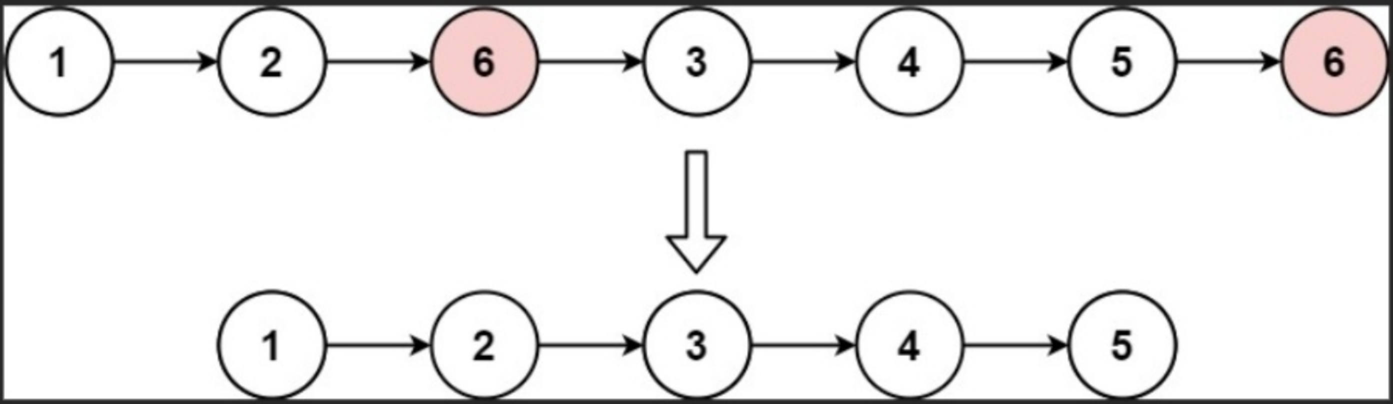
203. Remove Linked List Elements

Easy 7.7K 218

Companies

Given the `head` of a linked list and an integer `val`, remove all the nodes of the linked list that has `Node.val == val`, and return *the new head*.

Example 1:



```
Input: head = [1,2,6,3,4,5,6], val = 6
Output: [1,2,3,4,5]
```

Example 2:

```
Input: head = [], val = 1
Output: []
```

Example 3:

```
Input: head = [7,7,7,7], val = 7
Output: []
```

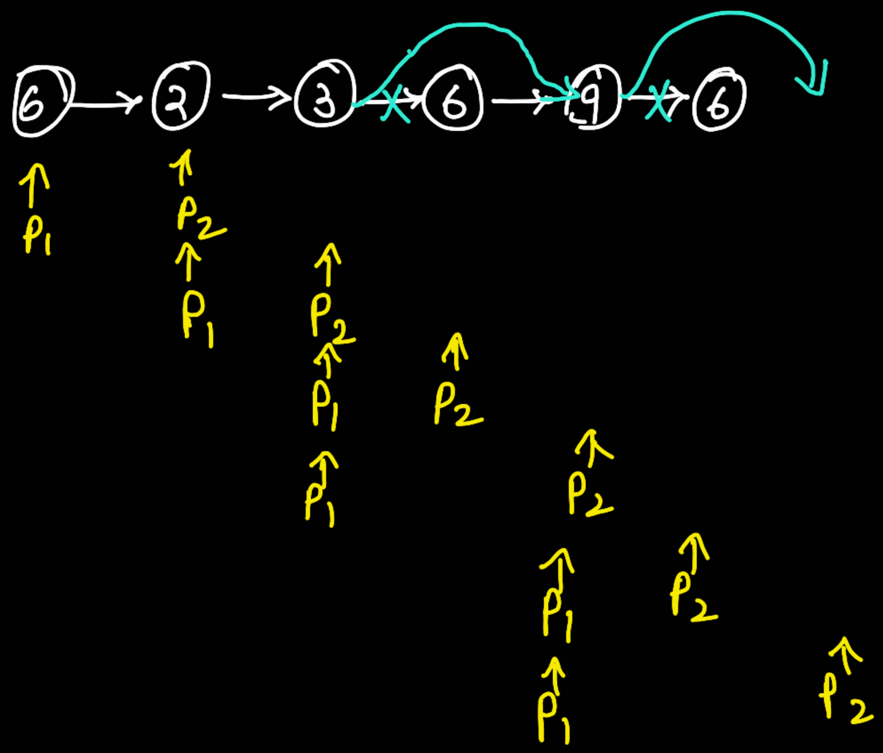
Constraints:

- The number of nodes in the list is in the range `[0, 10^4]`.
- `1 <= Node.val <= 50`
- `0 <= val <= 50`

Accepted 989.8K | Submissions 2.1M | Acceptance Rate 47.1%

Approach 1: Using two pointers

$x = 6$



As p_2 became NULL we stop.

Here a case was not handled if the first node value is x, then the above algorithm can't delete that.

So what we will do is after coming out of loop, we check

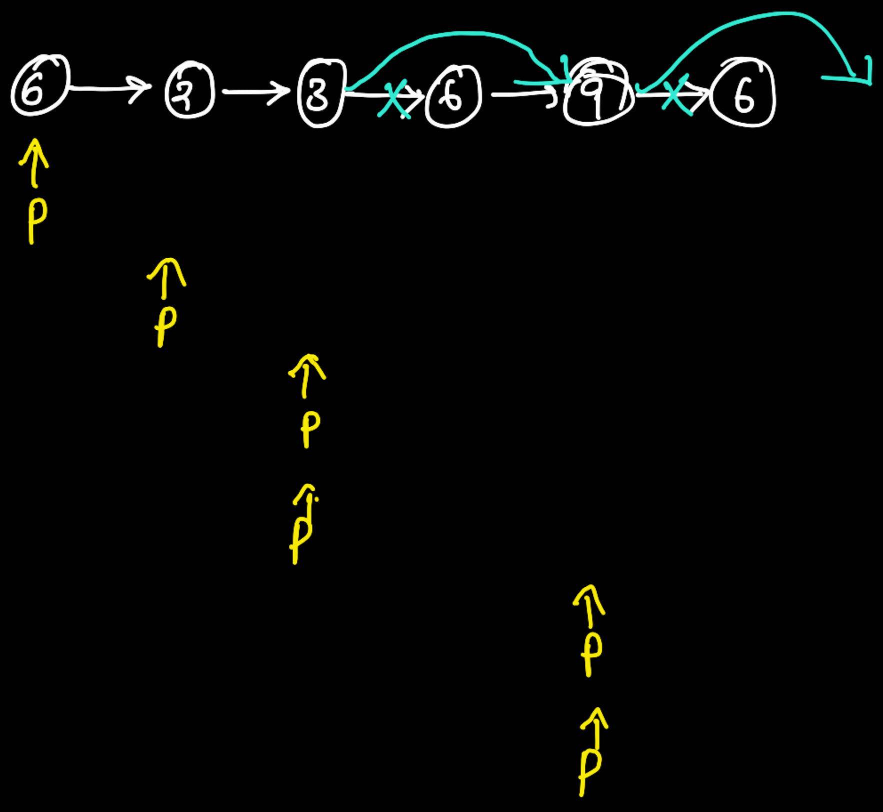
```
if( head→val == x)
    then make head = head→next

return head
```

$T(n) : O(n)$
$S(n) : O(1)$

Approach 2: Using only one pointer along with head

```
if(p→next→val == x)
    p→next = p→next→next
```



now as p→next is NULL we stop and check

```
if( head→val == x)
        head = head→next

return head
```

$T(n) : O(n)$

$S(n) : O(1)$