



**The University of Lahore**  
**Department of Computer Science & IT**

**CS-09204 Data Structures and Algorithm**  
**Fall 2025**

**Assignment # 2**

|                         |       |                            |
|-------------------------|-------|----------------------------|
| <b>Participant ID #</b> | _____ | <b>CLO:</b><br><b>PLO:</b> |
| <b>Total Marks:</b>     | 10    | <b>Obtained Marks:</b>     |

**Objective:**

Students will apply their knowledge of arrays, and linked lists, stack and Queue to solve a series of problems that simulate real-world scenarios.

---

**Question 1:**

A writing application wants to implement an **UNDO** feature that allows the user to reverse the most recent action.

Every time the user types a word, it is pushed onto a stack.

When the user presses UNDO, the recently typed word is removed.

**Task:**

Write a C++ program to implement the UNDO system using a **Stack implemented with an Array**.

**Your program must:**

1. Allow the user to input words and push them onto the stack.
2. On UNDO, pop the most recent word and display:

Undo performed. Removed: <word>

3. If the stack is empty and undo is attempted, show:

Nothing to undo.

4. Provide a menu for:
  - Adding a word (Push)
  - Performing Undo (Pop)
  - Displaying current text (Stack elements)
  - Exiting the program

### Operations in main():

- Push at least **5 words**
- Perform **2 undo operations**
- Display the remaining text on the screen

### Question 2:

A hospital wants to develop an **Emergency Check-In System** to manage incoming patients. Patients are treated on a **first-come, first-served** basis, but each patient record must store detailed information.

Each patient has the following attributes:

- **patient\_id** (integer)
- **name** (string)
- **age** (integer)
- **symptoms** (string)

Patients arrive one by one and are added to a queue.  
Doctors treat patients in the **exact order of arrival**.

#### Task:

Implement a **Queue using Singly Linked List** that manages emergency patient check-ins.

**Your queue must support these functions:**

#### 1. enqueue(patient\_id, name, age, symptoms)

Adds a new patient to the end of the queue.

#### 2. dequeue()

Removes and returns the patient at the front of the queue.

**If queue is empty, display:**

No patients waiting for treatment.

### 3. frontPatient()

Displays details of the patient currently at the front (next to be treated).

### 4. display()

Displays all patients in the queue in arrival order.

---

## Operations to Perform in main()

Perform the following operations in order:

1. Enqueue 3 patients:  
(101, "Ahmed", 45, "Chest pain")  
(102, "Sonia", 30, "High fever")  
(103, "Bilal", 60, "Breathing difficulty")
  2. Dequeue (treat) 1 patient.
  3. Enqueue 2 more patients:  
(104, "Mina", 28, "Migraine")  
(105, "Zaid", 50, "High blood pressure")
  4. Show the **patient currently at the front**.
  5. Display the **entire queue** with all remaining patients.
- 

## Sample Output (Example)

**Patient Added:** Ahmed

**Patient Added:** Sonia

**Patient Added:** Bilal

**Treating Patient:** Ahmed

**Patient Added:** Mina

**Patient Added:** Zaid

**Next Patient to Treat:**

ID: 102, Name: Sonia, Age: 30, Symptoms: High fever

**Current Emergency Queue:**

102 - Sonia (30) - High fever

103 - Bilal (60) - Breathing difficulty

104 - Mina (28) - Migraine

105 - Zaid (50) - High blood pressure