for staff.

Some decisions can be very difficult, and require careful balancing of competing needs – usually, cost against quality. A solution that is cheap and quick to produce may be barely competent, quickly wear out or be unpleasant to use. A superior solution that would lead to a solution with a long life and happy users will probably take longer to produce and cost more.

CASE STUDY

An artistic endeavour at Keen College: Choosing an innovative solution

Now that the AC students had scoped out the solution, they have to come up with the type of solution they are going to use. As they have a class of 21 students, they decide to split the class into three equal groups. One group will use the brainstorming technique to come up with innovative solution ideas, one group will try the mind-mapping technique and the other group will search the internet to see what has been done before to solve a similar opportunity. They think that by doing it this way they will come up with the most ideas.

At the end of all three techniques, they combine their ideas and compare them to the functional and non-functional requirements and the constraints. In this case they pruned down their list to the following five ideas.

1. Physical gallery with QR codes on each piece of artwork to supply the information
2. Physical gallery with AR markers on each piece of artwork to supply the information via mobile device app
3. Physical gallery with Bluetooth beacons that would supply different information depending on the range
4. Web page gallery
5. Printed catalogue with AR capability

The students then compared their cut-down list to the selection criteria to see how each fared (see Table 6.4). Each idea is rated on a scale of 1–10, where 1 is poor and 10 is excellent. The scores for each idea were then added to get a total.

TABLE 6.4 Each idea ranked against criteria

Criteria	Idea 1	Idea 2	Idea 3	Idea 4	Idea 5
How easy it is to use	7	6	8	8	8
Accessibility	4	4	4	4	2
How long it will take to implement	8	6	6	9	5
Scalability	8	9	6	9	5
Future modification and enhancement	9	9	7	9	4
Satisfies all requirements	6	6	6	9	8
Copes with constraints	8	8	8	9	7

Criteria	Idea 1	Idea 2	Idea 3	Idea 4	Idea 5
Ease of implementation	8	6	6	9	5
Development cost, future running and maintenance costs	9	5	3	8	4
The amount of labour required	8	3	6	8	4
The amount of disruption	8	8	8	7	9
Compatibility with existing hardware, software	6	7	4	9	6
Amount of training required for staff	7	6	5	8	7
Total	**96**	**83**	**78**	**106**	**74**

The class decided that a web page (Idea 4) would be the best solution for the users as it would make the artwork available to all users when they wanted to view it, and it would be ready on time and within budget.

Representing solution designs

A good solution will undergo a rigorous design stage before any attempt is made to implement it. The solution must address the stated need or opportunity and meet all the functional and non-functional requirements listed in the analysis stage.

Whereas in the analysis stage the focus is on what needs to be done, the design stage focuses on how it will be done. Designing a solution can occur without using a computer at all. Much time can be wasted if a design has not been properly planned before work starts on developing the solution.

In the problem-solving methodology, the two activities involved in the design stage are:

- designing the solution
- creating evaluation criteria.

To design the solution, a range of design tools are used to represent both the appearance and the functionality of the solution. The type of design tools used for a particular solution will vary depending on the format of the solution (website, database, drone, software application, etc.).

Types of design tools include (but are not limited to):

- layout and mock-up diagrams (see Chapter 2)
- pseudocode
- site maps
- storyboards.

Pseudocode

If you have decided to create a programming solution, then you would use pseudocode as the design tool to represent the functionality. **Pseudocode** is used to represent the logic or steps of processing that occur when a solution is run or executed. This logic or the steps of processing are often referred to as a solution's algorithm.

Pseudocode is frequently used when developing a solution using a programming language, but can also be used to represent the logic in solutions developed using other software types. Pseudocode combines the structure of programming language code with instructions or steps written in plain English, as it is created to be read by humans, not a computer.

Below is the pseudocode representing the steps of processing that will occur in a solution that a teacher will use to calculate a student's average grade and award based on their tests result.

```
ALGORITHM calculateResult()
BEGIN
     INPUT Mark 1
     INPUT Mark 2
     INPUT Mark 3
     Total = Mark 1 + Mark 2 + Mark 3
     PRINT Total
     Average = Total / 3
     PRINT Average
     IF Average >= 50 THEN
          PRINT Pass
     ELSE
          PRINT Fail
     ENDIF
END
```

Pseudocode is commonly used when creating solutions using a programming language. Advantages of pseudocode are that it allows the logic of the solution to be created before development begins; it allows the algorithm to be bench-tested; and changes can be made much more quickly and inexpensively than would be possible if changes were required to the solution when being developed.

Refer to Chapter 4 for more information on pseudocode.

Site maps

A site map is used to represent the functionality of a solution. A **site map** shows how the information has been structured within the solution, and how a user can navigate to find the required information. A site map is commonly used as a design tool for websites, but can also be used to show the structure of information presented using other formats, including games, multimedia, interactive solutions and smartphone applications.

Site maps help to display the information architecture of a solution. A site map can also be referred to as a hierarchy or tree diagram.

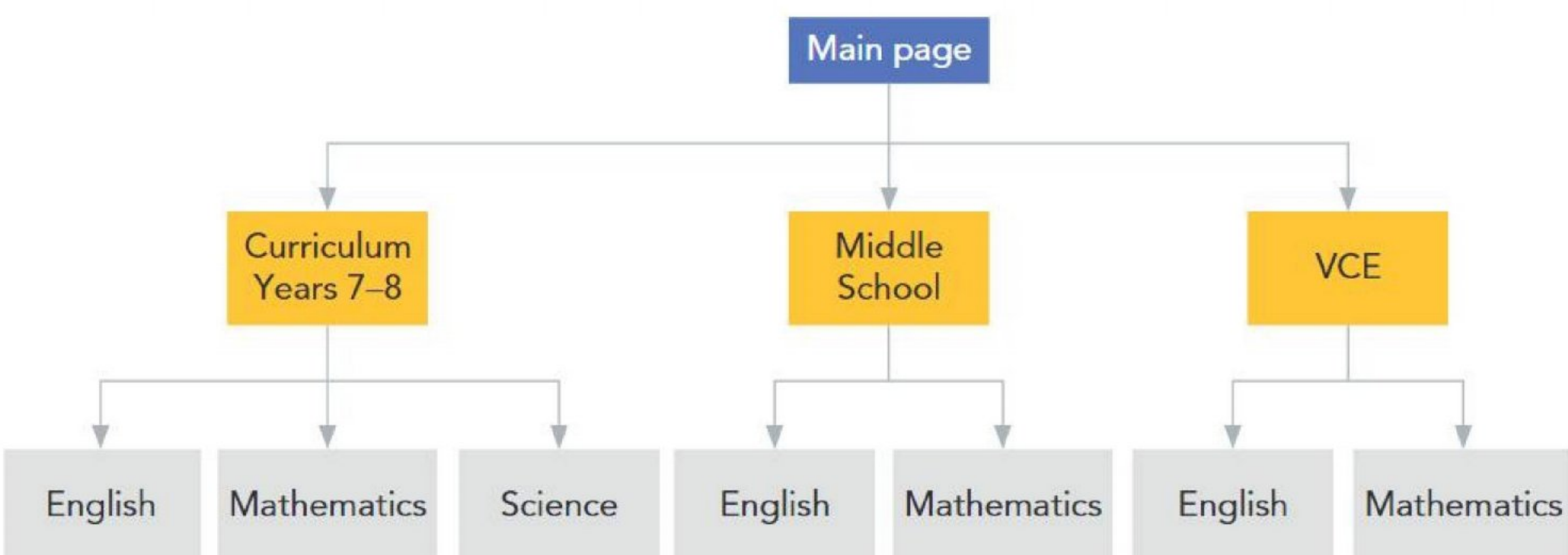


FIGURE 6.15 A site map for a subject selection website

Site maps are useful in that they help to identify how the information should be organised and presented, the number of sections needed within the solution (information architecture) and how a user can access the different sections (navigation).

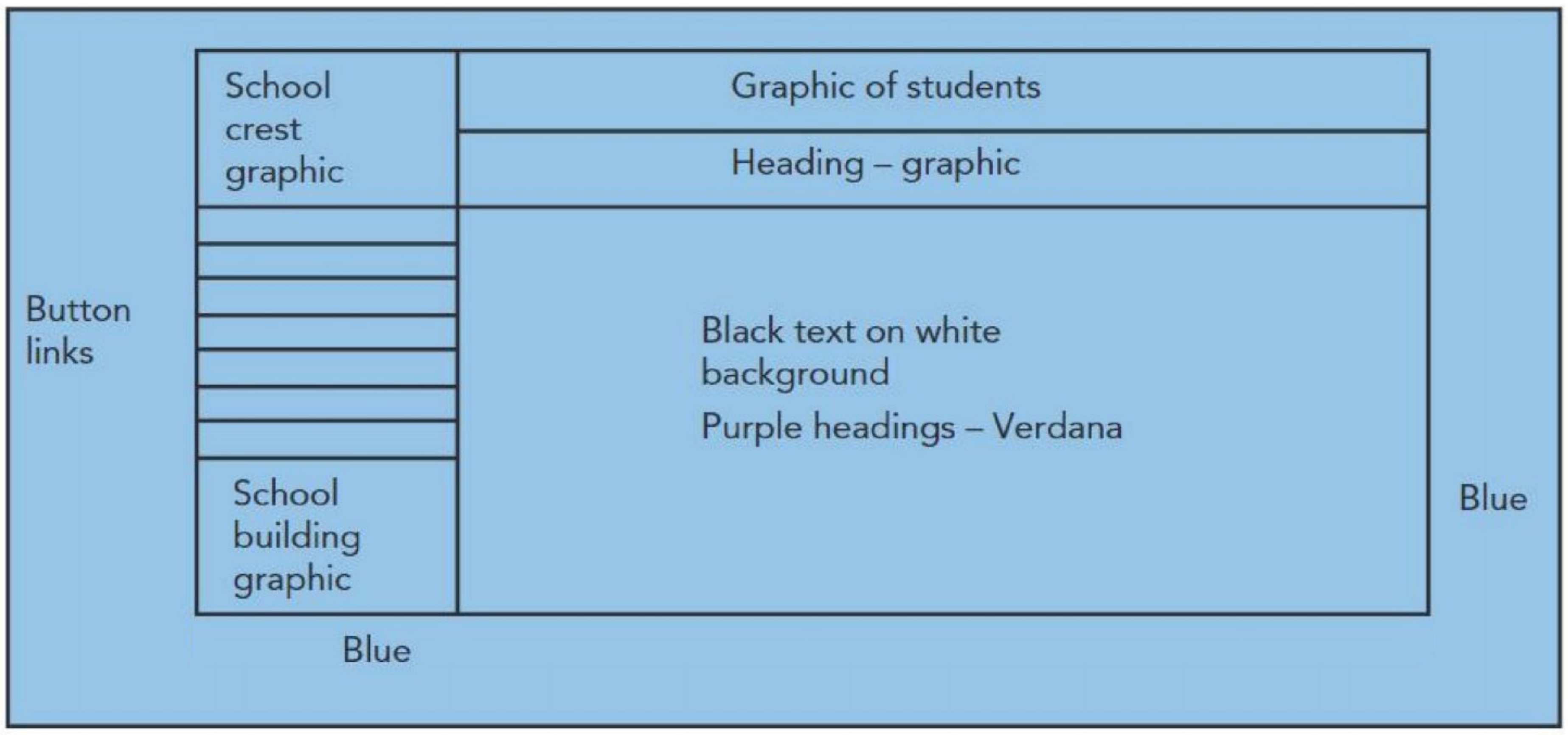
Storyboards

A storyboard is a design tool that can be used to represent the functionality of a solution, but also contains elements about the appearance of a solution.

A **storyboard** shows the features of an individual section of a solution. A series of storyboards can be used to represent the entire solution. Storyboards are commonly used when designing websites, but can be used for other software types.

A storyboard can contain a basic layout diagram showing how the section of the solution can appear.

Storyboard for Darby Vale	Page 1 of 43
Client William	Screen dimensions 1024 × 768



Purpose/description Introduction to Middle School page		Page title	Middle School
		File name	m_school
		Links	
		Link name	Link to
Graphics File name	Size (kb)	Subjects	Table of subject
		Year 7 & 8	Yr 7 & 8 page
DVC.jpg	23	VCE	VCE page
building.jpg	27	Policies	Policies page
heading.jpg	4	Contacts	Contacts page
		Site map	Site map page
Background(s)	Blue/white	Home	Home page

FIGURE 6.16 Storyboard for one page in a website

It also contains detailed information about:

- section name
- filename
- formatting features (e.g. background and foreground colours)
- purpose of the section
- media used in the section (e.g. images, videos)
- navigation to other sections of the solution.

CASE STUDY

An artistic endeavour at Keen College: Solution design

Now that the students had decided that a website was the best solution to the opportunity, they had to set about designing it so it met all the necessary functional requirements. They developed a site map to represent the structure of the website (Figure 6.17) and two layout diagrams to show how the final web pages would look (Figure 6.18 and Figure 6.19).

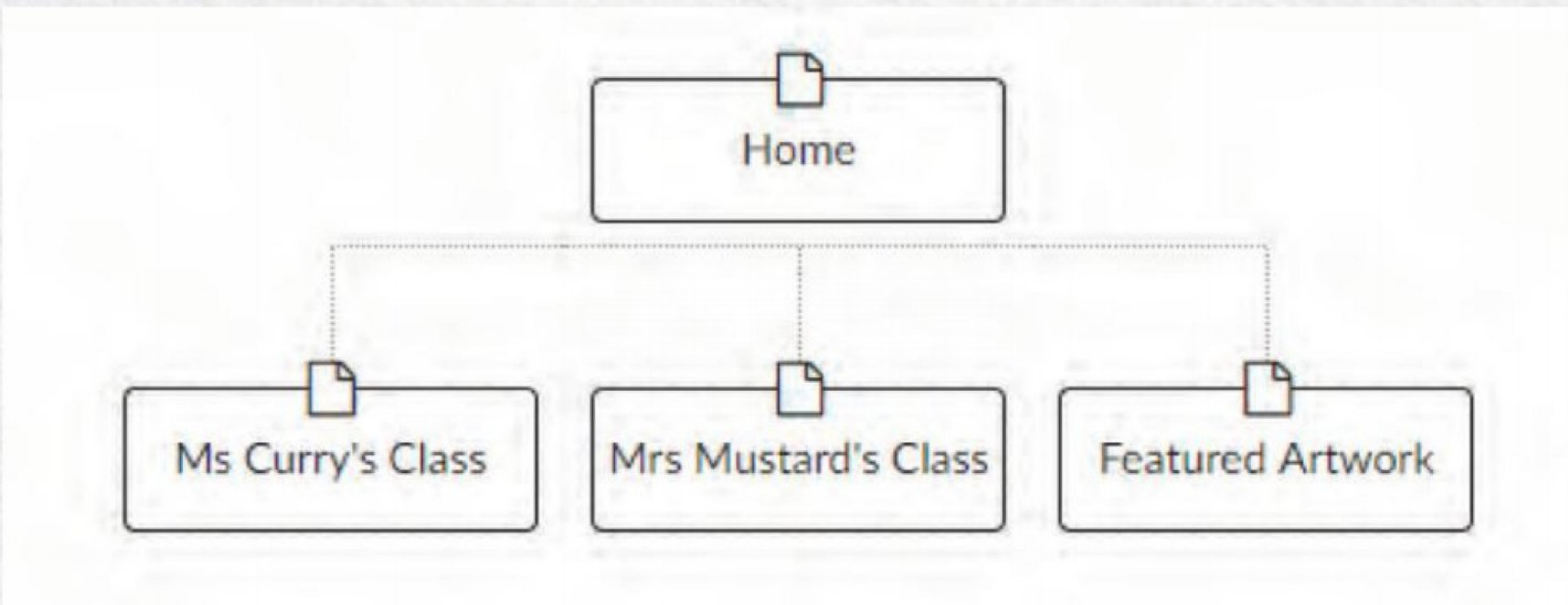


FIGURE 6.17 Site map of chosen web page solution

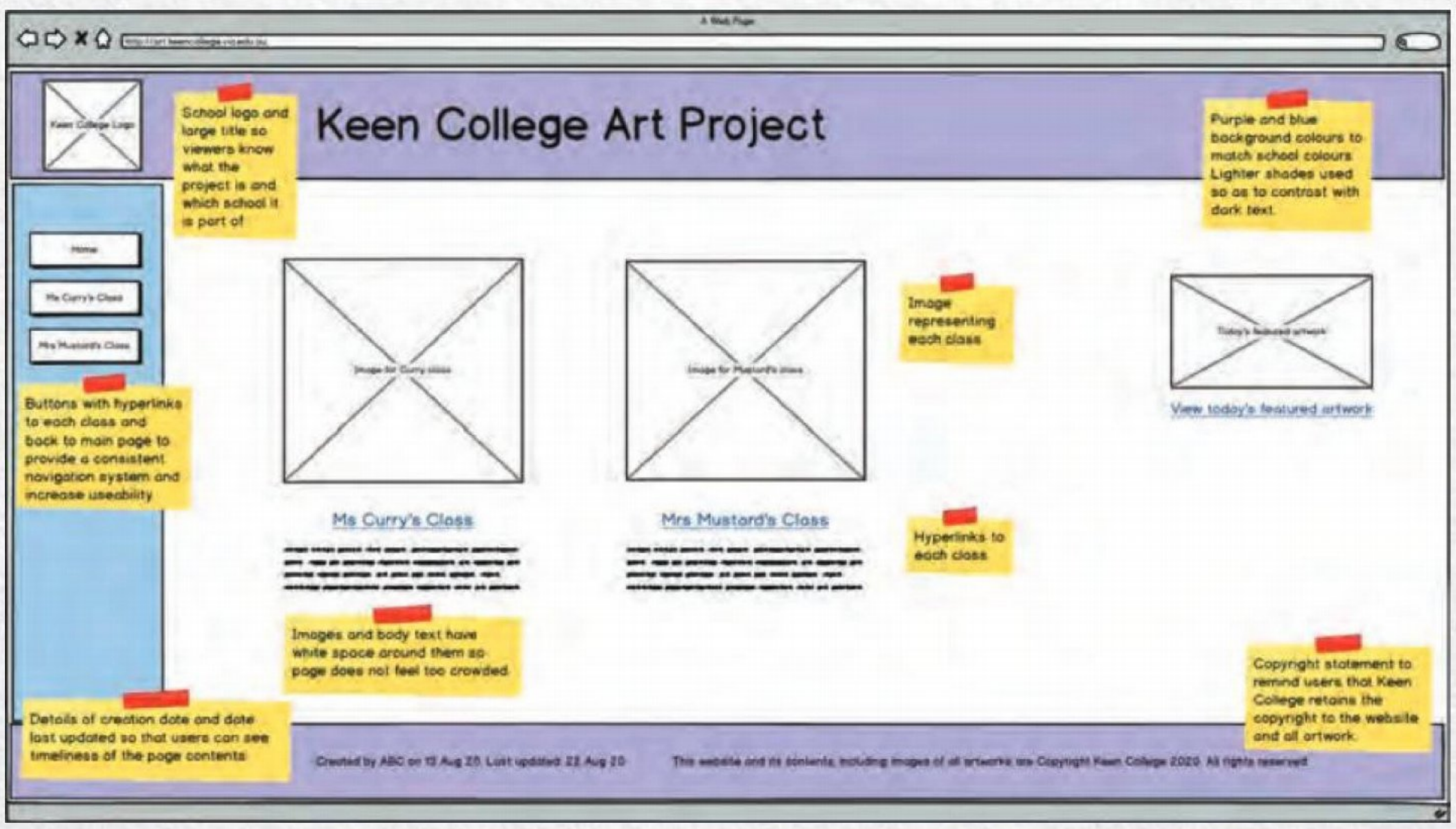


FIGURE 6.18 Layout diagram showing chosen web page solution

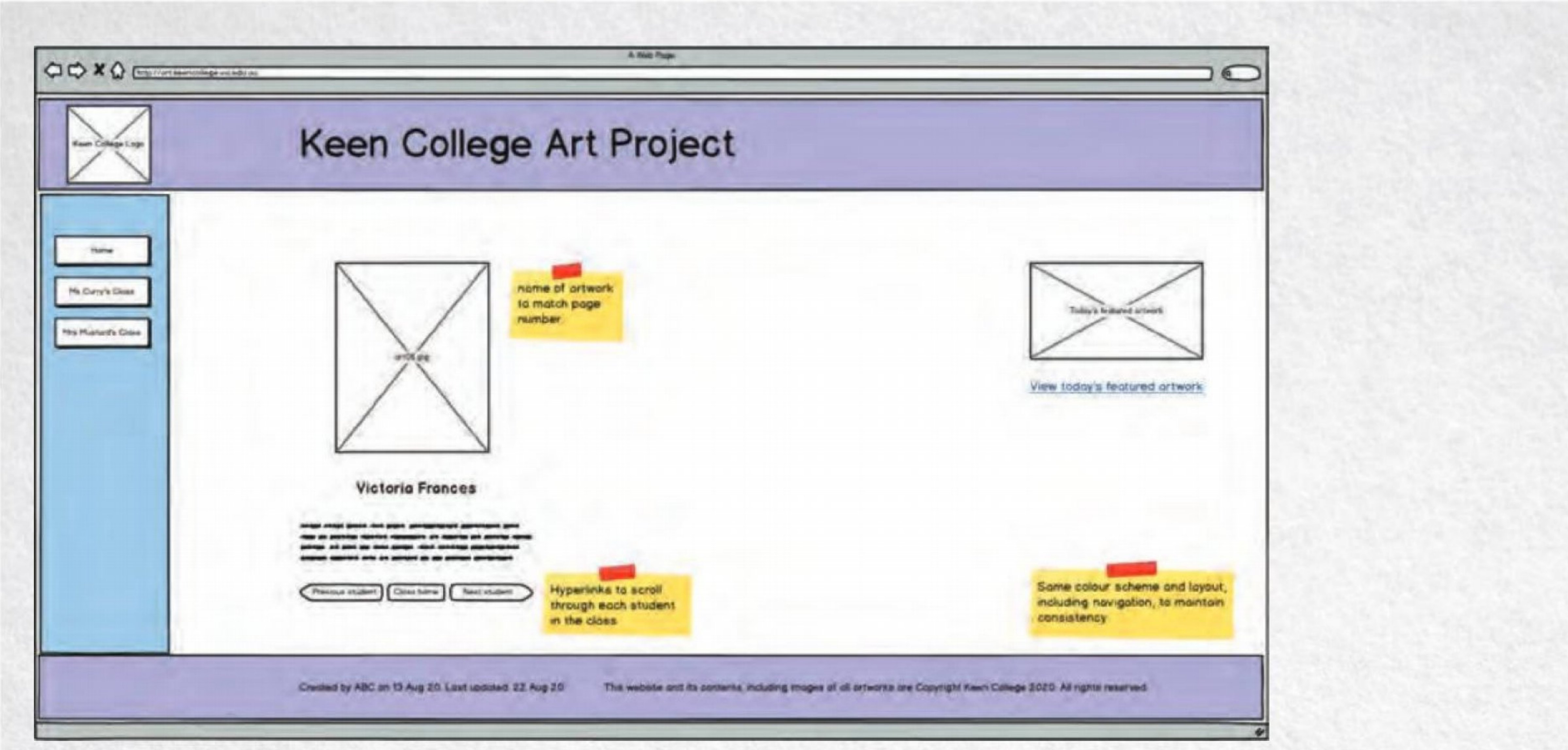


FIGURE 6.19 Layout diagram showing chosen web page solution

Validating and testing the solution

Remember:

- **Validation** checks the reasonableness of data inputs.
- **Testing** checks the accuracy of information outputs.

Validation

If you have designed a solution that requires data input, then you need to ensure that your solution is validated. Validation checks that input data are *reasonable* and *complete*. Validation does not, and cannot, check that inputs are accurate. How, for example, could validation tell whether a person is being honest when entering their age? However, validation can detect problems when a person enters their age as 152 years, or as 'banana', or nothing at all. You can perform validation manually (yourself) or allow software to do it for you.

Computers are particularly good at conducting validation checks.

1. **Existence checks** ensure that a value has been entered and the field is not blank, or <null>.
2. **Type checks** ensure data is of the right type; for example, the age that has been entered is actually a number.
3. **Range checks** ensure that data is within acceptable limits (for example, children enrolling in kindergarten must be 3–6 years old) or comes from a list of acceptable values (for example, small, medium or large).

People can perform manual validation, especially proofreading for sense, clarity, relevance and appropriateness. In addition, unlike spreadsheets, people tend to 'smell a rat' when values entered would pass electronic validation checks but are inaccurate because they are ridiculous.

THINK ABOUT APPLIED COMPUTING 6.12

If the person is expected to enter their age, what would be a reasonable range check?

a 5–50 years
b 15–80 years
c 0–100 years
d 1–200 years

9780170440806

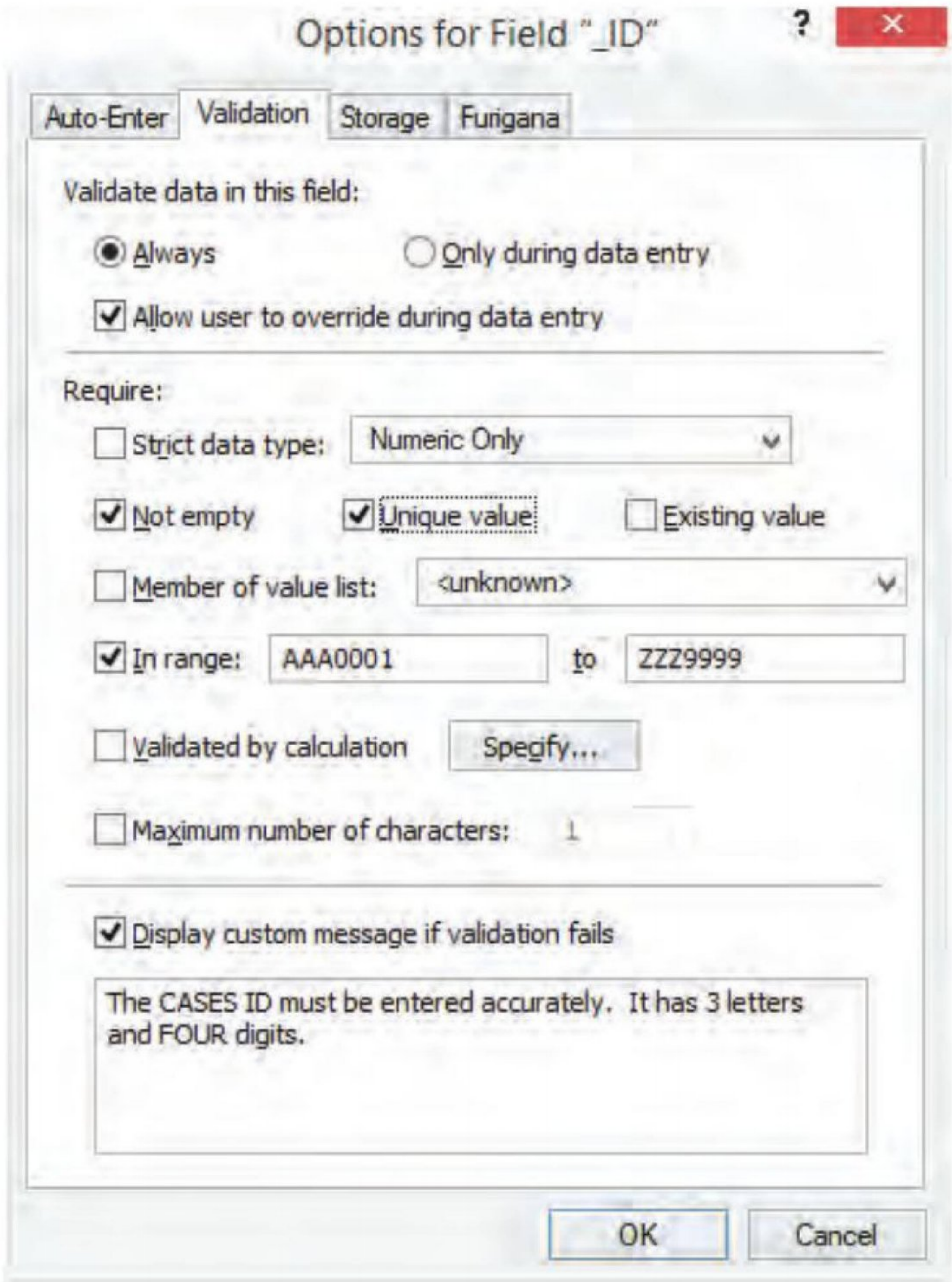


FIGURE 6.20 Validation rules in FileMaker database. Here an ID field is made compulsory ('Not empty') and unique, and within a defined range of values. The database is also told what error message to display if validation fails.

Similarly, Microsoft Word can find and identify words that are not in its dictionary, but it cannot advise a writer that a paragraph is boring or that the previous page was pretentious, misleading and irrelevant.

Testing

After designing and building your solution, you need to demonstrate that it has been thoroughly tested. You need to know what to test in your solution, so we plan the testing as part of the design stage.

If a solution fails, it could annoy or disadvantage users, so thorough and careful testing is necessary, whether the solution is a game, a website shopping cart, or an airliner's autopilot.

If your solution fails because of undiscovered faults, it may become difficult to use, or completely unreadable.

Testing checks that a solution does what it should do. Testing is not easy, quick or cheap – especially for a product such as an operating system, with megabytes of code in thousands of files created by hundreds of people.

The typical steps involved in testing are as follows.

1 Decide which tests will be conducted.
2 Create suitable test data.
3 Determine expected results.
4 Conduct the test.
5 Record the actual results.
6 Correct any errors.

There are many testing types, each intended to uncover different kinds of errors at different times during development. The types of testing relevant to your solution are listed in Table 6.5.

TABLE 6.5 Testing types

Type	What is tested?
Informal (alpha)	The part of the solution that has just been finished
User acceptance (beta)	Typical end users use their own equipment to check that the finished solution is acceptable under different user conditions
Component	A single part of a system works properly by itself (for example, a user entry form applies the correct delivery cost for a given destination)
Integration	Individual parts of a system work together (for example, the embedded Excel file correctly accesses the separate database table)
System	All components in the solution work properly as a single unit
Installation	The form control is installed correctly and working on your system, server or domain
Compatibility	The multimedia plug-in and its components are compatible with a variety of browsers and the main OS Note: ActiveX Form Controls will NOT work on the macOS
Usability	Whether users can operate your graphic solution quickly and simply
Accessibility	Whether users with special needs or disabilities can use your graphic solution

Test plan

A set of test data that will be used during development to ensure that the solution is functioning correctly is prepared in the design stage. The test data should be chosen to test all aspects of the solution, including identification and handling of unreasonable or incomplete data (validation). Once the solution has been shown to be functioning correctly, the test data is removed and the 'real' data relevant to the organisation is added. A test plan (developed during the design stage) is used to show all the functions to be tested, the sample test data and how the function is expected to handle the data. An example of a section from a test plan for a website is shown in Table 6.6. The actual testing process is covered in detail later in this chapter.

CASE STUDY

An artistic endeavour at Keen College: Test plan

TABLE 6.6 Part of the test plan for the Keen College website solution

What is to be tested?	Test data	Expected/Required result	Why?
Link from home page to Ms Curry's subject page menu	Starting page = index.html Link = Ms Curry's class (curryclass.html) Destination page = curryclass.html	When the link 'Ms Curry's class' is selected it should hyperlink to the page called 'curryclass. html'.	Because the link is to 'curryclass.html' and there is a page called 'curryclass.html'
Student artwork images load in the centre of the page	Page = curryclass05.html Image = art05.jpg (300 × 450 pixels) Alignment = 'center'	The image 'art05.jpg' must load in the centre of the page and aligned with the vertical middle of the text. Its height and width should be the full size of the image (300 × 450 pixels).	Because the code links to the 'art05.jpg' file in the images folder, its height is 450 and its width is 300 and its alignment is set to 'center'. This coding is placed before the text caption in the page body.
Screen layout on home page	Resolution = 1920 × 1080 keen_college_logo.jpg Navigation buttons (menu) Main text box	At 1920 × 1080 resolution the Keen College logo and navigation buttons should appear on the left of the screen, separated from the main text by a border. The main text area should fit on the rest of the page without the need for scrolling.	Because the page elements were set to be viewed at 1920 × 1080 resolution
Page weight for index.html less than 200 kB	Reported file size of: index.html = 4 kB keen_college_logo.jpg = 40 kB Navigation buttons = 60kB Main text area = 2 kB	Expected page weight = 4 + 40 + 60 + 2 = 106 kB, which is less than the 200 kB limit	Added up on a calculator
Home page download speed must be less than 2 seconds over NBN 25 Mbps connection	Page weight for index.html = 106 kB Theoretical NBN speed = 25 Mbps (8.3 MBps)	Expected download speed should be less than 2 seconds	Because 106 ÷ 8300 = 0.013 seconds (on calculator), which is less than 2 seconds
User acceptance of functionality	Procedures: Navigate to the home page and locate the link to Ms Curry's class page. Navigate to the page for Ms Curry's class and locate the link to Victoria Frances' work.	Users should be able to perform each of the procedures listed with minimal difficulty.	Interfaces have been designed to allow for easy navigation to these key areas (navigation panel on left of main page and menu on class page).

9780170440806

Media and plug-ins

Each picture, audio clip, video, graph and animation (that is, any non-textual information) must be inspected to see that it is displaying in the right place, at the right time, and at the right speed and volume in a variety of common environments, meaning different browsers and devices.

Hyperlinks

Every internal and external link in the solution needs to be manually clicked and the result noted. Create a list of links and tick off each one as it passes testing.

