Consider the function *mystery(N, M)* shown below:

```
def mystery(N, M):
    if N == 0:
        return 1
    else:
        x = M * mystery(N//2, M-1)
        return x
```

What is the recurrence relation for the time complexity of *mystery(N, M)* function?

Select one:

- A.

$$T(1) = b$$
$$T(N) = T(N//2) + c$$

- B.

$$T(1) = b$$
$$T(N) = M * T(N//2) + c$$

- C.

$$T(1) = b$$
$$T(M) = T(M - 1) + c$$

- D.

$$T(1) = b$$
$$T(M) = M * T(M - 1) + c$$

2
Marks

Consider the function *mystery(N, M)* shown below:

```
def mystery(N, M):
  if N == 0:
    return 1
  else:
    x = M * mystery(N//2, M-1)
    return x
```

What is the auxiliary space complexity of *mystery(N, M)* function? Explain your answer.

Report question issue

Notes

Question 2 Notes

Unsure

Notification

2

Marks

Consider a **Graph** ***G*** with **vertices** ***V*** and **edges** ***E***.

The pseudocode below attempts to store the **degree** for each vertex ***v*** **in** ***V***.

```
def incoming(graph G):
  for each vertex u of graph G:
    u.degree = 0
    for each edge <u,v,w> of vertex u:
      u.degree = u.degree + 1
```

What is the time and space complexity of the pseudocode above? Explain your solution.

Report question issue

Notes

Question 3 Notes

Unsure

Notification

2

Marks

Consider a prefix **Trie *T*** built from ***N* words** with the longest word having ***M* characters**. None of the words are stored in the **Trie** (even at the leaves). Each node only stores a single character (except the root, which does not store any characters).

For a given **query word *Q***, what is the time complexity to print out all the words in **trie *T*** with the **query word *Q*** as the prefix? Reason out your time complexity.

Report question issue

Notes

Question 4 Notes

Unsure

Notification

MONASH University

Questions attempted: 0/26

2

Marks

Given two **strings** of length *N* and *M*. What is the best and worst case time complexity to compare these strings in a comparison-based sorting algorithm?

Report question issue

Notes

Question 5 Notes

Unsure

Notification

3
Marks

Consider **radix sort** for a **list of *N* strings**, where the strings are sorted by running a count sort on each column, right to left. Discuss using examples why there is a need for the count sorts to be stable in ensuring correctness, and whether ensuring stability incurs any costs.

Report question issue

Notes

Question 6 Notes

Unsure

Notification

Questions attempted: 0/26

2 Marks

Consider a standard **Quick Sort** implementation with the following approaches:

- **Approach 1**: Perform **2-way partitioning** for < pivot and >= pivot
- **Approach 2**: Perform **3-way partitioning** for < pivot, == pivot and > pivot

State the best and worst-case time complexity for each of the approaches. Briefly explain when they occur.

Report question issue

Notes

Question 7 Notes

Unsure

Notification

3 Marks

Consider a **list of *N* positive integers.**

Describe a linear time algorithm (using high level plain language) to find the item in a list whose value is closest to the average of the *i-th* and *j-th* largest elements in a list.

You may assume that you have a **Quick Select** function that operates in linear time complexity.

Report question issue

Notes

Question 8 Notes

Unsure

Prev Page 1 2 3 4 5 6 7 8 9 10 Next Page Go To Summary

Notification

Questions attempted: 0/26

2

Marks

Consider the standard knapsack problem. You are given a set of items, each with a weight and a value, and each item can only be used once. You have a knapsack of some capacity, and you wish to determine which combination of items has:

- The highest total value
- Has a total weight less than or equal to your knapsack's capacity

This problem can be solved using Dynamic Programming. Write the definition for the set of overlapping sub problems which need to be solved in order to solve the knapsack problem.

