Problem 1.3.1: Merge Two Sorted Lists

Problem Statement: You are given the heads of two sorted linked lists list1 and list2. Merge the two lists into one sorted list. The list should be made by splicing together the nodes of the first two lists. Return the head of the merged linked list.

Code:

```
class Solution {
public:
    ListNode* mergeTwoLists(ListNode* list1, ListNode* list2) {
        if (list1 == nullptr) return list2;
        if (list2 == nullptr) return list1;
        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 (list1) current->next = list1;
        if (list2) current->next = list2;
        ListNode* result = dummy->next;
        delete dummy;
        return result;
    }
};
```

Problem 1.3.2: Remove Duplicates from Sorted List II

Problem Statement: Given the head of a sorted linked list, delete all nodes that have duplicate numbers, leaving only distinct numbers from the original list. Return the linked list sorted as well.

Code:

```
class Solution {
public:
    ListNode* deleteDuplicates(ListNode* head) {
        if (!head || !head->next)
            return head;
        ListNode* dummy = new ListNode(0);
        dummy->next = head;
        ListNode* prev = dummy;
        ListNode* current = head;
        while (current) {
            while (current->next && current->val == current->next->val) {
                current = current->next;
            }
            if (prev->next == current) {
                prev = prev->next;
            } else {
                prev->next = current->next;
            }
            current = current->next;
        }
        ListNode* result = dummy->next;
        delete dummy;
        return result;
    }
};
```