Links to external services

You should be able to completely test all parts of the solution under your control. You need to test the operation of any external connections to your product, to ensure data updates function as expected.

Readability

Use the checklist provided in Table 6.7 to test readability of your solution.

TABLE 6.7 Readability checklist

Checklist	Tick (✓)
Is the text large enough to read comfortably on a small device?	
Is contrast optimal, or at least satisfactory?	
Is the typeface a readable size?	
Are lines or paragraphs of a good length?	
Is text alignment attractive and readable on the page?	
Are the spelling, punctuation and grammar correct?	
Is the vocabulary appropriate and acceptable (inoffensive)?	
Is expression clear and unambiguous?	
Are headings clear and do they divide content into logical sections?	

Loading times

If the solution is online, clear your browser's cache to remove pre-loaded copies of files and media, and try loading the site via cable and wi-fi. Any page that takes more than a few seconds to load should be inspected and optimised. Another method is to use one of many online services that can measure the loading times for your pages. Online data repositories may have varying access times due to user demand.

Browser compatibility

Does your solution rely on a browser for presentation? Check that plug-ins and installed players and codecs (coder/decoders or compressor/decompressors) can read and display your chosen media. Browsers differ in their ability to interpret different media, and some systems may not have the right technology, such as HTML5, or the necessary plug-ins installed, such as Silverlight. Every piece of media must be checked on the dominant browsers to verify that they appear as expected. Remember that Flash will not play at all on many mobile devices.

You can manually test most site functionality yourself. If your solution is online, however, there are many services that can perform automated cross-browser compatibility checks using many new and previous browser versions.

Accessibility

Does your solution create unnecessary difficulty for users with poor eyesight or muscular control, or weak language skills, or other common disabilities? Is **alt text** applied to images? Are colour combinations considerate of colourblind people? Many colourblind-safe palettes are documented online.

W3C

There are several places online to test the accessibility of your solution. Try the World Wide Web Consortium (W3C) website.

Dynamic features

Every selection option item must be checked and its behaviour documented in a **testing table** (Table 6.8). If data entry forms are expected to work, data should be entered and its successful arrival at its destination should be documented. Any simulated functionality, such as a faked login box, should, as far as is practical, appear to work genuinely. Any coding, such as JavaScript, PHP/MySQL, Java, PERL, macros and Python, should be run using a variety of test data and the behaviour of the code recorded.

Classroom constraints

Your dynamic graphic solution may, because of constraints in classrooms and networks, not have access to updated data online. However, it should have the look, feel and apparent functionality of a real online solution, even though some features may have to be simulated because it is unreasonable to expect them to function under all working conditions.

THINK ABOUT APPLIED COMPUTING 6.13

How do you evaluate a new website, game or social media app? On what measures do you base your opinion?

Evaluation criteria

While **evaluation** is the final stage of the problem-solving methodology, the **evaluation criteria** are developed during the design stage. The purpose of evaluation is to check how well the solution is satisfying the needs of the user for which it was originally created.

Evaluation is not the same as testing; its purpose is distinctly different. By the time evaluation begins, the solution has already been proved to work properly and its functionality is no longer in question.

Evaluation can best be understood by saying what it does not do.

- Evaluation does *not* test that a solution is working properly. That should have been done during testing.
- Evaluation does *not* enter test data to check that output is accurate. That should have been done during testing.
- Evaluation does *not* use a stopwatch to time how long a process takes. That should have been done during testing.
- Evaluation does *not* perform checks with immediate results, such as pulling out the power plug to see if a system loses data. That should have been done during testing.

Evaluation looks at a solution's performance *over time* in terms of the **evaluation criteria**.

What to evaluate

Evaluation criteria are determined during the design phase of the problem-solving methodology and are based on the most important qualities that the solution is expected to have when it is designed.

Evaluation criteria fall under two headings: efficiency and effectiveness.

1 **Efficiency** can be measured in terms of speed or productivity (work produced in a given time), profitability (income generated versus running costs) and labour requirements (how much labour is required to achieve its productivity levels).

2 **Effectiveness** includes completeness, readability, attractiveness, clarity, accuracy, accessibility, timeliness, communication of message, relevance and usability.

CASE STUDY

An artistic endeavour at Keen College: Evaluation criteria

The Keen College AC students have created the following evaluation criteria for the planned website solution. They plan to perform the evaluation during the school holidays so they can respond to feedback when they get back.

TABLE 6.8 Evaluation criteria

Criterion	All student artwork can be viewed	Artwork images are downloaded within an acceptable timeframe	Website is easy to use to find student artwork
Time frame	During school holidays	During school holidays	During school holidays
Data required	Web page hit counter	Teacher, student and parent opinions	Teacher, student and parent opinions
Data-collection method	Website log files	Online survey	Online survey
Assessment	Check website logs to see if the artwork of each student has been accessed at least once	Ask teachers, parents and students if the downloading of artwork images occurs within an acceptable timeframe	Ask teachers, parents and students to rate the usability of finding student artwork out of 10
Judgement	If the logs show that all student artwork has been accessed at least once, then the criterion is achieved	If 95% of feedback states that the images are being received within an acceptable time frame, then the criterion is being achieved	If the average rating is 8 or better, then the criterion is being achieved

Developing your solution

Unit 2, Outcome 1, requires you to create an innovative solution in the form of a proof of concept, prototype or product. Here is the difference between them.

Proof of concept

A **proof of concept** involves demonstrating whether an idea or concept is feasible, possible or viable. It is used to show, even in theory, whether a solution has the capability and the right characteristics to meet the need or opportunity identified.

Steps that may be involved in creating a proof of concept include:

- explaining the emerging technology being used
- discussing any issues (e.g. economic, legal or ethical) surrounding the technology
- identifying the need or opportunity
- identifying the solution requirements, constraints and scope of the solution
- generating solution designs and creating evaluation criteria
- identifying the inputs and output required
- identifying the processing required
- identifying the hardware and software required
- identifying any validation and testing needed
- explaining how the solution will achieve the evaluation criteria.

By following these steps, although there will not be a working solution, the intention is to provide enough evidence to show that the solution is possible in meeting the need or opportunity.

Many vehicle manufacturers create prototypes of new vehicles before they start full production to make sure that clients would be happy with the end product. If there are any client 'pain points' with the prototype, they can be addressed before the vehicle goes into mass production.

Prototype

A **prototype** is a model or a solution with limited functionality. The purpose of a prototype is to display the look and feel of the completed solution but without full functionality. It also allows a solution to be created, so users can give feedback and so changes can be made without wasting all of the time, effort and money that would be required to develop a completed solution (Figure 6.21).

A prototype differs from a proof of concept in that a prototype is built using the same software that will be used for the completed solution, whereas a proof of concept only provides evidence that a solution is feasible and may not involve the use of the solution software at all.

THINK ABOUT APPLIED COMPUTING 6.14

Find out two other industries that use prototypes and why they use them.

Product

A **product** is a completed solution with full functionality. All capabilities and features of the software identified during the analysis and design stages are included in the solution, all appropriate data is validated and all functions are fully tested.

A product differs from a prototype in that all the functions of the solution are working as expected, whereas in a prototype only a limited number of functions are present.

CASE STUDY

An artistic endeavour at Keen College: The product

The Keen College AC students have decided to create a functional website as their solution. This is a product that will incorporate all elements identified in the scope. While creating the product, students will prototype it to ensure that the basic structure of the template is sound and that the navigation is usable.

9780170440806

Alamy Stock Photo/Dmitry Orlov

Alamy Stock Photo/Clarence Holmes Photography

FIGURE 6.21 Prototype (top) of a Tesla Model S, and the finished product (bottom)

Testing the solution

While a solution is under development, it will undergo extensive testing to ensure that it functions as expected. Much of the testing may be informal, but for the client a set of formal testing documentation needs to be created. You may make use of a testing table to summarise certain tests, and annotated screenshots to demonstrate the results of other tests.

Testing table

A testing table is a commonly used way to record evidence of functionality testing. A testing table for a digital game might look like this (Table 6.9).

TABLE 6.9 A testing table

What was tested	How it was tested	Expected result	Actual result	How it was fixed, if relevant
Ability to access apps on the watch by spinning a dial	Spun the dial on the watch to see if it cycled through apps	Apps will cycle one by one	✓	N/A
Pressing on an app icon on the watch face will open the app	Open each app by pressing on the app icon	Each icon will open the relevant app	Apps opened but some apps crashed	Not yet fixed
Readability	Asked five volunteers to read the text on the watch face and report on text size, contrast, alignment	Reports that text was easy to read	One volunteer suggested that italic fonts were hard to read	Removed all italic fonts from the watch system
Loading times for apps	Performed a speed test on loading of each app by recording the load time	All apps passing the speed test	All apps passed the speed test	N/A
Watch tracking number of steps taken by wearer	Asked a volunteer to wear the watch and take 100 steps	Watch should record 100 steps	Watch recorded 100 steps	N/A

How to document your testing

- Use a testing table (such as the one shown in Table 6.9).
- Seek a subjective report from a fellow student who tried out your solution's readability and usability.
- Capture screenshots of features that are not normally visible, such as drop-down menus and warning messages, showing that they work when needed.
- Make handwritten calculations, annotating printouts of screenshots of your solution's calculations to verify that the output has been checked for accuracy.
- Capture screenshots of the solution's validation rules responding properly to invalid data.

CASE STUDY

An artistic endeavour at Keen College: Validating and testing the solution

The content for the website is validated by the AC students. They check that there is a blurb and an image for every student. Then they manually proofread the text, looking to see that it makes sense and to correct any obvious spelling errors. Students also use an electronic spelling and grammar checker to assist with this process. If they are unsure of the spelling of a name, or of the description of an artwork, they ask the relevant student in order to clarify what the text is meant to be.

Once the website structure is complete and the images and text are placed in relevant folders, several students begin the task of formally testing the product. Using the test plan created in the design stage, the students check for such things as links working properly and that the correct images are loading on each page. They make use of both a testing table and annotated screenshots to test the website thoroughly.

TABLE 6.10 Keen College testing table

What is to be tested?	How it was tested/Test data	Expected/ Required result	Actual result	How it was fixed, if relevant
Link from homepage to Ms Curry's subject page menu	Starting page = index.html Link = Ms Curry's class (curryclass.html) Destination page = curryclass.html	When the link 'Ms Curry's class' is selected, it should hyperlink to the page called 'curryclass.html'	The link was broken and went to curryclas.html	Changed the link to be curryclass.html
Student artwork images load in the centre of the page	Page = curryclass05.html Image = art05.jpg (300 × 450 pixels) Alignment = 'center'	The image 'art05.jpg' must load in the centre of the page and aligned with the vertical middle of the text. Its height and width should be the full size of the image (300 × 450 pixels)	The image loaded in the centre of the page and was aligned with the vertical middle of the text. The height and width were correct.	N/A
Screen layout on home page	Resolution = 1920 × 1080 keen_college_logo.jpg Navigation buttons (menu) Main text box	At 1920 × 1080 resolution the Keen College logo and navigation buttons should appear on the left of the screen, separated from the main text by a border. The main text area should fit on the rest of the page without the need for scrolling.	The logo and navigation appeared on the left, separated by a border. The main text area did not fit on the rest of the page – it needed scrolling.	Reformatted the main text so that scrolling no longer occurred

TABLE 6.10 Keen College testing table (continued)

What is to be tested?	How it was tested/Test data	Expected/ Required result	Actual result	How it was fixed, if relevant
Page weight for index.html less than 200 kB	Reported file size of: index.html = 4 kB keen_college_logo.jpg = 40 kB Navigation buttons = 60 kB Main text area = 2 kB	Expected page weight = 4 + 40 + 60 + 2 = 106 kB, which is less than the 200 kB limit	Actual weight was 106 kB, less than the 200 kB limit	N/A
Home page download speed must be less than 2 seconds over NBN 25 Mbps connection	Page weight for index.html = 106 kB Theoretical NBN speed = 25 Mbps (8.3 MBps)	Expected download speed should be less than 2 seconds	Download speed was almost instant, much less than 2 seconds	N/A
User acceptance of functionality	Procedures: Navigate to the Home page and locate the link to Ms Curry's class page. Navigate to the page for Ms Curry's class and locate the link to Victoria Frances' work.	Users should be able to perform each of the procedures listed with minimal difficulty	Navigation to Victoria Frances' work was quick and easy	N/A

Evaluating the solution

During the design stage you create the evaluation criteria. This helps to guide the design and development of the solution. Once the solution has been finalised and installed, the formal evaluation process can be conducted. The evaluation might take place 3–6 months after the solution is implemented so that people have time to learn the new system and to give time for any minor bugs to be dealt with.

Evaluation methods

For each evaluation criterion, there must be a corresponding evaluation method that can measure the degree to which the criterion has been achieved.

- **Objective** (fact-based, measurable) results are solid facts that are hard to argue with. Measure whenever you can.
- **Subjective** results (emotions, opinions, personal judgements) can be gained from interviews, questionnaires and surveys. These should only be used when objective measurement is not possible or practical.

TABLE 6.11 Typical evaluation methods

Criterion	Method
Accuracy (effectiveness)	Check the complaints log and count the complaints from staff or customers about inaccurate information received over three months from the system.
Reliability (effectiveness)	Count the number of faults in the system's error log.
Security (effectiveness)	Count the number of successful and thwarted attempts made to penetrate system security.
Attractiveness, pleasure, comfort, confidence (effectiveness)	Interview users.
Productivity (efficiency)	Refer to system logs and count how many transactions the system handled over three months compared to the previous system.
Profitability (efficiency)	Ask the accountants to tally the new system's running costs over time. Check organisational profit figures and see if profit has increased.
Labour requirements (efficiency)	Count the number of staff hours spent operating and maintaining the system compared with the previous system.
Ease of use, usability (effectiveness)	Count the number of times online help was used (indicating that the solution may not have been intuitive). Add up how many errors were made by users. (A solution that is hard to use tends to cause users to make mistakes.) Check the help desk records to see how often users asked for help or complained about the solution. Give users a questionnaire asking about their feelings regarding the system's usability.

Remember: evaluation assesses your solution's performance over time. It is not instantaneous in the same way that testing is. Any emotional or judgemental feedback is only gathered on appropriate criteria. For example, it is pointless to ask interviewees questions such as, 'Is the new system faster than the old one?' Even if you received an answer to this question, you would not be able to trust its accuracy.

When to evaluate

Evaluation occurs after the solution has been in regular use for some time, so the solution is well 'bedded in' and its users are familiar and comfortable with it. A few months of regular, daily use is typical.

Evaluating a solution too soon can lead to negative feedback if users are not yet used to it and are slow and prone to making errors. Later, when they are comfortable and skilled with the solution, their feedback may be much more positive.

In cases when a system is used infrequently, but its success is critical to the organisation (such as creating school reports), evaluation may be done immediately after the system is used.

6.15 THINK ABOUT APPLIED COMPUTING

Why do software developers wait a period of time after the product is released before completing an evaluation?

CASE STUDY

An artistic endeavour at Keen College: Evaluation

Once the website solution has been completed and put onto the Keen College web server, it is made available to parents, students and teachers. The students allow all users time to learn to use and navigate the website during the school holidays, before conducting an evaluation. In their evaluation strategy the students have decided to survey all user groups to get feedback on the website's usability. The questions are based around the evaluation criteria created during the design stage. As a result of the evaluation process, the students identify a few small changes to make to the website, but overall it appears to be meeting the requirements of the users.

TABLE 6.12 Keen College evaluation results

Criterion	Time frame	Data required	Data-collection method	Assessment	Judgement
All student artwork can be viewed	During school holidays	Web page hit counter	Website log files	Check website logs to see if the artwork of each student has been accessed at least once.	The logs showed that every student page had been accessed at least once. This criterion was achieved.
Artwork images are downloaded within an acceptable timeframe	During school holidays	Teacher, student and parent opinions	Online survey	Ask teachers, parents and students if downloading artwork images occurs within an acceptable time frame.	96% of feedback indicated that downloading artwork was performed in an acceptable time frame. This criterion was achieved.
Website is easy to use to find student artwork	During school holidays	Teacher, student and parent opinions	Online survey	Ask teachers, parents and students to rate the usability of finding student artwork out of 10.	The average rating for ease of use was 9. This criterion was achieved.

6 CHAPTER SUMMARY

Essential terms

Act of Parliament a document that contains a set of conditions and rules about a particular area or topic

adjustments changes made to the project plan as a result of events that may have occurred as the project progressed

alt text text descriptions added to images on a web page, which describe the image for users who may not be able to access the image; 'alt' is an abbreviation of 'alternative'

annotations comments or notes placed on a Gantt chart to explain any changes that were made

brainstorming a technique to generate ideas to solve a problem

copyright gives the owner of an original work (e.g. software application, movie, song or images) the right to control who can use the original work, and how

Copyright Act 1968 an Australian federal law containing conditions and rules related to copyright

dependencies relationships between tasks, indicating when a task should begin or end in relation to another task or group of tasks

economic constraints factors that may limit or restrict the solution requirements and that are to do with money, including time and cost

effectiveness how well a solution meets the needs of its users, measured in terms of completeness, readability, attractiveness, clarity, functionality, accuracy, accessibility, timeliness, report formats, relevance, usability and communication of message

efficiency a measure of how much time, cost and effort is required to achieve the intended results

ethics accepted moral standards that guide behaviour; these standards may be common across a particular society or specific to a single organisation, and they apply to questionable activities over and above any legal requirements; ethics often provide us with a set of guidelines for appropriate behaviour

ethical issue a situation that arises when current behaviours or practices could be considered to be the wrong thing to do

evaluation the use of criteria created during the design stage to evaluate or judge whether the solution is meeting the needs of the user; the final stage of the problem-solving process

evaluation criteria measures that are used to assess or judge whether a solution is meeting the user's needs and/or requirements

existence checks checks carried out to ensure that a value has been entered and the field is not blank, or <null>

federal parliament a body of elected representatives responsible for creating legislation that applies to all individuals and organisations within Australia

functional requirements requirements of a digital solution that are directly related to what the solution will do

Gantt chart a project management tool that displays project schedule information by listing project tasks and their corresponding start and finish dates in a calendar format

Health Records Act 2001 a Victorian state law containing conditions and rules related to how and when an individual's personal health data and information can be collected by both Victorian government agencies and private organisations that either offer health services or handle health records

innovative solution a unique, original and previously unknown way to solve a problem

interoperability the ability of a device to connect and interact with other devices, particularly devices belonging to different information systems

9780170440806

interview a conversation between two or more people where one person (the interviewer) asks questions and the others (the interviewees) give their answers to the question

legislation a body of laws, created by governments, that individuals and organisations resident in the government's jurisdiction must follow

local councils elected bodies that can create by-laws, which apply within each individual local council area

mind mapping a technique for quickly generating and linking ideas

need something that it is essential to the solution so that the user may complete the task

non-functional requirements characteristics that the user or client would like the solution to have; often tied to solution constraints

non-technical constraints factors that may limit or restrict solution development, specifically related to legal, social, usability or economic factors

objective not influenced by personal feelings or prejudice; unbiased

opportunity something that occurs when circumstances or factors are present that may allow for a new solution, which does not already exist, or which people have not yet thought of

Privacy Act 1988 an Australian law that regulates the handling of personal information about individuals

Privacy and Data Protection Act 2014 a Victorian law that regulates the use, protection and storage of data and information in Victoria

proof of concept the process of demonstrating whether an idea or concept is feasible, possible or viable

product a completed solution with full functionality

project a series of interrelated tasks that need to be completed to be able to create a solution to a need or opportunity

project log similar to a diary in which events that affect each task in a project are written down to create a history of the project

prototype a model or solution with limited functionality

pseudocode a design tool combining the structure of programming language code with instructions written in plain English, used to represent the logic or steps of processing that occur when a solution is run or executed

range checks checks to ensure that data is within acceptable limits

scope of the solution boundaries or parameters outlining what the solution will contain

site map a way of displaying how information has been structured within a solution and how a user can navigate to find the required information

solution a way to solve a problem

solution constraints factors that may limit or restrict the solution requirements, both functional and non-functional

solution requirements what the client needs from the solution

state parliaments bodies of elected representatives responsible for creating legislation that applies to all individuals and organisations within each particular state

storyboard a tool used to show the features of an individual section of a solution

subjective based on feelings and emotions, opinions and tastes

survey a series of predetermined questions that can be sent to a participant to answer

technical constraints factors that may limit or restrict the solution requirements and that are related to the hardware and software available for the project

testing the process of checking that the innovative solution is working as expected

testing table commonly used way to record evidence of functionality testing

type checks checks to ensure that data is of the right type

validation the process of inspecting data that is being input into a solution to check if the data is reasonable

9780170440806

Important facts

1 **Legislation** is a body of laws, created by governments, that individuals and organisations resident in the government's jurisdiction must follow.

2 **Federal parliament** is responsible for creating legislation that applies to all individuals and organisations **within Australia**. **State parliaments** are responsible for creating legislation that applies to all individuals and organisations **within each particular state** such as New South Wales and Victoria, while **local councils** create by-laws that apply **within each individual local council area**.

3 The *Privacy Act 1988* is an Australian **federal law**. It contains conditions and rules related to how and when personal data and information can be collected by Australian Government agencies and private organisations that turn over more than $3 million annually.

4 The *Privacy and Data Protection Act 2014* is a **Victorian state law**. It contains conditions and rules related to how and when personal data and information can be collected by Victorian government agencies and private organisations that conduct contract work on behalf of the Victorian Government.

5 The *Health Records Act 2001* is a **Victorian state law** containing conditions and rules related to how and when an individual's personal health data and information can be collected by both Victorian government agencies and private organisations that either offer health services or handle health records.

6 The *Copyright Act 1968* is an **Australian federal law** containing conditions and rules related to copyright. **Copyright** gives the owner of an original work (e.g. software application, movie, song or images) the right to control who can use the original work, and how it can be used.

7 **Ethics** are beliefs, principles or standards that individuals, organisations and society have regarding acceptable behaviour.

8 Techniques used to **collect data** include **observation**, **interviews**, **questionnaires** and **surveys**.

9 **Solution requirements** are what the client needs from the solution. **Functional requirements** are directly related to what the solution will do. **Non-functional requirements** are other requirements that the user or client would like the solution to have but that do not affect what the solution does.

10 **Solution constraints** are factors that may limit or restrict the solution requirements, both functional and non-functional. **Economic constraints** include time and cost. **Technical constraints** are constraints related to the hardware and software available for the project. **Non-technical constraints** relate to areas other than hardware and software.

11 The **scope of the solution** outlines the boundaries or parameters of the solution so all stakeholders are aware of exactly what the solution will contain and not contain.

12 A **solution** involves formulating a way to solve a problem. A **need** is something that is essential for a user to complete a task. An **opportunity** occurs when circumstances or factors are present that may allow for a new solution, that does not already exist, or that people have not even thought of yet.

13 To qualify as an **innovative solution**, a solution must be unique and developed independently.

14 There are a variety of techniques available to generate solution ideas such as brainstorming and mind mapping.

15 A **successful problem-solver** will consider current functional and non-functional requirements and relevant constraints to develop an imaginative range of options from which the best solution can be chosen and developed into a detailed design.

16 Types of **design tools** include (but are not limited to):

- layout and mock-up diagrams (see Chapter 2)
- pseudocode
- sitemaps
- storyboards.

17 **Validation** checks the **reasonableness** of data inputs.

18 **Testing** checks the **accuracy** of information outputs.

19 A **proof of concept** involves demonstrating whether an idea or concept is feasible, possible or viable.

9780170440806

20 A **prototype** is a model or a solution with limited functionality.

21 A **product** is a completed solution with full functionality.

22 **Evaluation** is the final stage of the problem-solving methodology. It checks how well the solution is satisfying the needs of the user for which it was originally created. Evaluation occurs after the solution has been implemented.

23 **Efficiency** can be measured in terms of speed or productivity (work produced in a given time), profitability (income generated versus running costs) and labour requirements (how much labour is required to achieve its productivity levels).

24 **Effectiveness** includes completeness, readability, attractiveness, clarity, accuracy, accessibility, timeliness, communication of message, relevance and usability.

25 A **Gantt chart** provides a standard format for displaying project schedule information by listing project tasks and their corresponding start and finish dates in a calendar format.

26 A **project log** can be used to document the development of the solution.

9780170440806

TEST YOUR KNOWLEDGE

Review quiz

Key legislation and emerging technologies

1 Differentiate between federal, state and local laws.
2 Explain the relationship between an Act of Parliament and law.
3 List four Acts of Parliament that have an effect on emerging technologies.
4 Explain how each Act of Parliament you listed affects two chosen emerging technologies.
5 What is meant by an ethical issue?
6 Discuss one ethical issue related to the Internet of Things (IoT).
7 Explain how 3D printing is having an impact on the environment.

Collecting data

8 Explain why it is important to use a number of data-collection techniques when determining a user's needs.
9 What is the analysis stage of the problem-solving methodology about?
10 Explain the type of data that can be collected via both interview and survey.

Solution specifications

11 Explain how functional requirements differ from non-functional requirements.
12 Explain how two different types of constraints may affect solution requirements.
13 Discuss why it is important to define the scope of a solution.

Characteristics of solutions

14 Define an 'innovative solution'.
15 Explain how a need differs from an opportunity.
16 List the four steps in devising an innovative solution.
17 List three techniques used to generate creative ideas. Give an example of one of these techniques.
18 Explain a technique to select the best solution from a list of three.

Representing solution designs

19 List the two activities involved in the design stage of the problem-solving methodology.
20 Explain the purpose of pseudocode.
21 What is the difference between a site map and a storyboard?
22 Explain how a storyboard can be used to represent both the appearance and the functionality of a solution.

9780170440806

TEST YOUR KNOWLEDGE

Functions and techniques for developing innovative solutions

23 Describe what occurs in the development stage of the problem-solving methodology.

24 Explain how a proof of concept differs from a prototype.

25 Explain how a prototype differs from a product.

Validating and testing solutions

26 Explain how validation differs from testing.

27 List the four column headings of a testing table.

Developing your solution

28 What are the three alternatives for producing your solution for Unit 2, Outcome 1? What are the differences between them?

Evaluating the solution

29 Describe the purpose of evaluation criteria.

30 List the tasks involved in an evaluation strategy.

Coordinating and monitoring projects

31 List four elements of a project.

32 Describe the features found on a Gantt chart.

33 Explain the purpose of a project log.

34 Describe an advantage of annotations and adjustments of project plans.

9780170440806

APPLY YOUR KNOWLEDGE

A local cafe has approached you to come up with a solution to its marketing problem. Even though it has a large sign on the pavement outside the store, there is not much through traffic to generate the amount of business that it needs to stay afloat. The owner of the cafe wants something that will appeal to the younger generation, to 'put the cafe on the map' and to draw customers in. He has a budget of $10 000 and a time frame of 6 months before he will have to close his doors due to lack of trade.

1 Explain the techniques you would use to collect data.
2 Use the problem-solving methodology to analyse the cafe owner's needs and requirements.
3 Identify the opportunity.
4 Specify the functional and non-functional requirements.
5 What are the constraints?
6 Use the solution specifications and constraints to determine the scope of the project.
7 Present the cafe owner with three possible innovative solutions that you could go on to design.

9780170440806

PREPARING FOR

Unit 2

OUTCOME 1

Students are to work collaboratively to analyse, design, develop and evaluate an innovative solution to an identified need or opportunity involving a digital system.

Steps to follow

1. Create groups within the class.
2. Discuss a range of needs and opportunities and how each could be solved using emerging technology.
3. Select a topic.
4. Complete an analysis by identifying the solution specifications: functional and non-functional requirements, constraints and scope.
5. Complete the design of the solution by creating design tools that show the appearance and/or functionality of the proposed solution.
6. Complete the development of the solution by creating the solution using technology, including validation and testing.
7. Complete the evaluation of the solution by creating evaluation criteria and then collecting data that can be used to measure whether the criteria were achieved.

Documents required for assessment

Your teacher will advise you how to submit each section of the project. Elements that will require submission:

- Project plan
- Solution specifications
- Design tools
- Evidence of the completed solution
- Evaluation report

9780170440806

CHAPTER

7 Network security

KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

Digital systems

- applications and capabilities of LANs, Wide Area Networks (WANs) and Wireless Personal Area Networks (WPANs)
- functions and characteristics of key hardware and software components of networks required for communicating and storing data and information
- strengths and limitations of wired, wireless and mobile communications technology, measured in terms of cost, data storage options, data transfer rate, reliability and security
- technical underpinnings of intranets, the internet and virtual private networks
- design tools for representing the appearance of networks
- security threats to data and information, such as improper credential management, malicious software, outdated versions of software and weak passwords
- technical underpinnings of malware that can intentionally threaten the security of networks, such as denial of service attacks on websites, spyware, viruses and worms
- data and network protection strategies, such as authentication techniques and symmetric and asymmetric encryption methods
- preventative practices to reduce risks to networks, such as application of firmware, disaster recovery plans, operating system updates, software malware updates and staff procedures
- technical underpinnings of intrusion detection systems (IDS) and intrusion prevention systems (IPS)
- the role of ethical hacking

Interactions and impacts

- risks and benefits of using networks in a global environment
- key legislation that affects how organisations control the storage and communication of data and information: the *Health Records Act 2001*, the *Privacy Act 1988* and the *Privacy and Data Protection Act 2014*
- ethical issues arising from data and information security practices

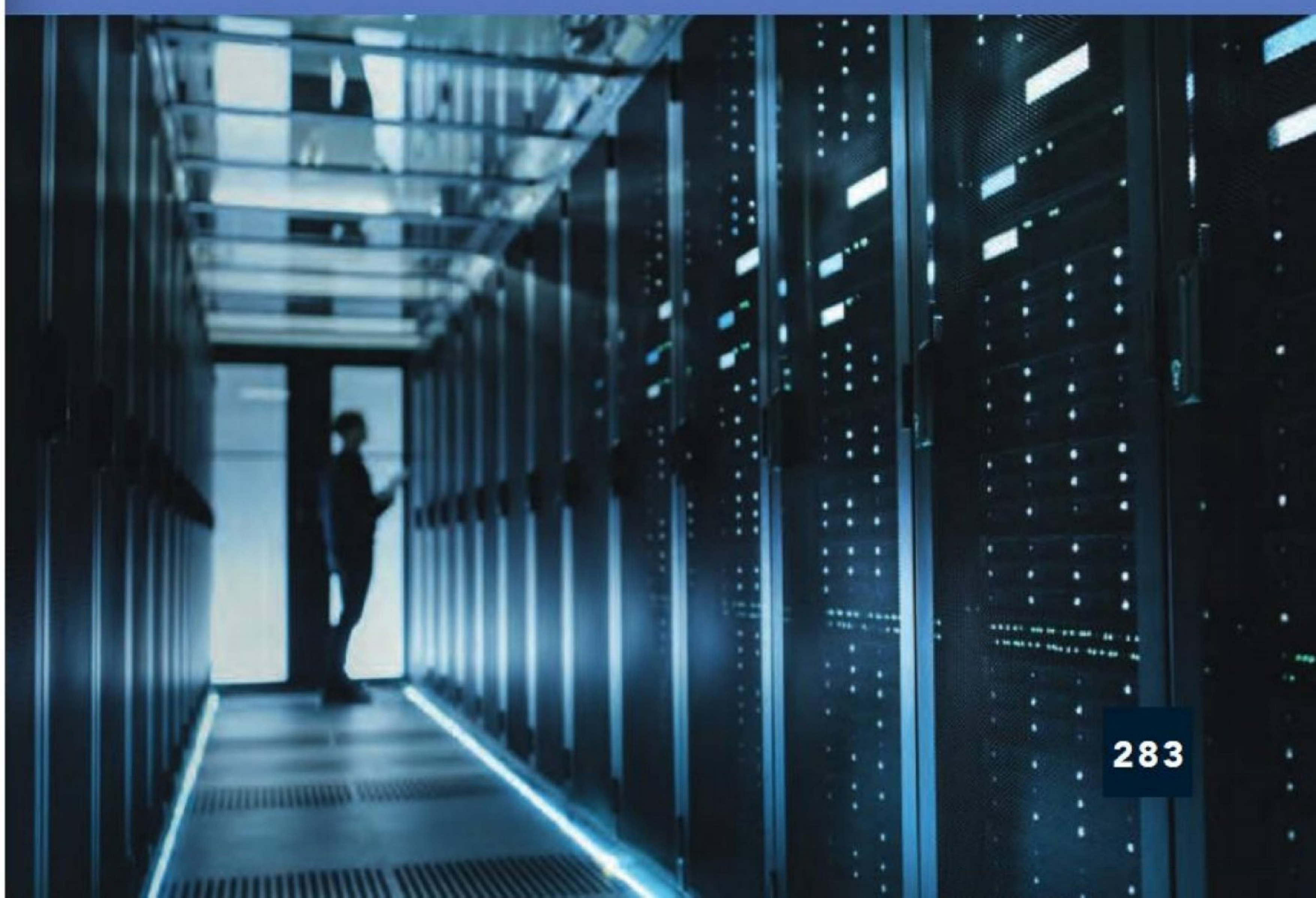
Reproduced from the VCE Applied Computing Study Design (2020–2023) © VCAA; used with permission.

FOR THE STUDENT

This chapter provides an overview of networks, with an emphasis on network security in a global environment. An explanation of networks, key network hardware and software components and communication methods is given. A range of security threats to data and information are discussed, in addition to a range of security measures that can be used to protect data and information from these security threats. The interactions and impacts created by networks are also discussed, with a focus on legal and ethical considerations required by organisations that operate in a global environment. For this outcome you will be required to design a Local Area Network (LAN), describe its components and methods used to communicate data and information. A range of security threats and strategies to protect data and information from these threats also need investigation.

