COMSATS University Islamabad (CUI)

Department of Computer Science
Lab MidTerm Examination Fall - 2024
BCS III SEMESTER

Course: CSC211-Data Structures **Instructor:** Tanveer Ahmed
Maximum Marks: 25 **Dated:** November 13, 2024 **Time Allowed:** 90 Minutes

(CLO – 4: Implement data structures and algorithms.)

Q#1: Time : 45 minutes [Marks =4+2*4=12]

TechGear Retail Solutions owns a warehouse where items are stored in aisles. Each aisle is organized so that items are accessible in the order they were placed, making tracking and retrieval easier.

In this system:

- Each item quantity is represented as a node in a linked list.
- Quantities at **even indices** represent **essential items** that require frequent replenishment (e.g., chargers, cables, common accessories).
- Quantities at **odd indices** represent **non-essential or premium items** that are restocked less frequently (e.g., high-end electronics or specialty equipment).

To maintain an optimal stock level and monitor replenishment needs, TechGear requires a summary of essential and non-essential inventory quantities. For efficient stock management, the following operations are implemented:

1. **Create_List_of_Item( ):** This function creates list of all items placed in a warehouse
2. **Total_essential_items()**: This function calculates the total quantity of all essential items.
3. **Total_non_essential_items()**: This function calculates the total quantity of all non-essential items.
4. **Summary_of_ essential_items at_ Beginning ( )**: Inserts a node at the start of the list representing the total count of essential items, providing a quick reference for essential stock levels.
5. **Summary_of_ non_essential_items_ at_End ( )**: Inserts a node at the end of the list representing the total count of non-essential items, allowing a quick assessment of premium stock levels without traversing the entire list.

Q#2: Time : 45 minutes [Marks = 4+ 3*3 = 13]

FinServe Solutions, a company specializing in managing and analyzing large volumes of financial transaction data for banks and other financial institutions, was recently tasked with tracking all transactions for a major bank. To support this essential data management and analysis task, the team has decided to use a Binary Search Tree (BST), where each node represents a transaction containing the following information:

- Transaction ID (int): A unique identifier for each transaction.
- Amount (float): The monetary value of the transaction.

When a transaction, especially a large one, is processed, it's added to the BST using an **insert** function, ensuring transactions are stored in an organized and efficient manner. FinServe has implemented the following operations to allow for rapid lookups, maintain ordered data, and manage transactions with minimal overhead:

1. **Identify_largest_transaction():**
   This function finds the transaction with the maximum amount, useful for auditing and identifying large or unusual transactions.
2. **Generate_a_report():**
   This function performs an in-order traversal of the BST, printing all transactions in ascending order, giving the bank valuable insights into transaction patterns over time.
3. **Quickly_verify_transaction():**
   This function checks whether a given Transaction ID exists, allowing customer service to confirm transaction details or investigate discrepancies efficiently