

# LEETCODE PROBLEM – 83

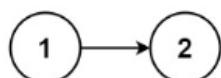
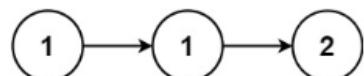
[Description](#) | [Editorial](#) | [Solutions](#) | [Submissions](#)

## 83. Remove Duplicates from Sorted List

[Easy](#) [Topics](#) [Companies](#)

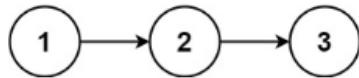
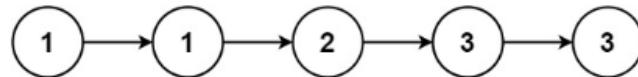
Given the `head` of a sorted linked list, *delete all duplicates such that each element appears only once*. Return the *linked list sorted* as well.

Example 1:



**Input:** head = [1,1,2]  
**Output:** [1,2]

Example 2:



**Input:** head = [1,1,2,3,3]  
**Output:** [1,2,3]

**Constraints:**

- The number of nodes in the list is in the range `[0, 300]`.
- `-100 <= Node.val <= 100`
- The list is guaranteed to be **sorted** in ascending order.

## </> Code



C ▾ 🔒 Auto

☰ ⌂ { } ⌂

```
1 struct ListNode* deleteDuplicates(struct ListNode* head) {
2     struct ListNode *curr = head;
3
4     while (curr != NULL && curr->next != NULL) {
5         if (curr->data == curr->next->data) {
6             curr->next = curr->next->next;
7         } else {
8             curr = curr->next;
9         }
10    }
11    return head;
12 }
13
```

Saved 🔒 Upgrade to Cloud Saving

Ln 13, Col

Testcase | > Test Result

**Accepted** Runtime: 0 ms

Case 1  Case 2

Input

```
head =
[1,1,2]
```

Output

```
[1,2]
```

Expected

```
[1,2]
```

Heart Contribute a testcase