FOR THE TEACHER

For this outcome students are required to describe the applications and capabilities of different networks and examine the impact of common network vulnerabilities. They are then required to design a network solution, including the identification of security threats, and to propose strategies to protect the security of the data and information. In addition, they need to discuss any possible legal or ethical issues that may arise from ineffective security practices. The content of this chapter will provide students with an understanding of the key knowledge required to complete the outcome.



Shutterstock.com/Gorodenkoff

9780170440806

For Unit 2, Outcome 2, you are required to describe the functions and capabilities of different types of networks.

Networks

A **standalone device** is a self-contained device or system in that it can perform its function without the need of another device, computer or connection. The device does not communicate with any other device.

THINK ABOUT APPLIED COMPUTING 7.1

List two advantages and two disadvantages of a standalone device.

FIGURE 7.1 A standalone fax machine; it does not need to be connected to a computer or network to perform its function.

To check the download speed of your home internet connection, visit the speed test weblink and run the speed test.

A **network** is a collection of computers and devices connected by communications channels that facilitates communications among users and allows users to share resources with one another. Examples of such resources are data, information, hardware and software. Networks can be internal to an organisation, or may cover the whole world by connecting to the internet.

Networks exist for sharing information, such as spreadsheet files, database records, email – indeed, anything that helps someone get their job done. The ability to share resources, such as servers, printers and software, also makes a network valuable.

As shown in Figure 7.2, for successful communications, a network needs:

- a sending device, such as a notebook computer, which initiates an instruction to transmit data, instructions or information
- a communications device, such as a wireless adaptor inside a notebook computer, to forward packets of data, instructions or information from a sending device via signals carried by a communications channel
- a communications channel or transmission media, such as a cable or radio waves, through which the digital signals travel
- a communications device, such as a wireless router, that receives the signals from the communications channel and forwards the packets to the receiving device
- a receiving device, such as a printer, which accepts the data, instructions or information.

Sending devices can usually also serve as receiving devices. Examples include notebook computers, personal computers, tablets and mobile phones. Such devices usually have a built-in communications device.

The primary function of a communications device such as a broadband router is to transmit data, instructions and information between a sending and a receiving device.

Data, instructions and information travel along a communications channel in the form of a digital signal.

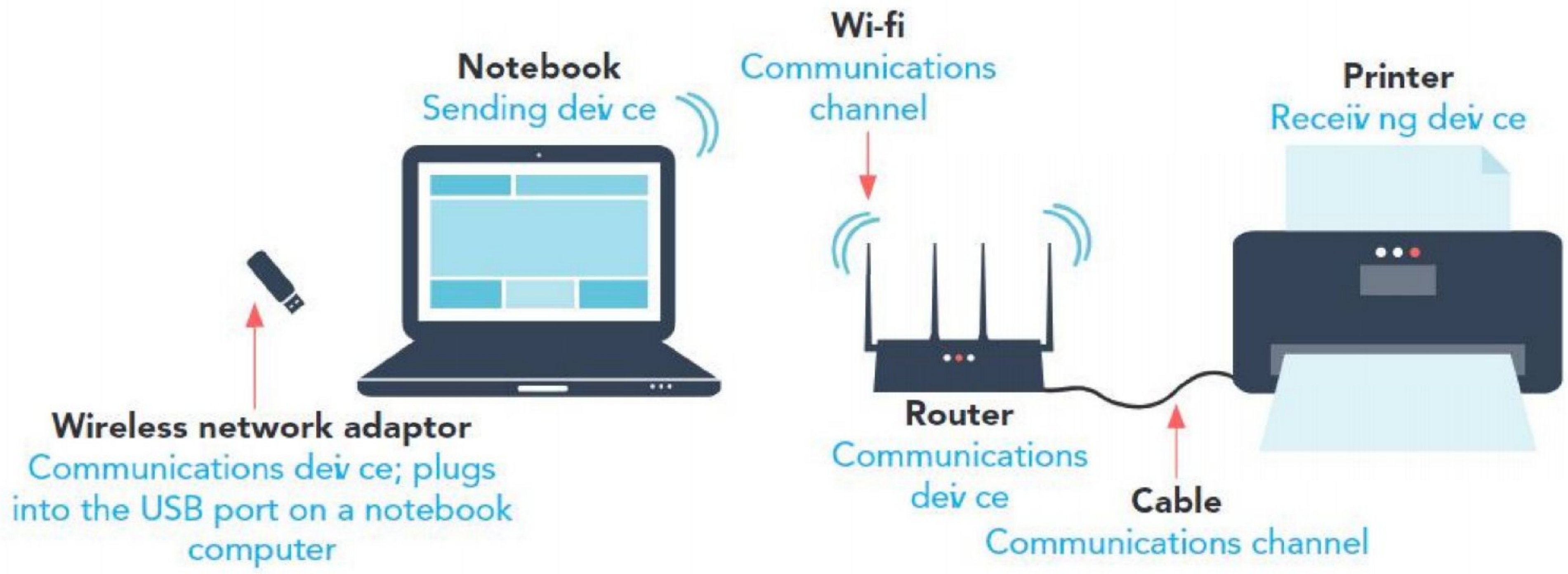


FIGURE 7.2 The notebook sends an instruction to the wireless adaptor (communications device), which sends a signal over radio waves (communications channel). The router (communications device) receives the signal and sends an instruction to print via cable (communications channel) to the printer (receiving device).

A **digital signal** consists of individual electrical pulses that represent the bits grouped together into bytes. Early networks used analog signals that consist of a continuous electrical wave. Computers process data as digital signals, so a device was needed to convert between analog and digital signals. That device is a modem.

Advantages of networks

Advantages of networks include their ability to:

- share data and information – once devices are connected, they are able to share data and information. Files on one device may be accessed by other devices on a network
- allow communication – users on network devices can communicate with each other. This may involve email, video conferencing, online chat or accessing shared calendars
- share hardware and software – once devices are connected and can communicate, they are also able to share other digital system resources.

THINK ABOUT APPLIED COMPUTING 7.2

List two benefits of connecting two or more devices together.

Types of networks

Networks can often be grouped into two types. The type of network refers to the area over which the network provides connectivity. A network is either a Local Area Network (LAN) or a Wide Area Network (WAN).

Local Area Network

A **Local Area Network (LAN)** connects computers and devices in a limited geographical area, such as a home, school, office building (Figure 7.3) or closely positioned group of buildings. Each computer or device on the network is a **node**. In many networks, nodes are connected to the LAN via cables; many new networks, however, use wireless transmission media. A wireless LAN (WLAN) uses no physical wires; instead, it uses wireless media, such as radio waves. Computers and devices that access a WLAN must have wireless capability, whether that is built-in or ad hoc using a USB wireless receiver. Usually a WLAN communicates with a wired LAN for access to its resources, such as software, hardware and the internet.

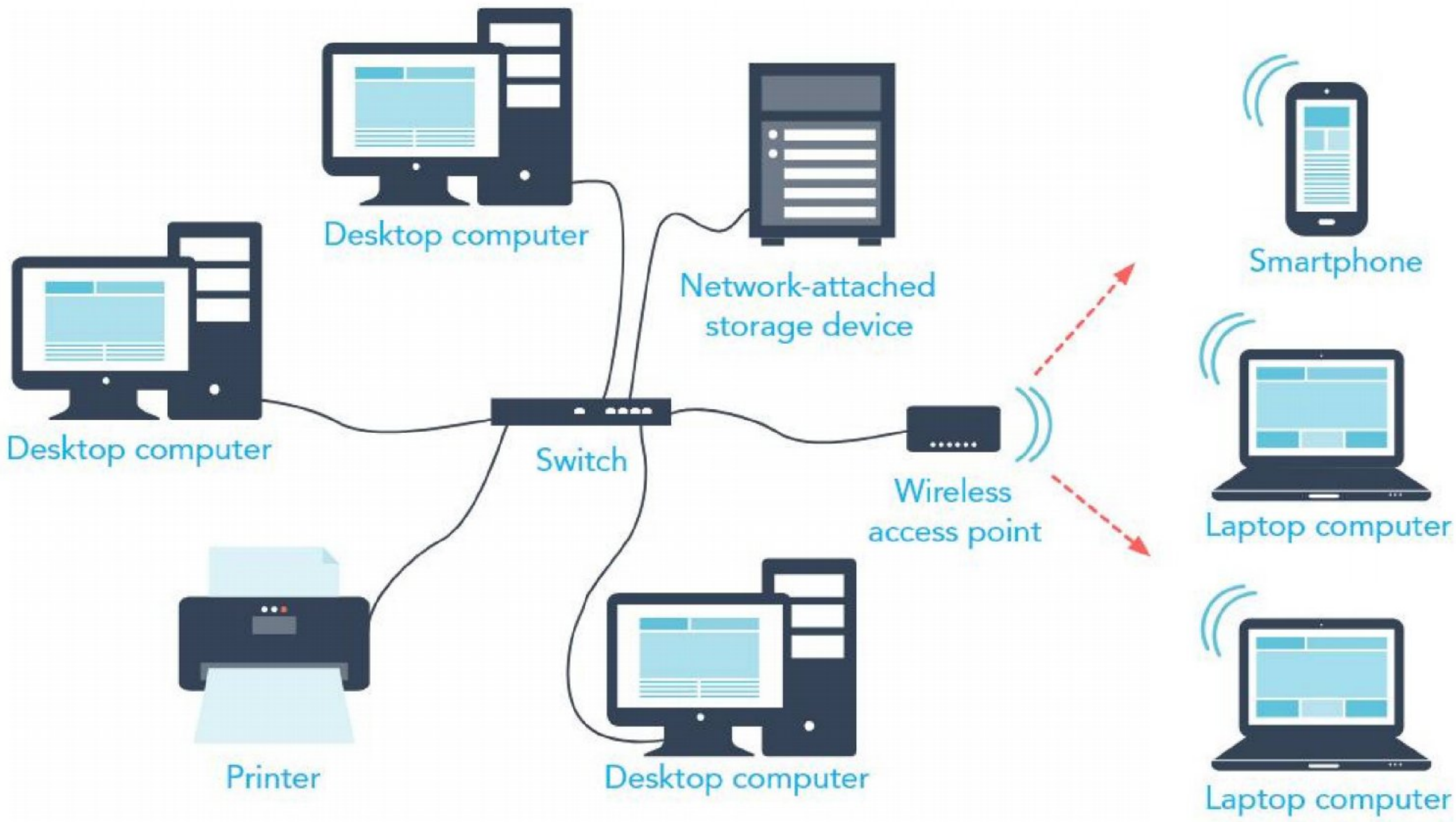


FIGURE 7.3 A Local Area Network operating within a confined geographical area

Network World – reviews of network operating systems and general information on networks

The logical design of the components of the network, including the number and types of servers, workstations and network resources, is known as the network architecture. It includes the communications devices and the types of physical and wireless transmission media used to connect components.

Home networks

If you have multiple computers in your home or home office, you can connect all of them together with a home network (see Figure 7.4). The advantages of a home network include the following:

- Desktop computers, notebooks, tablets and smartphones can all communicate with each other.
- All the computers can be connected to the internet at the same time.
- All computers can share peripherals, such as a scanner, printer or a network-attached storage device.
- Each networked computer is able to play multiplayer games with players on other computers in the house.
- Smart TVs can connect to the internet.

Wide Area Network (WAN)

7.3 THINK ABOUT APPLIED COMPUTING

Describe why an organisation that has offices in two different cities would use third-party transmission media rather than install its own dedicated connection.

A **Wide Area Network (WAN)** is situated in more than one geographical location. Often, it consists of two Local Area Networks in different locations that are able to communicate with each other using an internet connection. For example, two offices, each with its own Local Area Network, can communicate with each other using an internet connection (Figure 7.5).

One difference between a Local Area Network and a Wide Area Network is that a WAN generally uses third-party transmission media, often an internet connection, to connect the two locations. Third-party transmission media are transmission media that belong to another organisation. Often, this will be a telecommunications organisation such as Telstra or Optus. Once an organisation uses third-party transmission, it starts to lose some control over the network and the potential for data threats increases.

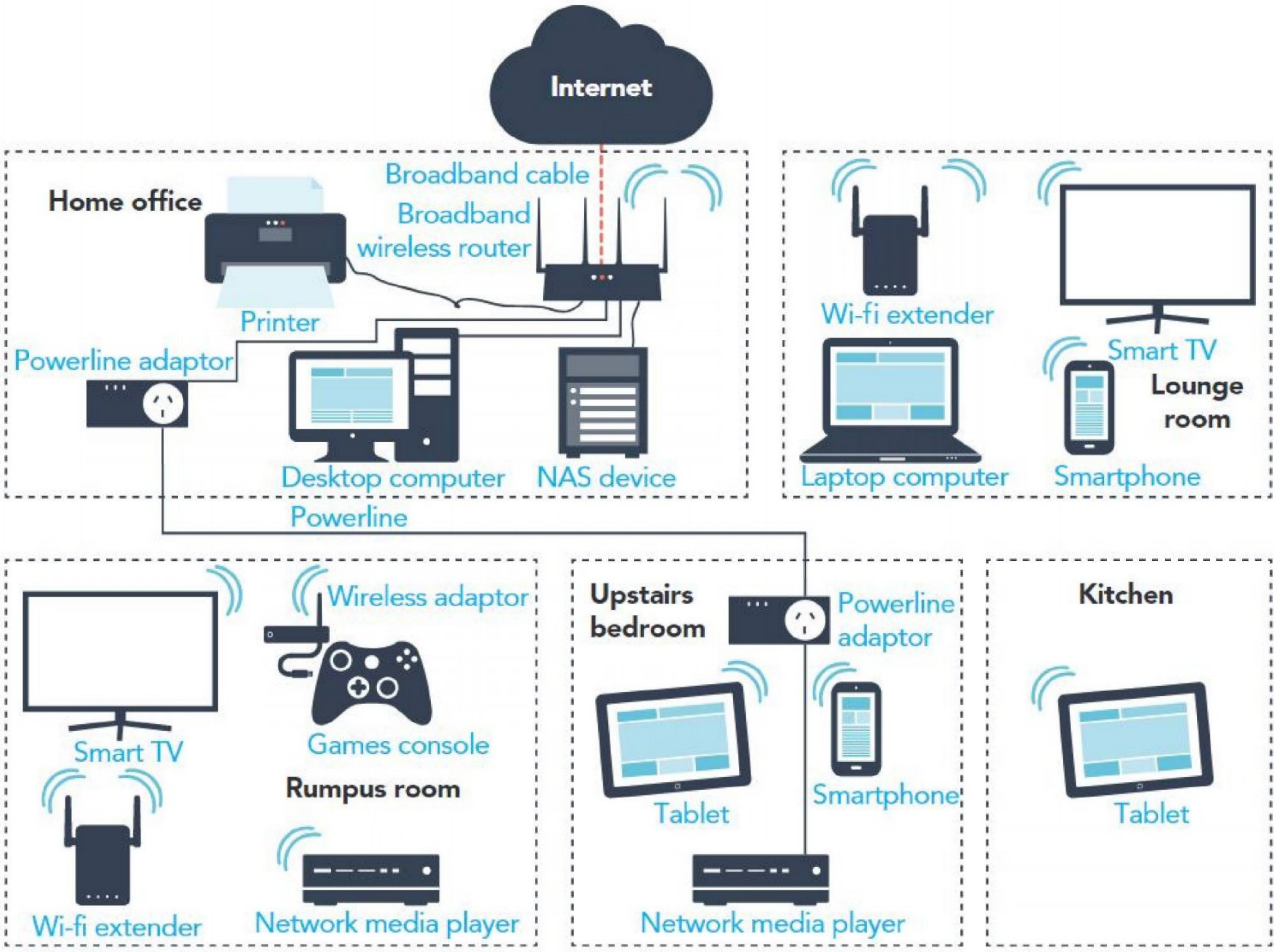


FIGURE 7.4 A home network

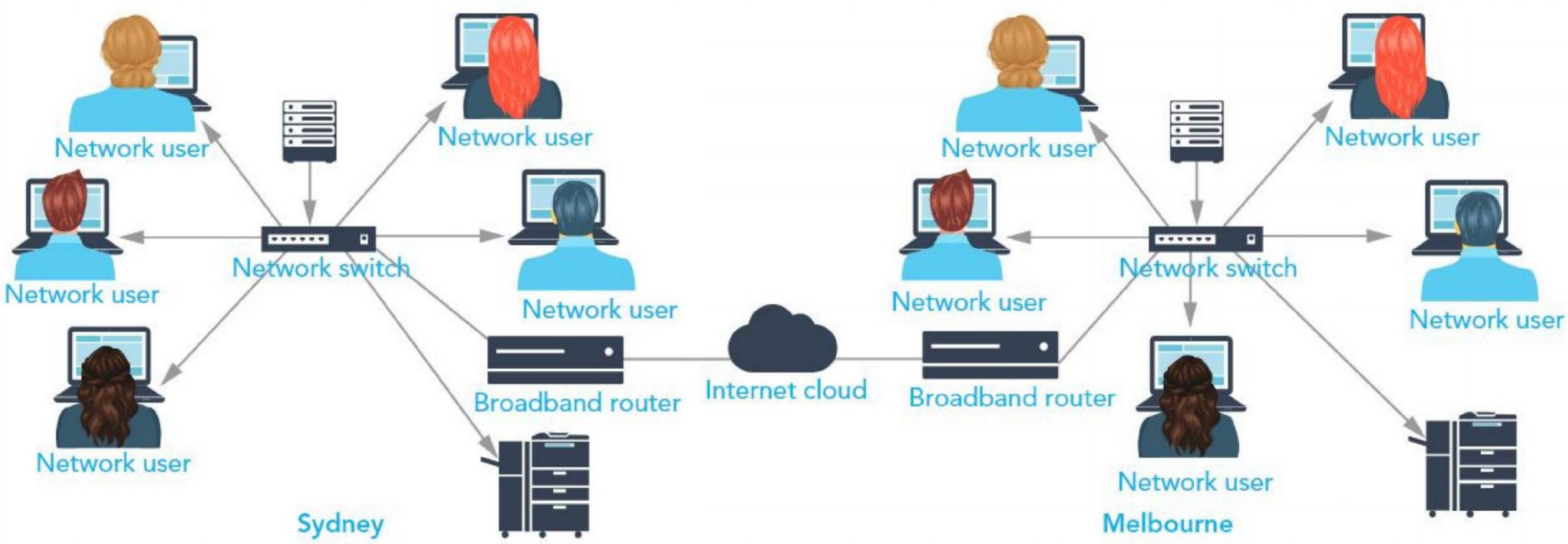


FIGURE 7.5 Two Local Area Networks connected via the internet to form a Wide Area Network

Network architecture

The term **network architecture** refers to the layout of the network, including the hardware, software, protocols and transmission media used. Each type of network can be further categorised by its architecture. Two common types of network architecture are a peer-to-peer network and a client–server network.

Peer-to-peer network

A **peer-to-peer network (P2P)** is a simple, inexpensive network that is typically used to connect organisations or groups of people with fewer than 10 computers. Each computer on a peer-to-peer network can share hardware (such as a printer), data or information that is

9780170440806

THINK ABOUT APPLIED COMPUTING 7.4

Suggest reasons why a peer-to-peer network would not be used in an organisation with more than 10 computers.

Network-attached storage devices (NAS) are discussed later in this chapter.

located on any other computer in the network (Figure 7.6). Each computer can store files on its own storage device or on another computer. Each computer in the network contains both a client operating system (like the 'Home' versions of Windows) with basic networking capability, and application software. All computers on the network share any peripheral device attached to any computer. For example, one computer may have a laser printer and a scanner, whereas another may have an ink-jet printer. They may also share a network-attached storage device (NAS) with a movie and music repository. Peer-to-peer networks are popular in homes where a login server is neither necessary nor practical.

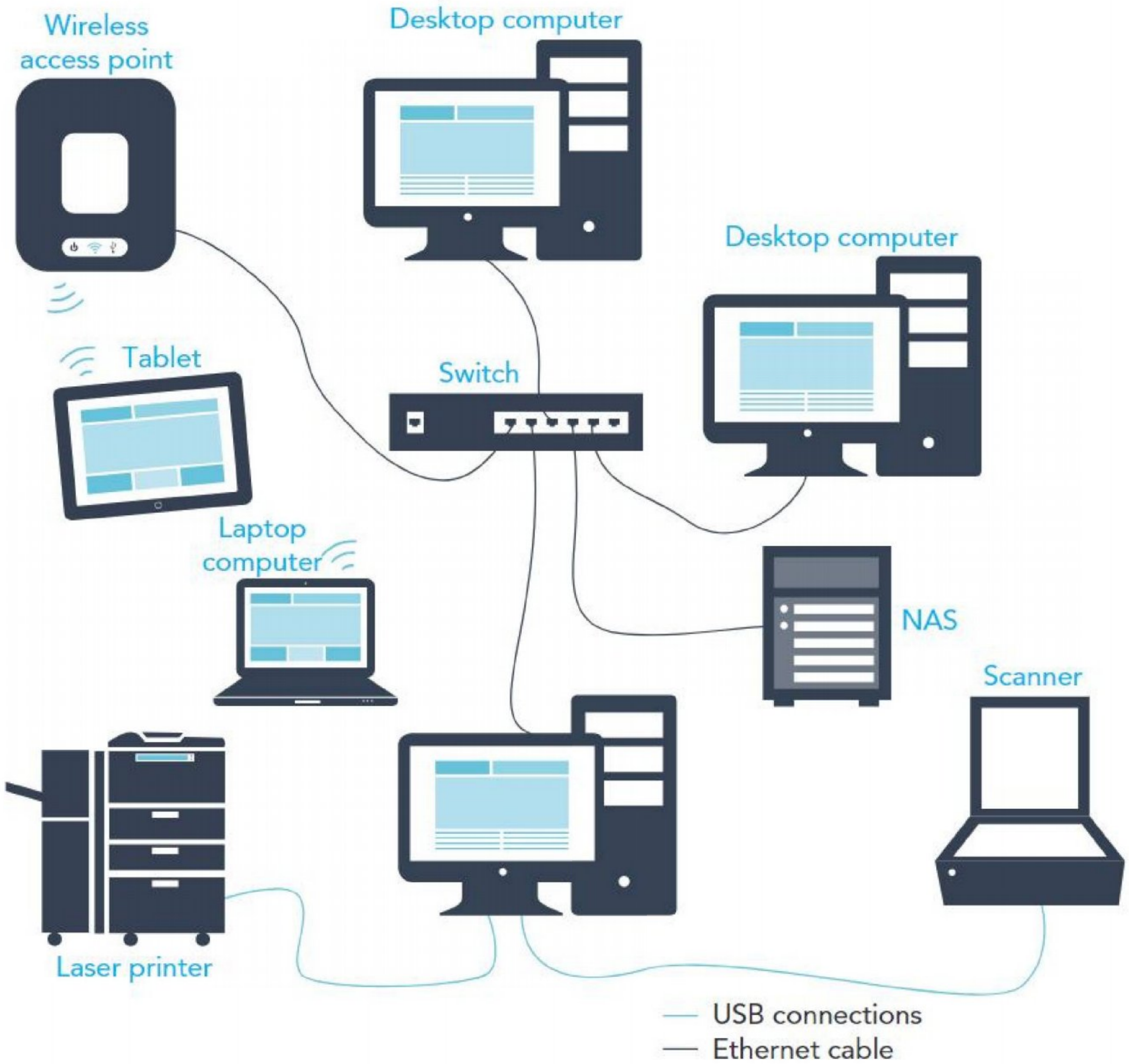


FIGURE 7.6 Each computer on a peer-to-peer network can access data from other users of the network and can share resources such as printers. A peer-to-peer network differs from a client–server network in that files can be directly transmitted between nodes, rather than from a server.

Client–server network

In a **client–server network**, a server, sometimes called the host computer, controls access to the hardware and software on the network and provides a centralised storage area for programs, data and information. Besides storage capacity, a server allows for file sharing, website hosting, email management and access to shared printers. The other computers on the network, called clients, rely on the server for these resources (Figure 7.7). For example, a server in a school's administration might store a database of student details. Every client on the network can access this database on the server.

The costs associated with a server-based network are significantly higher than those for a peer-to-peer network. Not only is the equipment necessary to build the network more expensive, but a client–server network also requires ongoing technical support to maintain the sophisticated hardware and software. On the positive side, however, there are clear economies of scale, as the cost of adding clients that share the server's resources becomes relatively less.

A client–server network provides better security than other configurations, because user access can be managed and logged.

The major difference between the server computer and the client computers is that the server has more storage space and power. Some servers, called **dedicated servers**, perform a specific task. For instance, a file server stores and manages files. Each user on the LAN can share files or programs stored on the file server. A print server manages printers and print jobs. Print jobs received from users on the LAN are queued on the print server in order of their arrival and fed to the various network printers one document at a time. A database server stores and provides access to a database. A network server manages network traffic. A number of servers are often configured in a type of rack, which makes it easier to manage the cables and power supplies. The server rack is often located in a specialised room where the temperature is kept cool to offset the amount of heat generated by the power-hungry servers.

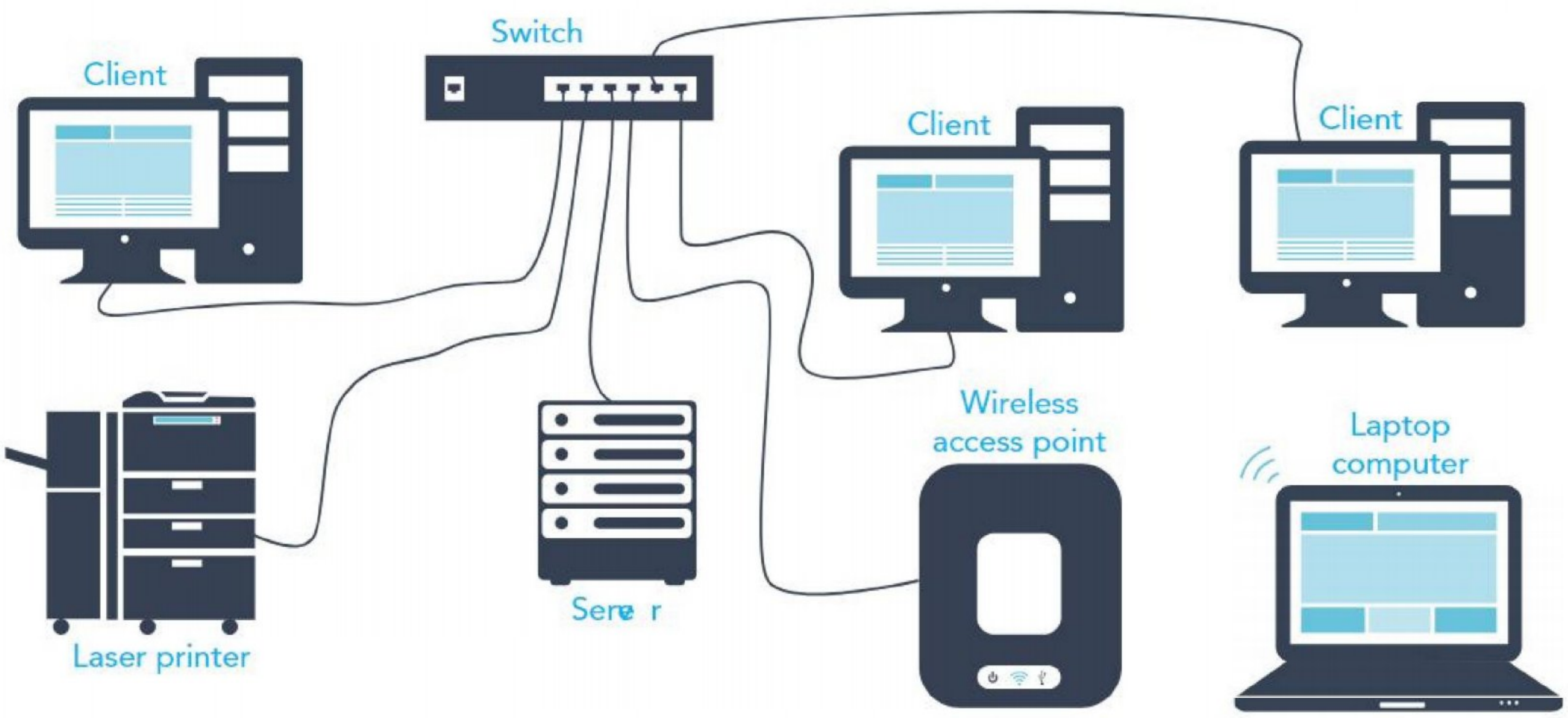


FIGURE 7.7 On a client–server LAN, one or more computers act as a server and the other computers on the network are called clients.

In the past, network administrators used to dedicate each server to a particular task. One application per server made it easier to track down any problems that arose. This approach, however, does not take advantage of the processing power of modern servers. Also, a larger storage room is required as the number of servers is increased. Many servers in networks are now virtual rather than physical. Specially designed software is used to convert one physical server into multiple virtual servers, with each virtual server responsible for a particular purpose.

Although a client–server LAN can connect a smaller number of computers, it is typically most efficient for it to connect 10 or more computers. Most client–server LANs have a network administrator because of their larger size. The **network administrator** is the operations person in charge of the network. Most schools have a client–server network installed. This allows staff and students to store data on shared drives.

THINK ABOUT APPLIED COMPUTING 7.5

List two advantages of using a client–server network in a school.

Wireless Personal Area Networks (WPANs)

A **Personal Area Network (PAN)** is a computer network for connecting an individual's devices within a limited range. A **Wireless Personal Area Network (WPAN)** is a type of personal network that uses wireless transmission media to transfer data between a user's personal devices. It allows an individual to connect their devices together and access the internet or a local network using wireless communication techniques. A WPAN may also be referred to as a short-range **wireless network**.

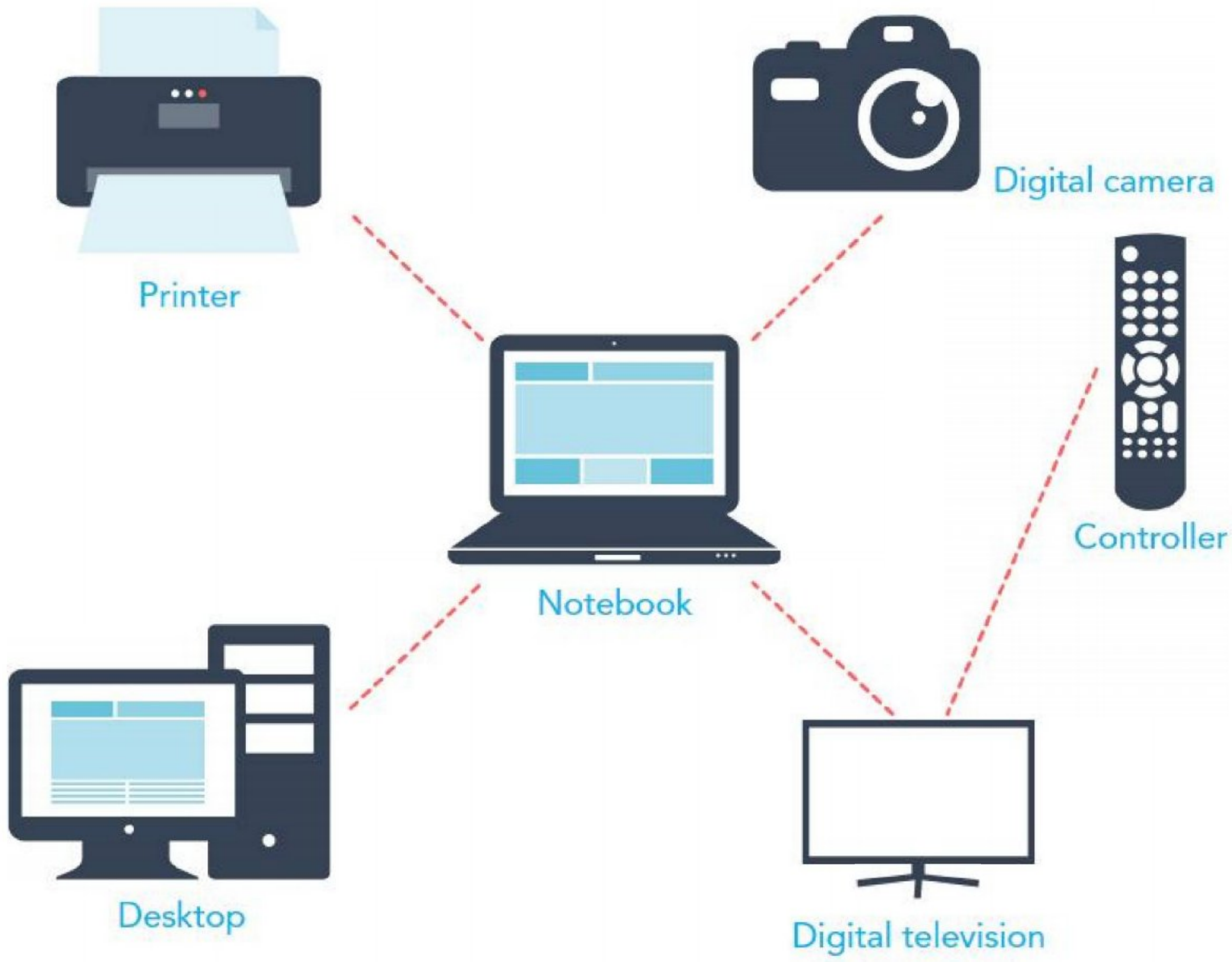


FIGURE 7.8 A Wireless Personal Area Network (WPAN)

Common devices found within a WPAN include desktops, laptops, smart televisions, wearable technology, smartphones and speakers.

