0258

National University of Computer and Emerging Sciences

Lahore Campus

Data Structures Lab (CL2001) **Final Exam**

Date: Dec 21st 2024

Total Time (Hrs): 2.5

Total Questions: 1

Total Marks: 40

Lab Instructor(s)

Mr. Junaid Hussain

Mr. Umar Farooq

Mr. Aqib Zeeshan

Mr. Durraiz Waseem

Ms. Hadiya Kashif

Ms. Sukhan Amir

Mr. Muhammad Ahmed

Ms. Marwa Khan

Ms. Anosha Khan

Ms. Hooria Najeeb

Mr. Muhammad Hashir

Roll No _______ Section _______ Student Signature _______

Submission Path: **\\Cactus1\Xeon\Fall 2024\Junaid Hussain\DS Lab\YourSection**

Attempt all the questions. No quires shell be entertained. If you have any confusion, write the assumptions in your code as comments. Drag and Drop the file into your respective submission folder, do not try to open it. Write your coder in a single .cpp file. The folder will close automatically on time, hence submission on time, submission of correct .cpp file with your rollNo is your responsibility. If you fail to do so, you will be awarded with zero marks. The Main file should display a proper menu. No Built-in Heap or Hasing is allowed.

CLO #: Understand and apply data structures such as heaps and hash maps to design and implement a dynamic leaderboard system for efficient real-time updates and queries

Q 1: Sports Car Racing Championship - Dynamic Leaderboard Management Using Heaps

Design a Dynamic Leaderboard Management System for a sports car racing championship. The leaderboard must handle real-time updates and queries efficiently. You are required to use Heap data structures (Min-Heap or Max-Heap) as a core component for optimizing operations. Use a Max-Heap to store cars based on scores (primary key) and carID (secondary key for tie-breaking). Use a Hash Map to store car details for efficient lookups and updates. Optionally, use a Min-Heap for top-3 queries if needed.

1. **Register a New Car (10 Marks)**
   - Each car has a unique identifier (carID) and starts with an initial score of 0.
   - The system should dynamically add new cars to the leaderboard during the championship.
2. **Update Scores (10 Marks)**
   - After each race, update the scores of one or more cars based on their performance.

- The system must efficiently handle bulk score updates and maintain heap properties.

3. **Top 3 Cars Query (10 Marks)**
   - Retrieve the top 3 cars with the highest scores, ranked in descending order.
   - If two cars have the same score, break ties using the lexicographical order of their carID.
   - The system should efficiently retrieve top cars without re-sorting the entire leaderboard.

4. **Dynamic Leaderboard Updates (10 Marks)**
   - The leaderboard should automatically maintain its order after any score update using the properties of heaps.

Example

Register Cars: {(A123, 0), (B456, 0), (C789, 0), (D101, 0)}

(a) **Update Scores:**

- A123 gains 15 points.
- C789 gains 20 points.
- B456 gains 15 points.
- D101 gains 20 points.

(b) **Heap Representation (Max-Heap):**

$$\{\{(C789, 20)\}, \{(D101, 20)\}, \{(A123, 15)\}, \{(B456, 15)\}\}$$

(c) **For Top 3 Cars: Output should be** $\{\{C789, 20\}, \{D101, 20\}, \{A123, 15\}\}$.

(d) **Dynamic Update:**

- If D101 gains 10 more points:
- Update leaderboard using heap operations.
- New Leaderboard: $\{\{D101, 30\}, \{C789, 20\}, \{A123, 15\}, \{B456, 15\}\}$.