UNIVERSITY OF RUHUNA

Faculty of Engineering

End-Semester 3 Examination in Engineering: March 2026

Module Number: EE3351 **Module Name: Data Structures and Algorithms (C-18)**

Part-A MCQ

[30 Minutes]

Instructions for candidates

- Write your index number on top of every page.
- Question paper contains 20 multiple choice questions. Each question carries 0.5 marks.
- Answer all questions. Each question has only one answer.
- **For each question, put an X mark on the letter: (a), (b), (c), or (d) which corresponds to the correct answer, by using a black or blue pen.**

---

1. An array stores elements in _____ memory locations.

   a) scattered
   b) contiguous
   c) dynamic
   d) random

2. Given `int A[5] = {1,2,3,4,5};`, what is the value of `*(A + 2)`?

   a) 2
   b) 1
   c) 4
   d) 3

3. What is the worst-case time complexity to delete an element from the middle of an array?

   a) O(log n)
   b) O(n)
   c) $O(n^2)$
   d) O(1)

4. The following code traverses a linked list. Fill in the blank:

```
Node* current = head;
while (current != NULL) {
    current = __________;
}
```

   a) head
   b) current→data
   c) NULL
   d) current→next

5. Which of the following is an advantage of linked lists over arrays?

a) Faster access by index
b) Fixed size
c) Efficient insertion and deletion
d) Less memory usage

6. The last node of a singly linked list points to _____.

a) head
b) NULL
c) itself
d) previous node

7. A stack follows the _____ principle.

a) FIFO
b) FILO
c) LIFO
d) LILO

8. In a stack where `top` points to the last element, overflow occurs when:

a) top < 0
b) top == 0
c) top == capacity
d) top == capacity - 1

9. Assume a stack is implemented using an array where the **top** pointer points to the **next empty position**. Which of the following correctly implements the **push** operation?

a)
```
top++;
data[top] = val;
```

b)
```
data[top] = val;
top++;
```

c)
```
top = val;
```

d)
```
data[top+1] = val;
```

10. Consider the following sequence of operations:

- Insert 10, 20, 30 into a queue (in that order)
- Remove one element from the queue and push it into a stack
- Remove another element from the queue and push it into the stack

What will be the output, if we pop one element from the stack?

a) 10
b) 20
c) 30
d) 10 20

11. In a queue, the next position of rear is calculated using:

a) rear % capacity
b) rear + 1
c) (rear + 1) % capacity
d) rear - 1

12. In a circular queue, if rear = 4 and capacity = 5, what is the next value of rear?

a) 5
b) 0
c) 4
d) 1

13. What is the average time complexity of searching in a balanced BST?

    a) O(n)
    b) O(log n)
    c) $O(n^2)$
    d) O(1)

14. In-order traversal of a BST produces elements in _____ order.

    a) random
    b) descending
    c) sorted
    d) reverse

15. Red-Black trees are _____ balanced compared to AVL trees.

    a) more strictly
    b) perfectly
    c) less strictly
    d) not

16. A collision occurs when:

    a) two keys are equal
    b) two keys map to the same index
    c) the table is full
    d) the key is NULL

17. In chaining, collisions are handled using a _____

    a) stack
    b) queue
    c) linked list
    d) array

18. In open addressing, which formula represents quadratic probing?

    a) $(h(k) + i) \bmod m$
    b) $(h(k) + i^2) \bmod m$
    c) $(h(k)i) \bmod m$
    d) $(h(k) - i) \bmod m$

19. Which statement correctly describes the difference between Breadth-First Search (BFS) and Depth-First Search (DFS)?

    a) BFS uses a stack and explores deepest nodes first, while DFS uses a queue and explores level by level.
    b) BFS uses a queue and explores level by level, while DFS uses a stack (or recursion) and explores along each branch before backtracking.
    c) Both BFS and DFS use a queue, but DFS ignores visited nodes.
    d) BFS works only on DAGs, while DFS works on all graphs.

20. What is the greedy choice in Kruskal's Algorithm for finding a Minimum Spanning Tree?

    a) Select the closest vertex to the source
    b) Select the maximum weight edge
    c) Select the smallest edge that does not form a cycle
    d) Select edges randomly until N-1 edges are chosen

UNIVERSITY OF RUHUNA

Faculty of Engineering

End-Semester 3 Examination in Engineering: March 2026

Module Number: EE3351 Module Name: Data Structures and Algorithms (C-18)

Part-B Essay

[1.5 Hours]

[Answer **all questions**, each question carries 10 marks]

Q1. a) Consider the following array of integers:

$$A = [3, 7, 12, 18, 25, 31, 42, 56]$$

i) Write a function to search for a given key in the array. The function should return the index of the key, if found. If the key is not found, return -1. You may use any Object Oriented Programming (OOP) language.

```
int search(int A[], int n, int key);
```

[1.0 mark]

ii) Calculate the best and worse case time complexities of your algorithm.

[1.0 mark]

b) Write a function to remove all occurrences of a given value `elem` from a singly linked list of integers. Assume the function is part of a LinkedList class with `head` as an available attribute. Use any OOP language.

```
void deleteWith(int elem);
```

The function should correctly handle all cases, including deletions at the beginning, middle, and end of the list.