A WPAN may use one of the following wireless communication standards:

- 802.11(x)
- Bluetooth
- infra-red (IR)
- satellite
- microwave.

Networking hardware and software

For Unit 2, Outcome 2, you are required to describe the function of hardware and software devices used to maintain a network in a global environment.

To be able to communicate data between devices on a network, both hardware and software are required. Hardware is the physical components of a network, while software contains instructions to the hardware to enable it to complete particular tasks.

Network hardware

Each network hardware component provides a different function to manage, control and secure data and information to be communicated on a network

Hardware used to manage, control and secure data includes:

- network interface card (or wireless adaptor)
- server
- wireless access point
- router
- switch
- modem.

Network interface card (or wireless adaptor)

Each device needs to have a **network interface card** (or wireless adaptor) installed to allow access to the network or information system. A network interface card slots into the motherboard of a device and then provides ports to allow the device to connect to a network. A **wireless adaptor** is connected to the motherboard and allows the device to send and receive wireless signals. Many devices come with both a wired network interface card and a wireless adaptor installed. This allows the device to communicate with other devices.

Shutterstock.com/Andrei Metelev

FIGURE 7.9 A network interface card

Server

A **server** is a device that is used to provide services to other devices connected to a network. For example, many networks have a file server. All of the files of the organisation may be stored on one or multiple servers; users operating other devices (known as clients) can then access the data from these servers.

Often, all the data and information related to an online or browser-based information system will be stored on a web server. When a user wishes to access the information system, they will be directed to the web server, where they will be able to access the data and information required.

Servers can be set up on a network to control and manage particular tasks on the network. The following are all types of servers:

- *file server* – used to store files in a central location, and manages the files so that authorised network users can access them from the central location. A file server generally does not perform any processing tasks, nor does it run any application software. Rather, its purpose is to enable users to store and retrieve data

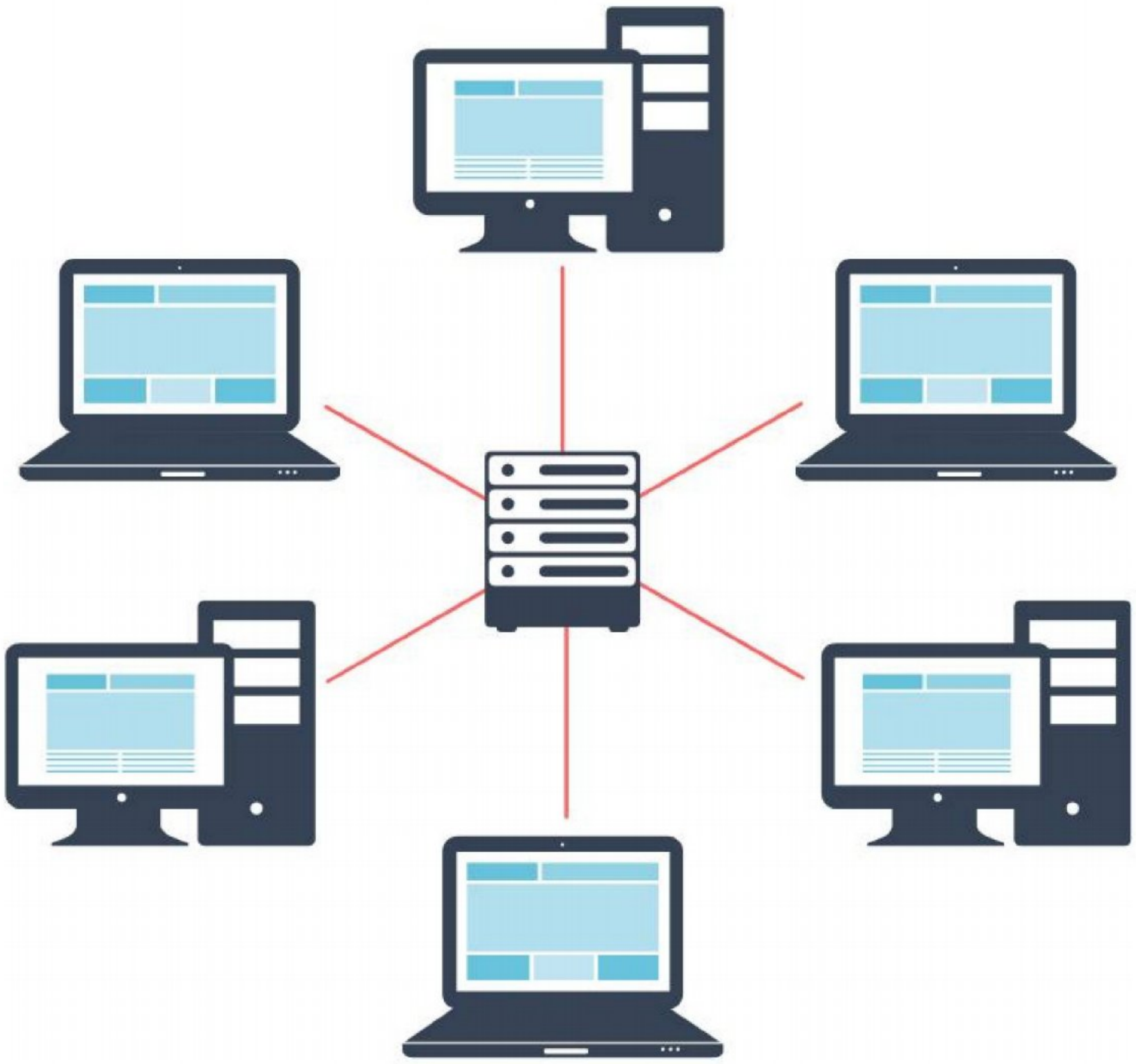


FIGURE 7.10 A Local Area Network with a dedicated server

THINK ABOUT APPLIED COMPUTING 7.6

Provide three reasons why users might choose to store their files on a central file server.

- *web server* – stores the files related to a website, a group of websites or an online browser-based information system. When a user requests a web page, the request is sent to the web server, which then sends the web page and associated files to the user using Hypertext Transfer Protocol (HTTP). The Universal Resource Locator (URL) of each web page is converted to an IP address that directs the request to the specific web server
- *proxy server* – manages all network requests for resources from the internet. Any request by a network user for resources from the internet will first be sent to the proxy server. The proxy server then sends the request for resources to the internet. When the proxy server receives the resources, it directs them to the correct user
- *database server* – stores a database application and the data used by the application. The user can access the data stored on a database server through 'front end' software, installed on their device, that retrieves the required data. In addition to storing data, the database server can perform analysis on the data
- *print server* – receives requests for printing from network users and sends these requests to the printers installed on the network. It allows requests to be centrally stored so that they can be managed and controlled
- *email server* – sends and receives all emails for users on a network. Each time a network user sends an email, it will be sent to the email server, which will then forward it on to the recipient. Conversely, when an email is sent to a user, it will first be sent to the email server, which will then deliver it to the user. A copy of all emails sent and received is normally stored on the email server.

Wireless access point

A **wireless access point (WAP)** is a communications device used on wireless Local Area Networks. It acts as a central transmitter and receiver of wireless radio signals. A wireless access point is often connected to a wired network backbone for faster transmission of data back to the network server.

Wireless access points are mostly used in business networks where larger buildings and spaces need wireless coverage. Home networks are generally small enough that a wireless router can provide sufficient coverage without the need for an access point.

If an area is too large to be covered by a single wireless access point, multiple access points can be used. There can be a momentary loss of connectivity when a user moves from the vicinity of one access point to another. Figure 7.11 shows a wireless network with overlapping access points. Overlapping access points provide a seamless area for users to move around in, using a feature known as 'roaming'. When a user moves from one area to another, the wireless networking hardware automatically jumps to the access point that gives the strongest signal.

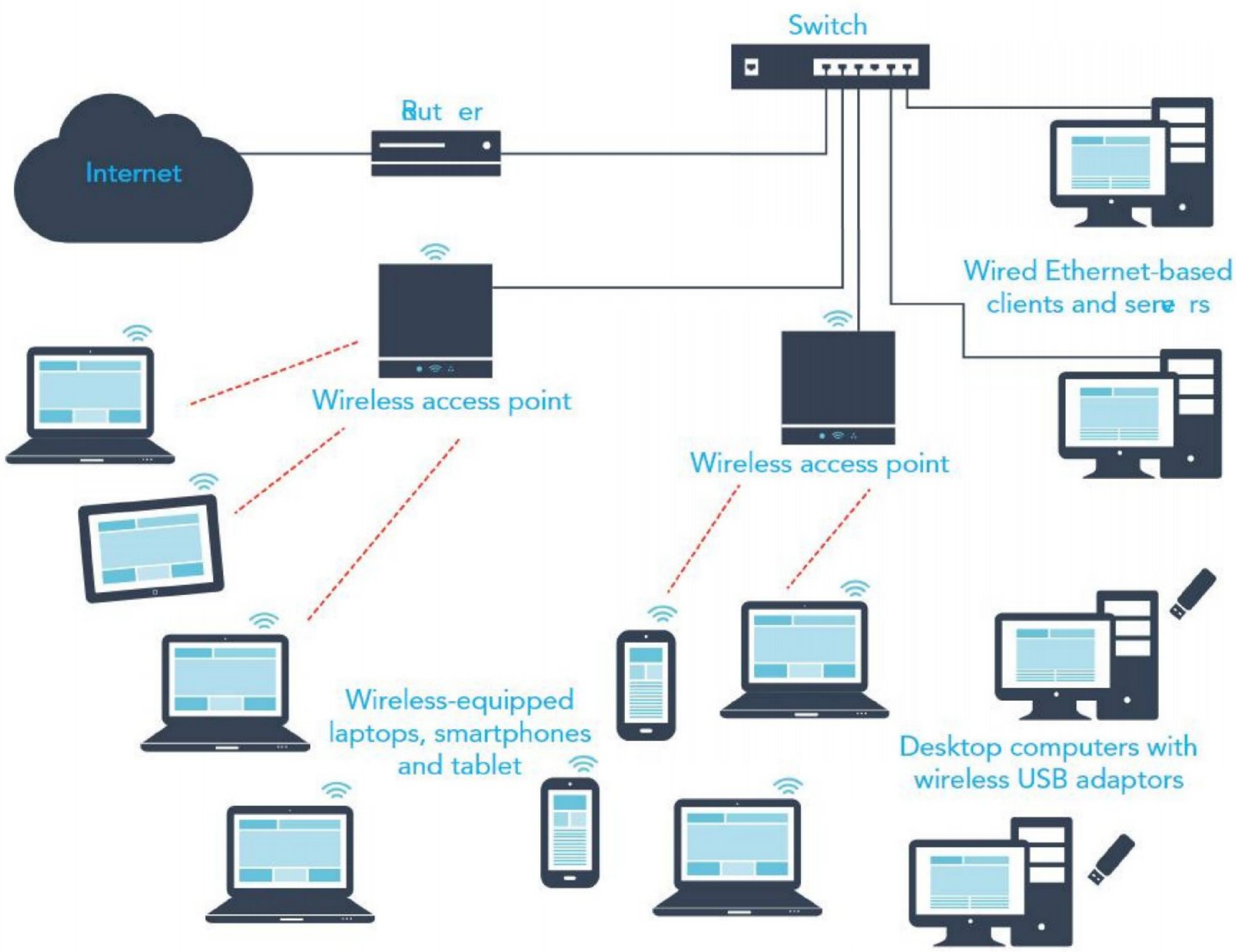


FIGURE 7.11 A business network using multiple wireless access points

Your school most likely operates a wireless network with this type of configuration.

Router

A **router** is a device used to connect multiple networks – several LANs, or a LAN and a WAN (such as the internet). A router will connect LANs and WANs irrespective of the hardware and network communications protocol used in each segment. On the internet or in a large corporate network, for example, routers receive TCP/IP packets, look inside each packet to identify the source and target IP addresses, and then forward these packets as needed to ensure the data reaches its final destination. Figure 7.12 looks at how a router can be used to communicate between two LANs operating within one organisation.

While a switch uses a node's Media Access Control (MAC) address to identify the path on which to send the packet, a router uses the IP, IPX or Appletalk address. The algorithm switches used to decide how to forward packets is different from the one used by routers. Most carriers, such as Vodafone, offer a 4G USB modem stick. Its small size makes it ideal to use with a mobile device, such as a laptop computer, to connect to the internet using the 4G phone network.

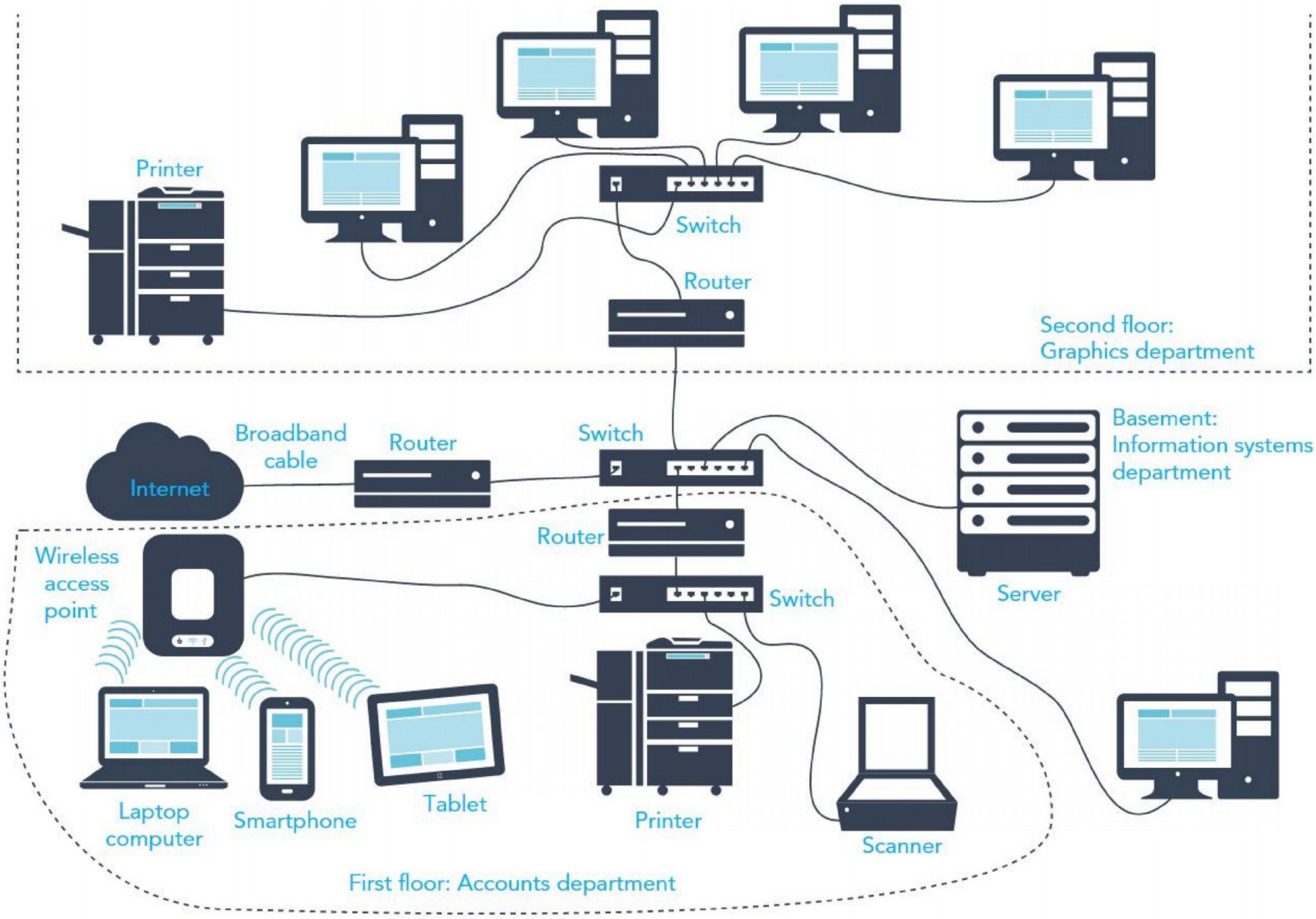


FIGURE 7.12 How a router can connect two LANs and provide internet access to both networks

Broadband routers

Routers for home networks (often called **broadband routers**) also can join multiple networks. These routers are designed specifically to join the home (LAN) to the internet (WAN) to share internet connections.

Wireless broadband routers combine the functions of a basic router (connecting the LAN to the internet), a switch (for devices, such as a desktop computer, that are connected by cable), a firewall (security measure) and a wireless access point (to allow wireless connectivity). Figure 7.13 shows a broadband router with an antenna for wireless connections, LAN ports for wired connections and a WAN port for connection to the internet.

The type of broadband router used to act as a bridge between a LAN and the internet depends on the type of connection provided by the internet service provider (ISP). A cable connection uses the high bandwidth available through the same broadband connection that delivers information to a television via a provider such as Foxtel. An **asymmetric digital subscriber line (ADSL)** connection provides internet access using copper wiring in telephone lines. The cost of an ADSL broadband router is between about $150 and $200. A cable broadband router is usually sourced from an ISP, since it has to be registered on the ISP's network before it will work.

iStock / Getty Images Plus / scanrail

FIGURE 7.13 Wireless broadband router

Switch

A **switch** is a device that provides a connecting point for cables in a LAN.

Network traffic in a LAN typically follows specific paths that connect members of a work group, such as the accounts department of a business. A switch stores the address of every device down each cable connected to it. When a packet arrives, the switch uses simple logic to detect the destination of the packet and sends the packet only down the cable that contains the addressed device. The result is that the packet reaches its destination quickly and without colliding with packets being sent to other nodes.

As packets are sent straight to the destination device through a switch, the two devices act as if they were directly connected. On a 100 **Mbps** switch, data can be sent to all nodes simultaneously and uninterrupted at a speed of 100 Mbps.

Encryption of data packets enhances the security of wireless transmissions. Setting up complex passwords for file sharing, routers and access points, as well as using encryption keys that are not obvious, greatly improves security.

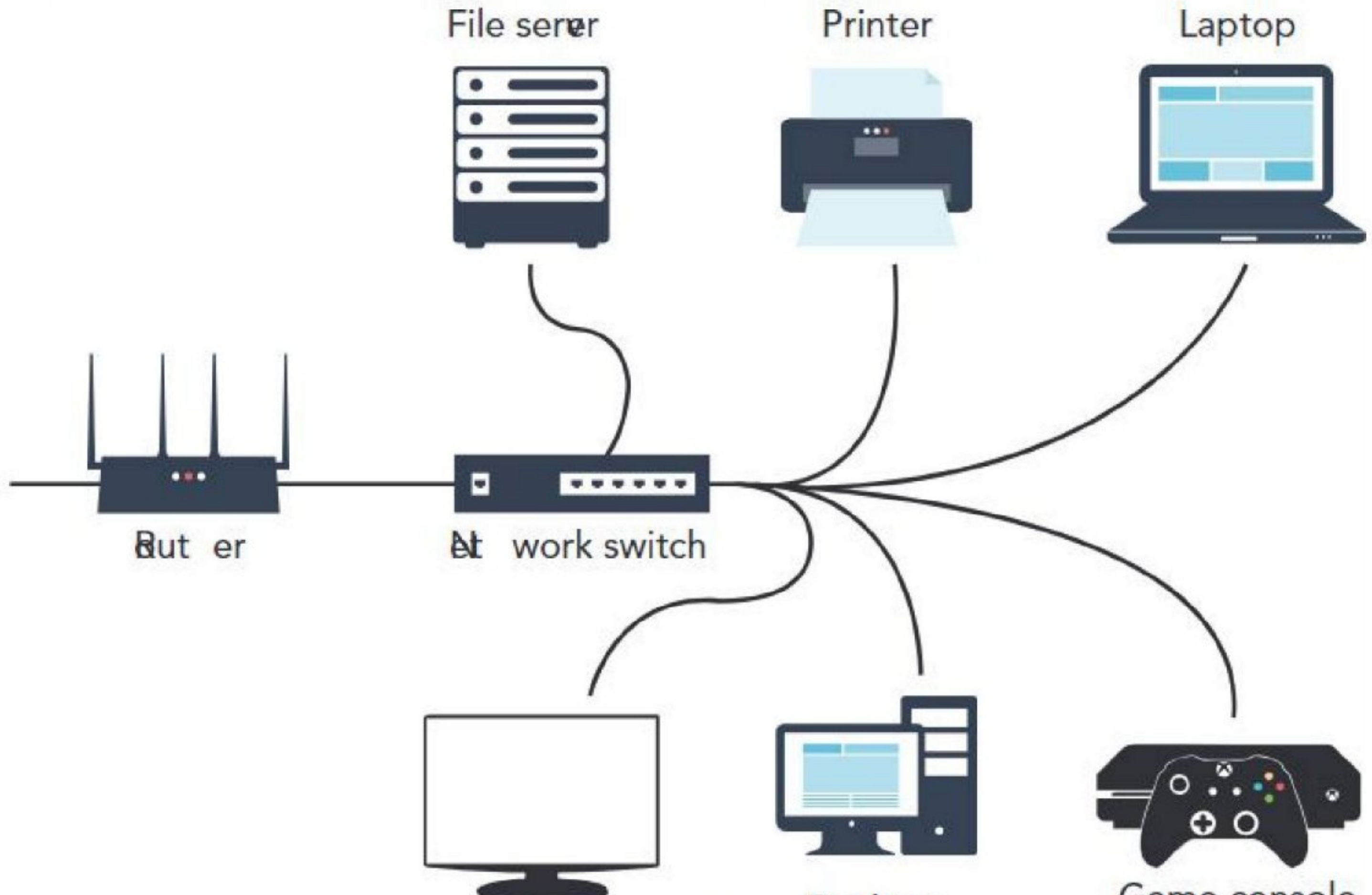


FIGURE 7.14 A switch is used to connect multiple devices.

THINK ABOUT APPLIED COMPUTING 7.7

What other types of connections are available over the NBN?

Modem

A **modem** is a device that is used to connect Local Area Networks to the internet. A modem converts digital signals (binary) into analog (audio) signals, and then back again, so data can be transmitted over a telephone line. In the early days of the internet, most households and organisations created a connection to the internet using a telephone line for data communication. The advantage of this was that most homes and businesses already had a telephone connection, which meant there was no need to install a new, dedicated connection at each location.

With the introduction of the National Broadband Network (NBN) a fibre-optic 'backbone' has been installed covering a large part of Australia. Many households, however, have a 'fibre to the curb' (FTTC) connection, whereby fibre-optic cabling is installed in the street, then a copper (**unshielded twisted pair**) cable connects the fibre-optic backbone to the house.

In these situations a modem (or NBN connection box) is still used to manage and control data transmitted over the internet connection, although in this situation it is not required to convert signals from analog to digital.

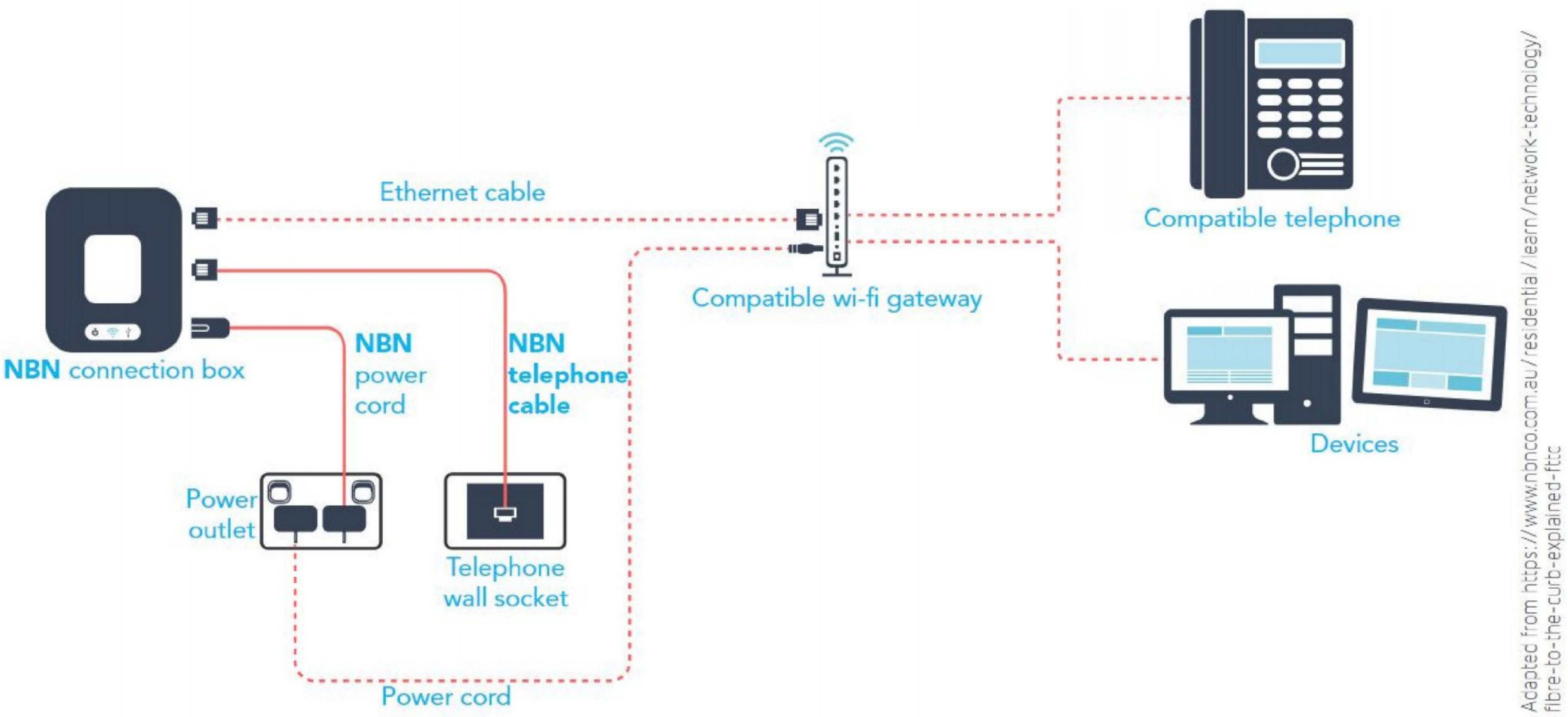


FIGURE 7.15 Equipment needed to create a FTTC NBN connection

Merging technology

Many of the devices discussed in the preceding pages have now been merged into a single device as technology has improved.

These include:

- modem/router
- router/wireless access point
- router/switch
- modem/wireless access point/router.

Network software

Software is used to control, manage and secure data in a network.

Network operating system

Just as your laptop or smartphone needs an operating system, each network needs a network operating system. A **network operating system** (also called a **network OS** or **NOS**) is the system software that organises, controls and coordinates the activities on a Local Area Network. A NOS controls the attached computer systems, any peripherals and the communication between them. The tasks that a NOS performs include:

- *administration* – adding, deleting and organising users and performing maintenance tasks, such as backup
- *file management* – locating and transferring files
- *device management* – coordinating print jobs and reports sent to specific printers on the network, ensuring resources are used correctly and efficiently
- *security* – monitoring and, when necessary, restricting access to network resources.

7.8
THINK ABOUT APPLIED COMPUTING

Check out the software installed on the network in your school. Find out from the network manager how many users are covered by a site licence for one of the software packages. Compare the cost of the site licence with that of the equivalent number of single-user versions (check the price of a single-user version online). What cost saving does the school gain by purchasing the network version rather than the single-user version? What disadvantage is there in purchasing site licence software compared with single-user packages?

Wired, wireless and mobile communications technology

There are a number of communication methods that can be used to transmit data and information over a network. These included wired cabling, wireless transmission and the use of existing mobile (or cellular) networks. The method used to transmit data will depend on the needs of the users of the network. Each method has strengths and weaknesses in terms of cost, data storage options, data transfer rates, reliability and security.

Wired communication technology

A **wired network** is a network in which the devices are connected using a physical cable.

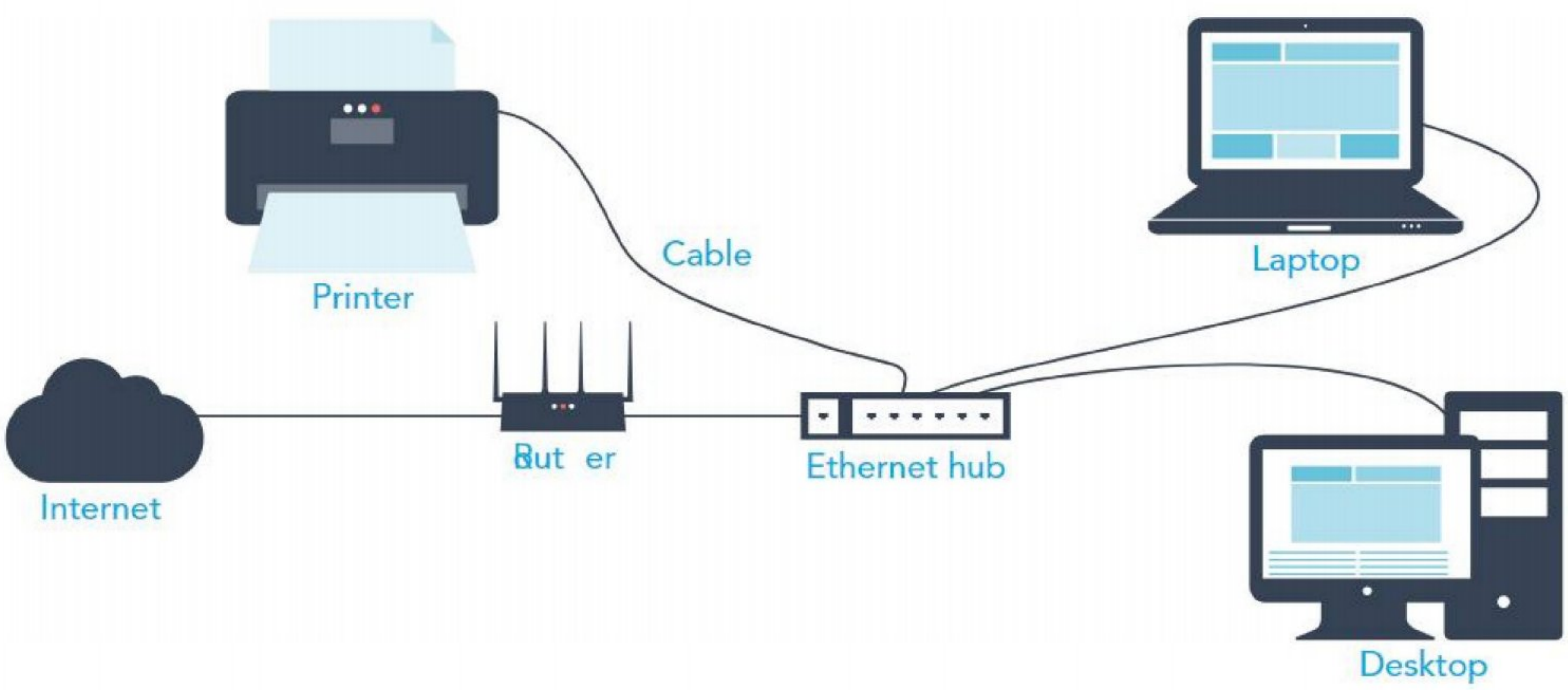


FIGURE 7.16 A wired network

Advantages of wired networks compared to wireless networks include:

- *faster data transfer speeds* – although wireless and mobile network data transmission rates are continually improving, it is still considered that wired networks generate superior speeds (particularly on average)
- *better security* – wired networks are considered more secure than wireless networks. It is much more difficult for a hacker to intercept the data in a wired network, particularly a Local Area Network, while there is also less chance that data will be lost due to signal interference
- *more reliable connection* – not only is a wired network considered more reliable in terms of data interference, but wired networks are considered to have more consistent data transfer rates than wireless networks, where the strength of the signal – and hence speed – may vary due to a range of factors.

Disadvantages of wired networks compared to wireless networks include:

- *lack of mobility* – as wired networks generally have fixed data points, which are used to connect devices, there is less flexibility regarding where devices can be used while staying connected to the network. As a result, devices remain stationary, limiting the ability of the user to roam while continuing to use their device
- *installation* – the cost of installing a wired network is considerably more expensive than the cost of a wireless network. The cost, as well as the time and effort, of purchasing and installing cabling is much greater than that of installing a wireless access point
- *maintenance* – from time to time, in a wired network, cabling may stop working correctly, or the organisation may require new data points or rearranging of existing data points. Just as the cost of installation can be significant, maintaining and making changes to an existing wired network is also time-consuming and costly.

Noise in a cable can be caused by cross-talk (electric currents between pairs of wires in the same cable) and by outside electrical fields, such as those created by power lines, motors and radio transmitters. The greater the cable's ability to resist internal and external noise, the longer the cable can be used to connect workstations and resources.

Twisted-pair cable

One of the more commonly used transmission media for network cabling and telephone systems is the **twisted-pair cable**. This cable contains one or more twisted-pair wires bundled together (see Figure 7.17). Each **twisted-pair wire** consists of two separate insulated copper wires that are twisted together. The wires are twisted together to reduce noise. Noise is an electrical disturbance that can degrade communications.

