Started on	Tuesday, 18 March 2025, 11:25 AM
State	Finished
Completed on	Tuesday, 18 March 2025, 11:38 AM
Time taken	13 mins 23 secs
Marks	6.00/10.00
Grade	**60.00** out of 100.00

Question 1

Complete

Mark 0.00 out of 1.00

What is the maximum number of nodes in a binary tree of height `h` (where height is counted as the number of edges from root to the deepest node)?

- a. (2^{h+1} - 1)
- b. (h log h)
- c. (2^h - 1)
- d. (h^2)

Question 2

Complete

Mark 0.00 out of 1.00

Consider the following pseudo-code for a function `func(Node root)` applied to a binary tree. What does it compute? Function func(Node root): if root is NULL: return 0 return 1 + func(root.left) + func(root.right)

- a. Number of nodes in the tree
- b. Height of the tree
- c. Sum of all node values
- d. Maximum depth of the tree

Question 3

Complete

Mark 1.00 out of 1.00

Which of the following is always true for a full binary tree with `n` nodes?

- a. The tree is always balanced
- b. The height of the tree is always `log n`
- c. Every level is completely filled
- d. Every node has either 0 or 2 children

Question 4

Complete

Mark 1.00 out of 1.00

Given a BST, which of the following elements will always be found in the left subtree of a node with value `x`?

- a. Elements equal to `x`
- b. Elements less than `x`
- c. Elements greater than `x`
- d. All elements in the tree

Question 5

Complete

Mark 1.00 out of 1.00

What is the output of the following function when applied to a BST? Function findMin(Node root): if root is NULL: return NULL if root.left is NULL: return root.data return findMin(root.left)

- a. The sum of all nodes
- b. The maximum value in the BST
- c. The height of the BST
- d. The minimum value in the BST

Question 6

Complete

Mark 0.00 out of 1.00

What is the worst-case time complexity of deleting a node in an unbalanced BST with `n` nodes?

- a. O(1)
- b. O(n log n)
- c. O(log n)
- d. O(n)

Question 7

Complete

Mark 1.00 out of 1.00

Which of the following statements is true for Dijkstra's Algorithm?

- a. It finds the shortest path between all pairs of nodes
- b. It guarantees the shortest path in all cases
- c. It works only for graphs with non-negative weights
- d. It works correctly with negative-weight cycles

Question 8

Complete

Mark 0.00 out of 1.00

What is the time complexity of Depth-First Search (DFS) on a graph with `V` vertices and `E` edges using an adjacency matrix?

- a. O(V + E)
- b. O(E log V)
- c. O(V)
- d. $O(V^2)$

Question 9

Complete

Mark 1.00 out of 1.00

Which traversal method should be used to determine if a directed graph contains a cycle?

- a. Kruskal's Algorithm
- b. Depth-First Search (DFS) with recursion stack
- c. Breadth-First Search (BFS)
- d. Dijkstra's Algorithm

Question 10

Complete

Mark 1.00 out of 1.00

What is the output of the following function when applied to an undirected graph represented as an adjacency list? Function fun(Node start): Queue Q Add start to Q While Q is not empty: Node u = Q.dequeue() print u For each neighbor v of u: If v is not visited: Mark v as visited Add v to Q

- a. Breadth-First Traversal
- b. Detection of cycles
- c. Finding the minimum spanning tree
- d. Depth-First Traversal