A B I U x_2 x^2 ABC 123

Report question issue

Notes

Question 9 Notes

Unsure

Notification

2

Marks

Give an example of a problem which can be solved using dynamic programming, where the space complexity of the DP algorithm which solves it is lower than the time complexity.

Briefly explain how this is achieved.

Report question issue

Notes

Question 10 Notes

Unsure

Notification

MONASH University

Questions attempted: 0/26

2

Marks

What is the best case and worst case time complexity to insert an item into a **Hashtable** containing ***N* items**. The Hashtable is implemented using **Separate Chaining with AVL Trees**. Explain your complexity.

Report question issue

Notes

Question 11 Notes

Unsure

Notification

4

Marks

Consider a **Hashtable** implemented using **Cuckoo Hashing** with the following criteria:

- There are 3 tables, A B and C.
- H1, H2, H3 are the hash functions for tables A, B and C respectively.

Describe in plain words how you would implement an algorithm to insert a <key, item> pair into the hashtable. Your answer must use A,B and C.

Report question issue

Notes

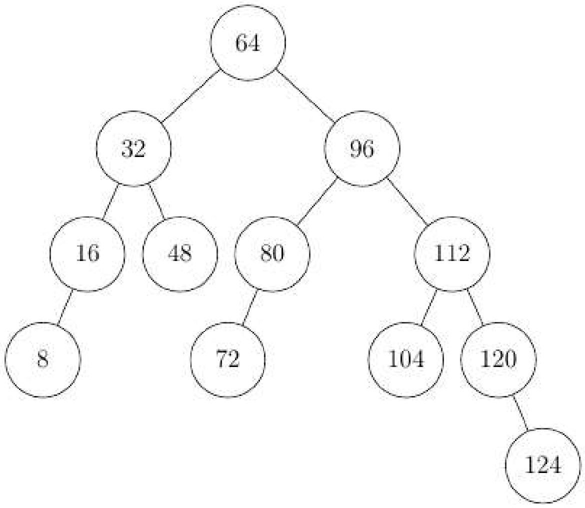
Question 12 Notes

Unsure

Notification

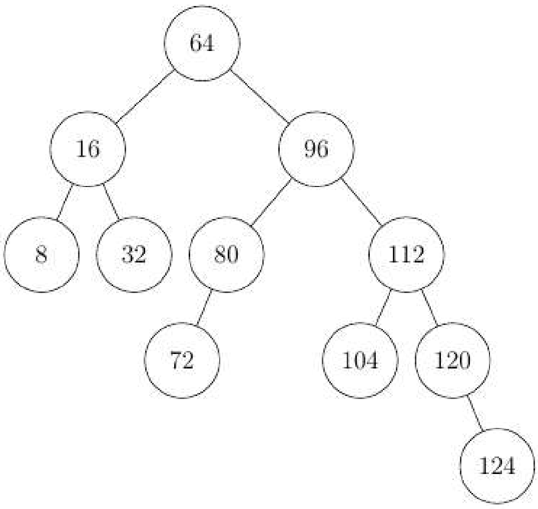
2
Marks

What is the state of the following AVL tree after the deletion of 48?

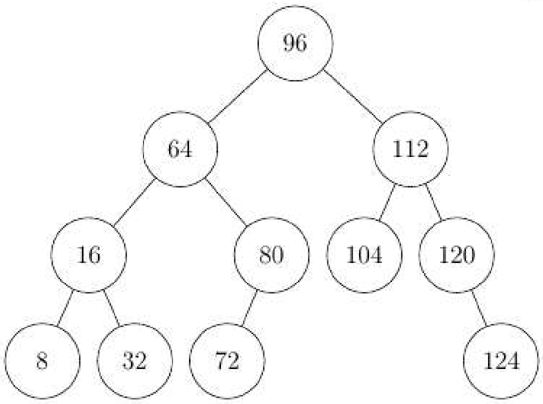


Select one:

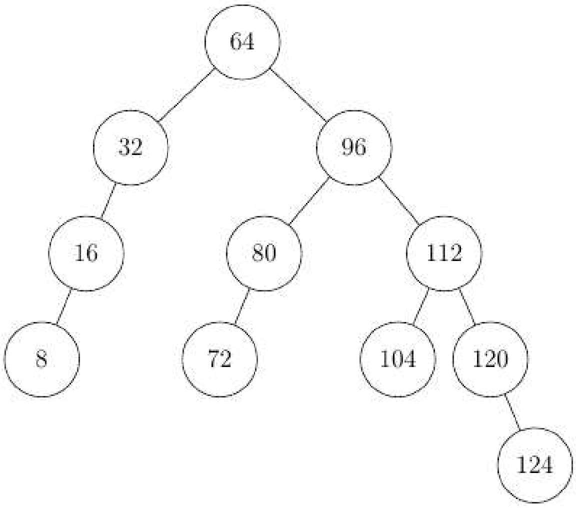
○ A.



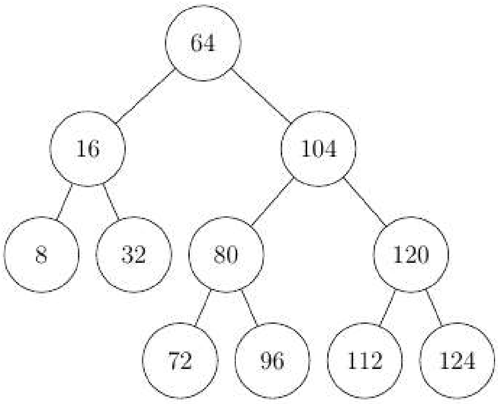
○ B.



○ C.



D.



Report question issue

Notes

Unsure

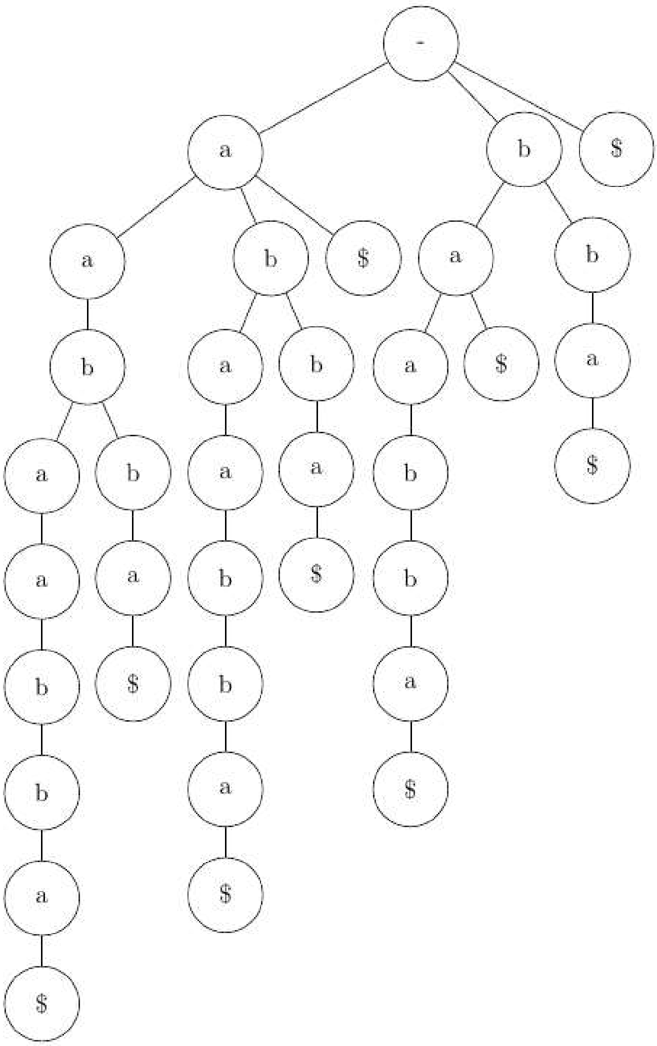
Notification

MONASH University

Questions attempted: 0/26

1 Mark

Consider the suffix **trie** of the string aabaabba (shown below). How many nodes (including the root) will the corresponding suffix **tree** have?



