Experiment-3

Student Name: Suryansh Gehlot **UID: 22BCS10900**
Branch: BE-CSE **Section/Group: KRG 2B**
Semester: 6th **Date of Performance: 5-2-25**
Subject Name: AP Lab-2 **Subject Code: 22CSP-351**

1. Aim: Merge Two Sorted Lists

2. Objective:

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.

Example 1:

Input: list1 = [1,2,4], list2 = [1,3,4]

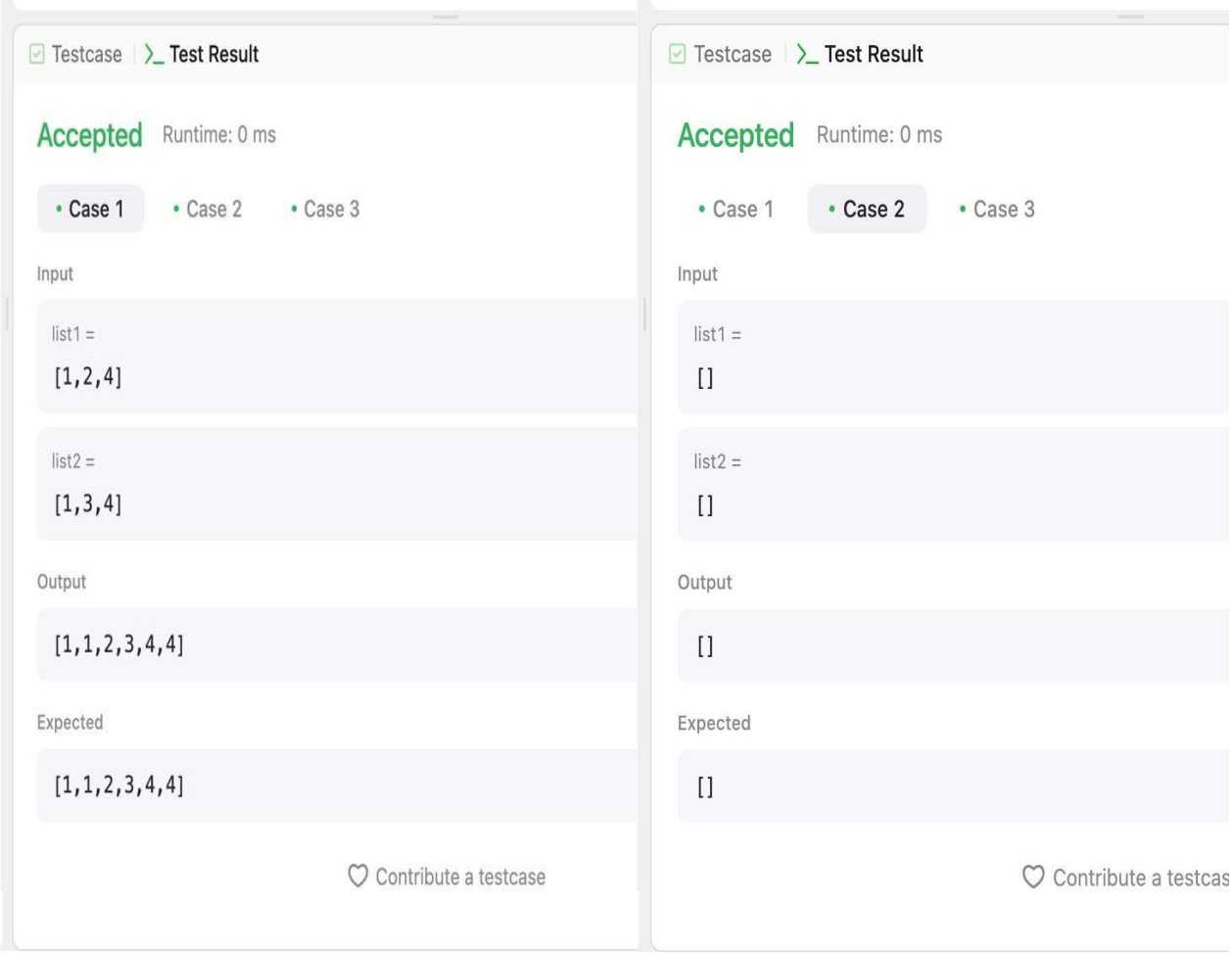
Output: [1,1,2,3,4,4]

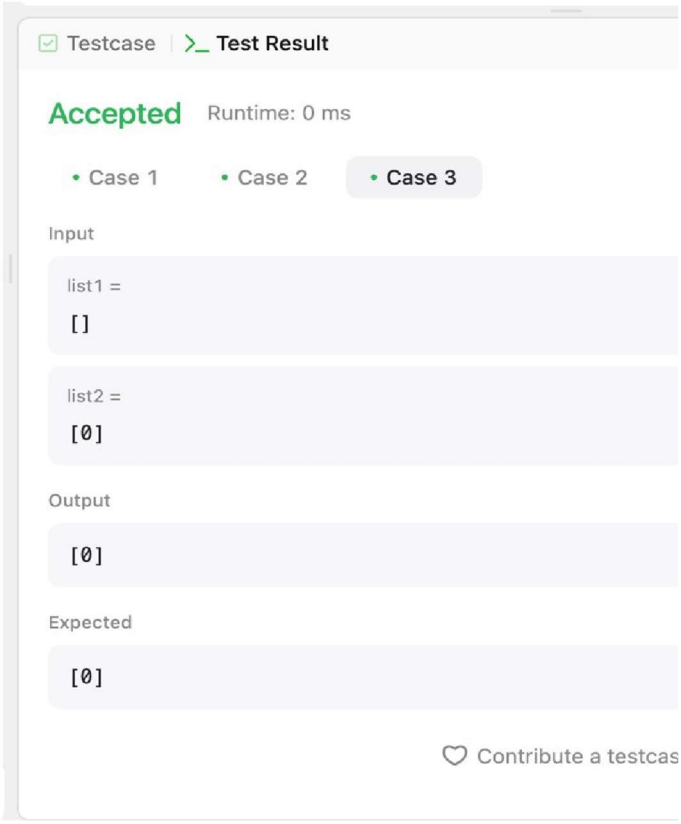
3. Implementation/Code:

```
class Solution {  public:
   ListNode* mergeTwoLists(ListNode* list1, ListNode* list2)
{        if (!list1)  return list2;          if (!list2)
```

```
        return list1;


        if (list1->val < list2->val) {              list1-
>next = mergeTwoLists(list1->next, list2);
return list1;
}  else
{
            list2->next = mergeTwoLists(list1, list2->next);
return list2;
        }

    }
};
```

