```cpp
#include <iostream>

struct ListNode {
    int val;
    ListNode* next;
    ListNode(int val) : val(val), next(nullptr) {}
};

ListNode* mergeTwoLists(ListNode* list1, ListNode* list2) {
    // Create a dummy node to simplify the code.
    ListNode* dummy = new ListNode(0);
    ListNode* current = dummy;

    while (list1 && list2) {
        if (list1->val < list2->val) {
            current->next = list1;
            list1 = list1->next;
        } else {
            current->next = list2;
            list2 = list2->next;
        }
        current = current->next;
    }

    // If there are remaining nodes in list1 or list2, append them.
    if (list1) {
        current->next = list1;
    } else if (list2) {
        current->next = list2;
    }

    return dummy->next;
}

// Utility function to print a linked list.
void printList(ListNode* head) {
    ListNode* current = head;
    while (current) {
        std::cout << current->val << " -> ";
        current = current->next;
    }
    std::cout << "nullptr" << std::endl;
}

int main() {
    //Example 1:
    ListNode* list1 = new ListNode(1);
    list1->next = new ListNode(2);
    list1->next->next = new ListNode(4);

    ListNode* list2 = new ListNode(1);
    list2->next = new ListNode(3);
    list2->next->next = new ListNode(4);

    ListNode* result = mergeTwoLists(list1, list2);
    printList(result); // Print the merged list
    //Output: 1 -> 1 -> 2 -> 3 -> 4 -> 4

    //Example 2
    ListNode* list3 = nullptr;
    ListNode* list4 = nullptr;

    ListNode* result2 = mergeTwoLists(list3, list4);
    printList(result2); // Print the merged list
    //Output: nullptr

    //Example 3
    ListNode* list5 = nullptr;
    ListNode* list6 = new ListNode(0);

    ListNode* result3 = mergeTwoLists(list5, list6);
    printList(result3); // Print the merged list
    //Output: 0 -> nullptr


    // Clean up the memory to avoid memory leaks
    while (result) {
        ListNode* temp = result;
        result = result->next;
        delete temp;
    }

    return 0;
}

```