Select one:

- A. 13
- B. 14
- C. 15
- D. 16
- E. 17

Report question issue

Notes

Notification

1
Mark

Consider a **Suffix Trie** T generated from a **String** S.

Describe how and why the suffix trie T can be used to determine if **query** Q is a substring of string S with a time complexity of $O(Q)$.

Report question issue

Notes

Question 15 Notes

Unsure

Notification

Monash University

Questions attempted: 8/26

3 Marks

Assume that we are constructing the **suffix array** for a **string *S*** using the **prefix doubling** approach. We have already sorted the suffixes for string *S* according to their first ***4* characters**; with the corresponding **rank array** shown below:

Suffix ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Rank	9	8	3	14	5	15	6	13	3	12	7	9	11	2	1

We are now sorting on the first **8** characters.

Describe how will you compare the following two suffixes on their first *8* characters in *O(1)*:

- Suffixes with ID 2 and ID 3
- Suffixes with ID 3 and ID 9

Report question issue Notes

Question 16 Notes

Unsure

Notification

Questions attempted: 8/26

2 Marks

Suppose you are performing the naive algorithm to invert of BWT. The given BWT string is

b$baaa

You have computed the following 2-mers.

$a
aa
ab
ab
b$
ba

Write down the 3-mers (in lexicographical order) which would be obtained after one more iteration of the algorithm. Write each 3-mer on a separate line, with no spaces between the characters

Report question issue

Notes

Question 17 Notes

Unsure

2
Marks

Consider the problem of searching for **all occurences** of the substring "ab" in the string "aaabaabbaaba". We want to solve this using the **Burrows Wheeler transform** substring search algorithm. The diagram below shows the **sorted characters of the string** in the third column and the **BWT of the string** in the fourth column. The second column shows the **indices**, and the first column shows the **suffix array**.

The general idea of the algorithm is:

1. You have a start and end character in the BWT, which define a *range* (initially 1 and the last position)
2. You contract that *range* appropriately
3. Use the LF mapping to obtain a new *range*
4. If you have processed the pattern, stop and return the appropriate suffix array indices.
5. Proceed by one character in the substring we are searching for, and go back to step 1

What are the start and end indices of the *ranges* during the search for "ab" **after each step 3**

Suffix ID	Index	Sorted Chars	BWT
13	**1**	$	a
12	**2**	a	b
1	**3**	a	$
9	**4**	a	b
2	**5**	a	a
5	**6**	a	b
10	**7**	a	a
3	**8**	a	a
6	**9**	a	a
11	**10**	b	a
8	**11**	b	b
4	**12**	b	a
7	**13**	b	a

Select one:

- A. (2,9), (10,12)
- B. (10,13), (7,9)
- C. (2,11), (5,10)
- D. (1,13), (2,11)
- E. (10,13), (2,4)

Report question issue Notes

Unsure

4

Marks

Consider the following 2 approaches to dealing with the priority queue when implementing Dijkstra's algorithm. In the following descriptions, "key" refers to the distance value of each vertex which is used to order the priority queue.

Approach 1 - Initialise the priority queue with all the vertices of the graph. Set their keys to infinity, and set the key of the source to 0. Whenever relaxation occurs, update the corresponding key in the priority queue.

Approach 2 - Initiliase the priority queue with the source vertex, with a key of 0. Whenever relaxation occurs for a vertex *v*, add a new element to the priority queue with key = the new distance for *v* and value = *v*. When removing an element from the priority queue, if that vertex has already been finalised (i.e. already been processed by the algorithm), just discard it.

