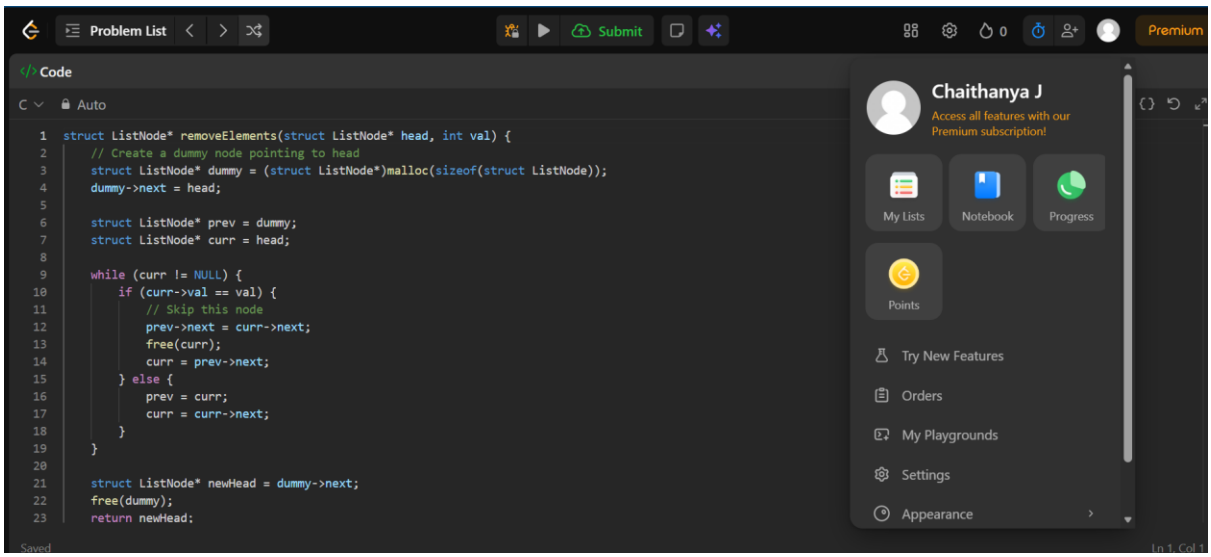
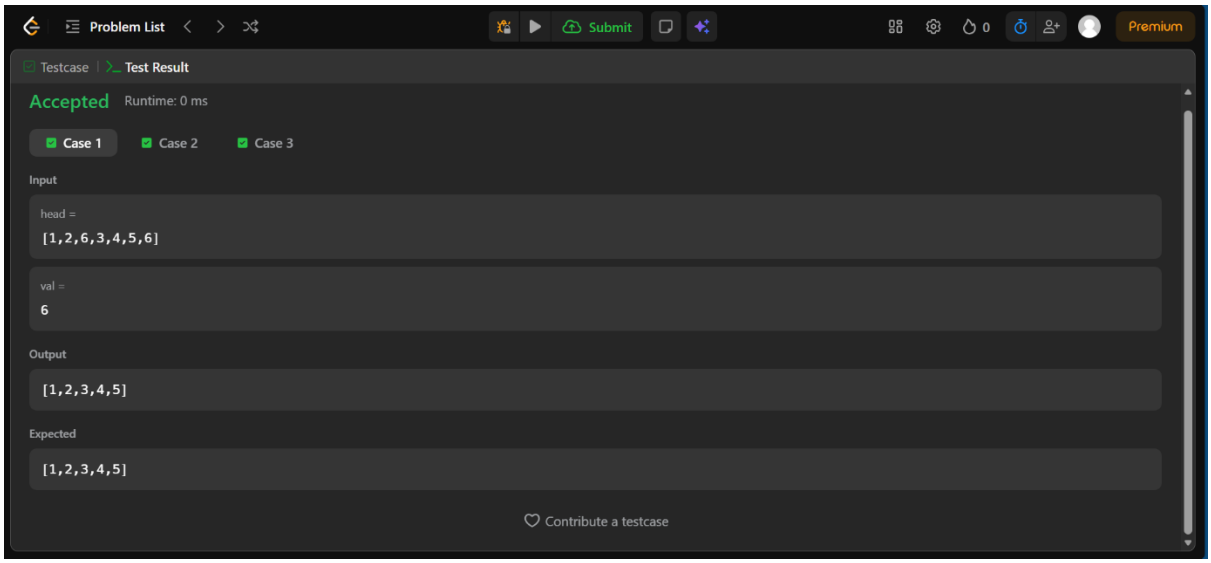
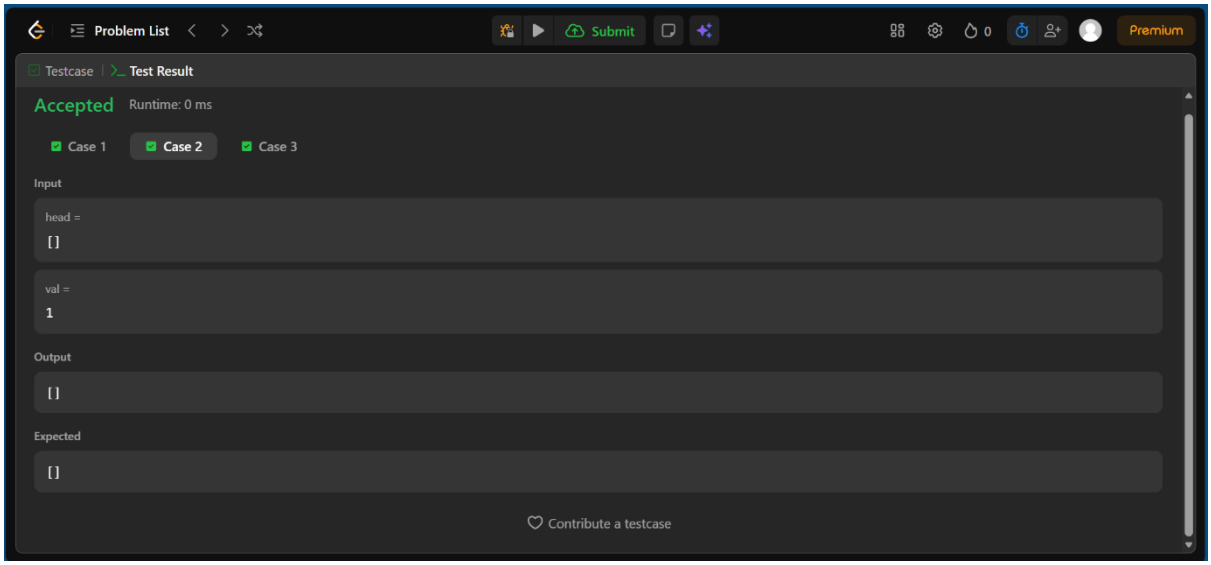
Leetcode:

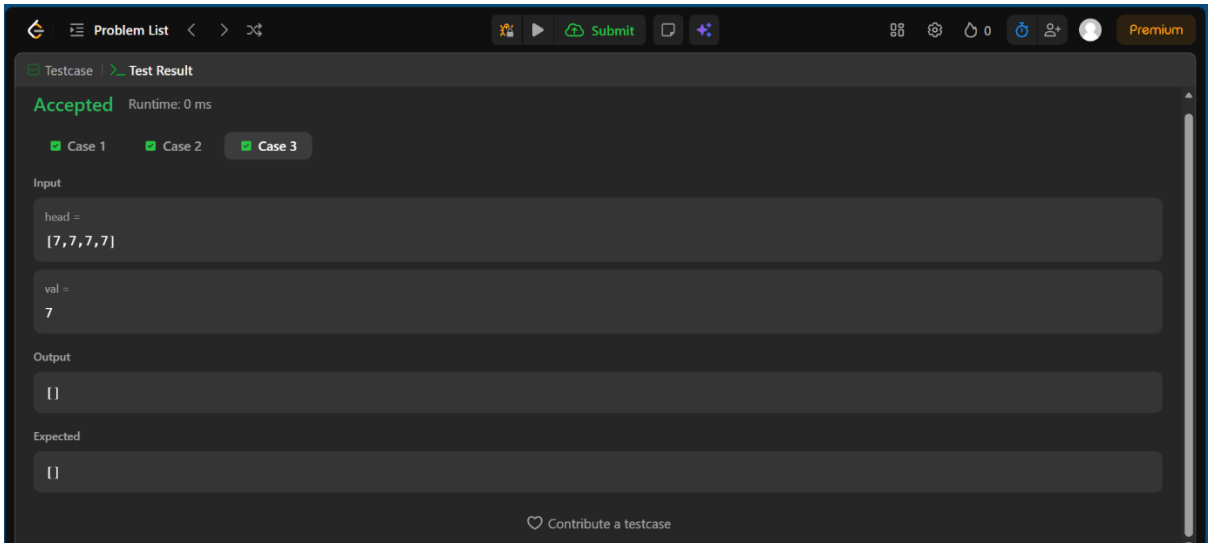
```
struct ListNode* removeElements(struct ListNode* head, int val) {
    // Create a dummy node pointing to head
    struct ListNode* dummy = (struct ListNode*)malloc(sizeof(struct ListNode));
    dummy->next = head;

    struct ListNode* prev = dummy;
    struct ListNode* curr = head;

    while (curr != NULL) {
        if (curr->val == val) {
            // Skip this node
            prev->next = curr->next;
            free(curr);
            curr = prev->next;
        } else {
            prev = curr;
            curr = curr->next;
        }
    }

    struct ListNode* newHead = dummy->next;
    free(dummy);
    return newHead;
}
```