For example, given the list: $5 \rightarrow 4 \rightarrow 5 \rightarrow 5 \rightarrow 8 \rightarrow 2 \rightarrow 5$

After calling deleteWith(5), the resulting list should be: $4 \rightarrow 8 \rightarrow 2$

[2.0 marks]

c) Stack and Queue are two logical data structures.

i) You are given two Queues, q1 and q2. Complete the following functions to implement a Stack using two Queues by filling in the blanks.
You may use the following queue operations:

- `enqueue(x)` $\rightarrow$ insert element
- `dequeue()` $\rightarrow$ removes and returns the front element
- `isEmpty()` $\rightarrow$ checks whether the queue is empty

```
Queue q1, q2;

// Push operation
void push(int x) {
    q2.enqueue(x);

    while (!q1.isEmpty()) {
        q2.enqueue( __________ ); // (1)
    }

    // Swap q1 and q2
    Queue temp = q1;
    q1 = __________ ;            // (2)
    q2 = __________ ;            // (3)
}

// Pop operation
int pop() {
    if (q1.isEmpty()) {
        return -1;
    }

    return __________ ;          // (4)
}
```

[2.0 marks]

ii) Calculate the time complexity of the push and pop operations in the above Stack data structure.

[1.0 mark]

d) A hash table of size 7 is used to store integer keys. The following hash function is defined:

$$h(k) = (k + \lfloor k/10 \rfloor) \bmod 7$$

The following keys are to be inserted into the hash table in the given order:

$$23, 45, 12, 39, 56, 78, 34$$

Collisions are resolved using linear probing.

i) Insert the given keys into the hash table. Show all intermediate steps, including how collisions are resolved. Draw the final hash table.

[2.0 marks]

ii) State the time complexity of insertion in the hash table *with and without* collisions.

[1.0 mark]

Q2. Trees are fundamental data structures used for efficient searching and organization.

a) Examine the tree structure illustrated below:

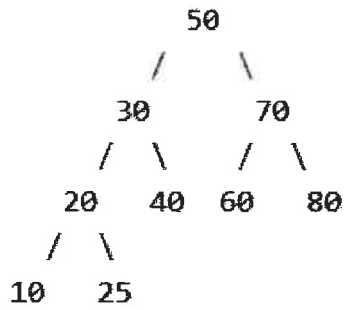

Figure Q2.a): A sample tree

Identify the following entities/properties for the tree above:

i) The Root node
ii) All Leaf nodes
iii) Parent of Node 20
iv) The total Depth of the tree (assuming the Root is at Level 1)

[2.0 marks]

Draw the resulting tree after deleting following nodes:

v) Node 70
vi) Node 50

[1.0 mark]

b) Consider the following sequence of keys: 10, 20, 30, 40, 50, 60

i) Construct a Binary Search Tree (BST) by inserting the keys in the given order. Show the tree after each insertion.

[1.0 mark]

ii) Construct an AVL Tree by inserting the keys in the given order. Show the required rotations and the corresponding tree after each insertion.

[2.0 marks]

iii) Construct a Red-Black Tree by inserting the keys in the given order. Clearly indicate the node colors and the corresponding tree after each insertion.

[2.0 marks]

iv) Compare and contrast the three trees obtained above in terms of their structure, height, insertion efficiency, and search efficiency.

[1.0 mark]

v) Perform an In-order traversal of the AVL tree constructed in part Q2.b).ii).

[1.0 mark]

Q3. A Graph Data Structure is a non-linear data structure where data elements do not follow a sequential or hierarchical order.

a) Draw a 3-regular graph with at least four vertices.

[1.0 mark]

b) State two differences between a graph and a tree data structure.

[1.0 mark]

c) Represent the directed and weighted graph given in Figure Q3.c) using an adjacency matrix. Clearly show how edge weights are stored.

[2.0 marks]

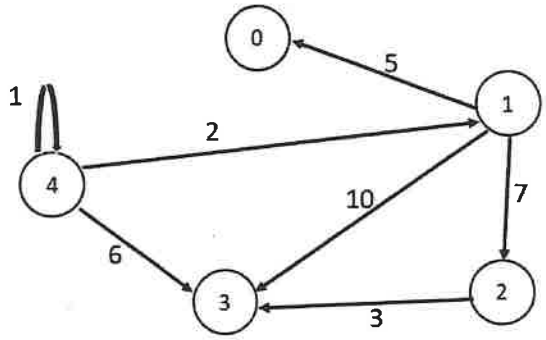

Figure Q3.c): A directed graph

d) Consider the weighted undirected graph shown in Figure Q3.d). Using Prim's Algorithm starting from **vertex 1**, find the Minimum Spanning Tree (MST) of the graph. Clearly show all the steps involved in constructing the MST, including: the edges selected at each step, the total cost calculation.

[2.0 marks]

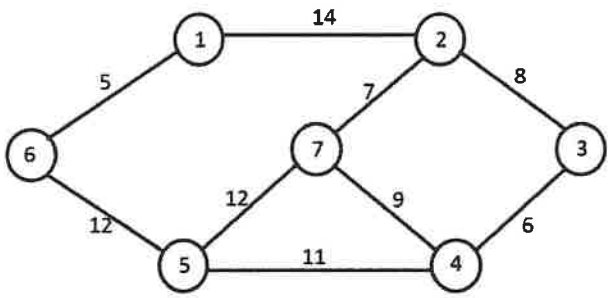

Figure Q3.d): An undirected graph

e) Algorithm analysis is the process of evaluating the performance of an algorithm in terms of the amount of time and space it requires to execute. Find the time complexity and the space complexity of the following code snippets with respect to input size **n**. Use the big O notation.

i)

```
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        count++;
    }
}
```

ii)

```
int i = 1;
while (i < n) {
    i = i * 2;
}
```

iii)

```
for (int i = 0; i < n; i++) {
    for (int j = 1; j < n; j = j * 2) {
        count++;
    }
}
```

iv)

```
int factorial(int n) {
    if (n == 0)
        return 1;
    else
        return n * factorial(n - 1);
}
```

[4.0 marks]