The University of Azad Jammu and Kashmir

Submitted by: M.Amir

Subject: DSA (discrete structure and algorithm)

Roll no: 23

Submitted to: Sir Zeshan

Task: Lab 5 (linked list)

Q. No.1: You are a software developer working for a hospital that manages patient check-ins.

Patients arrive, register, get treated, and leave. The hospital needs a dynamic system that can:

- Add new patients
- Remove treated patients
- Search a patient by ID
- Display all current patients

Since the number of patients changes continuously, the hospital wants a linked list–based system.

Your Task

Write a C++ program using a Singly Linked List where each patient has:

- Patient ID

Implement the following operations:

1. Insert a new patient at the end (new check-in)
2. Insert a patient at the beginning (emergency patient)
3. Display all patient

Solution

A Singly Linked List is a dynamic structure made of nodes.

Each node contains:

Patient ID

Pointer to the next node

Your task requires these operations:

1. Insert patient at the end → normal check-in

2. Insert patient at the beginning → emergency patient

3. Display all patients

C++ Program (Full Code With Comments)

```
#include
using namespace std;

struct Node {
int patientID;
Node* next;
Node(int id) : patientID(id), next(nullptr) {}
```

```
};

class LinkedList {
private:
Node* head;

public:
LinkedList() : head(nullptr) {}

// Insert patient at the beginning (emergency)
void insertAtBeginning(int id) {
Node* newNode = new Node(id);
newNode->next = head;
head = newNode;
}

// Insert patient at the end (normal check-in)
void insertAtEnd(int id) {
Node* newNode = new Node(id);
if (head == nullptr) {
head = newNode;
return;
}
Node* temp = head;
while (temp->next != nullptr) {
temp = temp->next;
}
temp->next = newNode;
```

```
}

// Display all patients in the list
void displayAll() {
if (head == nullptr) {
cout << "No patients in the list.\n";
return;
}

Node* temp = head;
cout << "Patients: ";
while (temp != nullptr) {
cout << temp->patientID;
if (temp->next != nullptr) cout << " -> ";
temp = temp->next;
}
cout << endl;
}
};

int main() {
LinkedList list;
int choice, id;

while (true) {
cout << "\nMenu\n";
cout << "1. Insert patient at end\n";
cout << "2. Insert patient at beginning\n";
cout << "3. Display all patients\n";
cout << "0. Exit\n";
cout << "Your choice: ";
cin >> choice;

switch (choice) {
case 1:
cout << "Enter patient ID: ";
cin >> id;
list.insertAtEnd(id);
break;

case 2:
cout << "Enter emergency patient ID: ";
cin >> id;
list.insertAtBeginning(id);
break;

case 3:
list.displayAll();
break;

case 0:
return 0;

default:
cout << "Invalid option.\n";
}
}
}
```

Output

```
Enter choice: 1
Enter patient ID to insert at end: 101
Inserted at end.

Enter choice: 2
Enter patient ID to insert at beginning: 200
Inserted at beginning.

Enter choice: 3
Current patients (IDs): 200 -> 101
```