National University of Computer and Emerging Sciences

Data Structures (CL-2001)

Lab Final Exam (B)

Date: December 2nd, 2024

Course Instructor(s)

Ms. Alishba Subhani

Total Time: 120 Minutes
Total Points: 100
Total Weightage: 50
Total Questions: 03
Semester: Fall 2024
Campus: Karachi
Department: AI

Student Name _____ Roll No _____ Section _____ Student Signature _____

General Instructions:

Carefully read the following instructions before attempting the paper.

- The Exam paper consists of 3 questions on 2 printed sides of 1 page.
- In case of any ambiguity, you may make assumptions, but your assumption must not contradict any statement in the question paper. Also mention your assumptions.
- **DON'T** share your program, if your code is matched to any member of your class, both will get **straight F** in the course without asking who shared or who magically copied.
- All your files must be named by your roll number along with question number e.g. **K23-XXXX_Q1.cpp.**

Question # 01 (LLO #: 1) ***[20 Points, 15 Weightage]***

You are managing a network of radio towers for a telecommunications company. Each tower broadcasts signals with varying strengths to provide coverage across a region. The strength of a signal decreases with distance from a central monitoring station located at (0, 0). The farther a tower is from the station, the weaker the signal becomes.

To optimize network performance, you decide to identify the **k towers closest to the central monitoring station** so you can prioritize boosting their signals and ensure strong coverage in the immediate vicinity.

Given an array of points representing the locations of radio towers (x, y), find the **k nearest towers** to the central monitoring station (0, 0) using a **min-heap**. Use the Euclidean distance from (0, 0) as the priority.

Tower Locations: [(1, 3), (2, -2), (5, 8), (0, 1), (7, 6)], **k=3**

K Closest Towers: [(0, 1), (2, -2), (1, 3)]

Question # 02 (LLO #: 2) ***[40 Points, 20 Weightage]***

You are developing a feature for a financial application that tracks transaction amounts in a balanced tree structure to ensure efficient data retrieval. Each transaction amount is an integer, and transactions are inserted into an AVL tree to maintain balance for quick lookups and updates.
The application has a reporting feature that allows users to query the total sum of transactions within a specific range [L, R]. For example, users might want to calculate the sum of all transactions between $100 and $500.

Design an AVL tree to store transaction amounts. Implement a function that efficiently calculates the sum of all transaction amounts within a given range [L, R].

- **Transaction amounts to insert:** [200, 50, 300, 400, 700, 150]
- **Query range:** [100, 500]
- **Output:** 200 + 300 + 400 + 150 = 1050

Question # 03 (LLO #: 3) ***[40 Points, 15 Weightage]***

A school wants to analyze the distribution of students' marks for a specific exam. Each student's marks are recorded, and the school needs to calculate how many students scored each mark.

1. Use a **hash table** to count the number of students who scored each unique mark:
    - Handle collisions using **double hashing.**
    - Resize and rehash when the load factor exceeds 0.7.
2. Sort the marks in **ascending order** using **MergeSort.** If two marks have the same frequency, sort them in ascending order.

Input Example:

Marks: 85, 90, 75, 85, 60, 90, 85, 75

Output Example:

Sorted Marks by Frequency:

1. 60: 1 student
2. 75: 2 students
3. 85: 3 students
4. 90: 2 students