

```

1  #include <iostream>
2  using namespace std;
3
4  struct Node {
5      int data;
6      Node* next;
7
8      Node(int value) {
9          data = value;
10         next = nullptr;
11     }
12 };
13 class LinkedList {
14 public:
15     Node* head;
16
17     LinkedList() {
18         head = nullptr;
19     }
20
21     // Function to append data to the linked list
22     void append(int value) {
23         Node* newNode = new Node(value);
24         if (head == nullptr) {
25             head = newNode;
26         } else {
27             Node* temp = head;
28             while (temp->next != nullptr) {
29                 temp = temp->next;
30             }
31             temp->next = newNode;
32         }
33     }
34
35     // Function to display the linked list
36     void display() {
37         Node* temp = head;
38         while (temp != nullptr) {
39             cout << temp->data << " ";
40             temp = temp->next;
41         }
42         cout << endl;
43     }
44 };
45
46 LinkedList mergeLists(LinkedList& list1, LinkedList& list2) {
47     if (list1.head == nullptr) return list2;
48     if (list2.head == nullptr) return list1;
49
50     Node* temp = list1.head;
51     while (temp->next != nullptr) {
52         temp = temp->next;
53     }
54     temp->next = list2.head;
55
56     return list1;
57 }
58
59 int main() {
60     // Create Linked List A
61     LinkedList listA;
62     listA.append(1);
63     listA.append(3);
64     listA.append(5);
65
66     // Create Linked List B
67     LinkedList listB;
68     listB.append(2);

```

```
67     listB.append(4);
68     listB.append(6);
69
70     // Merge A and B
71     LinkedList mergedList = mergeLists(listA, listB);
72
73     // Display the merged list
74     cout << "Merged Linked List: ";
75     mergedList.display();
76
77     return 0;
78 }
```