NAME: SANIA MEMON
ROLLNO : 0091
LABS= DSA LAB03
LEETCODE SOLUTION

TASK01:

```java
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode(int x) {
 *         val = x;
 *         next = null;
 *     }
 * }
 */
public class Solution {
    public ListNode getIntersectionNode(ListNode headA, ListNode headB) {
        if(headA==null || headB==null){
            return null;
        }
        ListNode cA=headA;
        ListNode cB=headB;
        while(cA != cB){
            cA=(cA==null)? headB:cA.next;
            cB=(cB==null)? headA:cB.next;
        }
        return cA;
    }
```

TASK02:

```
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode() {}
 *     ListNode(int val) { this.val = val; }
 *     ListNode(int val, ListNode next) { this.val = val; this.next = next;
 * }
 */
class Solution {
    public ListNode deleteDuplicates(ListNode head) {
        ListNode current=head;
        while(current != null && current.next!=null){
            if(current.val==current.next.val){
                current.next=current.next.next;
            } else{
                current=current.next;
            }
        }
        return head;
    }
```

Accepted Runtime: 0 ms

• Case 1 • Case 2

Input

head =

[1,1,2]

Output

[1,2]

Expected

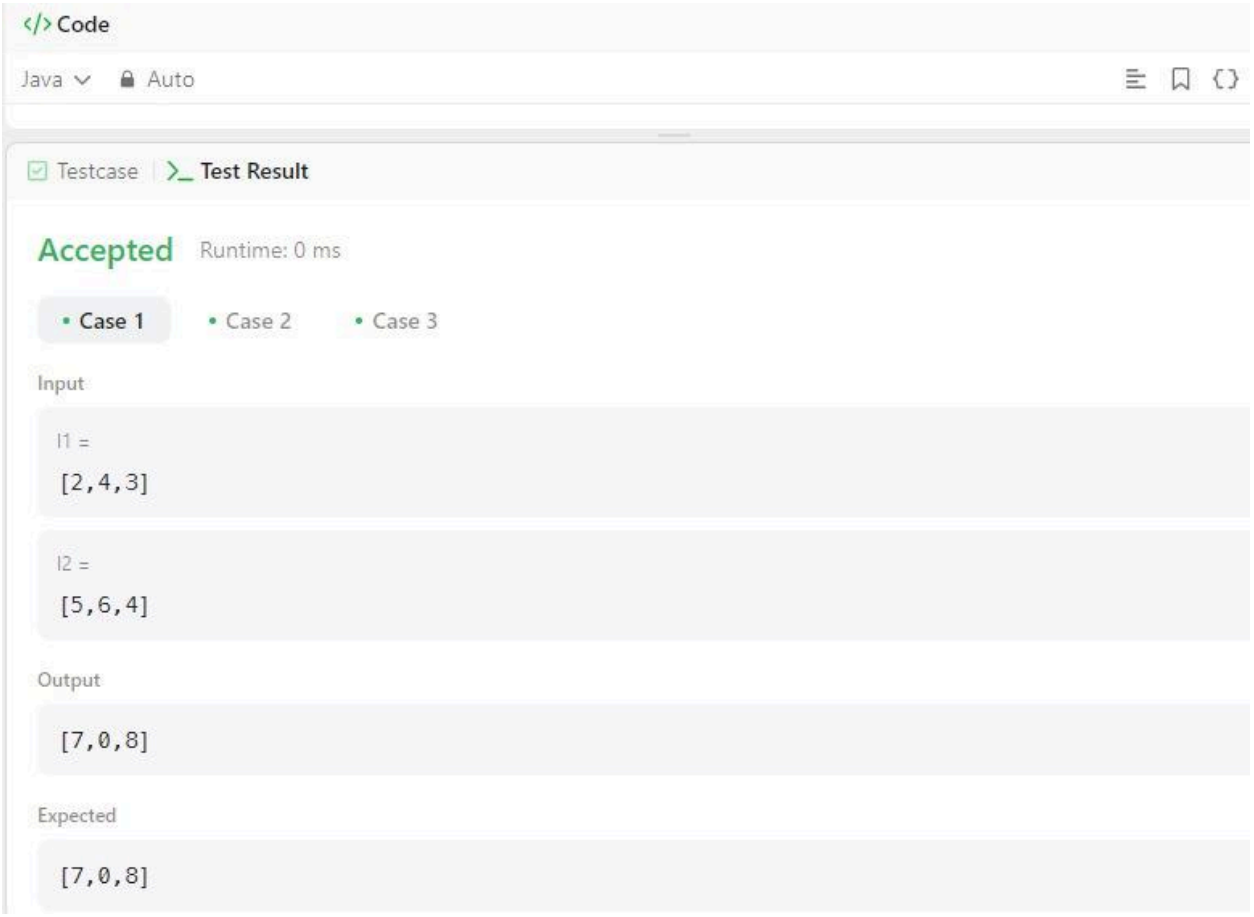
TASK03:

</> Code

Java Auto

```java
class Solution {
    public ListNode mergeTwoLists(ListNode list1, ListNode list2) {
        ListNode list = new ListNode(-1);
        ListNode current = list;
        while (list1 != null && list2 != null) {
            if (list1.val < list2.val) {

                current.next = list1;
                list1 = list1.next;
            } else {
                current.next = list2;
                list2 = list2.next;
            }
            current = current.next;
        }
        if (list1 != null) {
            current.next = list1;
        }
        if (list2 != null) {
            current.next = list2;
        }
        return list.next;
    }
}
```



TASK04:

```java
class Solution {
    public ListNode addTwoNumbers(ListNode l1, ListNode l2) {
        ListNode list=new ListNode(0);
        ListNode current=list;
        int carry=0;
        while(l1 != null || l2 != null || carry != 0){
            int val1=(l1 != null) ? l1.val : 0;
            int val2=(l2 != null) ? l2.val : 0;
            int sum = val1+val2+carry;
            carry=sum/10;
            int digit = sum%10;
            current.next=new ListNode(digit);
            current=current.next;
            if(l1 != null){
              l1=l1.next;
            }
            if(l2 != null) {
                l2=l2.next;
            }
        }
    }
    return list.next;
```

Accepted Runtime: 0 ms

• Case 1 • Case 2 • Case 3

Input

l1 =

```
[2,4,3]
```

l2 =

```
[5,6,4]
```

Output

```
[7,0,8]
```

Expected

```
[7,0,8]
```