Data Structures and Algorithms
CS-F22
LAB-04

Issue Date: July 19,2024

Start Time: 8:15 AM
Completion Time: 11:15 M

Total Marks : 60

Objective:

In this lab, students will understand about priority and circular queues working. The objective is to reinforce understanding of different functionalities performed by different types of queue.

Instructions:

1) Follow the question instructions very carefully, no changes in function prototypes are allowed.
2) Your laptops must be on airplane mode.
3) Anyone caught in an act of plagiarism would be awarded an "F" grade in this Lab.

TASK 01: Implement real-life scenario of a hospital emergency room (ER) [20 Marks]

A **hospital emergency room (ER)** where patients need to be treated based on the severity of their condition. Patients with more critical conditions should be treated before those with less severe conditions.

Patient's class:

```
class Patient                                                    [2 Marks]
{
        string name;
        string disorder;
        string registrationNo;
        int severity;
public:
        Patient (string name, string disorder, string registrationNo, int severity);
}
```

ER's class:

```
class EmergencyRoom
{
 private:
        priorityQueue erQueue;
 public:
        void addPatient(const Patient& patient);   for adding patient to queue          [6 Marks]
        Patient treatNextPatient();                for removing recovered patient from the queue  [6 Marks]
        bool hasPatients() const;                  to check if any patient is left in the queue   [3 Marks]
        void displayAllPatients() const;           display all patients waiting in queue          [3 Marks]
                                                   (Display info of all patients in a clean and ordered format
                                                   (highest to lowest severity))
 };
```

TASK 02: Merge two sorted arrays using priority queue [15 Marks]

You are given with two arrays and their sizes respectively, you have to merge them in an array but with using priority queue. (You clearly don't have liberty to use any array sorting method like bubble, insertion, quick, merge sort, etc.)

Function Prototype: **int * merge (int arr1[], int arr2[], int m, int n)**
//m and n are the sizes of arr1 and arr2 respectively

Sample run:
Input:
arr1 = [1,2,3] m = 3
arr2 = [2,4,7] n = 3
Output:
Resultant = [1,2,2,3,4,7]

TASK 03: Design Front Middle Back queue using CDLL [25Marks]

Design a queue that supports push and pop operations in the front, middle, and back.

Implement the FrontMiddleBack class:

- FrontMiddleBack() Initializes the queue. [1 Mark]
- void pushFront(int val) Adds val to the front of the queue. [4 Marks]
- void pushMiddle(int val) Adds val to the middle of the queue. [4 Marks]
- void pushBack(int val) Adds val to the back of the queue. [4 Marks]
- int popFront() Removes the front element of the queue and returns it. If the queue is empty, return -1. [4 Marks]
- int popMiddle() Removes the middle element of the queue and returns it. If the queue is empty, return -1. [4 Marks]
- int popBack() Removes the back element of the queue and returns it. If the queue is empty, return -1. [4 Marks]