* VIRTUAL SCHOOL VICTORIA

2024

NAME:

VSV Student ID:

ALGORITHMICS UNIT 3

SAC 2: Algorithms (Weeks 1 to 5)

Outcome 2

Date of Completion: 13-17 May 2024

Reading Time: 5 minutes
Writing time: 55 minutes
TOTAL (60 minutes)

QUESTION AND ANSWER BOOK

Type	*Number of questions*	*Number of questions to be answered*	*Number of marks*
Short/ Extended Response	10	10	38
		Total	38

Materials supplied

- Question and answer book of 8 pages

Materials permitted

- Pens/Stationary and one Scientific Calculator permitted.

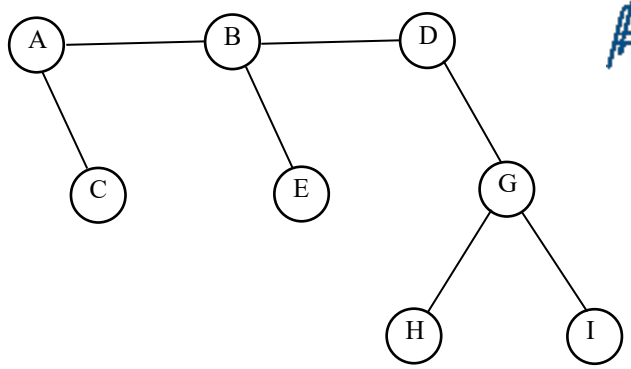
No Reference material permitted.

Instructions

- Write your **name** in the space provided above on this page.
- All written responses must be in English, point form is preferred.

Students are NOT permitted to bring mobile phones and/or any other unauthorised electronic devices into the test room.

1. Consider the graph below:



(a) Carry out a depth first search traversal, stating the order in which vertices are added, adding items to the stack/queue in alphabetical order where multiple options exist.

ABDGIHEC if not added in alphabetical order

(2 marks)

(b) Carry out a breadth first search traversal, stating the order in which vertices are added, adding items to the stack/queue in alphabetical order where multiple options exist.

(2 marks)

2. Maureen is responsible for providing free Wifi to a large city. The city is made up of a number of suburbs, and each suburb has a Wifi access point. These are connected to a central computer by underground cables. The access points are represented as nodes on a graph. The computer needs to be stored at the same place as one of the access points.

To work out the distance from the central computer to each Wifi access point, the following functions are available:

`Dijkstra(start, end)` – uses Dijkstra to return the distance between start and end.
`BellmanFord(start)` – uses Bellman-Ford to return an array containing the distances between the start node and every other node.

Maureen wants to find out where to place the central computer to minimise the average distance between the central computer and each Wifi access point. In addition, where there are multiple solutions with a similar average distance, she would like the central computer to be as close to the suburb “Porttown” as possible.

Describe an algorithmic approach that Maureen can take to achieve these goals. Explain your choice of Dijkstra/BellmanFord at each stage, and why this gives you the most efficient solution.

First use BF on every node. Find the average distance from the array it returns, and choose the location that gives the lowest average distance. BF is better here as we want to find the distance to all nodes from a particular path, and we don’t care about reconstructing the path.

If there are multiple locations with the same average distance, run Dijkstra starting at each of those locations and ending at Porttown. Dijkstra is better here because we only want the shortest distance between a pair of nodes.

(4 marks)

3. Consider the following recursive algorithm. It uses the binary operator % such that a % b returns the remainder when a is divided by b. For example, 10 % 3 = 1.

```
Algorithm Surprise(a,b):

     rem = a % b.
     if rem = 0:
          return b
     else:
          return Surprise(b, rem)
```

Showing your working, find what Surprise(56, 12) returns.

56 % 12 = 8

12 % 8 = 4

8 % 4 = 0 → return 4

(2 marks)

4. Write a recursive function in pseudocode to check that a given string is a palindrome. (i.e. it reads the same forwards and backwards. TACOCAT is a palindrome.)

```
begin algorithm(string):
  if len(string) = 0 or 1:
    return True
  else:
    if first letter of string = last letter of string:
      remove first and last letters from string
      return algorithm(string)
    else return False
```

Not recursive = no points

(4 marks)

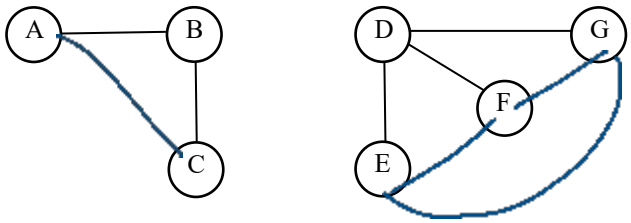
5.

(a) Explain the purpose of Floyd Warshall's Transitive Closure algorithm.

To identify if an indirect path exists between pairs of vertices in a graph.

(1 mark)

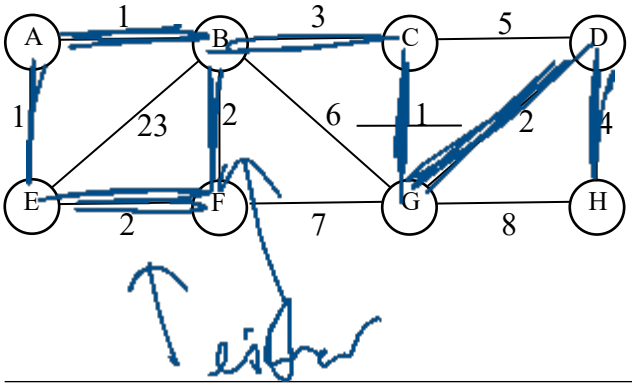
(b) If this algorithm is run on the following graph, add edges to show the resulting output graph.