Problem List
Submit
Premium
Code
Auto
struct ListNode* removeElements(struct ListNode* head, int val) {
// Create a dummy node pointing to head
struct ListNode* dummy = (struct ListNode*)malloc(sizeof(struct ListNode));
dummy->next = head;
struct ListNode* prev = dummy;
struct ListNode* curr = head;
while (curr != NULL) {
if (curr->val == val) {
// Skip this node
prev->next = curr->next;
free(curr);
curr = prev->next;
} else {
prev = curr;
curr = curr->next;
}
}
struct ListNode* newHead = dummy->next;
free(dummy);
return newHead;
Chaithanya J
Access all features with our Premium subscription!
My Lists
Notebook
Progress
Points
Try New Features
Orders
My Playgrounds
Settings
Appearance

OUTPUT:


Problem List
Submit
Premium
Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Case 3
Input
head =
[1,2,6,3,4,5,6]
val =
6
Output
[1,2,3,4,5]
Expected
[1,2,3,4,5]
Contribute a testcase





OBSERVATION:

21/12/25

Prg- 5b.

Leetcode

Remove Linked List Elements :-

Code :

```
struct ListNode* removeElements(struct ListNode*
                                head, int val) {
    struct ListNode* dummy = (struct ListNode*)
                malloc(sizeof(struct ListNode));
    dummy->next = head;

    struct ListNode* prev = dummy;
    struct ListNode* curr = head;

    while (curr != NULL) {
        if (curr->val == val) {
            prev->next = curr->next;
            free(curr);
            curr = prev->next;
        }
        else {
            prev = curr;
            curr = curr->next;
        }
    }

    struct ListNode* newHead = dummy->next;
    free(dummy);
    return newHead;
}
```

o/p

Case 1 :-

I/p => head = [1,2,6,3,4,5,6]

val : 6

o/p : [1,2,3,4,5]

Case 2 :-

I/p : head = []

val : 1

o/p : []

Case 3 :-

I/p : head = [7,7,7,7]

val : 7

o/p : []