4. Output


Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Case 3
Input
list1 =
[1,2,4]
list2 =
[1,3,4]
Output
[1,1,2,3,4,4]
Expected
[1,1,2,3,4,4]
Contribute a testcase
Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Case 3
Input
list1 =
[]
list2 =
[]
Output
[]
Expected
[]
Contribute a testcas



5. Learning Outcome:

a)We learn About the use of linked lists.
b)We learn About the use of recursive functions.
c)We learn About the use of loops.
e)We learn About the Calling For the function.

Question 2.

1. **Aim:** Remove Duplicates from Sorted List II

2. **Objective:**

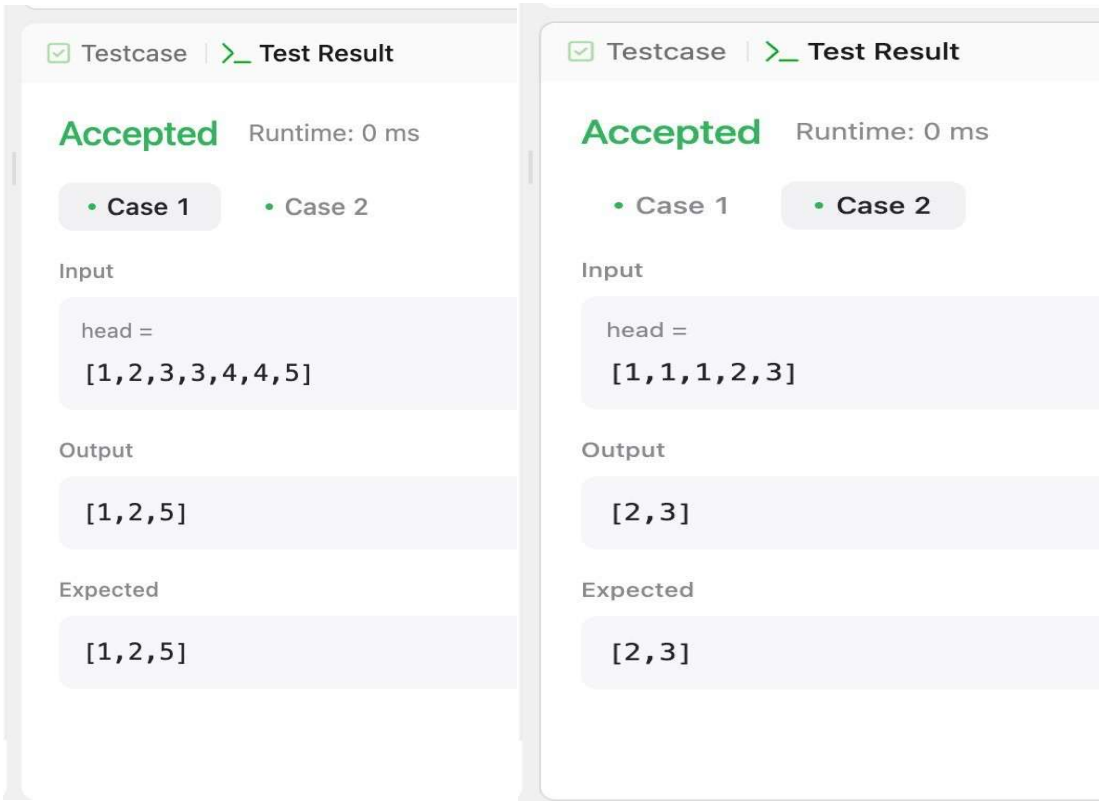
Given the head of a sorted linked list, delete all nodes that have duplicate numbers, leaving only distinct numbers from the original list. Return the linked list sorted as well.
Input: head = [1,2,3,3,4,4,5]
Output: [1,2,5]

3. **Implementation/Code:**

```
class Solution {  public:
    ListNode* deleteDuplicates(ListNode* head) {

        ListNode* dummy = new ListNode(0, head);
        ListNode* prev = dummy;

        while (head) {

            if (head->next && head->val == head->next->val) {

                while (head->next && head->val == head->next->val) {
                        head = head->next;
                }

                prev->next = head->next;
            } else {
```

```
            prev = prev->next;
        }

        head = head->next;
    }

    return dummy->next;
  }
};
```

4. Output



5. Learning Outcome:

1. We Learn About the use of linked lists.
2. We Learn About the use of nodes.
3. We Learn About the use of head , prev and next nodes.
4. We Learn About the Calling For the libraries.