State, for each approach, the **total** complexity (i.e. the total cost over the lifetime of the algorithm) of performing

1. The extract_min operations
2. The relaxation operations (and the associated priority queue updates)

Justify your answers

Report question issue

Notes

Question 19 Notes

Unsure

Notification

2

Marks

The Bellman-Ford algorithm for computing single-source shortest paths in a graph with negative weights can be optimised so that it has the ability to terminate early.

Explain how this can be done and state the best case time complexity of the resulting algorithm.

Report question issue

Notes

Question 20 Notes

Unsure

Notification

The Floyd Warshall all pairs shortest path algorithm works as follows:

- Iterates over all the vertices in the graph. Call the current vertex "k".
- For each *k*, look at every pair of vertices *i,j* . Try to find a path from *i* to *j* passing through *k*, which is shorter than the current shortest path from *i* to *j*.

Let us call *k* the "detour" vertex.

Suppose that you have the following distance matrix just before using vertex D as the detour vertex.
What is the state of the matrix after the iteration in which D is used as the detour vertex?

	A	B	C	D
A	0	-7	1	-5
B	∞	0	8	2
C	∞	∞	0	∞
D	4	∞	5	0

Please format your answer like this:
x,x,x,x;x,x,x,x;x,x,x,x;x,x,x,x
Where the "x" are replaced by the values in the distance matrix. "," indicates a new element, and ";" indicates the end of a row.
The values should be left to right, top to bottom. Represent infinity as "inf".

For example, the state shown in the diagram above would be:
0,-7,1,-5;inf,0,8,2;inf,inf,0,inf;4,inf,5,0

write your answer here

Questions attempted: 0/26

3

Marks

Consider the following **Union-Find data structure** (**Linked Trees with Size**) for a **Kruskal's** algorithm at some **step** ***k*** in finding a **Minimum Spanning Tree** (**MST**):

Vertex ID	0	1	2	3	4	5	6	7	8	9
Parent	1	-4	0	-1	6	1	-2	-1	-2	8

Then in plain words, answer the following:

- What is the size of the largest tree? What are the vertices in the largest tree?
- Describe the steps which would occur when performing the union(2,9) operation.

Report question issue

Notes

Question 22 Notes

Unsure

Notification

3

Marks

<u>Describe an algorithm</u> to determine if a **graph *G*** with ***V* vertices** and ***E* edges** have a <u>unique</u> **Minimum Spanning Tree (MST)** in a time complexity of <u>*O(E log V)*</u>.

If a unique MST exist, return the edges that form the MST. Else, return False.

A B I U x_2 x^2 ABC 123

Report question issue Notes

Question 23 Notes

Unsure

Notification

2

Marks

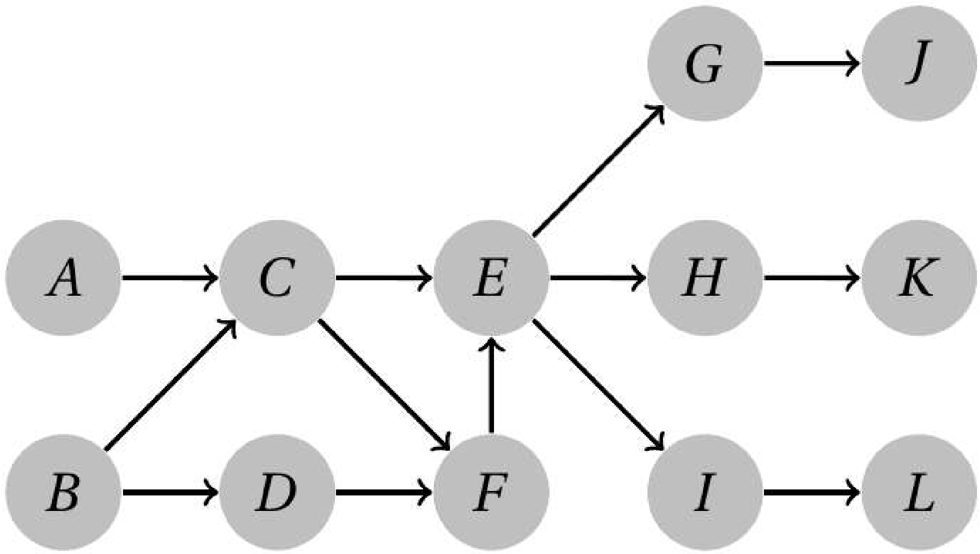
You are overseeing the development of an app. There are many features to implement, and some features rely on other features. Consider a **Directed Acyclic Graph** *G* with **vertices** *V* and **edges** *E*, where each vertex corresponds to a feature. There is an edge from *feature A* to *feature B* if *feature A* must be implemented **before** work can start on *feature B*.

