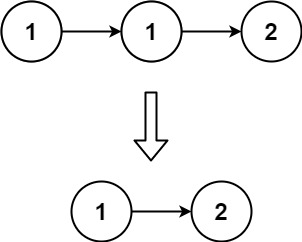
**DAY-8 TASK**

Problem-1

Link: https://leetcode.com/problems/remove-duplicates-from-sorted-list/description/?envType=problem-list-v2&envId=linked-list

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*.

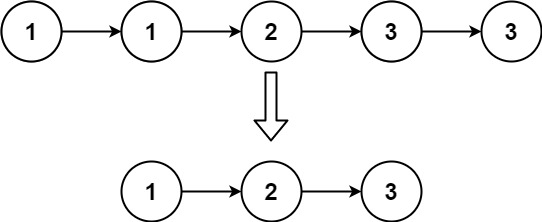
**Example 1:**



**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.

Problem-2

Link: https://www.geeksforgeeks.org/problems/detect-loop-in-linked-list/1

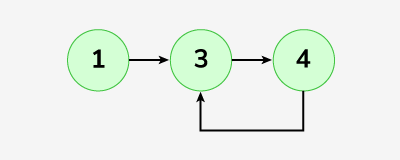
You are given the **head**of a singly linked list. Your task is to determine if the linked list contains a **loop**. A loop exists in a linked list if the next pointer of the last node points to any other node in the list (including itself), rather than being null.

**Custom Input format:**  
A**head**of a singly linked listand a**pos**(1-based index) which denotes the position of the node to which the last node points to. If **pos**= 0, it means the last node points to null, indicating there is no loop.

**Examples:**

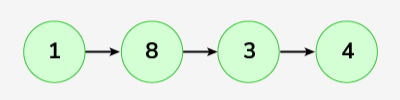
**Input:** head: 1 -> 3 -> 4, pos = 2

**Output:** true

**Explanation:** There exists a loop as last node is connected back to the second node. ****

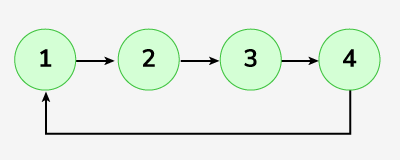
**Input:** head:1 -> 8 -> 3 -> 4, pos = 0

**Output:** false

**Explanation:** There exists no loop in given linked list. ****

**Input:** head: 1 -> 2 -> 3 -> 4, pos = 1

**Output:** true

**Explanation:** There exists a loop as last node is connected back to the first node.

**Constraints:**  
1 ≤ number of nodes ≤ 104  
1 ≤ node->data ≤ 103  
0 ≤ pos ≤ Number of nodes in Linked List

Try more examples