# 83. Remove Duplicates from Sorted List



April 18, 2016 | 108.6K views

Average Rating: 4.69 (75 votes)

Given a sorted linked list, delete all duplicates such that each element appear only *once*.

**Example 1:**

```
Input: 1->1->2
Output: 1->2
```

**Example 2:**

```
Input: 1->1->2->3->3
Output: 1->2->3
```

## Solution

### Approach 1: Straight-Forward Approach

**Algorithm**

This is a simple problem that merely tests your ability to manipulate list node pointers. Because the input list is sorted, we can determine if a node is a duplicate by comparing its value to the node *after* it in the list. If it is a duplicate, we change the `next` pointer of the current node so that it skips the next node and points directly to the one after the next node.

Java

Copy

```java
public ListNode deleteDuplicates(ListNode head) {
    ListNode current = head;
    while (current != null && current.next != null) {
        if (current.next.val == current.val) {
            current.next = current.next.next;
        } else {
            current = current.next;
        }
    }
    return head;
}
```

**Complexity Analysis**

- Time complexity : $O(n)$. Because each node in the list is checked exactly once to determine if it is a duplicate or not, the total run time is $O(n)$, where $n$ is the number of nodes in the list.
- Space complexity : $O(1)$. No additional space is used.

**Correctness**

We can prove the correctness of this code by defining a *loop invariant*. A loop invariant is condition that is true before and after every iteration of the loop. In this case, a loop invariant that helps us prove correctness is this:

> All nodes in the list up to the pointer `current` do not contain duplicate elements.

We can prove that this condition is indeed a loop invariant by induction. Before going into the loop, `current` points to the head of the list. Therefore, the part of the list up to `current` contains only the head. And so it can not contain any duplicate elements. Now suppose `current` is now pointing to some node in the list (but not the last element), and the part of the list up to `current` contains no duplicate elements. After another loop iteration, one of two things happen.

1. `current.next` was a duplicate of `current`. In this case, the duplicate node at `current.next` is deleted, and `current` stays pointing to the same node as before. Therefore, the condition still holds; there are still no duplicates up to `current`.
2. `current.next` was not a duplicate of `current` (and, because the list is sorted, `current.next` is also not a duplicate of any other element appearing *before* `current`). In this case, `current` moves forward one step to point to `current.next`. Therefore, the condition still holds; there are no duplicates up to `current`.

At the last iteration of the loop, `current` must point to the last element, because afterwards, `current.next = null`. Therefore, after the loop ends, all elements up to the last element do not contain duplicates.

Rate this article:

Previous Next

## Comments: 65

Sort By

Type comment here... (Markdown is supported)

Preview Post

belfazt ★28 October 15, 2018 11:34 PM

Here's a python solution

```
class Solution(object):
    def deleteDuplicates(self, head):
        """
```

Read More

27 Share Reply

SHOW 5 REPLIES

terriblewhiteboard ★3833 May 6, 2020 6:56 AM Report

I made a video if anyone is having trouble understanding the solution (clickable link)
https://www.youtube.com/watch?v=gfgJjrkR-W4

Read More

19 Share Reply

wakil ★16 July 7, 2018 11:38 AM

easy java solution

16 Share Reply

EEEETTTT ★11 August 4, 2019 5:03 PM Report

easy cpp solution

```
class Solution {
```

Read More

7 Share Reply

SHOW 1 REPLY

amit12A ★7 January 14, 2018 9:43 AM

yes, we need else statement otherwise it won't work in case of duplicate values occur more than two times.

7 Share Reply

SHOW 2 REPLIES

kannanparamasiva1986 ★5 September 17, 2018 3:18 PM Report

In the solution mentioned here, I assume there are two issues.

1. The tail node is not checked, what if the duplicate value is in tail node.
2. When deleting a node, they are not cleaning up the references from the node which is going to be deleted, especially the next pointer of the node which is going to be deleted. Cleaning up the next

Read More

4 Share Reply

SHOW 1 REPLY

zhengzhicong ★294 November 20, 2018 6:51 AM

python3:

```
# Definition for singly-linked list.
# class ListNode:
#     def __init__(self, x):
```

Read More

6 Share Reply

SHOW 4 REPLIES

OlegStadnichenko ★6 September 2, 2018 8:35 PM Report

It's not clear should we change the input structure or create new. i mean in description.

2 Share Reply

SHOW 1 REPLY

TengLong_12138 ★4 December 10, 2017 8:32 PM

@alex1461 I think it's necessary as head always returns you the first node of your linkedlist.

2 Share Reply

yaaantinnng ★3 August 4, 2018 7:58 PM Report

why does it return head, not current?

3 Share Reply

SHOW 3 REPLIES