Your superior has decided the project will be broken up into "**phases**". Each **phase** consists of implementing four features, but it does not matter which four, the phases are purely for marketing hype.

So the first four features will comprise phase 1, and the next four will comprise phase 2, etc. Note that if *feature B* requires *feature A,* they can both be implemented in the same phase, provided A is implemented first.

Describe an algorithm that returns the **first** phase in which you have a choice about which features to implement.

As an example, consider the following DAG:



F cannot be started until C and D are complete. E cannot be started until F and C are complete, so A,B,C,D must be completed before F or E can be started. Although they can be completed in different orders, phase 1 will always comprise A, B, C and D, so there is no choice.

In Phase 2, we must complete F, then E, and then we have 3 features we could implement next, but we only need 2 more features to finish phase 2. Therefore, we have a choice in what features we implement in phase 2, so the solution would be 2.

Notification

Questions attempted: 0/26

4

Marks

You are running a conference. The plan is to have the attendees eat at various local restaurants.

- Each attendee has indicated several places they would be happy to eat dinner.
- Each dining place has informed you of the most conference attendees that can eat there.
- You are concerned that it will not be possible to allow everyone to eat where they would prefer, but you want to let as many people as possible eat at one of their preferred places.

What is the largest number of attendees who can eat at a restaurant for which they have a preference?

1. Describe how to model this problem as a maximum flow problem.
2. State how you would solve the problem, once it was modeled as a maximum flow problem.

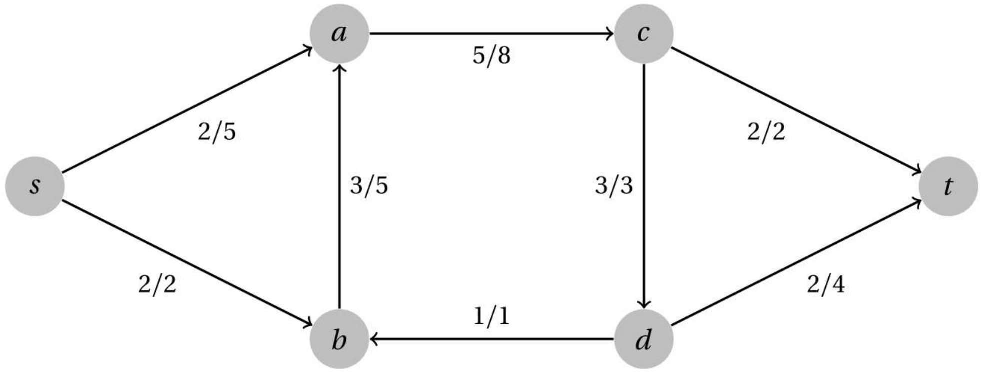
Report question issue Notes

Question 25 Notes

Notification

1 Mark

Consider the following flow network with source s and sink t. Currently there is a flow of 4 units in this network. Determine the maximum flow which can be sent through this network. Write your answer as a number, using digits (not words).



write your answer here

Report question issue

Notes

Unsure

Notification