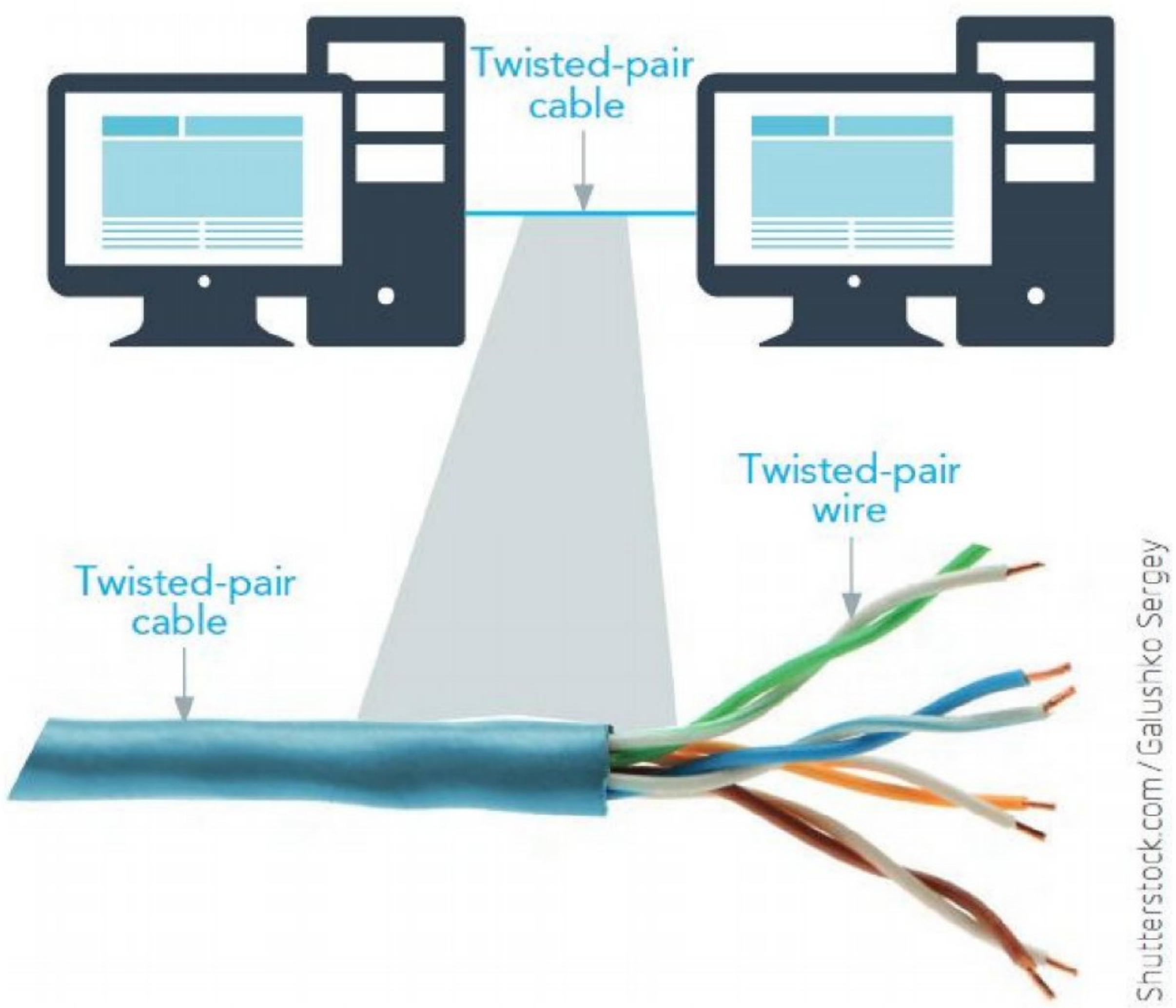


FIGURE 7.17 A twisted-pair cable consists of one or more twisted-pair wires; each twisted-pair wire is usually colour-coded for identification. Telephone networks and LANs often use twisted-pair cables.

Fibre-optic cable

Fibre-optic cabling is a wired transmission media that contains shards of glass, which reflect pulses of light generated by small lasers or light-emitting diodes (LEDs).

The core of a fibre-optic cable consists of dozens or hundreds of thin strands of glass or plastic that use light to transmit signals. Each strand, called an **optical fibre**, is as thin as a human hair. Inside the fibre-optic cable, an insulating glass cladding and a protective coating surround each optical fibre (Figure 7.18).

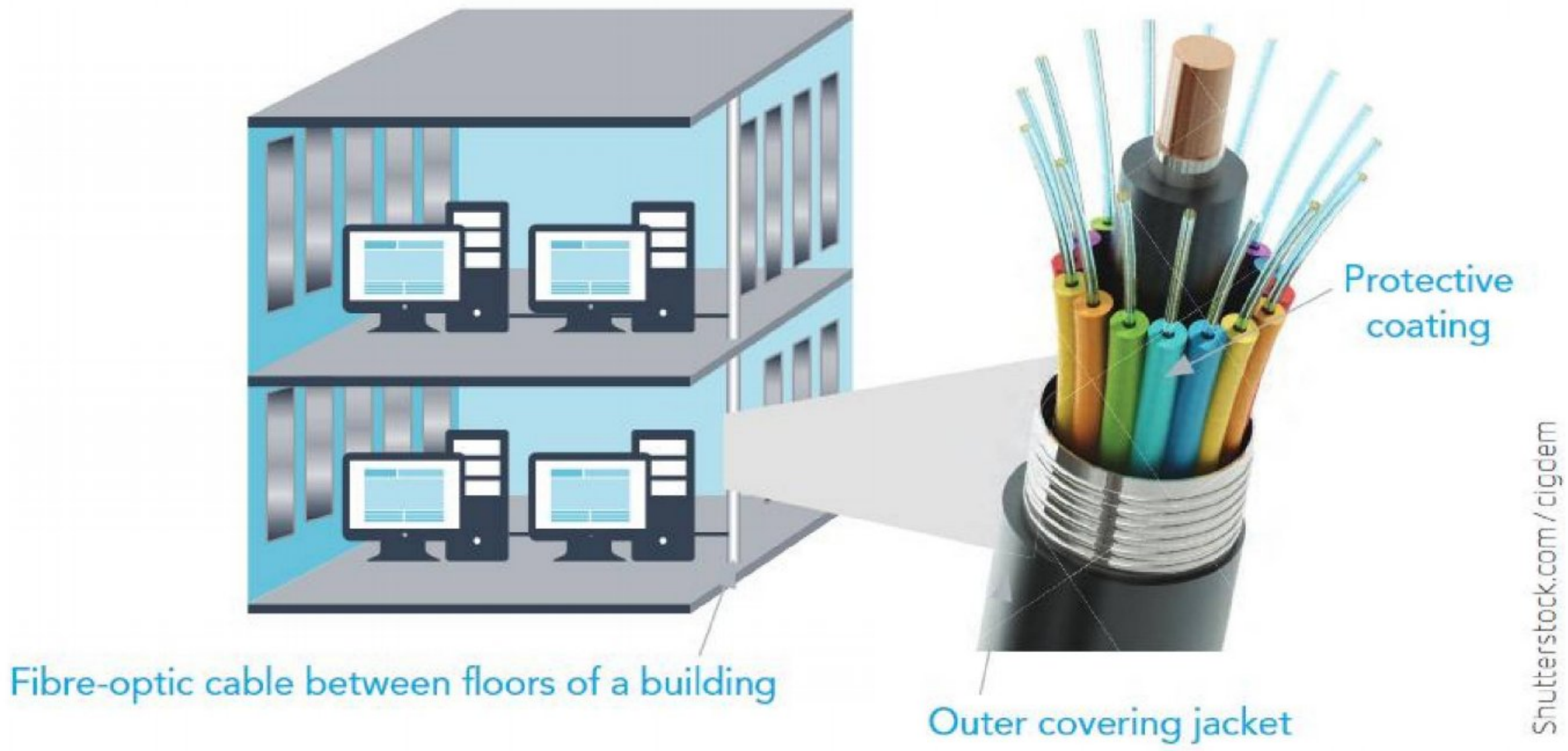


FIGURE 7.18 A fibre-optic cable consists of hair-thin strands of glass or plastic that carry data as pulses of light.

Fibre-optic cables have several advantages over twisted-pair cables. These advantages include:

- the ability to carry significantly more signals than wire cables
- faster data transmission
- less susceptibility to noise (interference) from other devices, such as a copy machine
- better security for signals during transmission as they are less susceptible to noise
- their smaller size (much thinner and lighter).

The material used for fibre-optic cable has optical properties that cause light in the cable to totally internally reflect from the outer surface. This allows all the light to progress down the cable rather than escape to the surrounding air. As fibre-optic cables transmit pulses of light, they have total immunity from electrical noise. Signals can therefore be sent over much larger distances than with twisted-pair cables.

The disadvantages of fibre-optic cable are that it costs much more than twisted-pair cable, and it can be difficult to install and modify. Despite these limitations, many local and long-distance telephone companies and cable television operators are replacing existing telephone and coaxial cables with fibre-optic cables. Many businesses are also using fibre-optic cables in high-traffic networks or as the main cable in a network.

7.9 THINK ABOUT APPLIED COMPUTING

Find out why fibre-optic cabling is not normally used inside a room.

National Broadband Network

The **National Broadband Network (NBN)** is designed to provide the infrastructure to deliver affordable and reliable high-speed internet and telephone access to all Australians. The geography of Australia, and especially its size, means that a variety of technologies are required to deliver the NBN. As originally proposed, the NBN was to deliver communications at a speed of 100 Mbps, with fibre-optic cable connecting homes directly to the internet in a majority of locations. The scheme has been modified, and it is now planned that fibre-optic cable will reach a node in the street (called the 'street cabinet'); from there it will be

View the progress and extent of the NBN rollout at the NBN website.

7.10
THINK ABOUT APPLIED COMPUTING

Some future speculators believe that, by 2023, households will on average have more than 100 devices connected to the internet. Devices such as lights, air conditioners, heating systems, door locks, irrigation systems, motion detectors, smoke detectors, home entertainment systems and connections that monitor family members, pets and vehicles will all require bandwidth. Online video chatting, which requires fast download and upload speeds, will become common practice. Do you think that speeds of 25 Mbps will provide adequate bandwidth in five to 10 years' time? What devices do you expect to be connected in your home in 10 years' time?

split, and hybrid cables or existing copper telephone networks will be used to carry signals to homes and businesses. Other types that may be used include coaxial cabling, foxed wireless and SkyMaster connections.


AAP Image/NBNCC

FIGURE 7.19 The rollout of the NBN

Households where it is impractical to use fibre-optic cable will connect to the NBN via fixed wireless and satellite technologies. Up to 10 per cent of homes have needed wireless or satellite connections.

Wireless communication technology

A wireless network is a computer network that uses wireless data transfer between network devices. Wireless networks generally use radio communication to transmit data.

Wireless networks can include:

- wireless Local Area Networks (WLANs)
- mobile networks
- satellite networks
- microwave networks.

Wireless transmission media are used when it is inconvenient, impractical or impossible to install cables. The faster speeds afforded by the 802.11ax standard place wireless connections are on a par with wired networks in terms of speed. Wireless transmission media used in communications include broadcast radio, cellular radio, microwaves, communications satellites and infra-red.

An 802.11ac wireless router operates on the 5 GHz bandwidth but can still run on the 2.4 GHz network simultaneously. Some vendors may quote a wireless 802.11ac router as a single-speed device operating at 1.75 Gbps. This figure is an amalgamation of the 5 GHz and 2.4 GHz capabilities of the router. That is, it is gained by adding 1.3 Gbps from the 5 GHz connection to 450 Mbps from the 2.4 GHz network.

TABLE 7.1 Transfer rates of various wireless transmission media

Transmission medium	Transfer rate (maximum)
Bluetooth	1–2 Mbps
HomeRF	1.6–10 Mbps
802.11a	54 Mbps
802.11b	11 Mbps
802.11g	54 Mbps
802.11n (Wi-Fi 4)	108–600 Mbps
802.11ac (Wi-Fi 5)	867–1300 Mbps
802.11ax (Wi-Fi 6)	3.5 Gbps
Cellular radio	
2G	9.6–14.4 Kbps
3G	200 Kbps – 3.1 Mbps
4G	100–300 Mbps
5G	10–30 Gbps
Microwave radio	150 Mbps
Communications satellite	1 Gbps
Infra-red	115 Kbps – 4 Mbps

Wi-fi communications

For **wi-fi communication** transmissions, you need a transmitter to send the radio signal and a receiver to accept it. To receive the signal, the receiver has an antenna that is located in the range of the signal. Some networks use a transceiver, which both sends and receives signals from wireless devices. Wi-fi communication is slower and more susceptible to noise than physical transmission media, but it provides flexibility and portability.

Wireless computer network components typically use radio signals in either a 2.4 GHz range or a 5 GHz range. A 5 GHz network can carry more data than a 2.4 GHz network; however, the higher the frequency of a radio signal, the shorter its range. So a 2.4 GHz network covers a much larger range than a 5 GHz network. The higher frequency is not as good at penetrating solid obstacles such as walls. On the other hand, there are a number of household devices, such as cordless phones, that operate on the 2.4 GHz band and that could interfere with a broadcast transmission at that frequency. The 5 GHz band does not compete with other common household devices. A number of components now come with dual-band capacity to get the best of both worlds.

Bluetooth

Bluetooth uses short-range radio waves to transmit data among Bluetooth-enabled devices. These devices contain a small chip that allows them to communicate with other Bluetooth-enabled devices. Examples of such devices are desktop personal computers, notebook computers, hand-held computers, mobile telephones, fax machines and printers. To communicate with one another, they must be within a specified range (about 10 metres, but the range can be extended to 100 metres with additional equipment). A popular use of Bluetooth is to enable hands-free chatting on mobile phones or to play music through a speaker or in the car. Most new cars are now sold with a built-in Bluetooth station that the user can synchronise with their mobile phone. Bluetooth and wi-fi communications technologies use radio signals.

Bluetooth gets its name from a legendary Viking king (one of the main developers was the Norwegian communications company Ericsson). One advantage of Bluetooth is that it can be used to set up a network on the spur of the moment. A group at a meeting can all network their computers to share files, access data from mobile telephones, and send documents to printers and fax machines, all without cables or additional network interface cards. Of course, all of the devices must be Bluetooth-enabled. Data transmission using Bluetooth is fairly slow compared with other wireless transmission media.

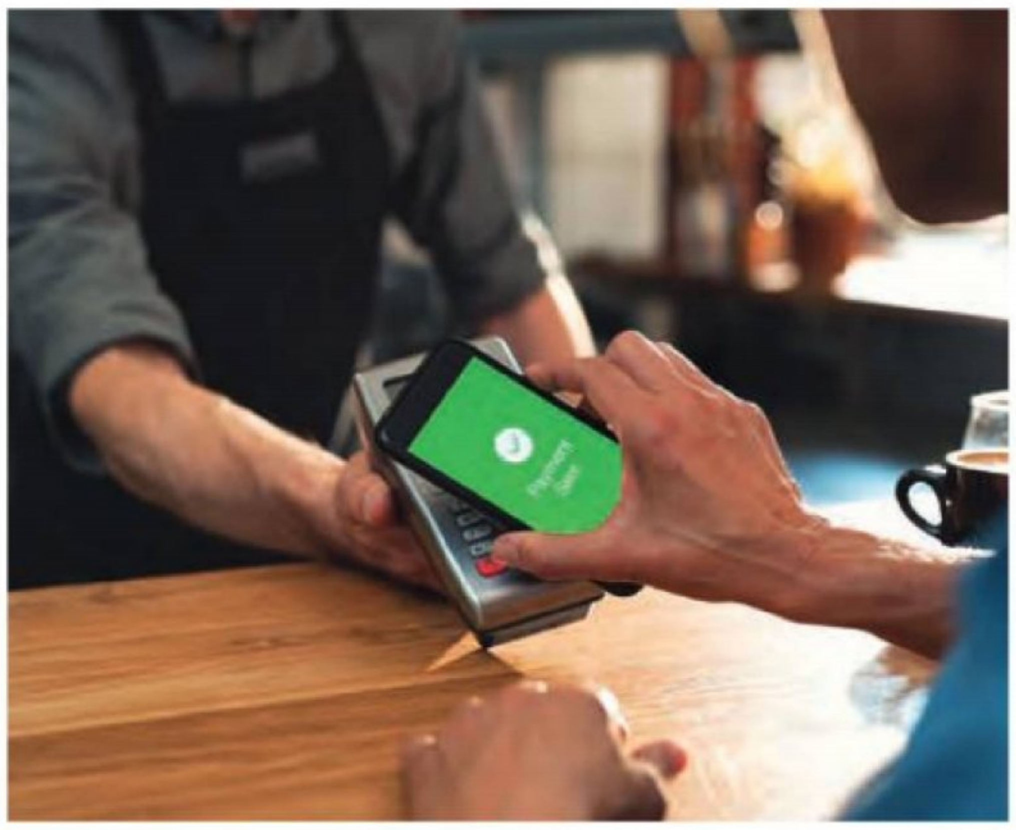
THINK ABOUT APPLIED COMPUTING 7.11

Use the internet to explore creative ways in which people are using NFC tags.

Near-field communication

Near-field communication (NFC) technology is a form of contactless communication between devices such as smartphones and tablets. A user is able to wave their smartphone over an NFC-compatible receiver to send information without needing to touch the devices together or set up a formal connection. In this way it is possible to pay for goods purchased in stores or pay for a parking meter (the parking meter can even send messages to the smartphone indicating how much time is left).

An unpowered chip, called a tag, can be used with an NFC device, such as a smartphone with NFC capability. The tag uses electromagnetic induction to draw its power from the device that reads it. A smartphone can be paired with an NFC tag, which can be programmed by apps on the phone to automate tasks. For example, tapping a smartphone on a smart tag contained in a poster will transfer information from an embedded chip in the poster onto the smartphone. In this way a user tapping a movie advertising poster will receive on their smartphone comprehensive details about the film, such as session times, biographies of leading actors, reviews and more. Tapping on a menu in a restaurant could load the menu into the phone along with nutritional information and cooking notes.



Shutterstock.com / Rido

FIGURE 7.20 An example of near-field communication use for short-range contactless communication

iStock Editorial / Getty Images Plus / CraigRJD

FIGURE 7.21 The myki public transportation ticketing system uses near-field communication to read a commuter's details from their smart card or smartphone.

NFC tags are small and cheap to produce, so are suited for a range of uses involving mobile payments and creative marketing. More interesting uses will emerge as people become more aware of the capabilities of NFC.

PayWave transactions and Victoria's myki transportation system (Figure 7.21) are examples of the use of cards with NFC technology. Waving the card near a card reader allows data to pass from the card to the reader, and hence the transaction is completed with little effort and in almost no time.

Microwaves

Microwaves are radio waves that provide a high-speed signal transmission. Microwave transmission involves sending signals from one **microwave station** to another (Figure 7.22).

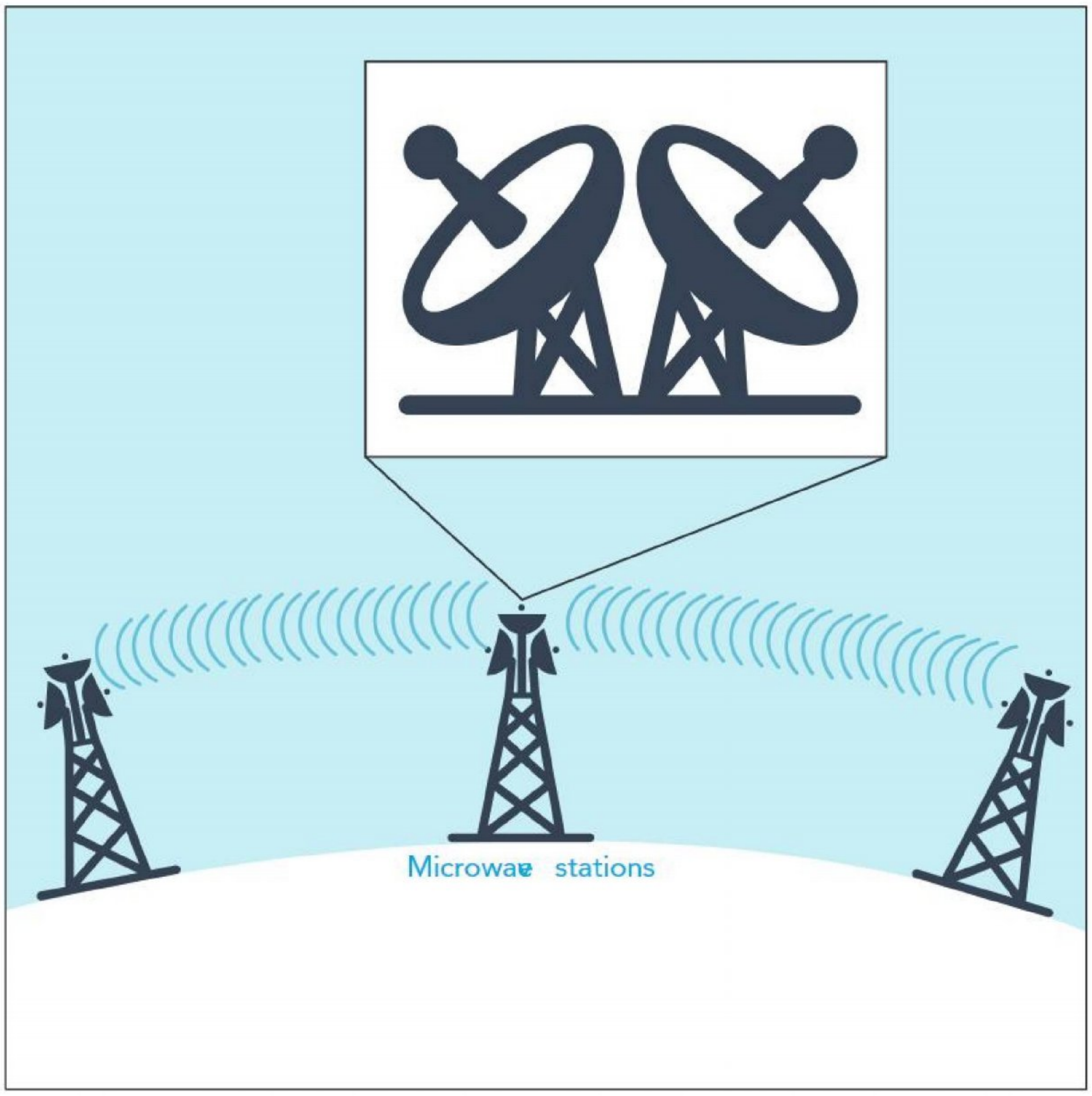


FIGURE 7.22 A microwave station is an Earth-based reflective dish that contains the antenna and other equipment necessary for microwave communications; the dish collects the signals and redirects them to the central collector.

Microwaves use line-of-sight transmission, which means that microwaves must transmit in a straight line with no obstructions between microwave antennas. To avoid possible obstructions, such as buildings or mountains, microwave stations often sit on the tops of buildings, towers or mountains.

Electromagnetic radiation, such as light and radio waves, travels almost as fast through the air as it does through a vacuum (about 300 000 km per second). This means that microwave communication is significantly faster than fibre-optic transmissions, which send laser light

There are four types of NFC tags. Type 1 tags store 96 bytes and operate at 106 Kbps. The biggest and fastest tag is Type 4, which can store up to 32 KB and transmit at 424 Kbps. Type 1 and 2 tags can be written to multiple times, and can use encryption, which permanently locks them so the data cannot be manipulated. Type 3 and 4 tags can be written to once only.

Microwave transmissions are often a viable alternative to cabling where an organisation has premises on different sides of a major road. The cost of digging a tunnel under the road can be expensive and repairs are difficult if there is a break.

An organisation is likely to prefer a licensed microwave link (operating at frequencies between 7 GHz and 42 GHz) if the data being transferred is 'mission critical', if the connection is long-distance or it operates in a high-density area where interference is a problem at lower frequencies.

pulses down glass strands. The glass slows the light beam by 30 to 40 per cent. Microwave transmission is used in environments where installing physical transmission media is difficult or impossible, where the organisation that will be using it occupies a large site and where line-of-sight transmission is available.

Although satellite internet connections are more expensive than cable internet, they are still the only high-speed option in remote areas.

Communications satellites

A **communications satellite** is a space station that receives microwave signals from an **Earth-based station**, amplifies (strengthens) the signals and broadcasts the signals back over a wide area to any number of Earth-based stations (Figure 7.23). These Earth-based stations are often microwave stations. Other devices, such as hand-held computers and GPS receivers, can also function as Earth-based stations. Transmission from an Earth-based station to a satellite is an **uplink**. Transmission from a satellite to an Earth-based station is a **downlink**.

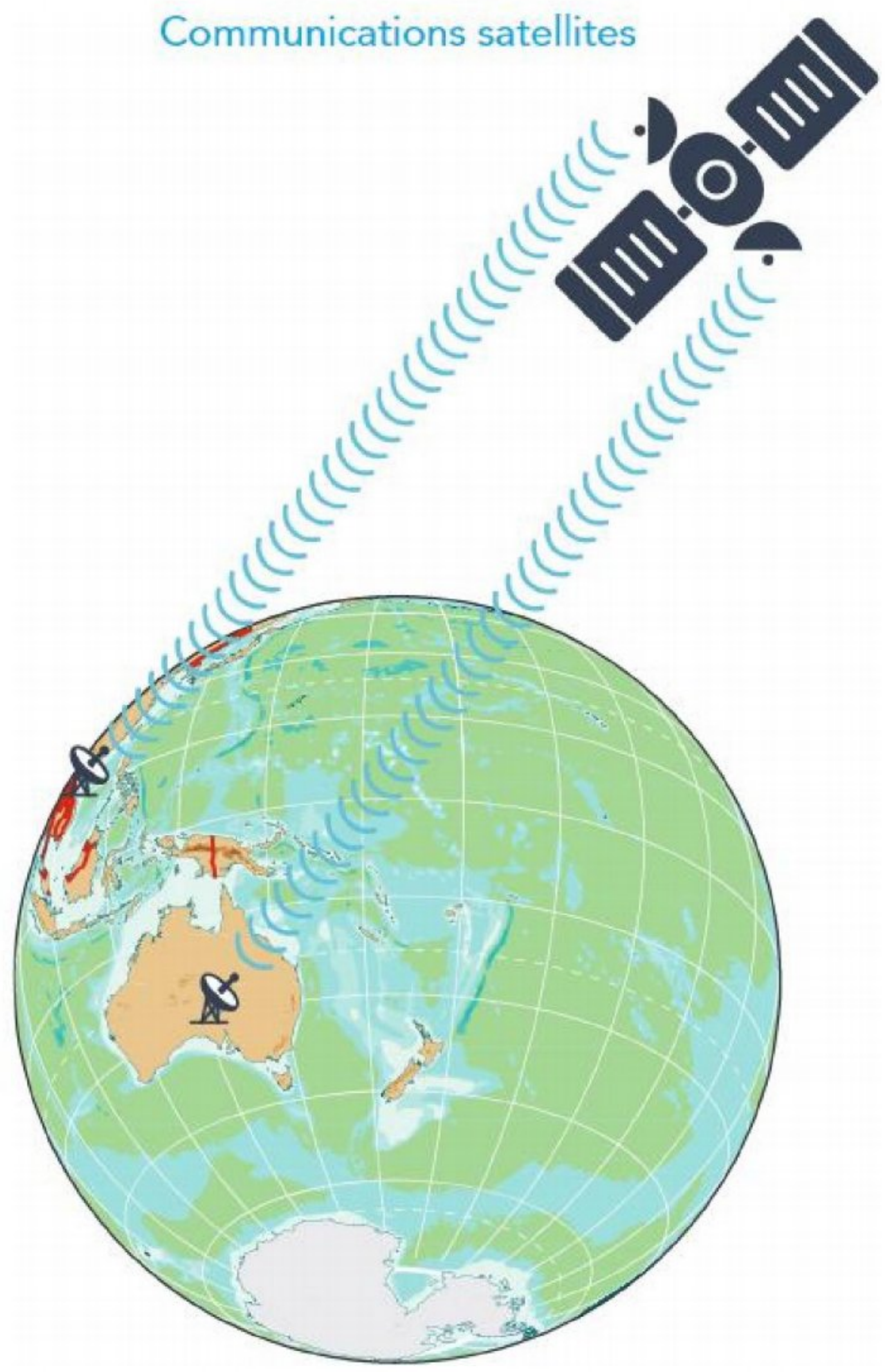


FIGURE 7.23 Communications satellites are placed approximately 37 000 kilometres above Earth's equator.

Mobile communication technology

A **mobile network** (also known as a cellular network) uses telecommunication networks to allow users to communicate using their mobile device. A mobile network consists of a number of mobile phone towers (or base stations), which send and receive signals. Data coverage of these mobile phone towers often overlaps, allowing users to stay connected to the network as they move around an area.

THINK ABOUT APPLIED COMPUTING 7.12

There are areas where the coverage of mobile phone towers does not overlap. What are the consequences for a user who is in an area where they are not in reach of a mobile phone tower?

The range of each mobile phone tower can vary depending on its location and the number of other towers located in the area. In large cities, each tower may only have a short range (e.g. 2 or 3 km), but often these towers overlap with numerous others. In more isolated areas, the tower may have a larger coverage area (e.g. approximately 10 to 50 km, depending on the terrain).

A user's mobile device will connect to the nearest base station. Each base station is then connected to a digital exchange where the communication is sent over a wired network. Mobile networks allow the user to send and receive voice, data, images and text messages. As most (if not all) mobile networks belong to a telecommunications organisation, there is usually a cost associated with transmitting data using these networks.

Internet of Things

Chapter 5 introduced the **Internet of Things (IoT)**. The Internet of Things is a network formed by 'smart devices' such as mobile phones, wearable devices, headphones, heating systems, refrigerators and other devices that have an 'on-off' switch to the internet, and/or to other IoT devices.

There are a number of ways in which these devices can communicate data with each other.

Device-to-device

Device-to-device communication occurs when two devices communicate directly with one another, independently of any other device. These devices communicate with each other using the same communication protocol – for example, Bluetooth.

Device-to-device communication is commonly used in systems that require the transfer of small data packets between devices at a relatively low data transfer rate. This could be a virtual assistant, smart speakers or light dimmers that send small amounts of information to each other.

> **7.13**
> **THINK ABOUT APPLIED COMPUTING**
> What other IoT devices would communicate device to device?

Device-to-gateway

Device-to-gateway communication involves an IoT device connecting to a 'gateway', which then forwards the signals to another device in the local area. The gateway device may be a router or a smartphone, with appropriate software installed. The router or smartphone usually has internet connectivity to allow data to be sent and received over the internet.

The software used on a smartphone that acts as a gateway may be a software application. This application may incorporate security measures to assist in the protection of data and information communicated, and assists to manage the transfer of data from the device to other devices.

Device-to-cloud

Device-to-cloud communication involves an IoT device connecting directly to the internet (or cloud). Devices that transmit data directly to the cloud require a component that is able to transmit data directly to the internet. This could be via a wired, wireless or cellular connection.

Cloud connectivity is usually used with devices that do not require any human interaction or the ability to communicate with other devices locally; for example, a smart electricity meter that is used to upload to the cloud the amount of electricity consumed in a house each month.

Technical underpinnings of networks

Intranets

An **intranet** is a private network accessible only to users within an organisation.

An organisation's internal intranet is unlike the internet in that a wide range of information and services that are available on it, including calendars, policies, procedure manuals and technical support files, are unavailable to the public. An intranet also allows access to documents required by users within the organisation.

Many intranets have search engines, user profiles, blogs, notifications, and event planning functionality.

An intranet is generally a website-based information system, made up of a number of HTML pages that are stored on a web server (similar to any other website). Users of the intranet access the information using a web browser, such as Chrome, Safari or Firefox.

An intranet will often include a connection to the internet, allowing employees access to information from the web. Intranet pages will often include links to internet sites, with information relevant to the organisation.

An intranet provides the following efficiencies:

- It facilitates communication by allowing employees to work in groups.
- Users can access information faster since data does not need to pass through a router, and loading graphics and images becomes much quicker than on the internet.
- An intranet reduces paper waste because companies are able to move their documents and processes onto the intranet.
- An intranet improves ease of use. Point-and-click technology allows easy access to hyperlinked company documents.

The effectiveness benefits of an intranet include the following:

- An intranet allows restricted access to company information.
- An intranet provides dynamic information. Company documents stored on an intranet can be updated more easily and quickly than hard-copy versions.
- An intranet allows connection across different network platforms. Users of complex networks that employ a number of different operating systems – such as Microsoft Windows, Macintosh and UNIX – are able to communicate easily within an intranet using their browser software. The HTML code used by the web pages is universal across all platforms.
- An intranet makes data more accessible. A user who is authorised to connect to the internal network can access the information stored on an intranet from anywhere in the world via the internet.
- An intranet supports the ability to communicate using audio and video files rather than just text and graphic images.

Internet

The **internet** is a worldwide network in which billions of devices are connected together for the purposes of communication and data exchange.

The internet uses common communication protocols to link the network devices with each other. Two major concepts that allow devices to communicate on the internet are IP addresses and the protocol TCP/IP.

IP addresses

Every device that is connected to the internet must have a unique address. This address is called the IP address. The IP addressing standard is four numbers, each between 0 and 255, separated by full stops.

IP stands for Internet Protocol. In computing, a protocol is a set of rules. The Internet Protocol is a set of rules that are followed to allow devices to communicate over the internet.

An IP address is similar to a street address for a house. When we post a letter, the receiver's address is written on the front of the letter. This allows the mail service to deliver the letter to the correct address. Also, the sender's address is often written on the letter, so the receiver knows who sent the letter. On the internet, an IP address plays a similar role.

There are two types of IP addresses: private IP addresses and public IP addresses.

A private IP address is assigned by the user. There are three blocks of IP addresses that are dedicated for private use, usually ranging between 192.168.1.1 to 192.168.1.254. A private IP address is not globally unique as anyone can use the IP address within their own private network.

7.14
THINK ABOUT APPLIED COMPUTING

Open the command prompt on your device and input the command **ipconfig** to display your private IP address.