(2 marks)

6.

Use Prim's algorithm to find a MST for the graph below, starting at node A.



(3 marks)

7. Charles wants to compress a range of strings To compress a string means to make its length smaller whilst preserving the original information, using some clever tricks.

He is using the following pre-existing functions:

`compress(X)`: takes a string `X`, compresses it and returns a string `Y`.

`length(X)`: takes a string `X` and returns its length in bytes.

It is possible to run 'compress' more than once to make a file even smaller. Because of the way the function works, Charles cannot predict how much smaller the compressed file will be in relation to its original file. Sometimes the function cannot make a file any smaller.

His aim is to reduce any given string to 100 bytes or less.

Using the functions `compress(X)` and `length(X)`, write pseudocode which:

- takes a string input
- compresses it
- checks to see if it is smaller than or equal to 100 bytes
- if it is, then return the compressed string along with the integer '1' to indicate it was compressed once
- if not, compress it again
- keep compressing it until the file size does not change, i.e. maximum compression has been achieved
- returns the compressed string, along with an integer showing the number of times it was compressed

```
begin algorithm (plaintext):
  oldlen = length(plaintext)
  running = True
  counter = 0
  while running is True:
    compressed = compress(plaintext)
    counter ++
    if length(compressed) <= 100:
       running = False
    if length(compressed) = oldlen:
       running = False
    oldlen = length(compressed)
    return compressed, counter
```

```
Implied that we keep checking for <= 100 bytes but not explicitly
stated
```

(4 marks)

8. Gita is constructing a social network called Snapgram. A snapgram user can follow another user, and can view the users that any given user follows. A user's popularity increases with the number of followers they have, and with the popularity of those followers.

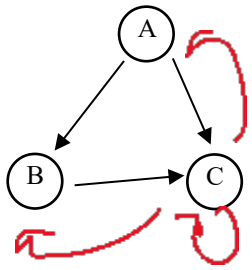
(a) Explain how three features of this scenario can map onto the PageRank algorithm.

User = web page, follower = link between a web page, damping factor = probability that a user follows a user they already follow, rather than choosing a new one at random

(allowed discussion of popularity since Q didn't explicitly state they don't have to follow a user currently followed)

(3 marks)

(b) A small subset of the network is shown here:



The PageRank of Page A is given by

$$\mathrm{PR}(A) = \frac{(1-d)}{N} + d\left(\frac{\mathrm{PR}(B)}{\mathrm{L}(B)} + \frac{\mathrm{PR}(C)}{\mathrm{L}(C)} + \frac{\mathrm{PR}(D)}{\mathrm{L}(D)}\right)$$

$d = 0.8$

Find Pr(C) after the second iteration to three decimal places. (Initialisation does not count as an iteration).

Init all to 0.3. $Pr(A) = \frac{0.2}{3} + 0.8\left(\frac{0.3}{3}\right) = 0.15$

$Pr(B) = \frac{0.2}{3} + 0.8\left(\frac{0.3}{2} + \frac{0.3}{3}\right) = 0.28$

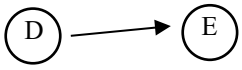
$Pr(C) = \frac{0.2}{3} + 0.8\left(\frac{0.3}{2} + 0.3 + \frac{0.3}{3}\right)$

Second $Pr(C) = \frac{0.2}{3} + 0.8\left(\frac{0.15}{2} + 0.28 + \frac{0.5}{3}\right) = 0.5$

$= 0.506$

(4 marks)

(c) An even smaller subset of the network is shown here:

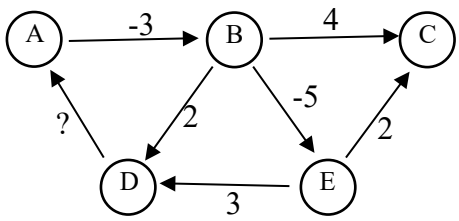


Deduce the value of *d* such that Pr(D) = Pr(E) for all iterations.

$d = 0$

(1 mark)

9. Consider the following graph:



Explain whether or not each of the following algorithms can be run on this graph, specifying the range of possible values for the weight of edge *AD* if appropriate.

(i) Bellman Ford

Yes if AD > 5 (permit >= 5)

Works if negative edges exist as long as there are no negative cycles

Note tolerance for ambiguity odf question – consistency of understanding is examined

(ii) Dijkstra's

Yes if AD > 1 (permit >= 1)

Still may not return the optimal path.

Technically should both be "yes" for all values but may not return the right solution.

Correct but reversed: 1m

(3 marks)

10. The coin change problem asks what the minimum number of coins is required to make a certain amount of money.

Suppose we take the following approach: at each stage, use the largest coin available which does not exceed the remaining amount.

(a) State the name of this algorithm design pattern.

Greedy

(1 mark)

For example, given the coins [50c, 30c, 5c, 1c], we want to make 60c change.

(b) What solution does the above approach provide?

50c + 5c + 5c → 3 coins

(1 mark)

(c) Explain whether or not this approach is guaranteed to be optimal.

No, since we can clearly choose 2 coins 30c + 30c

(1 mark)

END OF TEST