ourdiu.com | Shared by Kazi Shajidul Islam Sobuj

# Daffodil International University

**Faculty of Science & Information Technology**
**Department of Computer Science & Engineering**
**Final Examination, Spring 2026**
**Course Code: CSE123, Course Title: Data Structure**
**Level: 1 Term: 3 Batch: 69**

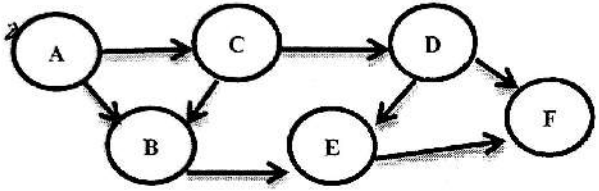
**Time: 02:00 Hrs** **Marks: 40**

## Answer ALL Questions

***[The figures in the right margin indicate the full marks and corresponding course outcomes. All portions of each question must be answered sequentially.]***

| | | | | |
|---|---|---|---|---|
| **1.** | | A tech company is developing a system to manage a large collection of unique user IDs. The records are stored in a simple rule: values smaller than a node are placed on one side and values greater than the node are placed on the other side. Since the system frequently performs search operations, the engineers want to maintain a tree structure that keeps the height as small as possible after every insertion to ensure consistently fast lookup times. The company has provided the following unique user IDs: (40, 20, 10, 60, 50, 70, 25, 30, 80) | | |
| | *a)* | Construct the tree step-by-step after inserting all the elements in the given order. | 5 | CO3 |
| | *b)* | Draw the updated tree after deleting the element 20 while maintaining height as small as possible. | 3 | |
| **2.** | | A library is organizing its catalog of unique book accession numbers for internal record-keeping. The records are stored in a hierarchical structure where each number is placed according to a simple rule: values smaller than a node are placed on one side and values greater than the node are placed on the other side. The structure is built based on the order in which the accession numbers are received, without applying any additional balancing or restructuring. The library system receives the following accession numbers: (55, 25, 75, 15, 35, 65, 85, 30, 45) | | CO2 |
| | *a)* | Write a C program to create the tree from the given data elements and display all the package IDs in ascending order based on the tree's ordering property. | 6 | |
| | *b)* | Write a function (deleteRecord) to remove a specific accession number from the structure while preserving the ordering rule. | 3 | |
| | *c)* | Write a function to find how many leaf nodes are there in Q2(a) tree. | 3 | |
| **3.** | | A cloud computing company is designing a system to manage task execution times. To ensure that the task with the smallest execution time is always processed first, the engineers store all task times in a special tree-based structure where the value at the top is always the smallest among all elements and the system allow duplicate element. To test the system, the following task execution times (in milliseconds) are inserted in order: (20, 15, 80, 25, 66, 50, 15, 90, 30, 10) | | CO3 |
| | *a)* | Build the tree step-by-step after inserting all execution time in the given order. | 4 | |
| | *b)* | Show the inorder, preorder and postorder of Q3(a) tree. | 3 | |

ourdiu.com | Shared by Kazi Shajidul Islam Sobuj

| | | | | |
|---|---|---|---|---|
| | *c)* | Draw the updated tree after deleting the smallest execution time related node from the tree. | 3 | |
| 4. | |  | | |
| | *a)* | Show the step-by-step procedure for finding the shortest path from node **A** to node **F** using either BFS or DFS and justify your choosing technique. | 6 | **CO3** |
| | *b)* | Apply the concept of topological sort to find the orderings of the above graph. | 4 | |