# Lab Task 7

## Code Explanation:

Doubly Linked List implementation with insertions at start, end, and a specific position, along with forward and reverse display.

## Code:

#include <iostream>  
using namespace std;  
  
class Node {  
public:  
 int data;  
 Node\* prev;  
 Node\* next;  
 Node(int val) : data(val), prev(nullptr), next(nullptr) {}  
};  
  
class DoublyLinkedList {  
private:  
 Node\* head;  
 Node\* tail;  
 int size;  
  
public:  
 DoublyLinkedList() : head(nullptr), tail(nullptr), size(0) {}  
  
 void insertFirst(int val) {  
 Node\* newNode = new Node(val);  
 if (!head) head = tail = newNode;  
 else {  
 newNode->next = head;  
 head->prev = newNode;  
 head = newNode;  
 }  
 size++;  
 }  
  
 void insertLast(int val) {  
 Node\* newNode = new Node(val);  
 if (!tail) head = tail = newNode;  
 else {  
 tail->next = newNode;  
 newNode->prev = tail;  
 tail = newNode;  
 }  
 size++;  
 }  
  
 void insertAt(int val, int pos) {  
 if (pos < 1 || pos > size + 1) return;  
 if (pos == 1) return insertFirst(val);  
 if (pos == size + 1) return insertLast(val);  
  
 Node\* newNode = new Node(val);  
 Node\* temp = head;  
 for (int i = 1; i < pos - 1; i++) temp = temp->next;  
  
 newNode->next = temp->next;  
 newNode->prev = temp;  
 temp->next->prev = newNode;  
 temp->next = newNode;  
 size++;  
 }  
  
 void insertCenter(int val) {  
 insertAt(val, (size / 2) + 1);  
 }  
  
 void display(bool reverse = false) {  
 Node\* temp = reverse ? tail : head;  
 while (temp) {  
 cout << temp->data << " ";  
 temp = reverse ? temp->prev : temp->next;  
 }  
 cout << endl;  
 }  
};  
  
int main() {  
 DoublyLinkedList dll;  
 dll.insertFirst(10);  
 dll.insertLast(30);  
 dll.insertAt(20, 2);  
 dll.insertCenter(25);  
 dll.display();  
 dll.display(true);  
 return 0;  
}

## Output:

```  
10 20 25 30  
30 25 20 10  
  
```