

|  |
| --- |
| **LAB 7** of DSA LAB |

**Submitted by:**

**Name:** Muhammad shuraim

**Roll no:** SU92-S24-BSSEM-103

**Section:** 3A

**Submitted to**

Sir Rasikh Ali

**Question 1**

#include <iostream>

using namespace std;

class Node {

public:

int data;

Node\* next;

Node\* prev;

Node(int d) {

data = d;

next = nullptr;

prev = nullptr;

}

};

class DoublyLinkedList {

public:

Node\* head;

DoublyLinkedList() {

head = nullptr;

}

void insertAtStart(int d) {

Node\* newnode = new Node(d);

if (head == nullptr) {

head = newnode;

return;

}

newnode->next = head;

head->prev = newnode;

head = newnode;

}

void insertAtEnd(int d) {

Node\* newnode = new Node(d);

if (head == nullptr) {

head = newnode;

return;

}

Node\* temp = head;

while (temp->next != nullptr) {

temp = temp->next;

}

temp->next = newnode;

newnode->prev = temp;

}

void insertAtSpecLoc(int pos, int d) {

if (pos < 1) {

cout << "INVALID POSITION!" << endl;

return;

}

if (pos == 1) {

insertAtStart(d);

return;

}

Node\* newnode = new Node(d);

Node\* temp = head;

int count = 1;

while (temp != nullptr && count < pos - 1) {

temp = temp->next;

count++;

}

if (temp == nullptr) {

cout << "INVALID POSITION!" << endl;

return;

}

newnode->next = temp->next;

if (temp->next != nullptr) {

temp->next->prev = newnode;

}

temp->next = newnode;

newnode->prev = temp;

}

void insertAtCenter(int d) {

if (head == nullptr) {

insertAtStart(d);

return;

}

Node\* turtle = head;

Node\* tiger = head;

while (tiger->next != nullptr && tiger->next->next != nullptr) {

turtle = turtle->next;

tiger = tiger->next->next;

}

Node\* newnode = new Node(d);

newnode->next = turtle->next;

newnode->prev = turtle;

if (turtle->next != nullptr) {

turtle->next->prev = newnode;

}

turtle->next = newnode;

}

void display() {

Node\* temp = head;

while (temp != nullptr) {

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

temp = temp->next;

}

cout << "NULL" << endl;

}

void displayReverse() {

if (head == nullptr) return;

Node\* temp = head;

while (temp->next != nullptr) {

temp = temp->next;

}

while (temp != nullptr) {

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

temp = temp->prev;

}

cout << "NULL" << endl;

}

};

int main() {

DoublyLinkedList obj;

obj.insertAtStart(10);

obj.insertAtStart(20);

obj.insertAtEnd(30);

obj.insertAtEnd(40);

obj.insertAtSpecLoc(3, 25);

obj.insertAtCenter(15);

cout << "Doubly Linked List in Order: " << endl;

obj.display();

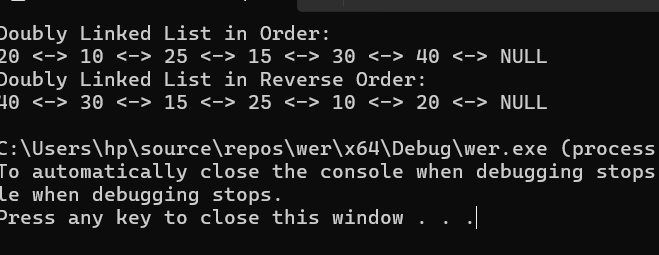
cout << "Doubly Linked List in Reverse Order: " << endl;

obj.displayReverse();

return 0;

}

**OUTPUT**



Explanation

1. Node Structure:

• The Node class represents a node with data, next (pointer to the next node), and prev (pointer to the previous node).

2. DoublyLinkedList Class:

• insertAtStart(int d) → Inserts a node at the beginning.

• insertAtEnd(int d) → Inserts a node at the end.

• insertAtSpecLoc(int pos, int d) → Inserts a node at a specific position.

• insertAtCenter(int d) → Inserts a node in the middle using the tortoise and tiger approach.

• display() → Prints the list from head to tail.

• displayReverse() → Prints the list from tail to head.

3. Main Function:

1. Inserts values (20, 10 at start), (30, 40 at end), 25 at position 3, 15 in the middle.

2. Displays the list in forward and reverse order.