A public IP address is a globally unique IP address that is assigned by an ISP. The ISP assigns a public IP address to a network device connected to the internet either dynamically or statically.

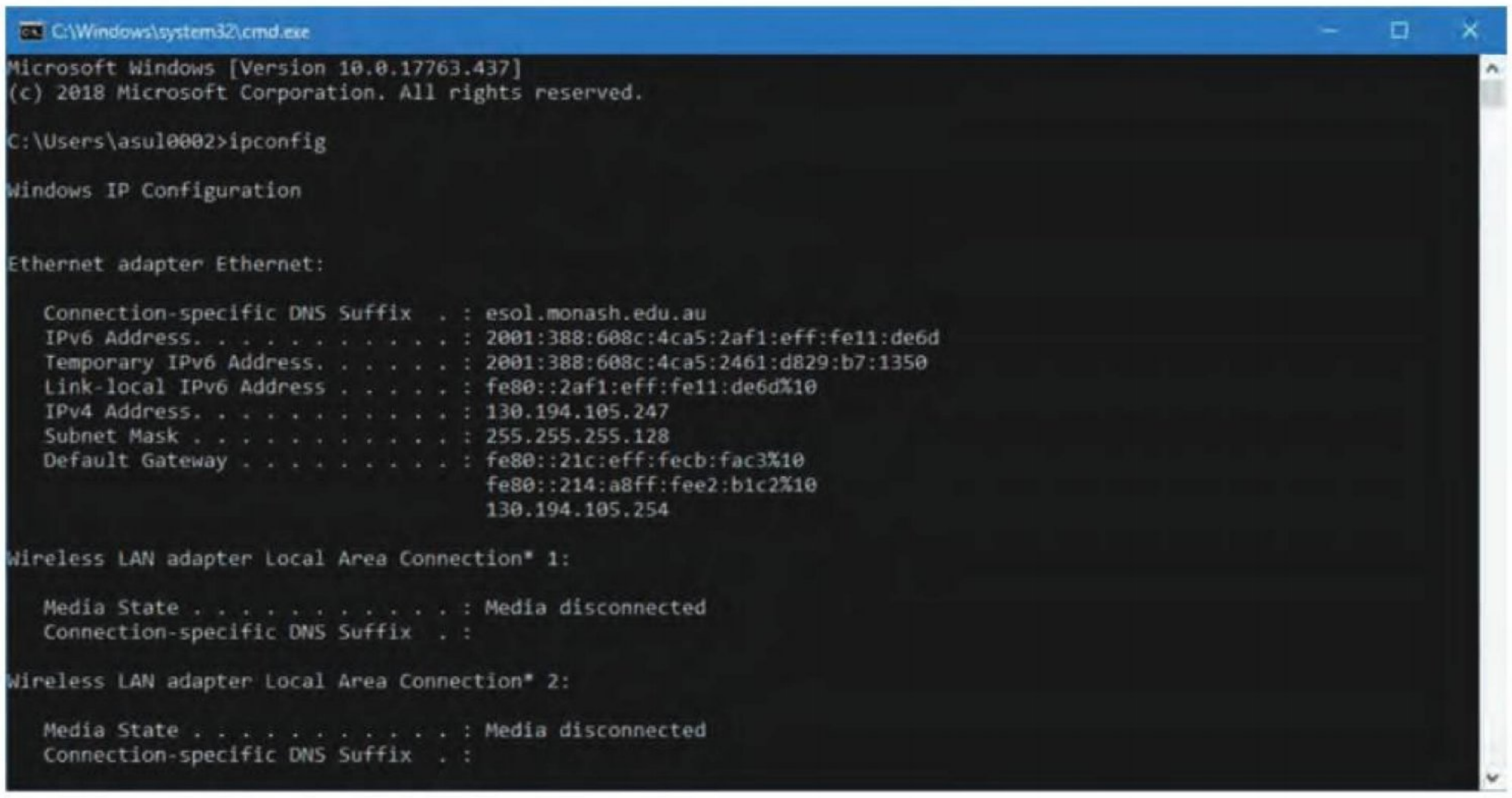


FIGURE 7.24 Screenshot of console window showing a private IP address

> **7.15 THINK ABOUT APPLIED COMPUTING**
>
> Use a website such as What Is My IP? to identify your public IP address.

What Is My IP?

The protocol IPv6 offers a larger range of IP addresses than earlier versions of the protocol.

TCP/IP

The **Transmission Control Protocol/Internet Protocol (TCP/IP)** is a set of rules that allow communication between two networked devices.

TCP/IP is a combination of two protocols (TCP and IP). It ensures that the messages travelling over the internet reach the destination IP address. TCP/IP can also be used on Local Area Networks (LANs).

Transmission Control Protocol

When data is sent over the internet, TCP:

- converts the data into packets
- sends each packet towards the destination
- arranges the packets in the correct order when they arrive
- reassembles the packets back into the correct format at the destination.

TCP is also responsible for resending any packets lost during the communication process.

Internet Protocol

IP is responsible for addresses and the routing of the packets to the correct destination. It uses packet switching to do this.

Packet switching

When data is sent across the internet, TCP breaks it down into small packets. When IP directs each packet across the internet, it is possible that each packet may travel by a different route to the destination.

As each individual packet is sent, IP identifies the most efficient route to the destination at that time. As the next packet is sent, the most efficient route is recalculated, which improves the data transfer speed of sending data over the internet.

Accessing websites

A website consists of a number of web pages that can be accessed over the internet by entering the site's domain name or internet address. All the resources of the website (e.g. pages, files, multimedia content) are stored on a web server. Behind each domain name is an IP address that directs the request to the location of the web browser where the resources are stored.

Virtual private network (VPN)

A **virtual private network (VPN)** allows a user to use public networks such as the internet to send and receive data and information as if they were using a private network, such as an intranet.

Virtual private networks were developed to allow users to access an organisation's applications and resources securely from remote locations. Security is maintained by employing protection strategies such as requiring users to authenticate their identity with passwords or similar techniques when they attempt to log in to the VPN.

A VPN can also be used to avoid geo-restrictions and censorship, and to protect users' anonymity and conceal their location while they are surfing the internet.

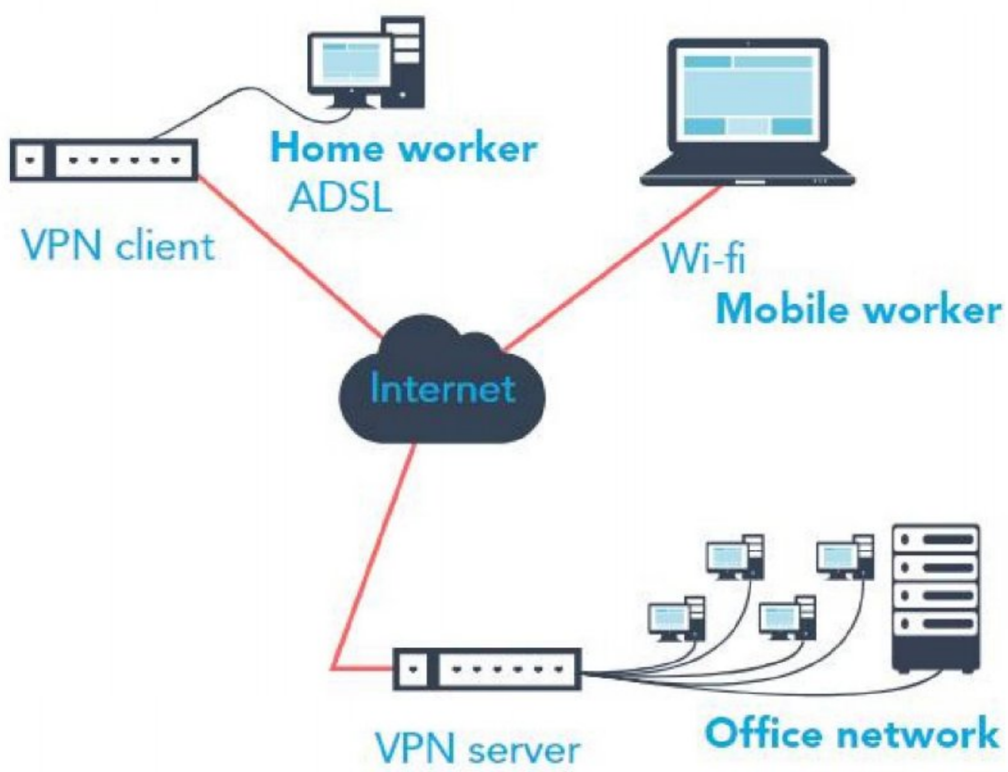


FIGURE 7.25 A virtual private network allows remote workers to connect to the office network.

Network diagrams

A common design tool used to represent the appearance of a network is a layout diagram. A layout diagram can be used to create a network diagram that allows a topology view of a network.

A **network diagram** is a visual representation of a network that shows the nodes that make up a network and how they interact (Figure 7.26). This can include devices such as desktop, laptops and mobile devices, network hardware including servers, routers, switches and modems, and communication methods including wired, wireless and mobile communication. Security controls used, including authentication techniques, firewalls and system protection software and protocols, can also be included or annotated on the diagram.

Note that Figure 7.26 is not based on the plans of the building. This would make the diagram cumbersome and difficult to read. The physical buildings are not important in a network diagram, but it is necessary to identify work areas, such as the dispatch department and the marketing department in this particular example.

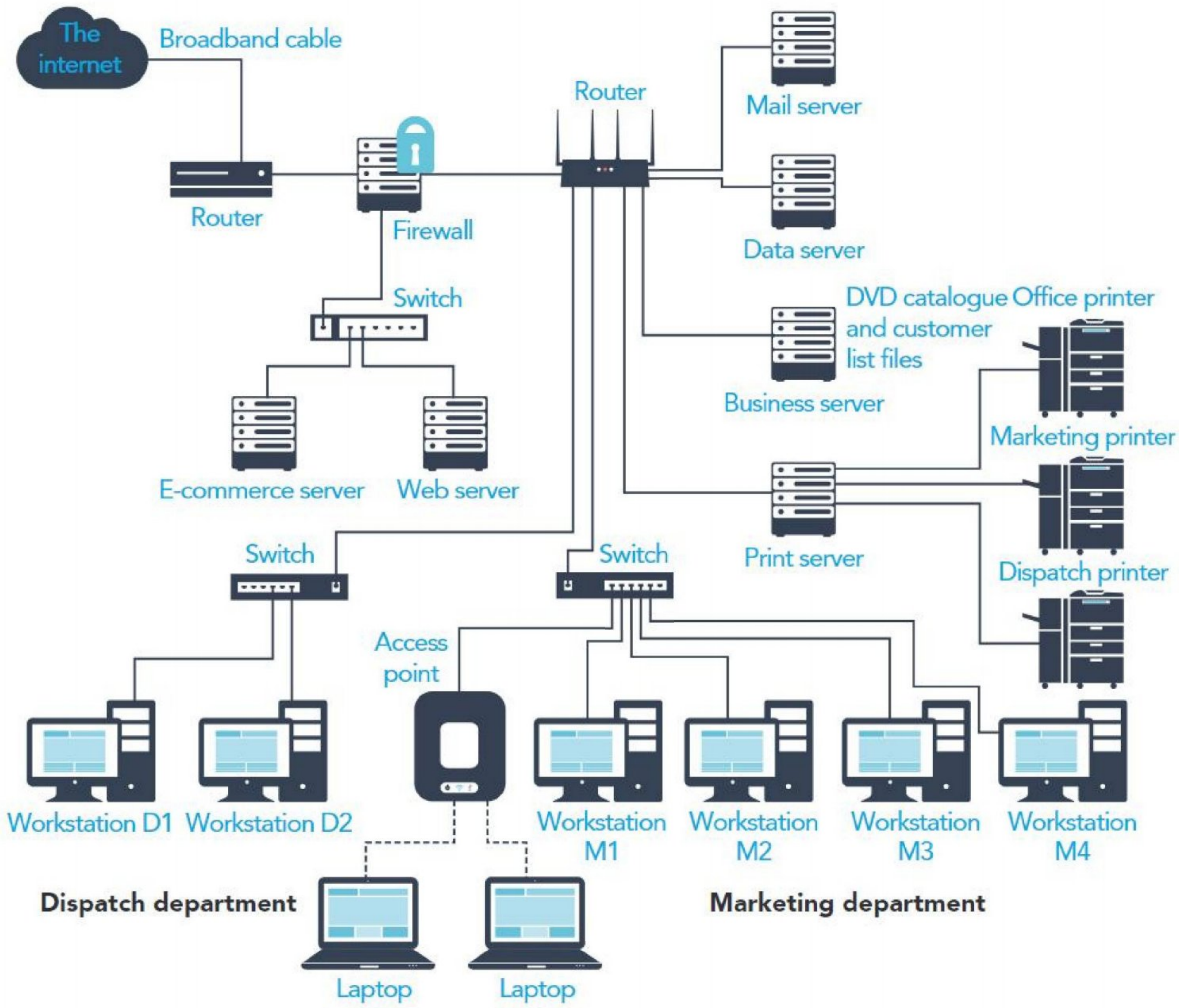


FIGURE 7.26 Network diagram

Security threats

A **threat** to a network is anything that has the potential to cause harm to data and information stored and communicated between information systems. A threat may or may not occur, but it has the potential to result in the loss, theft or damage of data and information. A number of measures can be used to minimise the chances of a security threat.

Credential management

Credential management involves checking a user's credentials to determine whether they are the person they claim to be. This is an important element of the management of data and networks. Authentication techniques such as usernames and passwords, biometric security and swipe cards are used to reduce the chance of unauthorised users gaining access to data and information. Additional methods to authenticate a user's identity can include asking for personal details such as date of birth, driver's license number or residential address, or a combination of them. Online systems can also use these methods to ensure a user is who they claim to be. Users may forget their username and password (some do so frequently) and require them to be reset. **Two-factor authentication** (using two forms of identification) is becoming increasingly common online.

Each of these methods needs good management if it is to remain effective. Staff turnover within organisations may require system administrators to be constantly adding or removing users from network access. Staff may move between roles, and may require changes to their

> **7.16 THINK ABOUT APPLIED COMPUTING**
>
> How does your school/institution manage user access once students complete their VCE studies?

levels of access rights or permissions as a result. Users of online services can change address and personal circumstances. Strategies need to be in place to verify that the person who asked for a password to be reset was in fact the user.

Malicious software

Malware (short for 'malicious software') is software that is designed to damage, disrupt or gain unauthorised access to an information system. The term malware is used to group together a range of software threats to data and information.

Each type of malware functions differently. Below is a brief description of the functionality of some common forms of malware.

Viruses

A virus is a software application that is written to cause some type of negative effect on a device or network. When a virus is executed, it normally duplicates itself in a range of locations on a device. A virus may be designed to damage, steal, modify or corrupt data.

Worms

A worm is software that, once installed, self-replicates. Often, the damage a worm does arises because the worm continues to replicate itself over and over again, causing the performance of a device or network to slow down as the software starts to drain all of the system resources.

Some worms also contain a 'payload' that may cause some type of damage to either hardware or software.

Spyware

Spyware is software that collects data about a user's activity and sends that data to another location without the user's knowledge. Spyware can also include functions that allow network or security settings on a user's device to be changed. Like other malware, spyware can be downloaded inside a Trojan horse (a type of virus) or after clicking on a link in an email.

Denial-of-service attacks

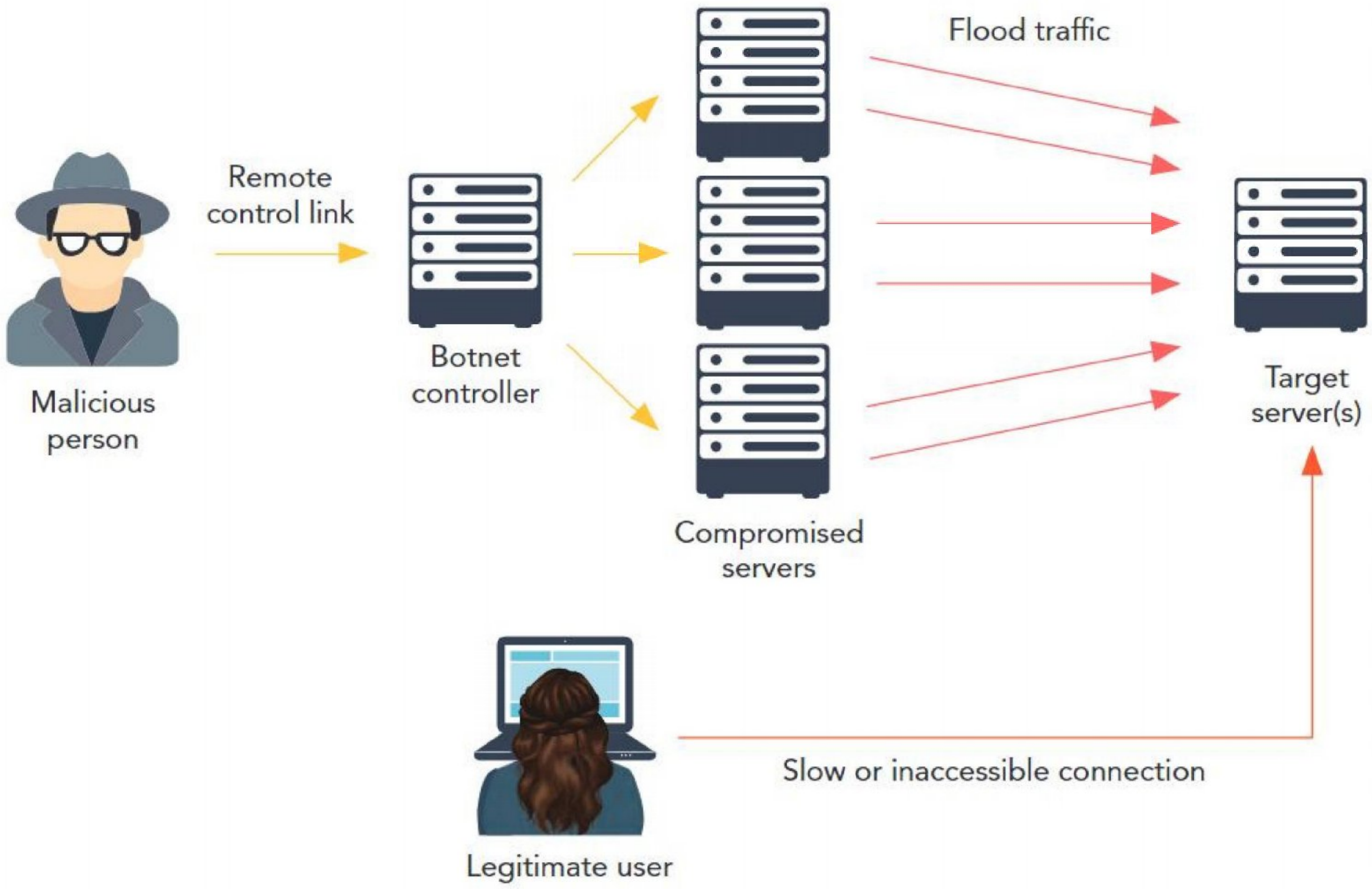
A **denial-of-service (DoS) attack** occurs when hackers attempt to prevent legitimate users from accessing a service from an information system or network.

A DoS attack usually takes the form of a very large number of messages, which ask the network or server to authenticate requests, but that appear to be sent from non-existent addresses. When the network or server responds by sending authentication approval, it is unable to find the invalid return addresses. It keeps trying, causing the server to wait before closing the connection.

When the server finally closes the connection, the attacker sends more authentication requests, again with invalid return addresses. The network or server is kept busy, trying to respond but unable to find the return addresses. This prevents legitimate users from accessing the services.

Hackers may also send fake data packets to every computer on a network instead of only one, or may carry out a SYN (synchronisation) flood, in which a server connection request is sent to every open port but the acknowledgement process or 'handshake' is never completed, which leaves no ports open for legitimate users.

Distributed denial-of-service (DDoS) attacks (Figure 7.27) are similar to DoS attacks except that they originate from many sources instead of one. This increases the effectiveness of the attack as the amount of traffic on the network is increased significantly. It also makes it much more difficult to shut down the attack and to identify its source.



Adapted from https://blogs.msdn.microsoft.com/mlserver/2017/07/11/understanding-server-traffic-logs-and-detecting-denial-of-service-attacks/, Madhura Raju, Microsoft

FIGURE 7.27 Visual representation of a DDoS attack

Username and password

A combination of a **username and password** is one of the most widely used security controls. Each user of an information system is allocated their own unique username, while, in most cases, a user is able to create their own password. Collectively, a username and password is often referred to as the login.

Some information systems require users to create a password that meets particular requirements. Commonly, for example, the password:

- must consist of at least eight characters
- must contain at least one upper-case letter
- must contain at least one lower-case letter
- must contain at least one symbol
- must not contain spaces
- cannot be 'password'
- cannot be the same as the username.

Short or single-word passwords are a threat to security. These are vulnerable to brute force attacks. Obvious passwords such as *password*, or users who use the same password for multiple information systems, are particularly susceptible.

A **brute force attack** involves making repetitive attempts to access an information system using a variation of possible usernames and/or passwords. Often an attempt will be based on an established username; for example, an email address.

A bot can be developed that will attempt to guess the password that matches the username. On the first attempt the password input could be 'a', then 'b', then 'c', right through to 'z', then the bot will start over again but this time at 'aa', then 'ab', 'ac', etc.

TABLE 7.2 Password protection: the time it takes to discover a password

Password		Average time to discover[b]	
Number of characters[a]	**Possible combinations[b]**	**Human[c]**	**Computer[d]**
1	36	3 minutes	0.000 018 second
2	1300	2 hours	0.000 65 second
3	47 000	3 days	0.02 second
4	1 000 000	3 months	1 second
5	60 000 000	10 years	30 seconds
10	3 700 000 000 000 000	580 million years	59 years

a Possible character combinations include the letters A–Z and numbers 0–9.
b Average time assumes the password would be discovered in approximately half the time it would take to try all possible combinations.
c Human discovery assumes one try every 10 seconds.
d Computer discovery assumes 1 million tries per second.

Access rights

Access rights are a form of permission that has been granted to a user, or to an application, to read, write and delete files on a network.

By failing to apply access rights, or by not managing access rights correctly, an organisation will fail to restrict the number of users that have rights to access data and information. This increases the chance of both user error and deliberate tampering with the data and information.

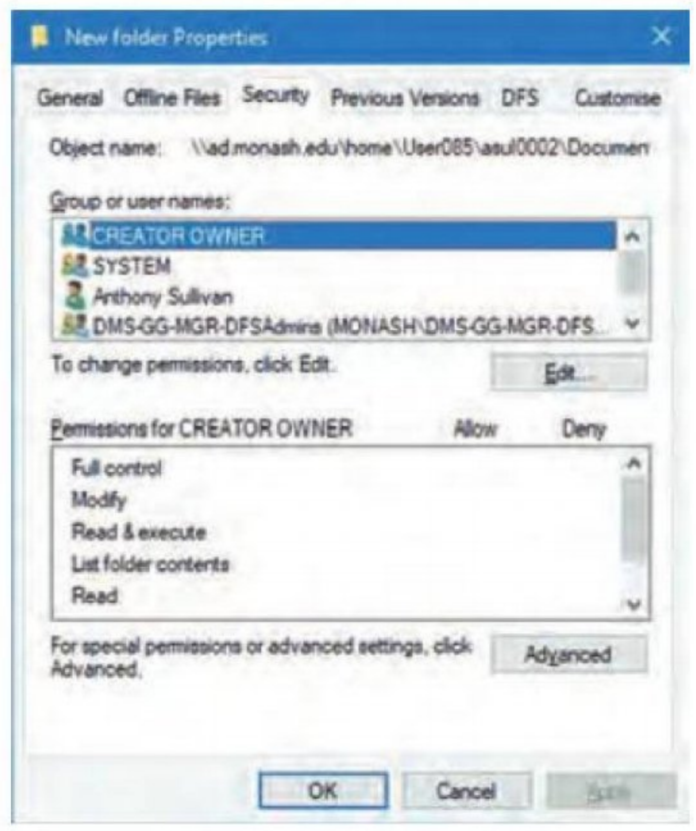


FIGURE 7.28 Folder properties showing access rights

Out-of-date software

A **software update**, sometimes referred to as a patch, is generally a small file that is downloaded and that fixes any bugs, improves any functionality issues, makes minor changes to the user interface or addresses any security issues with existing software applications installed on a device or network. Operating system updates are a common example of a software update. These updates attempt to address any newly identified security threats that may make a digital system vulnerable to hackers or malware. Smartphone applications rely on regular software updates.

A **software upgrade** is a new version of the software. An upgrade generally involves a large number of significant changes to the application, including the addition of improved security and functionality and significant changes to the user interface. Upgrading from Windows 8.1 to Windows 10 is an example of a software upgrade.

One method used to distinguish a software update from a software upgrade is the use of a version number. Version numbers generally follow a convention whereby they are made up of three numbers in the form 'X.Y.Z'.

The first number (X) indicates the version number, and this changes when significant changes have been made to the software. The second number (Y) indicates a minor version upgrade where small changes have been made to functionality, while Z indicates an update addressing bugs or errors in the software. There are a wide range of version numbering conventions used.

Failing to update or upgrade software can lead to the following issues:

- vulnerability against newfound security risks
- missing out on new functionality of the software
- lower battery performance or slower processing times
- lower productivity
- decreased functionality related to the use of software that contains bugs or errors.

Data and network protection strategies

It is important to use a variety of security measures to prevent unwanted access to networks and data.

Encryption

Encryption is a security control that is used when communicating data and that can also be used to protect stored data. **Encryption** involves encoding or changing data so that it can only be decrypted by the receiver if they have the correct **key**. An unauthorised user might gain access to the data and information, but without the correct key they would not be able to understand it.

A credit card number used to purchase goods online is an example of data that would benefit from encryption.

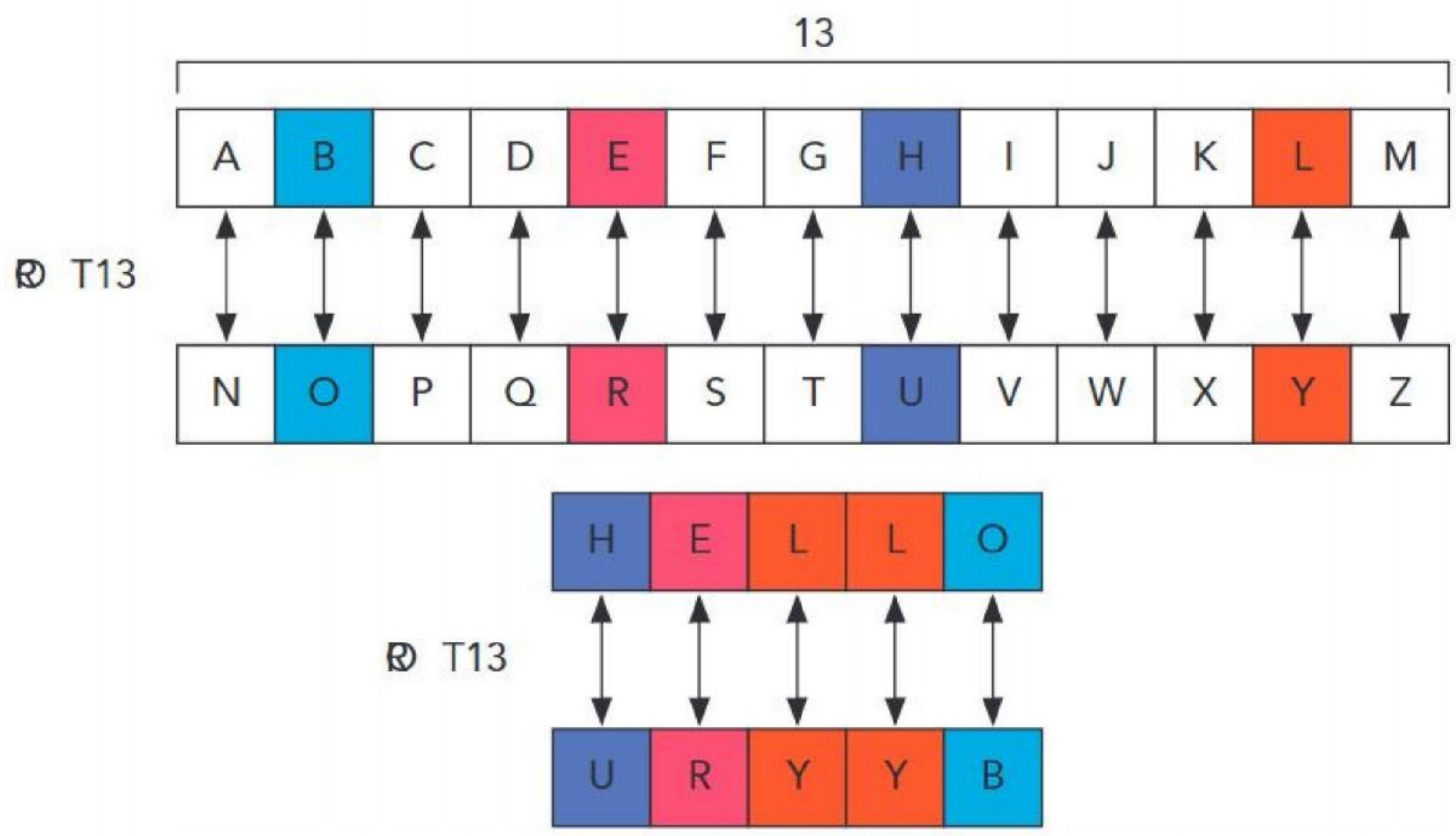


FIGURE 7.29 A basic example of a method used to encrypt data. The alphabet is broken into two halves. Each letter is paired with another letter. When data is encrypted, each letter in the data is replaced by its pair. The data 'HELLO', once encrypted, would be transmitted or saved as 'URYYB'.

7.17 THINK ABOUT APPLIED COMPUTING

1. How would the data 'YESTERDAY' be stored after it was encrypted using the method shown in Figure 7.29?
2. How would the data 'CNFFJBEQ' appear after it was decrypted?

Private and public keys

To be able to encrypt (and decrypt) data, a key is needed. A key can be used when there is data that requires protection. A key is a series of bits (0s or 1s in binary code) that is used to apply an algorithm to the data packet to scramble (encrypt) and unscramble (decrypt) the data. In other words, a key is used to 'lock' and 'unlock' the data. Encryption software is used to create a key.

A key is a long random number, such as:

4356 0241 00C9 18DF CF8D EB2D EFD5 FD37 89B9 E069 EA97 FC20 5E35 F567 RE31 C4FB C6E4 4811 7D86 BC8F BAFA 362F 922B F05R 2F88 C133 2654 C0DR 2881 D673 BN2B 4673 C266 E2CD CB02 0301 3451

Common key sizes include 128, 192 and 256 bits. The larger the key, the more secure the encryption.

Two types of encryption are symmetric and asymmetric encryption. **Symmetric encryption** requires both the sender and the receiver to use the same private key. A **private key** is a key that is only known by a user who has the key installed on their device.

The sender uses the key to encrypt the data packet, and the receiver needs the same key to decrypt that data packet. A private key is also referred to as a secret key.

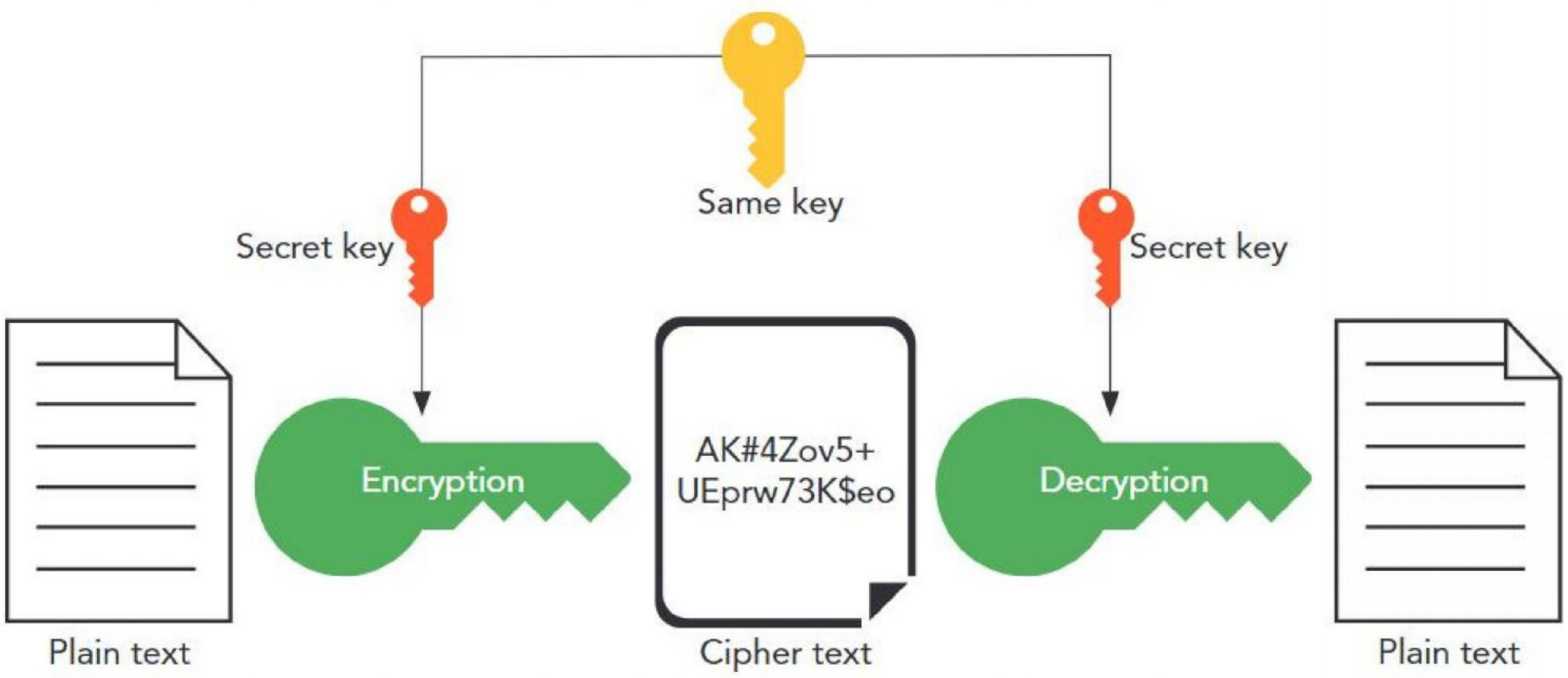


FIGURE 7.30 Visualisation of symmetric encryption

Asymmetric encryption requires the user to install two keys on their device: a private and a public key. A **public key** is a key that is made available to other devices and is used to encrypt data packets that are being sent to the user, while the private key is used on the receiving device to decrypt the data.

