**NAME : MALIHA SHAHID**

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

**LAB 8**

**Question # 02**

#include <iostream>

using namespace std;

class DoubleNode {

public:

int value;

DoubleNode\* nextNode;

DoubleNode\* prevNode;

DoubleNode(int v) {

value = v;

nextNode =NULL;

prevNode = NULL;

}

};

class DoubleLinkedList {

public:

DoubleNode\* firstNode;

DoubleLinkedList() {

firstNode = NULL;

}

void append(int val) {

DoubleNode\* newNode = new DoubleNode(val);

if (firstNode == NULL) {

firstNode = newNode;

}

else {

DoubleNode\* temp = firstNode;

while (temp->nextNode != NULL) {

temp = temp->nextNode;

}

temp->nextNode = newNode;

newNode->prevNode = temp;

}

}

void showList() {

DoubleNode\* temp = firstNode;

while (temp != NULL) {

cout << temp->value << " <-> ";

temp = temp->nextNode;

}

cout << "NULL" << endl;

}

};

DoubleNode\* mergeLists(DoubleNode\* list1, DoubleNode\* list2) {

if (!list1) return list2;

if (!list2) return list1;

DoubleNode\* mergedHead = NULL;

if (list1->value < list2->value) {

mergedHead = list1;

list1 = list1->nextNode;

}

else {

mergedHead = list2;

list2 = list2->nextNode;

}

DoubleNode\* temp = mergedHead;

while (list1 && list2) {

if (list1->value < list2->value) {

temp->nextNode = list1;

list1->prevNode = temp;

list1 = list1->nextNode;

}

else {

temp->nextNode = list2;

list2->prevNode = temp;

list2 = list2->nextNode;

}

temp = temp->nextNode;

}

if (list1) temp->nextNode = list1, list1->prevNode = temp;

if (list2) temp->nextNode = list2, list2->prevNode = temp;

return mergedHead;

}

int main() {

DoubleLinkedList dll1, dll2;

dll1.append(1);

dll1.append(3);

dll1.append(5);

dll2.append(2);

dll2.append(4);

dll2.append(6);

cout << "First Doubly Linked List: ";

dll1.showList();

cout << "Second Doubly Linked List: ";

dll2.showList();

DoubleLinkedList mergedDLL;

mergedDLL.firstNode = mergeLists(dll1.firstNode, dll2.firstNode);

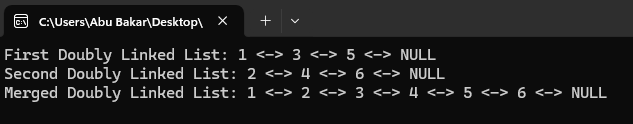
cout << "Merged Doubly Linked List: ";

mergedDLL.showList();

return 0;

}

**OUTPUT**

****

**EXPLANATION**

* **DoubleNode Class**: Represents a node with a value, a pointer to the next node, and a pointer to the previous node.
* **DoubleLinkedList Class**: Manages a doubly linked list with functions to append and display nodes.
* **append(int val):** Adds a new node to the end of the doubly linked list.
* **showList():** Prints the linked list in a readable format using <-> to indicate links.
* **mergeLists():** Merges two sorted doubly linked lists into a single sorted list.
* **main() Function**: Creates two doubly linked lists and fills them with values.
* **Merging**: Calls mergeLists() to combine the two lists in sorted order.
* **Output:** Displays the original and merged doubly linked lists.