National University of Computer and Emerging Sciences

Data Structures (CL2001)

Lab Final Exam

Date: December 4th 2024

Course Instructor(s)

Sir Shafique, Ms. Zainab Asif Jawed, Sir Misbah

Total Time: 120 minutes
Total Marks: 100
Total Questions: 03

Semester: FA-2024
Campus: Karachi
Dept: Computer science

Paper A

- You must comment your student ID on top of each file in the .cpp and word file.
- Name the .cpp file for each question according to Roll_No e.g. k23-xxxx_Q1.cpp, k23- xxxx _Q2.cpp etc.
- Create a ZIP folder of all your solutions and copy it in the local storage with the title K23-xxxx_A.
- Submission are on local storage that can be accessed using win+r keys and entering **\\172.16.5.43**
- address in the dialog box.
- Enter your username as khifast\K23xxxx and its assigned password.
- Zip folder needs to be pasted in the "Exam Submission\teacherName\CourseName.

Student Name	Roll No	Section	Student Signature

CLO # 1: Use & explain concepts related to basic and advanced data structures and describe their usage in terms of common algorithmic operations. **Marks: [15 wtg, 20 marks]**

Imagine a network system where a server needs to send data packets containing certain characters (string). Each character in the data stream represents a resource or a type of operation (such as a task). Each character's frequency is directly tied to how often it appears in the data stream, impacting network performance.

The system has the ability to remove some characters from the data stream to reduce the overall "**load**" on the system, but there's a limit to how many characters (denoted as k) can be removed from stream (string). The "**load**" of a string is defined as the sum of squares of the count of each distinct character. For example, consider the string "saideep", here frequencies of characters are s=1, a=1, i=1, e=2, d=1, p=1 and load of the string is 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 2^2 = 9. The goal is to minimize the "load" of the stream on the system by removing k characters from stream(string). Implement this using heap.

Example:

Input: A data packet stream abccc with a limit to remove 1 character (K = 1).

Output: After removing one c, the remaining characters are abcc, and the resulting "cost" or "value" is 1^2 + 1^2 + 2^2 = 6.

Input : str = aaab, K = 2

Output : 2

CLO # 2: Compare different data structures in terms of their relative efficiency and design effective solutions and algorithms that make use of them. **Marks: [20 wtg, 40 marks]**

You are tasked with implementing a Warehouse Management System where items in the warehouse are represented by unique integer IDs and descriptions. The items will be stored in a Binary Search Tree (BST), where each node contains: A unique integer representing the item. Description: A string representing the item's description. Your task is to Insert: Add an item (node) with a unique ID and description into the BST. Print Items in Range: Print the k items whose IDs does not fall within a specified range [L, R]. Find Closest Node: Identify the node whose ID is as far as possible to a given target ID. Return and print those nodes which does not contain the character provided by the user. Delete Multiples: Delete the m items whose IDs are not multiples of a given integer n.
For this implementation:

k: Number of nodes to print within a given range [L, R].
m: Number of nodes to delete.
n: Integer used to find multiples for deletion.

CLO # 3: Use & explain concepts related to basic and advanced data structures and describe their usage in terms of common algorithmic operations. **Marks: [15 wtg, 40 marks]**

You are tasked with developing a library management system for a bookshelf that holds a maximum of 20 books. Each book is identified by a name consisting of two uppercase letters (e.g., AB, CD, EF). The system must ensure that no two books with the same name occupy the same position. Each book's name will be mapped to a specific position on the shelf, which has 20 available spots. If multiple books map to the same position, a new shelf should be created for the colliding books. When adding a new book, the system checks if the position is already occupied. If it is, the book will be placed in the newly created shelf at that position. To facilitate efficient retrieval, books within each shelf must be organized by name using an appropriate sorting method based on the divide-and-conquer approach. The system must also support deleting books, ensuring that if a shelf becomes empty after deletion, it is either left empty or reused. Finally, the system should allow displaying all books, sorted by name, across all positions. Edge cases such as managing up to 20 books and handling empty positions must also be addressed.

Choose the appropriate type of hashing and sorting algorithm for the above scenario.