The algorithms used by the two keys are related, so that data encrypted by a public key can only be decrypted by a private key.

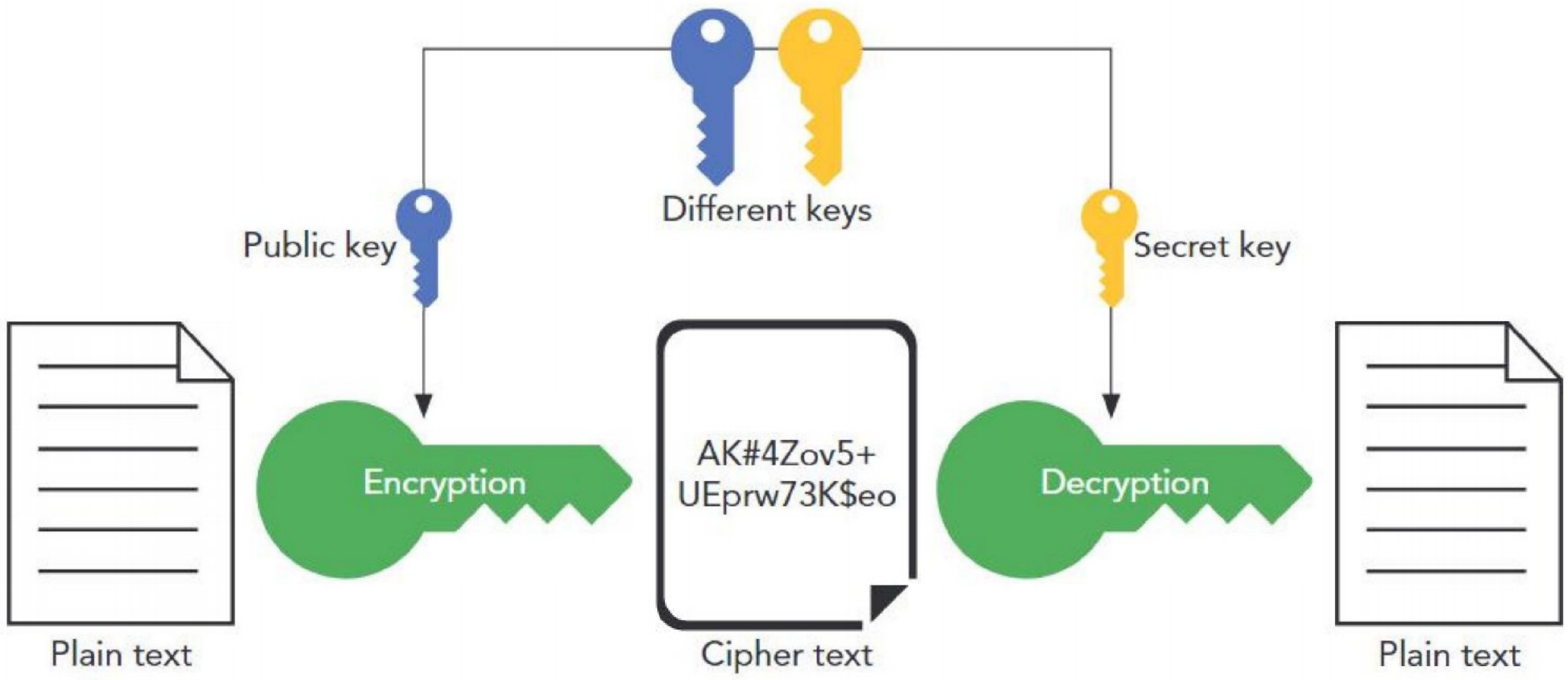


FIGURE 7.31 Visualisation of asymmetric encryption

Authentication techniques

As already discussed, **authentication** involves confirming or verifying that a user of an information system is the person they claim to be. Usernames and passwords are discussed on page 311. Other authentication techniques are discussed here.

Biometrics

Biometric authentication is an authentication technique that uses an individual's unique biological characteristics to verify that they are who they claim to be.

Types of biometric authentication include face, fingerprint, iris, signature and voice recognition.

Biometric authentication requires users to initially submit relevant biological data, which is then stored in a database. Then, when the user attempts to access a room, building or digital device, they are required to input their biological data again. The input data is then compared with the data previously stored in the database. If both sets of biometric data match, authentication is confirmed.

Biometric authentication is becoming far more widely used as the hardware and software it depends on becomes more affordable. It offers greater security than usernames and passwords, but biometric software is still not 100% reliable. Systems can still throw up errors, and an authorised user may fail, or an unauthorised user may pass, the authentication.

Digital signatures

A **digital signature** is an authentication technique used to verify whether an electronic document (such as an email, a data file or a web page) that has been received is genuine. This includes confirming the identity of the sender, ensuring that the sender confirms ownership of the document and checking that the document has not been altered during transmission.

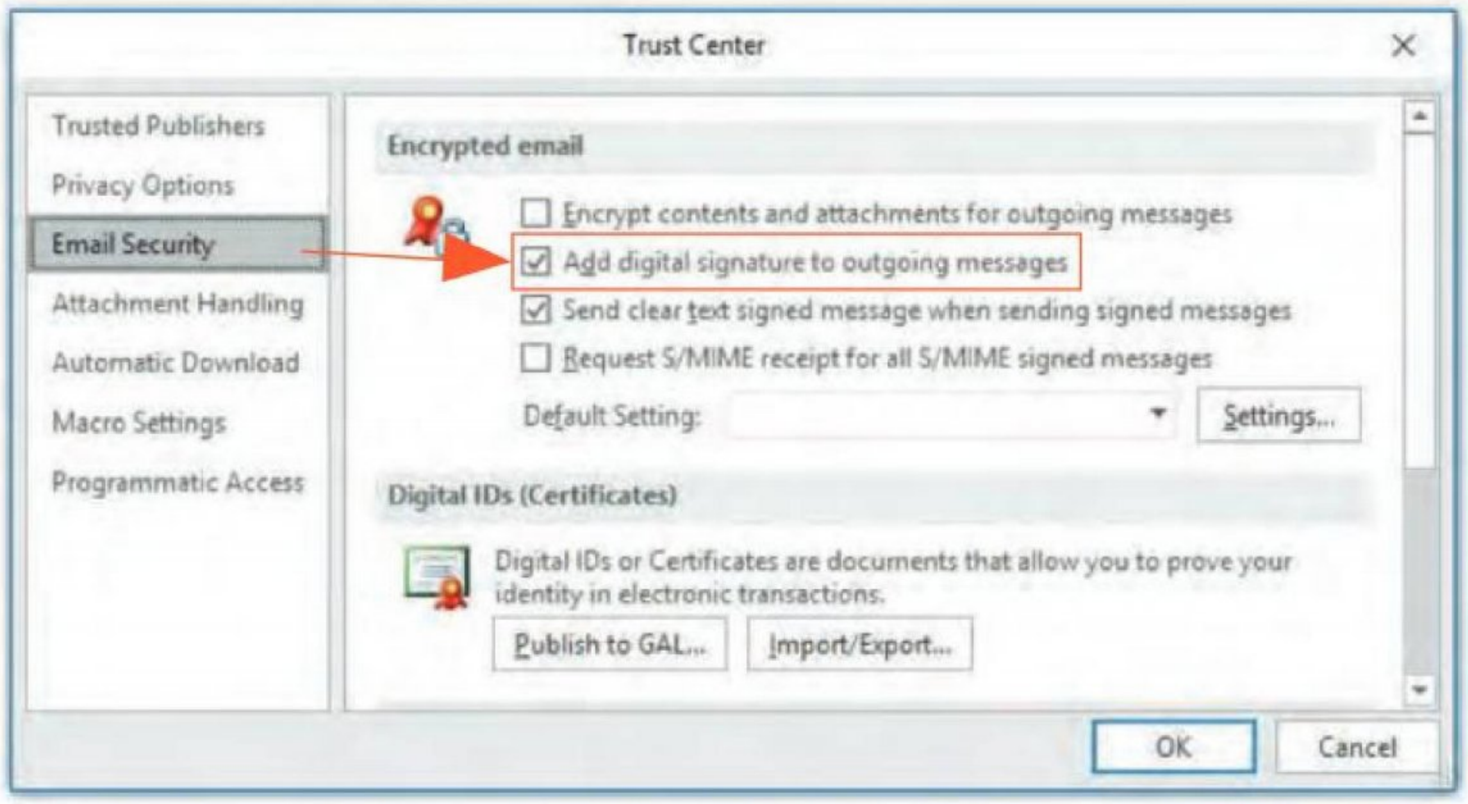


FIGURE 7.32 Settings to enable a digital signature to be sent with an email

Digital signatures use asymmetric encryption techniques, with both private and public keys. In digital signature authentication, firstly, a hashing algorithm is used to generate a value from the data contained within the data packet, then the data packet is encrypted using a private key and transmitted. Once received, the data is then decrypted using the private key, then a hashing algorithm is used to convert the value back into its original form. The combination of a hash value and encryption can create a unique format (the digital signature), which makes it possible to verify the identity of the sender. The hash value can also be used to establish whether the document has been changed, edited or tampered with during transit.

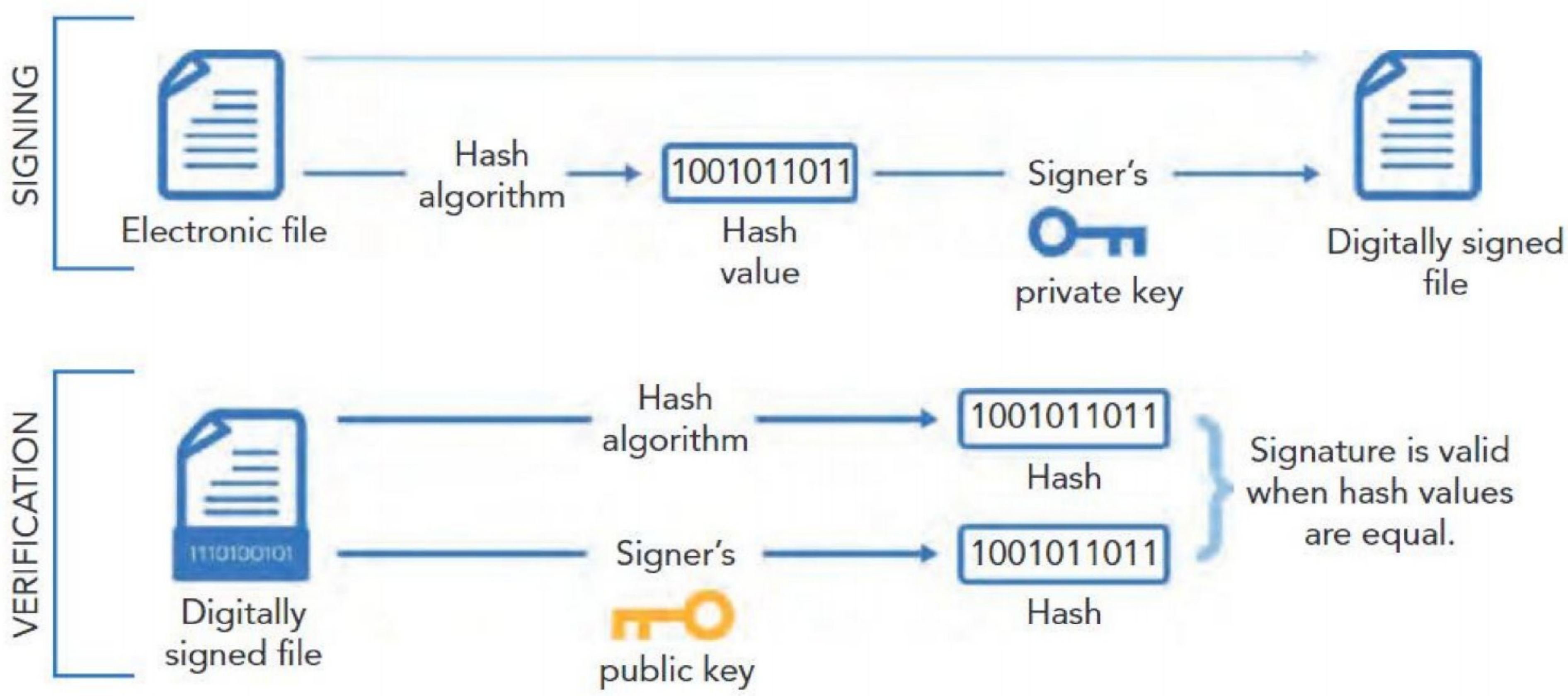


FIGURE 7.33 Flow diagram showing the use of digital signatures

Preventative practices to reduce risks

Preventative practices are intended to reduce the chance of data loss or theft due to security risks. Such practices may include software-based, process-based or hardware-based measures, each attempting to reduce the chance of different types of risks. Like all security measures, preventative practices aim to reduce the risk of data loss or theft, but they cannot absolutely guarantee security.

Software maintenance

Software maintenance involves making changes to software applications, after they have been installed, to fix faults, improve performance or add future functionality to the software. Software maintenance involves making changes to the programming code used to create the application.

Maintenance of software can be required for a variety of reasons. The changes made may be:

- *adaptive changes* – made to the software to adjust to changes in the operating environment
- *perfective changes* – made to the software to improve the functionality of the application
- *corrective changes* – made to fix bugs or errors in the application
- *preventative changes* – made to improve the reliability of the application or reduce the need for future maintenance.

Software updates and upgrades (see pages 312–13) are examples of software maintenance. These files are made available to users via a download from the software developer.

Maintaining software helps to address any new security threats and improves the reliability of the software. This reduces the chance of data loss and improves functionality, allowing users to be more productive when using the application.

Operating system updates

An **operating system update** is a form of software update. Operating system updates are frequently released by organisations such as Apple, Google and Microsoft to improve the usability, functionality and security of an operating system.

When a new version of an operating system is released, users are quick to provide feedback about areas of the operating system that could be improved. As well, cybercriminals and hackers are quick to find vulnerabilities in operating systems, and the developers of the operating system thus constantly need to make changes to the code.

Operating system updates can also include updates to associated applications installed on the device.

Operating system updates aim to improve:

- protection against spyware, viruses and other malware
- protection against cyberattacks
- the overall performance of the device.

Virus definitions

As discussed on page 310, a **virus** is a software application that is written to cause some type of negative effect on a device or network. When a virus is executed, it normally duplicates itself in a range of locations on a device. A virus may be designed to damage, steal, modify or corrupt data.

System protection software contains antivirus functionality: the software will scan a device to detect whether any viruses have infiltrated the device. Antivirus software is used to scan a device or network in search of viruses. Antivirus software needs to be aware of the characteristics of a virus before it is able to identify a virus on a device or network. To do this, antivirus software uses virus definitions.

A **virus definition** is a collection of binary code that is used to identify a particular virus. The definition for each known virus is stored in a database attached to the antivirus software. As the software scans a device or network, it compares each file against the binary code stored in the virus definition database. If a virus is found, the antivirus software will either delete or quarantine the virus.

If the antivirus software is not aware that a virus exists, then it will not be able to detect that particular virus if it is present on the device or network. Since new viruses are being written every day, antivirus software needs to be constantly updated in order to be effective. This often requires the software to download updated virus definitions. For many antivirus software applications, this updating occurs automatically (often daily).

Firmware

Firmware is software that has been programmed onto a hardware device such as a television controller, DVD player, hard drive, mouse or keyboard. The software is designed to give instructions to communicate with other devices in a system. It is semipermanent software: it stays the same unless it is updated. It cannot be uninstalled or removed. Firmware is subject to security threats and needs to be updated as required by the manufacturer.

Disaster recovery plans

A **disaster recovery plan** is a list of procedures and actions for an organisation to follow in the event of a disaster.

The purpose of a disaster recovery plan is to allow an organisation to start to function as normally as possible after a disaster has occurred, and to minimise data loss.

A disaster recovery plan is often written as a series of step-by-step instructions, clearly outlining the procedure to take. Often, a disaster recovery plan will consist of a number of sub-plans, including emergency, backup, recovery and test plans.

Emergency sub-plan

An emergency sub-plan consists of step-by-step instructions regarding what to do in the event of an emergency. It contains information that will be relevant if a disaster occurs, including:

- emergency procedures
- evacuation procedures
- emergency services contact details
- equipment shutdown procedures.

iStock/Getty Images Plus/steved_np3

FIGURE 7.34 An example of an emergency plan for an organisation

Backup sub-plan

The backup sub-plan is different from the organisation's backup strategy or plan, but the two are related. The backup sub-plan in a disaster recovery plan outlines specific instructions for how to restore the backup. This may include:

- a list of locations where the backup is stored
- the type of backup, and transmission time needed to restore the backup
- step-by-step instructions about how to restore the backup
- contact details of key staff who need to be contacted regarding the backup.

Recovery sub-plan

The recovery sub-plan contains instructions for how to recover after a disaster occurs. This can include procedures to follow when:

- restoring power to a building
- replacing damaged equipment
- restarting hardware
- reinstalling software
- restoring the backup.

Test sub-plan

The test sub-plan contains steps to follow to fully test that the disaster recovery plan is working as expected. Often, a disaster drill will be acted out (similar to a fire drill) during which all sub-plans of the disaster recovery plan will be carried out.

During testing, data will be captured about steps that may need to be added to or removed from the disaster recovery plan or otherwise changed.

THINK ABOUT APPLIED COMPUTING 7.18

Research features of the disaster recovery plan that is in place at your school or organisation.

Staff procedures

Staff procedures are steps or processes that staff should follow in a given situation when dealing with data and information. Many organisations require staff to follow a series of procedures related to the use of data and information within the organisation, including procedures relating to:

- leaving devices unattended
- password requirements
- file naming conventions
- acceptable use of the system
- email protocols
- internet usage and downloading guidelines.

Training staff to follow these procedures when using data and information reduces the chance of data loss or threat.

Common signs of intrusion

One threat to devices, information systems and networks is that of unauthorised users gaining access. These unauthorised users, or **hackers**, may have a variety of reasons to attempt to gain access without authorisation: simply for fun, for a challenge, out of curiosity, or to steal data or cause damage.

Signs that a device or network has been exposed to an unauthorised intrusion can include:

- increased use of system resources
- new software installed
- changed passwords
- sending of spam email
- unknown applications requesting access
- system protection software uninstalled
- a device completing tasks by itself
- changes to a web browser home page.

It must be remembered that there could be a wide range of reasons why these behaviours might occur, with unauthorised intrusion being one of them. If any of the signs are present, a range of tests should be applied to attempt to identify the cause of the issues.

Increased use of system resources

For an unauthorised user to gain access to a device or network, they first must create a connection. This connection will increase use of the digital system's resources. Once connected, the unauthorised user will start to complete tasks. This activity will consume further system resources and can affect data transfer rates, processing speeds, memory consumption and graphics performance.

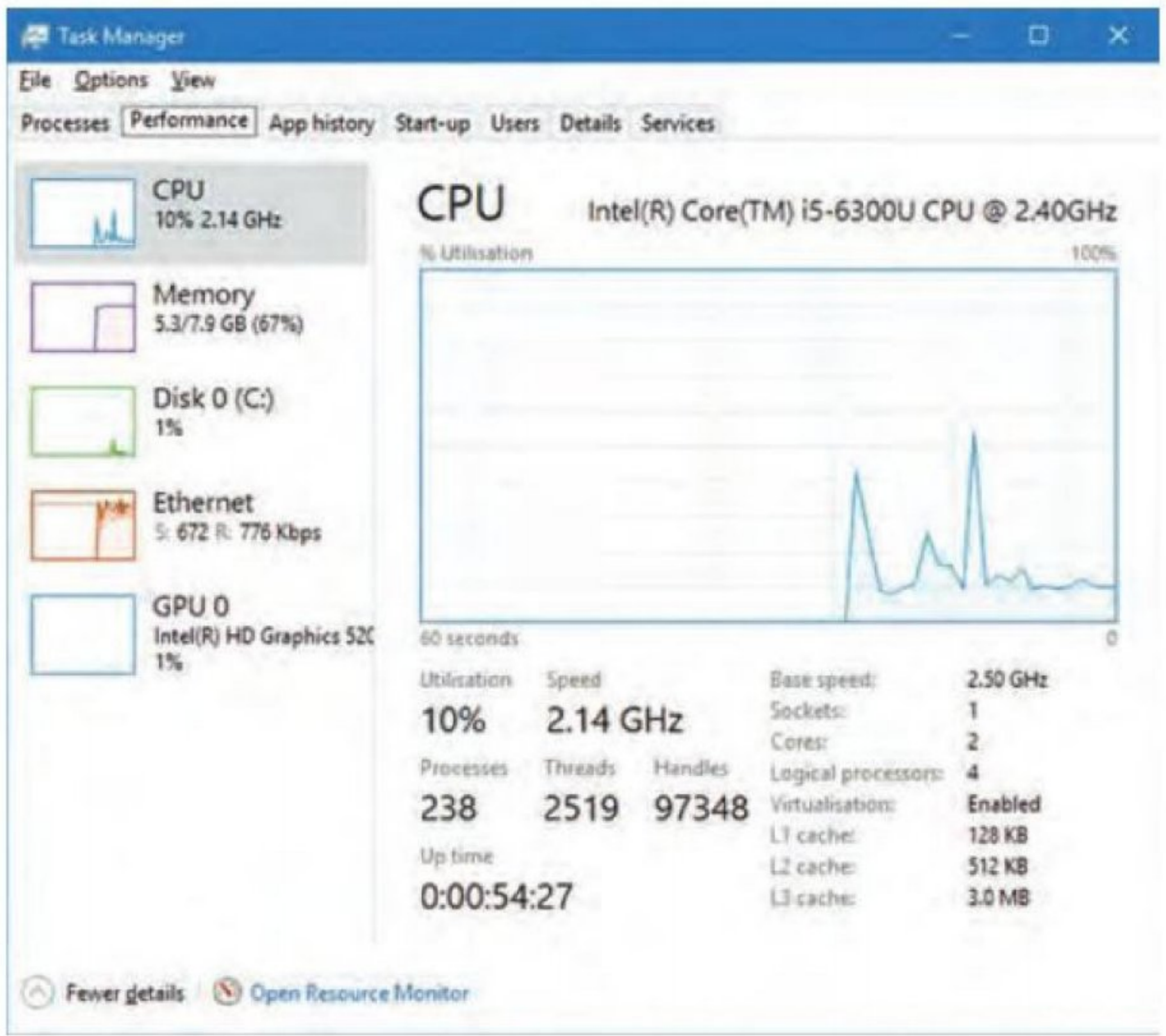


FIGURE 7.35 Task manager showing system resources consumed

New software installed

If new applications have been installed on a device or network without the user's knowledge, this is a strong indication of an unauthorised intrusion. Often, the software installed will contain some elements of malware, such as viruses, Trojans, spyware, adware or keylogging software.

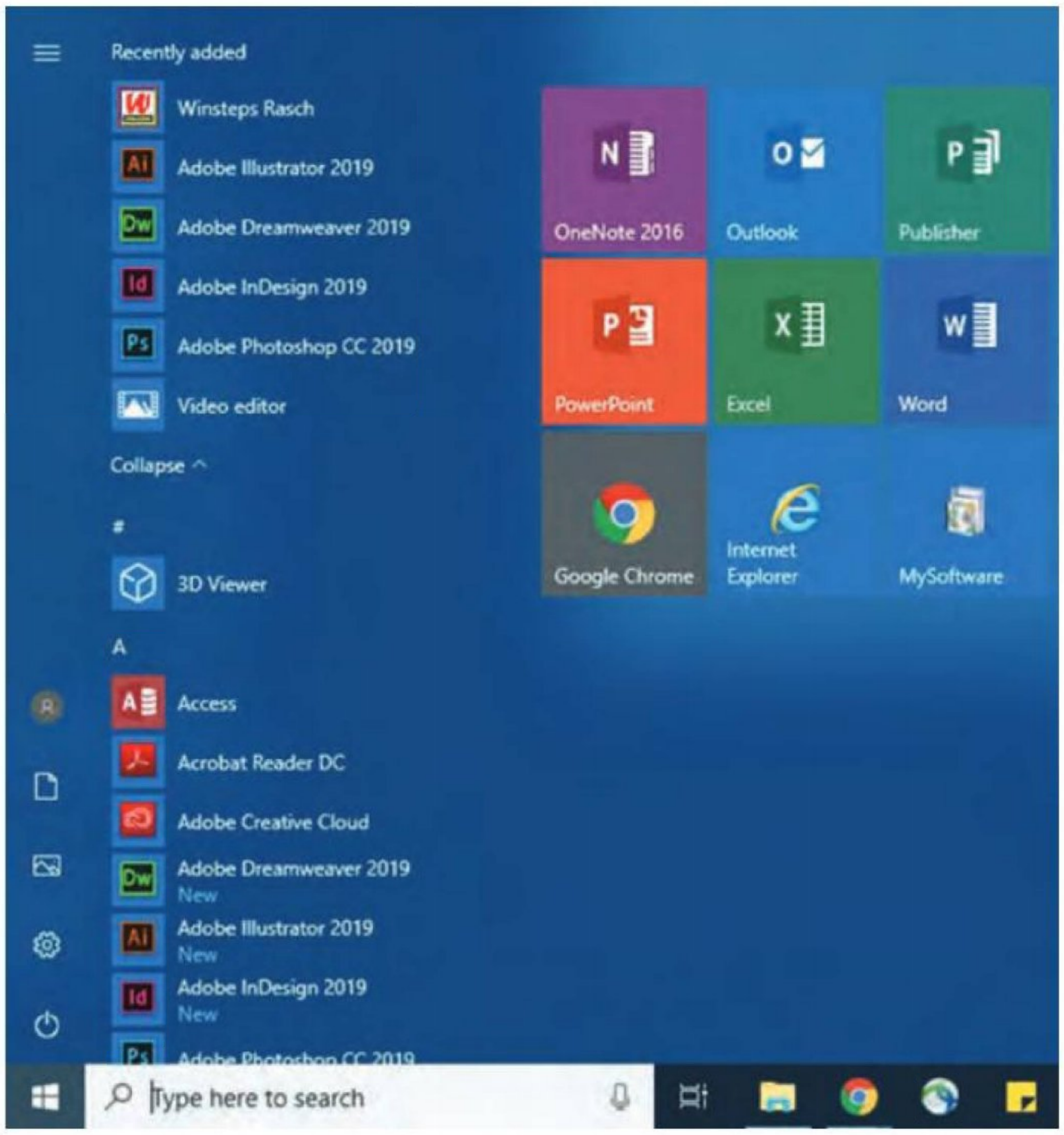
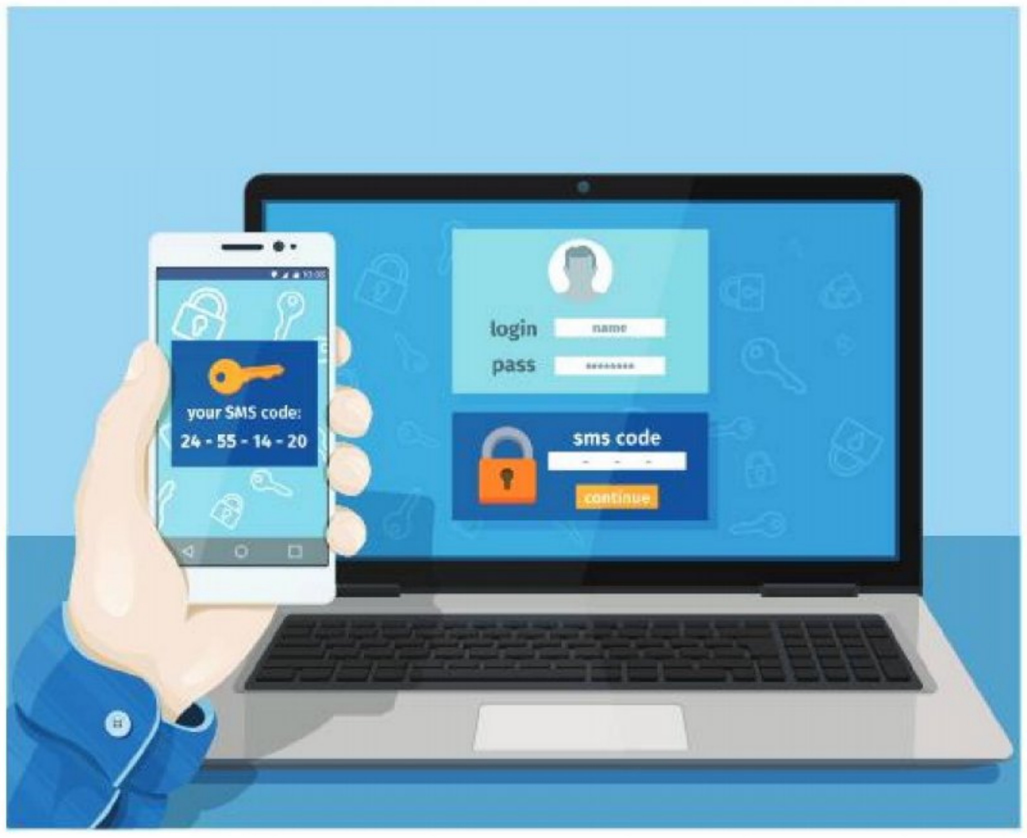


FIGURE 7.36 Task manager showing new software installed on a device

Changed passwords

A common sign of unauthorised intrusion is that passwords have been changed. Those responsible for the intrusion will often change passwords in an attempt to gain control of the digital system for a sustained period of time, allowing them more opportunity to conduct whatever tasks they wish to perform. Many organisations now have two-factor authentication in place to make it more difficult for an unauthorised user to make changes to accounts.



Shutterstock.com/Stocker top

FIGURE 7.37 Two-factor authentication

Sending of spam email

Once an unauthorised intrusion has occurred, it is common for the hacker to gain access to the email system and to start sending emails to every contact in the email address book.

These emails will appear to be sent by the authorised user of the account and are likely to contain some type of malicious payload including **phishing** activities. At the least, the emails will contain some form of **adware** promoting a service or product. The recipients of the email will believe that the email has come from a trusted source, and thus may not be sufficiently vigilant to follow a commonsense approach when dealing with the email. An occurrence such as this can also be damaging for the user who has been hacked, as their reputation among friends and colleagues may suffer.

Unknown applications requesting access

One role of a firewall on a digital system is to scan incoming data packets to ensure the packets have been requested. After an unauthorised intrusion, the firewall may start to detect unrequested data packets that are originating from unknown applications. The unauthorised user may have installed software that is requesting data from external applications. If these applications have not been used before, the firewall may detect this activity.

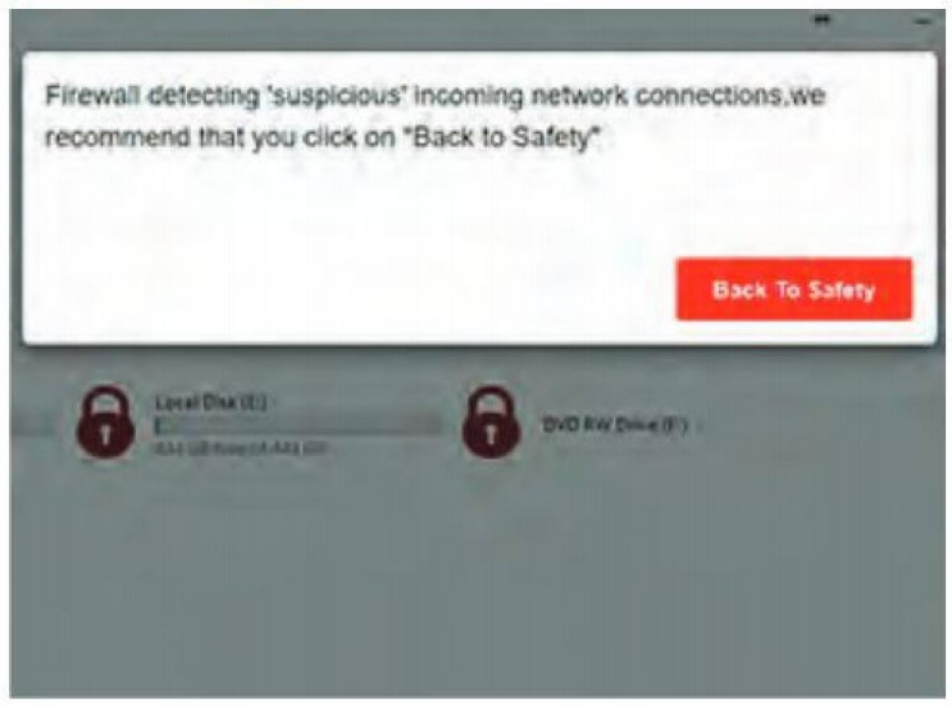


FIGURE 7.38 Message alert from firewall

System protection software uninstalled

If it is discovered that security software has been turned off without the user's knowledge, this could indicate that an unauthorised intrusion has occurred. Affected software could include system protection software or particular functions of the software. An unauthorised user may turn off security functionality in an attempt to avoid drawing any attention to their activities. Without functioning security software there may be a delay in detecting the intrusion, allowing the hacker to conduct further malicious activity, or the intrusion may go undetected indefinitely.

