Difficulty: **Category:** Linked Lists **Successful Submissions:** 23,565+

Remove Duplicates From Linked List

You're given the head of a Singly Linked List whose nodes are in sorted order with respect to their values. Write a function that returns a modified version of the Linked List that doesn't contain any nodes with duplicate values. The Linked List should be modified in place (i.e., you shouldn't create a brand new list), and the modified Linked List should still have its nodes sorted with respect to their values.

Each `LinkedList` node has an integer `value` as well as a `next` node pointing to the next node in the list or to `None` / `null` if it's the tail of the list.

Sample Input

```
linkedList = 1 -> 1 -> 3 -> 4 -> 4 -> 4 -> 5 -> 6 -> 6 // the head node with value 1
```

Sample Output

```
1 -> 3 -> 4 -> 5 -> 6 // the head node with value 1
```

Hints

Hint 1

The brute-force approach to this problem is to use a hash table or a set to keep track of all node values that exist while traversing the linked list and to simply remove nodes that have a value that already exists. This approach works, but can you solve this problem without using an auxiliary data structure?

Hint 2

What does the fact that the nodes are sorted tell you about the location of all duplicate nodes? How can you use this fact to solve this problem with constant space?

Hint 3

Since the linked list's nodes are sorted, you can loop through them and, at each iteration, simply remove all successive nodes that have the same value as the current node. For each node, change its next pointer to the next node in the linked list that has a different value. This will remove all duplicate-value nodes.

Optimal Space & Time Complexity

O(n) time | O(1) space - where n is the number of nodes in the Linked List