```cpp
#include<bits/stdc++.h>
using namespace std;

struct node
{
    int data;
    node* next;

    node(int value)
    {
        data = value;
        next = nullptr;
    }
};

class LinkedList
{
public:
    node* head;

    LinkedList()
    {
        head = nullptr;
    }

    void append(int value)
    {
        node* newnode = new node(value);
        if(head == nullptr)
            head = newnode;
        else
        {
            node* temp = head;
            while(temp->next != nullptr)
            {
                temp = temp->next;
            }
            temp->next = newnode;
        }
    }

    void display()
    {
        node* temp = head;
        while(temp != nullptr)
        {
            cout << temp->data << " ";
            temp = temp->next;
        }
        cout << endl;
    }
};

LinkedList mergeSortedLists(LinkedList& ListA, LinkedList& ListB)
{
    LinkedList resultList;
    node* headA = ListA.head;
    node* headB = ListB.head;

    node* temp = nullptr;

    // Start merging the lists
    while (headA != nullptr && headB != nullptr)
    {
        if (headA->data <= headB->data)
        {
```

```
            if (resultList.head == nullptr)
            {
                resultList.head = headA;
                temp = headA;
            }
            else
            {
                temp->next = headA;
                temp = temp->next;
            }
            headA = headA->next;
        }
        else
        {
            if (resultList.head == nullptr)
            {
                resultList.head = headB;
                temp = headB;
            }
            else
            {
                temp->next = headB;
                temp = temp->next;
            }
            headB = headB->next;
        }
    }

    // Append the remaining elements from ListA or ListB
    if (headA != nullptr)
        temp->next = headA;
    if (headB != nullptr)
        temp->next = headB;

    return resultList;
}

int main()
{
    LinkedList ListA;
    LinkedList ListB;

    // Creating sorted list A
    ListA.append(1);
    ListA.append(3);
    ListA.append(5);

    // Creating sorted list B
    ListB.append(2);
    ListB.append(4);
    ListB.append(6);

    LinkedList mergedList = mergeSortedLists(ListA, ListB);

    cout << "Merged Sorted LinkedList: ";
    mergedList.display();

    return 0;
}
```