

# Problem 82: Remove Duplicates from Sorted List II

## Problem Information

Difficulty: **Medium**

Acceptance Rate: 50.85%

Paid Only: No

Tags: Linked List, Two Pointers

## Problem Description

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.

**Example 1:**



**Input:** head = [1,2,3,3,4,4,5] **Output:** [1,2,5]

**Example 2:**



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

**C++:**

```

/**
 * Definition for singly-linked list.
 * struct ListNode {
 *   int val;
 *   ListNode *next;
 *   ListNode() : val(0), next(nullptr) {}
 *   ListNode(int x) : val(x), next(nullptr) {}
 *   ListNode(int x, ListNode *next) : val(x), next(next) {}
 * };
 */
class Solution {
public:
    ListNode* deleteDuplicates(ListNode* head) {

    }
};

```

## Java:

```

/**
 * 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) {

    }
}

```

## Python3:

```

# Definition for singly-linked list.
# class ListNode:
#     def __init__(self, val=0, next=None):
#         self.val = val
#         self.next = next
class Solution:

```

```
def deleteDuplicates(self, head: Optional[ListNode]) -> Optional[ListNode]:
```