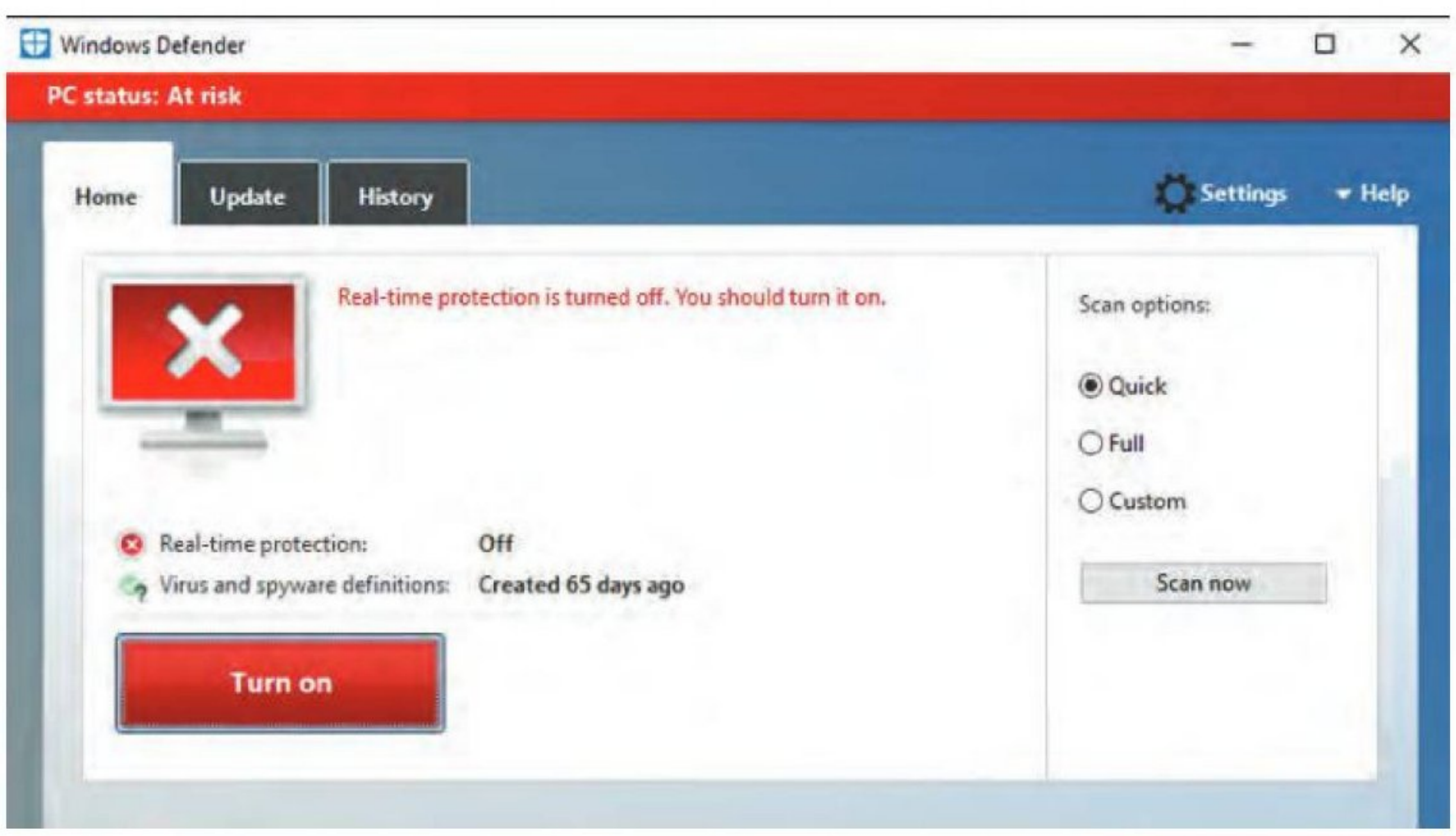


FIGURE 7.39 System protection software

Device completing tasks by itself

Remote access software, once installed, allows a user to control a device from a remote location, using an internet or network connection, without having physical access to the device.

If a user notices that the device is completing tasks by itself, this could indicate the presence of remote access software. Visible activity may include the mouse cursor moving, software applications opening and closing, or the typing of documents or emails without any obvious user interaction.

Changes to web browser home page

If the default home page setting of a web browser changes without warning, this may indicate an unauthorised intrusion. This is known as home page hijacking.

There are a number of reasons that hackers employ this practice, including:

- attempting to increase traffic to a particular website
- directing users to a website that contains links to malware and other security threats
- using the 'new' home page to collect and transmit personal data from the user
- displaying advertisements and other adware pop-ups
- encouraging users to click on a hyperlink that purports to fix home page hijacking, but in fact directs users to a fake website that could be full of viruses.

Systems used to detect intrusions

Systems used to minimise the risk to data and information due to intrusions include intrusion detection systems and intrusion prevention systems.

Intrusion detection systems

An **intrusion detection system (IDS)** is used to monitor network traffic in an attempt to detect intrusions. An IDS can be either hardware-based or software-based. Two types of IDS are host intrusion detection systems (HIDS) and network intrusion detection systems (NIDS).

A host intrusion detection system (HIDS) is installed on individual devices, including key devices on the network, and monitors and analyses all data packets sent to or from the device. An alert will be sent if suspicious activity is detected. It works by comparing current system files with previous system files. If changes are identified, an alert is sent.

Network intrusion detection systems (NIDS) are installed at a strategic point on the network. They function by monitoring passing network traffic, analysing data packets and checking whether the data packets match any in a library of known intrusion attacks.

An IDS performs a similar function to a firewall but, where a firewall focuses on examining data packets that are passing through an organisation's firewall connection, an IDS focuses on data packets that are travelling within the network.

Intrusion prevention systems

An **intrusion prevention system (IPS)** is often considered an extension of an IDS. Where an IDS focuses on identifying an intrusion and alerting network staff, an intrusion prevention system also attempts to block or stop the threat without the need for human intervention.

An IPS attempts to block intrusions by deleting suspicious data packets, resetting network connections or blocking data packets originating from a particular IP address. An IPS can be configured to follow a set of rules when a particular activity is identified.

Ethical hacking

A hacker is a skilled computer expert. Hacking involves using these skills to gain access to data and information on a device or network without authorisation. A so-called 'black hat hacker' aims to steal the data and information to sell or ransom; a 'grey hat hacker' does it for the challenge or prestige.

Ethical hacking occurs when a person (known as a 'white hat hacker') is given authorisation to hack a device or network. The aim of ethical hacking is to identify weaknesses with the current security strategy. The hacker can inform the organisation of these weaknesses, so changes can be made to the security techniques used to improve overall security.

Organisations are often eager to have individuals with little inside knowledge of a network or information system attempt to gain access or find weaknesses in the existing security controls. The thinking behind this is that if a white hat hacker can gain access to a computer system, then so can a black hat hacker. This is important so that vulnerabilities in systems can be identified and fixed.

Common techniques used in ethical hacking are penetration testing (attempting to access a network with authorisation – for example, by attacking ports) and sending phishing emails to members of the organisation.

Ethical hackers must respect the privacy of the organisations and the individuals concerned, and they must report all vulnerabilities found to the network owner.

Using networks in a global environment

The use of networks has become widespread as the technology it requires has become more widely available and the use of the internet and social media has become a seamless part of our lives. There are many clear benefits in using a network, from sharing hardware and software to accessing the internet. There are also a number of risks associated with the use of networks, particularly in the context of a global environment. In this section, we look at the benefits to individuals and organisations of using a network, and then at the risks that may be encountered.

Benefits

Establishing a network has a number of benefits over running standalone computers and resources. These include access to peripherals (for example, printers), lower set-up costs (terminals are cheaper than standalone computers) and the speed of communications. Further advantages of using a network are discussed below.

Sharing hardware

Each networked computer can access and use hardware on the network. For example, suppose several personal computers on a network each require the use of a laser printer. If the personal computers and a laser printer are connected to a network, the personal computer users can each access the laser printer on the network when they need it.

Businesses and home users network their hardware for one main reason – it may be too costly to provide each user with the same piece of hardware, such as a printer.

Sharing data and information

In a networked environment, any authorised computer user can access data and information stored on other computers in the network. For example, a large company might have a database of customer information. Any authorised person, including a mobile user using a smartphone to connect to the network, can access this database.

The capability to provide access to, and storage of, data and information on shared storage devices is an important feature of many networks. Project teams can share data, even if they are geographically remote, by using an organisation's virtual private network (VPN), which uses the internet to make global connections.

Networks support collaborative work practices through services such as cloud computing, email and file transfer.

Sharing software

Users connected to a network can access software (programs) on the network. To support multiple-user access of software, most software vendors sell network versions of their software. In this case, software vendors issue a site licence.

A network licence is a legal agreement that allows multiple users to run the software package simultaneously. The site licence fee is usually based on the number of users or the number of computers attached to the network.

Sharing software via a network usually costs less than buying individual copies of the software package for each computer.

Transferring funds

Electronic funds transfer (EFT) allows users connected to the internet (an example of a Wide Area Network) to transfer money from one bank account to another via transmission media. Consumers can use credit cards or an online payment system such as PayPal to make purchases over the internet. Businesses can use the internet to deposit their employees' salaries directly into their bank accounts. Both businesses and consumers pay bills online, which involves instructing their bank to use EFT payment to pay creditors. Global networks are a boon to online retailers, who can effectively trade continuously, without interruption.

Risks

The risks associated with using a network relate to inconvenience caused by any fault in the network devices, and damage or loss of sensitive data caused by breaches of security.

Breaches of security

Networks with inadequate security systems are likely to be attacked by malware or hackers. These threats can result in valuable information being accessed, stolen, damaged or deliberately altered for fraudulent purposes. Hackers have been able to access customer credit card details by finding an opening in a commercial corporation's network. Viruses and other malware can deliberately sabotage the operation of computers and software.

Wireless networks without suitable encryption security run the risk of outsiders eavesdropping on messages or accessing important files transmitted between users.

User dependence

Network users rely on a network to operate correctly so they can access files, applications and resources. If a component, such as a file server, develops a fault, users will not be able to run applications or access shared data. This limits the effectiveness of a worker and impacts on the productivity of the business.

Social networks

The popularity of social networking sites such as Facebook, Twitter, Instagram and Snapchat has increased rapidly in recent years. There are a number of negative effects that overuse of social networking can have on users. These include:

- distracting users from more purposeful tasks, such as studying
- reduced learning and research capability as students rely on information that is easily accessible from social network sites but is often unverified
- reduction in traditional communication between people, such as face-to-face conversations
- adoption of poor language skills and underdevelopment of creative writing skills
- negative impact on health due to skipping meals and not participating in physical activity
- reliance on virtual world experience rather than from the real world.

> **THINK ABOUT APPLIED COMPUTING 7.19**
>
> Have a look at the list under the heading 'Social networks'. Which of these risks apply to you?

Social media are now important tools for businesses to market their services and keep in touch with their customer base. Customers expect that organisations they do business with will be contactable via sites such as Twitter, Facebook and LinkedIn. Social media risks faced by organisations come under three areas: operational, regulatory and reputational.

- Operational risk relates to employees posting material in breach of copyright; the monitoring of employees on social media; and ownership of material posted.
- Regulatory risk involves company disclosures – market-sensitive data – that may be revealed on social media before a public announcement.
- Reputational risk includes what the business or employees say online, or what customers may say about the company.

Key legislation

There are several key laws that relate to the information systems and telecommunications industries. These laws govern the collection and use of private information by both government and non-government organisations at both state and federal levels. Employers and government agencies have a legal responsibility to ensure that these laws are implemented within their organisations. Organisations must make employees and customers aware of their rights, as well as their responsibilities, in relation to these laws.

Privacy Act 1988

The *Privacy Act 1988* has been discussed in detail in Chapter 1. This chapter deals with how the Act affects the way organisations control, store and communicate data.

The Privacy Act includes the following:

- 13 Australian Privacy Principles (APPs) that apply to the handling of personal information by most Australian and Norfolk Island Government agencies and some private sector organisations (see pages 28–30)
- credit reporting provisions that apply to the handling of credit-related personal information that credit providers are permitted to disclose to credit reporting bodies for inclusion on individuals' credit reports
- the collection, storage, use, disclosure, security and disposal of individuals' tax file numbers
- the handling of health information for health and medical research purposes in certain circumstances, where researchers are unable to seek individuals' consent
- the Information Commissioner to approve and register enforceable APPs codes that have been developed
- provision for a small business operator, who would otherwise not be subject to the APPs, to opt-in to being covered by the APPs.

For an individual, the Privacy Act gives people more control over the way their personal information is handled. The Privacy Act allows individuals to:

- have the option of not being identified, or the use of a pseudonym in certain circumstances (APP 2)
- know why personal information is being collected, how it will be used and who it will be disclosed to (APP3)
- discontinue receiving unwanted direct marketing (APP 7)
- ask for access to personal information (including health information) (APP 12)
- ask for personal information that is incorrect to be corrected (APP 13)
- make a complaint about an entity covered by the Privacy Act, if personal information has been mishandled.

As part of the Privacy Act, the Australian Privacy Principles (APPs) were devised to set out the standards, rights and obligations for collecting, handling, holding, accessing, using, disclosing and correcting personal information.

The Australian Privacy Principles apply to all federal government agencies and certain non-government organisations. They do not apply to local councils, or state or territory governments. Some states have their own privacy laws, such as Victoria's *Privacy and Data Protection Act 2014*.

The APPs oversee the handling of personal information by:

- Australian and Norfolk Island Government agencies
- all private health service providers
- businesses that have an annual turnover of $3 million or those that trade personal information.

Privacy and Data Protection Act 2014

The *Privacy and Data Protection Act 2014* (Vic) was discussed in Chapter 6. The 10 Information Privacy Principles (IPPs) are shown in Table 6.1 on page 236. The IPPs that relate to the control of storage and communication of data and information are:

IPP 4: Information must be protected from misuse, loss, unauthorised access, modification or disclosure. Reasonable steps must be taken to destroy or de-identify personal information that is no longer needed.

IPP 5: The organisation needs to be transparent about what it does with information.

IPP 7: Organisations can use unique identifiers (often these are numbers) only when able to show that the unique identifier is essential to the efficient performance of functions.

IPP 9: If your personal information travels outside Victoria, your privacy protections must travel with it.

Health Records Act 2001

The Victorian *Health Records Act 2001* was discussed in Chapter 6. The 11 Health Privacy Principles are summarised in Table 6.2 on page 238.

The Act protects the confidentiality of patients' healthcare information by allowing the information to be used only for the primary purpose for which it was gathered. This means that information about medical test results and your medical history may be used by your doctor, the hospital and any other health professionals only for the purpose of your immediate or ongoing care. Without your consent, this information would not be disclosed to a third party (for example, your medical insurance company or another hospital) for a 'secondary' purpose. Health information may, however, be provided to third parties without your consent under certain, and strictly limited, circumstances that include requests by family members in an emergency when you cannot give your consent and your life is threatened; where there is a serious threat to public health and welfare; research in the public interest; investigation of unlawful activity; and as part of a legal claim.

An individual who believes that the Health Records Act has been breached can make a complaint to the Health Services Commissioner, who will try to achieve a resolution

RESEARCH

Patient privacy

Jordan is a medical student working as a receptionist at a local doctor's surgery during his mid-term vacation. His job is to greet patients as they arrive at the surgery, retrieve their medical records and alert the doctor of their arrival. At the end of their consultation he collects the fee and makes another appointment if required.

The surgery database has detailed medical records on its 215 patients. Each record is stored under the patient's name and address. Jordan has downloaded a copy of the surgery's patient database onto his laptop as he wants to be able to refer to real-life examples when he is completing a group assignment for his university studies. He intends to share the medical records with the two other students in his group. Jordan's next rotation is at a hospital in regional Victoria and he intends to take his laptop with him.

1 Refer back to Table 6.1 on page 236. List the IPPs that are relevant to this case study, with a comment on how each is relevant.

2 Has Jordan breached any other Act? If yes, which one has he breached, and how?

by discussion between the parties. If a satisfactory resolution cannot be reached, the Commissioner may then serve a compliance notice on the organisation that has breached the Act to inform the organisation which area of the Act has been breached and that it must correct its procedures. The maximum penalty for an organisation is currently 3000 penalty units and 600 penalty units for non-corporate cases.

Ethical issues

People who have access to computer networks also have access to large amounts of confidential data. Yet the training of computer specialists does not usually focus on privacy or other ethical issues in the way that other professions do.

The following questions are examples of privacy issues you may need to consider as a computer professional.

1 Should you read the emails of network users?
2 Should you monitor websites visited by network users?
3 Should you place keyloggers on computers to capture what network users are typing?
4 Should you read documents stored on a person's computer or on the file server?
5 Should you view graphics stored on a person's computer or on the file server?

A computer professional who has full network access has been given a position of considerable responsibility by the organisation that they work for. Computing is still a relatively young profession, especially when compared to medicine and law, and does not have a strong code of ethics.

7.20
THINK ABOUT APPLIED COMPUTING

You are a network administrator for a small company. You notice an email exchange between a staff member of your company and someone from a competing company that talks about an upcoming product. What do you do?

For more on ethics, refer to Chapter 1, page 31, and Chapter 6, page 239.

7

CHAPTER SUMMARY

Essential terms

access rights permissions that are granted to a user, or to an application, to read, write and delete files on a network

adware a form of malware that presents advertisements on your computer

asymmetric digital subscriber line (ADSL) a digital line alternative for the small business or home user. ADSL transmits on existing standard copper telephone wiring. The ADSL2+ technology allows a faster transfer rate than the older ADSL rate – 20 Mbps compared with 8 Mbps.

asymmetric encryption the use of public and private keys to decrypt messages

authentication processes that aim to confirm or verify that a user of an information system is the person they claim to be

biometric authentication an authentication technique that uses an individual's unique biological characteristics

Bluetooth a protocol that uses short-range radio waves to transmit data along enabled devices, such as notebook computers, mobile telephones and printers

broadband router a basic router that connects a LAN to the internet, also functioning as a switch, a firewall and a wireless access point

brute force attack a continual trial and error method to find out a password and username

client–server network a network consisting of a central server – a large computer that stores data or provides services – and a number of other computers, called clients, that rely on the server for these resources

communications satellite a space station that receives microwave signals from Earth, amplifies the signals, and retransmits them back to Earth

credential management the process of checking the credentials of a user of a system to ensure they are the person they claim to be

dedicated server a server that performs a specific task, such as a file server, a print server, a database server or a network server

denial-of-service (DoS) attack a malicious act that consists of flooding a network with traffic to make it unusable

device-to-cloud a form of communication that uses a gateway to enable non–internet connected devices to reach cloud services

device-to-device direct communication between devices through the use of radio technology

device-to-gateway communication between an IoT device and a gateway device such as a router or a smartphone, which then forwards the signals to another device in the local area

digital signal individual electrical pulses that represent the bits that are grouped together to form characters

digital signature an authentication technique used to verify if an electronic document (e.g. email, data file or web page) that has been received is genuine

disaster recovery plan a list of procedures and actions for an organisation to follow in the event of a disaster

downlink transmission from a communications satellite to a receiving Earth station

Earth-based station a communications facility that uses large, dish-shaped antennas to transmit and receive data from satellites

encryption the process of encoding or changing data so that an unauthorised user who reads the data would not be able to understand it

9780170440806

ethical hacking attempting to gain access to a device or network without authorisation, with the intention of informing the organisation of weaknesses with the current security strategy so changes can be made that improve overall security

fibre-optic cabling a wired transmission medium containing shards of glass that reflect pulses of light generated by small lasers or light-emitting diodes (LEDs)

firmware software that has been programmed onto a hardware device

hacker a person who gains unauthorised access to a computer motivated by profit or a challenge, or in order to view restricted data

host intrusion detection system (HIDS) monitors a computer's infrastructure, analyses traffic and logging any malicious behaviour

internet a worldwide network in which billions of devices are connected together for the purposes of communication and data exchange

Internet of Things (IoT) collectively, the billions of devices around the world that are connected to the internet, collecting, receiving and sending data

intranet a private network accessible only to authorised users within an organisation

intrusion detection system (IDS) a system, either hardware-based or software-based, that is used to monitor network traffic in an attempt to detect intrusions

intrusion prevention system (IPS) a system with the functionality of an IDS that also attempts to block or stop threats without the need for human intervention

Local Area Network (LAN) a network that is created when two or more devices are connected together in the same geographical area

key a series of bits (0s or 1s in binary code) that is used to apply an algorithm to a data packet to scramble (encrypt) and unscramble (decrypt) data that requires protection

malware short for 'malicious software'; programs designed to infiltrate and cause damage, disruption or access to a device or network without the user's knowledge or consent; includes viruses, worms, Trojan horses, adware, spyware, logic bombs and keyloggers

microwaves radio waves that can be used to provide high-speed transmission of both voice and data; data is transmitted through the air from one microwave station to another in a manner similar to the way radio signals are transmitted

microwave station Earth-based reflective dish that contains the antenna, transceivers and other equipment necessary for microwave communications

Mbps short for megabits per second; a unit of data transfer speed

mobile network also known as a cellular network; a network that uses telecommunications networks to allow users to communicate using their mobile devices

modem a device used to connect a device to a network using a copper cable

National Broadband Network (NBN) a fibre-optic backbone for the transmission of digital data covering a large part of Australia

near-field communication (NFC) technology digital technology using radio waves that allows contactless communication between devices over a very small distance, without the need to configure settings; examples include payWave and Myki

network a collection of two or more digital system devices, connected by communications channels, that facilitates communications among users and allows users to share resources with one another

network administrator the person who oversees the operations of a client–server network

network architecture the layout of a network, including the hardware, software, protocols, and transmission media used

network diagram a visual representation of a network that shows the nodes that make up a network and how they interact

network interface card also known as a wireless adaptor; slots into the motherboard of a device and then provides ports to allow the device to connect to a network

network intrusion detection system (NIDS) monitors all traffic to and from all devices connected to a network and alerts when any suspicious traffic is detected

9780170440806

network operating system (NOS) system software that organises, controls and coordinates the operation of a network

node a network connection point, such as a desktop or mobile computer, a peripheral such as a printer or scanner, or a portable device such as a smartphone

operating system update a form of software update released to improve the usability, functionality and/or security of an operating system

optical fibre a strand of glass or plastic, as thin as a human hair, that uses light to transmit signals

peer-to-peer network a computer network with no central server, on which each workstation shares its files equally with the others

Personal Area Network (PAN) a computer network for connecting an individual's devices within a limited range

phishing sending an email to a user, falsely claiming to be an established enterprise of some kind, in an attempt to scam the user into giving up private information that will be used for identity theft

private key a key that is only known by a user who has the key installed on their device

public key a key that is made available to other devices and is used to encrypt data packets that are being sent

remote access software a type of software that allows a user to connect to a remote computer or network

router a device that is used to connect two separate networks together

server a device that is used to provide services to other devices connected to a network

software maintenance the process of making changes to software applications, after they have been installed, to fix faults, improve performance or add future functionality to software

software update sometimes referred to as a patch; generally, a small file that is downloaded and that fixes any bugs in a piece of software

software upgrade replacing an old version of software with a new version

staff procedures steps or processes staff should follow in a given situation when dealing with data and information

standalone device any piece of computing equipment that can perform its function without the need for another device, computer or connection

switch a device that is used to connect multiple devices to a network

symmetric encryption use of one key to both encrypt and decrypt a message

threat anything that has the potential to cause harm to data and information stored and communicated between information systems

Transmission Control Protocol/Internet Protocol (TCP/IP) a network standard that manages the transmission of data by breaking it up into packets and transmitting the packets over the internet

twisted-pair cable cable made up of twisted-pair wires

twisted-pair wires two separate insulated copper wires that are twisted together to reduce interference

two-factor authentication an extra layer of security to an online account

unshielded twisted pair *see* **twisted-pair wires**

uplink transmission from an Earth-based station to a satellite

username and password widespread security control that identifies a user of a network, also known as a login

virtual private network (VPN) extends a private network across a public network and allows a user to use public networks such as the internet to send and receive data and information as if they were using a private network, such as an intranet

virus a computer program that can destroy files and alter the performance of the operating system; once a virus is in a computer, it can spread over a network to other connected computers

virus definition a collection of binary code that is used to identify a particular virus

Wide Area Network (WAN) a network that is situated in more than one geographical location

wi-fi communication a wireless transmission medium that distributes radio signals through the air over long distances

9780170440806

wired network a network in which the devices are connected using a physical cable

wireless access point (WAP) a communications device that allows devices to wirelessly connect to an existing wired network

wireless adaptor *see also* network interface card; slots into the motherboard and allows the device to send and receive wireless signals

wireless broadband router a device that combines the functions of a basic router (connecting the LAN to the internet), a switch (for devices, such as a desktop computer, connected by cable), a firewall (security measure) and a wireless access point (to allow wireless connectivity)

wireless network a computer network that uses wireless data transfer between network devices

Wireless Personal Area Network (WPAN) a type of personal network that uses wireless transmission media to transfer data between a user's personal devices

Important facts

1 A **network** is a collection of computers and devices connected by communication channels that facilitates communication among users.

2 **Types of networks** include **Local Area Network**, situated in one location, and **Wide Area Network**, in multiple locations.

3 Types of network architecture include **peer-to-peer**, where each device shares data and information with other devices, and **client–server networks**, which contain a server that provides data and services to client devices. A **Personal Area Network** is a network in which a user's individual devices are connected to each other.

4 **Hardware** required on a network includes a **network interface card** (or wireless adaptor), which is used to connect a device to a communication channel; a **server**, used to provide services to clients; a **switch**, to connect devices; a **router**, used to connect separate networks; and a **modem**, which allows data communication over a telephone line.

5 A **network operating system** is software used to control and manage network performance.

6 Communication methods used to transmit data include **wired networks**, **wireless transmission** – either wi-fi (802.11x) or Bluetooth – and **mobile network standards** (3G, 4G and 5G). Wired networks can use either unshielded twisted-pair or fibre-optic cabling.

7 The **Internet of Things** is a network formed by 'smart devices' that have an 'on-off' switch to the internet, and/or to other IoT devices, while an **intranet** is a private network accessible only to users within an organisation. The **internet** is a worldwide network, and a **virtual private network (VPN)** allows a user to send and receive data as if their computing devices were directly connected to the private network from a remote location.

8 A **network diagram** is used to visually represent a network.

9 A **threat** is anything that has the potential to cause harm to data and information stored and communicated between information systems.

10 **Credential management** involves determining if a user is the person they are claiming to be.

11 **Malware** is software that is designed to damage, disrupt or gain unauthorised access to an information system.

12 **Security controls** are used to reduce the effectiveness of threats include usernames and passwords, access rights, software updates, and encryption, which involves encoding or changing data so that the data is unreadable by others.

13 **Encryption** uses keys. A key is used to scramble and unscramble data. A private key is a key that is only known by a user, and a public key is a key that is made available to other devices and is used to encrypt data packets.

14 **Authentication** involves confirming or verifying that a user of an information system is the person they are claiming to be. Techniques include usernames and passwords, biometric techniques and digital signatures.

15 **Preventative measures** include software maintenance, operating system and virus definition updates, a disaster recovery plan and staff procedures.

16 An **intrusion detection system (IDS)** is used to monitor network traffic in an attempt to detect intrusions, while an **intrusion prevention system (IPS)** contains the functionality of an IDS it also attempts to block or stop threats, without the need for human intervention.

9780170440806

17 **Ethical hacking** involves attempting to gain access to a device or network without authorisation, with the intention of identifying weaknesses with the current security strategy, to inform the organisation about those weaknesses.

18 **Benefits of using networks in a global environment** include the ability to share hardware, software, data and information, and the ability to complete financial transactions online; risks include breaches of security, user dependence and issues related to the use of social networks.

19 Laws related to the handling of data and information include the *Privacy Act 1988* (Privacy Act), the *Privacy and Data Protection Act 2014* (Vic) and the *Victorian Health Records Act 2001*.

9780170440806

TEST YOUR KNOWLEDGE

Review quiz

Networks

1 How is a Local Area Network different from a Wide Area Network?

2 What are one advantage and one disadvantage of a peer-to-peer network?

3 Describe the role of a server on a network.

4 List three devices that may be found on a Wireless Personal Network.

Network hardware and software

5 Describe the role of a network interface card.

6 Explain how a router differs from a switch.

7 Outline the role of a modem on a network.

8 State one function of a network operating system.

Communication methods

9 Explain an advantage of unshielded twisted-pair cabling compared with fibre-optic cabling.

10 Describe one advantage of wireless communication over wired communication.

11 Why may data transmission on a mobile network incur a cost?

Types of networks

12 Describe one similarity and one difference between an intranet and the internet.

13 List three devices that may be found on the Internet of Things.

14 Explain when a user would require the use of a virtual private network.

15 Explain the use of a network diagram.

Security threats

16 Provide three examples of malware.

17 Give an example that illustrates why the credentials of users need to be managed.

Security controls

18 How do access rights protect data and information within an organisation?

19 Write a sentence using the encryption method explained in Figure 7.29. Give it to someone else to decrypt using the same key.

9780170440806

TEST YOUR KNOWLEDGE

Authentication

20 Describe two different methods of authentication.

21 Explain the purpose of a digital signature.

Preventative measures

22 List the four sub-plans of a disaster recovery plan.

23 What is a virus definition? How can it be used to detect viruses?

Network intrusion

24 How does a host intrusion detection system (HIDS) differ from a network intrusion detection system (NIDS)?

25 How does an intrusion prevention system block threats to data and information?

Ethical hacking

26 Outline the purpose of ethical hacking.

Laws

27 Which law regulates the handling of personal data for all Australians?

28 Explain the Information Privacy Principles (IPPs) of use and disclosure.

Ethical issues

29 Describe one issue that may arise regarding the privacy of users of a network.

30 Explain an issue that may occur as a result of ethical hacking.

9780170440806

APPLY YOUR KNOWLEDGE

Medical Answers is an organisation that provides administration and database support for small- to medium-sized medical clinics across Australia. Services offered by Medical Answers include:

- training for clinic staff in the use of DBMS software
- database maintenance and management
- 24-hour customer support.

At present, Medical Answers employs three staff and has a turnover of around $2.5 million per year, though the business has been growing over the last couple of years and this growth is expected to continue over the next few years.

At present, the computer office network at Medical Answers consists of three laptops, a printer, a file server, a switch, a router, a modem and a wireless access point.

The file server, router and wireless access point are all connected to the switch using unshielded twisted-pair cables. The three laptops then use wireless transmission to connect to the network via the wireless access point, while the router is connected to a modem, which is then connected to a telephone line used to connect the network to the internet.

To provide database maintenance and management, staff of Medical Answers require remote access to the network of each medical centre network.

At present, the staff at Medical Answers do not use usernames and passwords on their network in the office. Neither the laptops, the wi-fi connection nor the file server require any authorisation.

The network has a firewall installed on the internet connection, and computer protection software is used to scan the network for malware, but this software has not been updated for more than 12 months.

When accessing a medical clinic's network remotely, all staff use a generic username and password. These are currently set as 'admin' and 'password' for each clinic, to make them easy to remember. Staff then download the database from a medical clinic to perform database maintenance, store a copy on the file server locally, then upload the updated database back to the medical clinic.

1. Describe the type of network used by Medical Answers.
2. Draw a network diagram of Medical Answers' network. Annotate the diagram to explain the function of each component of the network and the communication media and security techniques used.
3. Describe three threats to the data and information either communicated by, or stored on, Medical Answers' network. Evaluate the significance of each threat.
4. Discuss three legal or ethical issues that may arise from Medical Answers' current security practices.
5. Describe three security techniques that could be introduced to the network. Justify how each technique will improve security. Annotate the network diagram to show where these techniques would be used.

9780170440806

PREPARING FOR

Unit 2

OUTCOME 2

Students are to respond to a teacher-provided case study to examine the capabilities and vulnerabilities of a network, design a network solution, discuss the threats to data and information, and propose strategies to protect the security of data and information.

Skills required

You will be required to demonstrate the following skills:

1. Describe the applications and capabilities of a range of networks.
2. Examine the impact of common network vulnerabilities.
3. Design a network solution with wireless capability.
4. Identify and evaluate threats to the security of data and information.
5. Propose and justify strategies to protect the security of data and information within a network.
6. Identify and discuss possible legal and ethical issues arising from ineffective data and information security practices.

Documents required for assessment

You will be required to submit your written responses to the case study to your teacher.

9780170440806

Index

9780170440806

9780170440806

9780170440806

9780170440806

9780170440806

nelsonnet

ACE THIS SUBJECT

Want to go further with your learning?
Unlock your **NelsonNet** resources now.

Study **anywhere, anytime** with your downloadable **NelsonNetBook**

Go further with **weblinks** to explore outside the classroom

Chapter review **quizzes** help you **prepare for exams** and in-class tests

TURN THE PAGE FOR YOUR ACCESS CODE!

PLUS! GRAB YOUR FREE TIMETABLE

Scan this QR code to download and use it to help you plan your revision

www.nelsonnet.com.au

ISBN 978-0170440806