**NAME : MALIHA SHAHID**

**ROLL NO:SU92-BSSEM-S24-071**

**LAB 4**

#include <iostream>

using namespace std;

class ListNode {

public:

int data;

ListNode\* nextNode;

ListNode(int val) {

data = val;

nextNode = NULL;

}

};

class LinkedList {

public:

ListNode\* first;

LinkedList() {

first = NULL;

}

void insertAtPosition(int position, int value) {

if (position < 1) {

cout << "Invalid position!" << endl;

return;

}

ListNode\* newNode = new ListNode(value);

if (position == 1) {

newNode->nextNode = first;

first = newNode;

return;

}

ListNode\* temp = first;

int index = 1;

while (temp != NULL && index < position - 1) {

temp = temp->nextNode;

index++;

}

if (temp == NULL) {

cout << "Position out of bounds!" << endl;

delete newNode;

return;

}

newNode->nextNode = temp->nextNode;

temp->nextNode = newNode;

}

void printList() {

if (first == NULL) {

cout << "The list is empty!" << endl;

return;

}

ListNode\* temp = first;

while (temp != NULL) {

cout << temp->data << " -> ";

temp = temp->nextNode;

}

cout << "NULL" << endl;

}

};

int main() {

LinkedList myList;

myList.insertAtPosition(1, 5);

myList.insertAtPosition(2, 10);

myList.insertAtPosition(3, 15);

myList.insertAtPosition(2, 20);

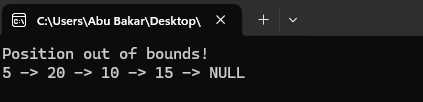
myList.insertAtPosition(10, 25);

myList.printList();

return 0;

}

**OUTPUT**

****

**EXPLANATION**

* ListNode Class – Represents a node with data and nextNode.
* LinkedList Class – Manages the linked list using first.
* insertAtPosition – Inserts a new node at a specific position.
* Handles Invalid Positions – Prevents out-of-bounds insertions.
* printList – Displays the linked list.
* main Function – Inserts nodes at different positions.
* Final Output – Prints the updated linked list.