Data Structures and Algorithms Lab

CS-F23

LAB-08

Issue Date: March 24, 2025

Start Time: 11:45 AM
Submission Time: 12:45 AM

Total Marks: 25

The objective of this lab is to:

This lab aims to enhance students' understanding of **queues** and their applications in solving real-world problems efficiently.

Instructions:

1) Use **queues** to solve both tasks efficiently.
2) Do not use built-in stack libraries
3) Ensure correctness by testing with multiple **input cases**.
4) Implement each function **efficiently** to minimize time and space complexity.

Task (Bank Queue System)

The goal of this simulation is to model a bank queue system where different types of customers arrive at regular intervals and are served based on First In, First Out (FIFO) order. There are multiple service counters, each dedicated to specific customer types, and various rules govern how customers are assigned, how long they wait, and when additional counters open.

Customer Types & Assigned Counters

The bank has three types of customers, each assigned to specific counters:

1. **Senior Citizens & Disabled Customers (id 3)** → Served at Counter 1 & Counter 2
2. **Business Customers(id 2**) → Served at Counter 3
3. **Regular Customers (id 1)**→ Served at Counter 4 & Counter 5

Customer Arrival & Service Requests

- Each customer requests one of the following services, with different processing times:
  - Cash Deposit → 3 minutes
  - Cash Withdrawal → 4 minutes
  - Cheque Clearance → 6 minutes
  - Account Opening → 10 minutes
  - Loan Consultation → 15 minutes

How Customers Are Served?

1. FIFO Order in Assigned Queues

- Customers are assigned to their respective queues and are served in the order they arrive.
- Each counter serves one customer at a time.

2. Overcrowding Leads to Temporary Counters

- If a queue has more than 10 customers waiting, a temporary counter opens to speed up processing (only one extra counter can open).

Final Output (Simulation Results)

At the end of the simulation, the system should output:

✓ **Customers served by Counter c** → Total customers count that given counter number served.

✓ **If extra counter used or not** → Print that if extra counter used or not. (Just print)

Function Prototype:

int simulateBankQueue(Customer customers[], int size, int c);

```
struct Customer
{
  int id;
  int arrivalTime;
  int serviceTime;
};
```

Example
Input:

Customers = [{1,0,3}, {2,2,15}, {3,4,4},{1,6,6}, {3,8,10}] , c = 4

Customer ID	Arrival Time (mins)	Service Type	Service Time
1 (Regular)	0	Cash Deposit	3
2 (Business)	2	Loan Consultation	15
3 (Senior)	4	Cash Withdrawal	4
1 (Regular)	6	Cheque Clearance	6
3 (Senior)	8	Account Opening	10

Assign Customers to Their Queues

- Regular Customers (ID: 1, 4) → Assigned to Counter 4 & 5 (prioritize 4 if both are idle).
- Business Customers (ID: 2) → Assigned to Counter 3.
- Senior Customers (ID: 3, 5) → Assigned to Counter 1 & 2 (prioritize 1 if both are idle).

Processing Customers

1. Time = 0
   - Regular Queue → {1, 0, 3} starts at Counter 4 (finishes at T=3).
2. Time = 2
   - Business Queue → {2, 2, 15} starts at Counter 3 (finishes at T=17).
3. Time = 3
   - Regular Queue → {1, 6, 6} arrives but waits.
   - Counter 4 is free.
4. Time = 4
   - Senior Queue → {3, 4, 4} starts at Counter 1 (finishes at T=8).
5. Time = 6
   - Regular Queue → {1, 6, 6} starts at Counter 4 (finishes at T=12).
6. Time = 8
   - Senior Queue → {3, 8, 10} starts at Counter 1 (finishes at T=18).

Handling Extra Counters

- At no point does any queue have more than 10 customers waiting.
- No extra counter is opened in this simulation.

Final Output:

Customers served by counter 4 : 2
Prints: No extra counter used

Good Luck!

Note: You must complete all your tasks individually. Absolutely NO collaboration is allowed. Any case of plagiarism/cheating would result in 0 marks in sessional activities.