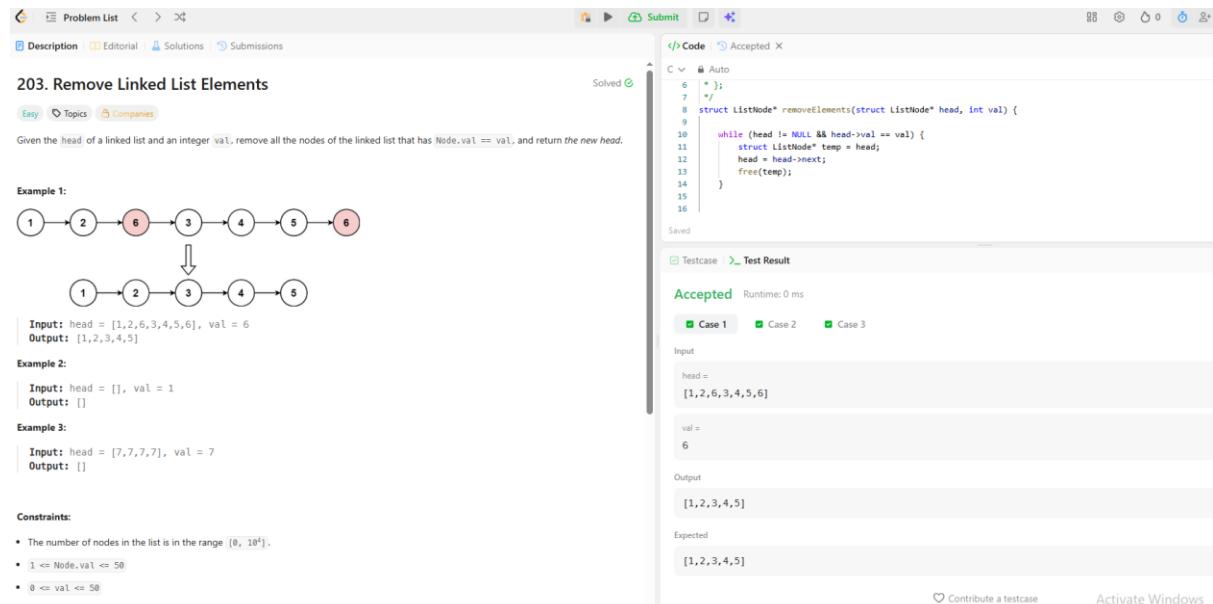


## Remove Linked List Elements

```
struct ListNode* removeElements(struct ListNode* head, int val) {  
  
    while (head != NULL && head->val == val) {  
        struct ListNode* temp = head;  
        head = head->next;  
        free(temp);  
    }  
  
    struct ListNode* curr = head;  
    while (curr != NULL && curr->next != NULL) {  
        if (curr->next->val == val) {  
            struct ListNode* temp = curr->next;  
            curr->next = curr->next->next;  
            free(temp);  
        } else {  
            curr = curr->next;  
        }  
    }  
  
    return head;  
}
```

## Output



The screenshot shows a code editor interface with the following details:

- Code Area:** The code provided in the question is pasted here.
- Test Result:** The code is marked as "Accepted" with a runtime of 0 ms.
- Testcases:** Three test cases are shown:
  - Case 1:** Input: head = [1,2,6,3,4,5,6], val = 6. Output: [1,2,3,4,5]. A diagram shows a linked list with nodes 1, 2, 6, 3, 4, 5, 6. Node 6 is highlighted in red, and an arrow points down to the result list [1, 2, 3, 4, 5] where node 6 is missing.
  - Case 2:** Input: head = [], val = 1. Output: []
  - Case 3:** Input: head = [7,7,7,7], val = 7. Output: []
- Constraints:**
  - The number of nodes in the list is in the range  $[0, 10^4]$ .
  - $1 \leq \text{Node}_i.\text{val} \leq 50$
  - $0 \leq \text{val} \leq 50$