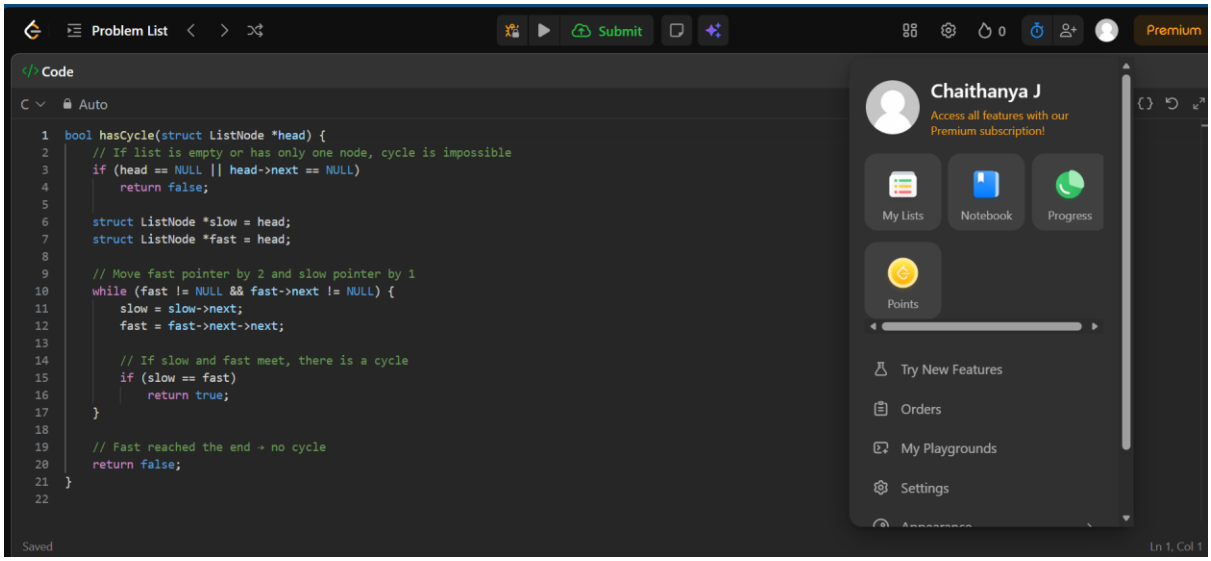
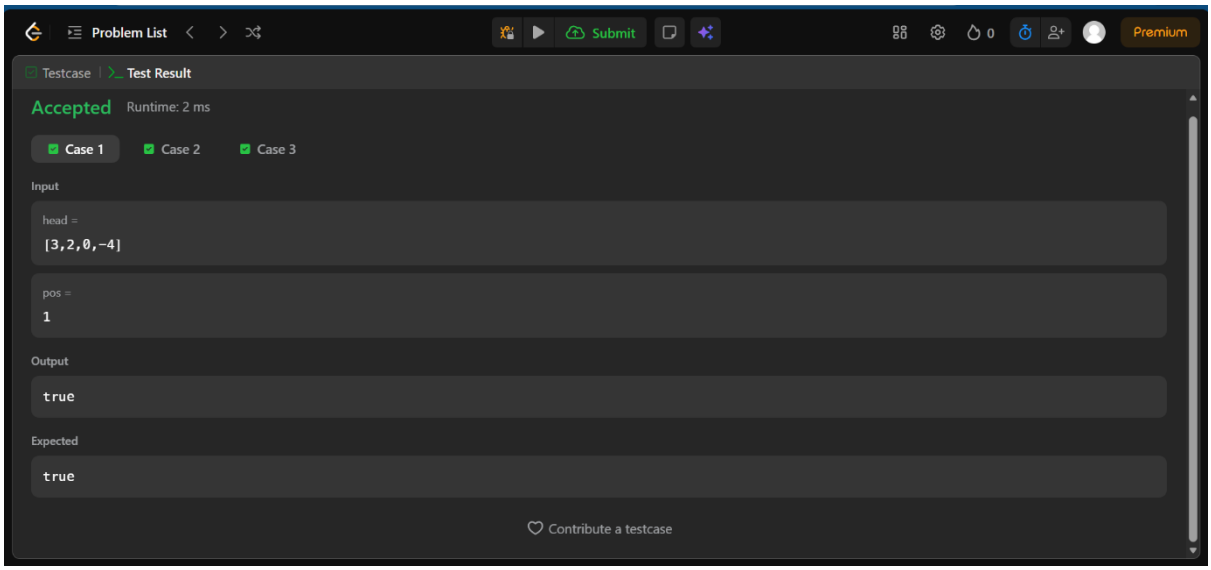
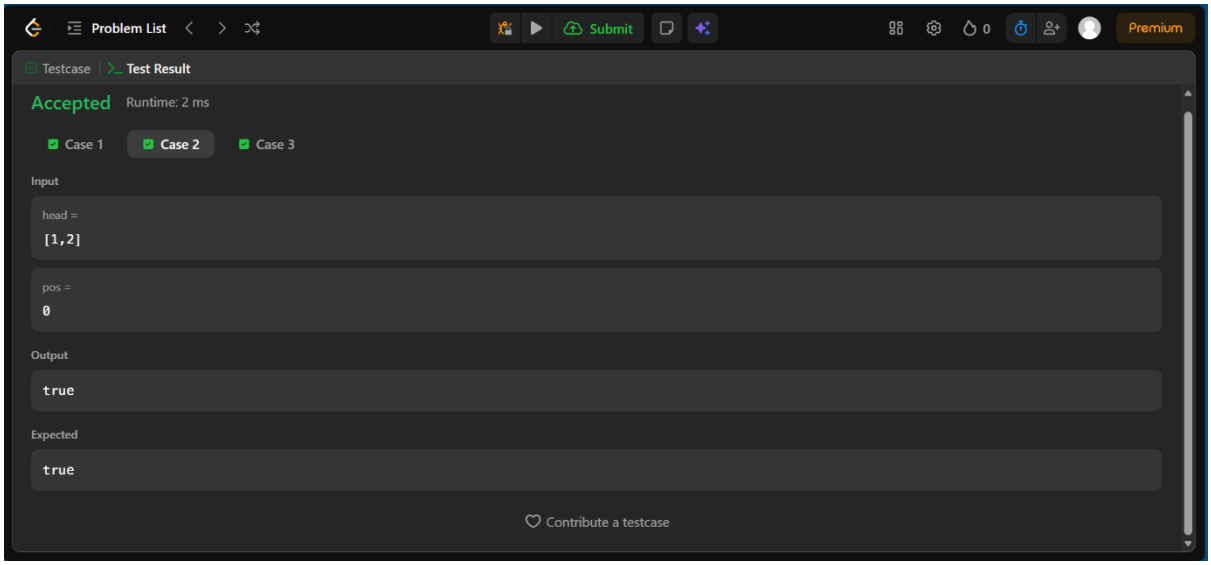
Leetcode:

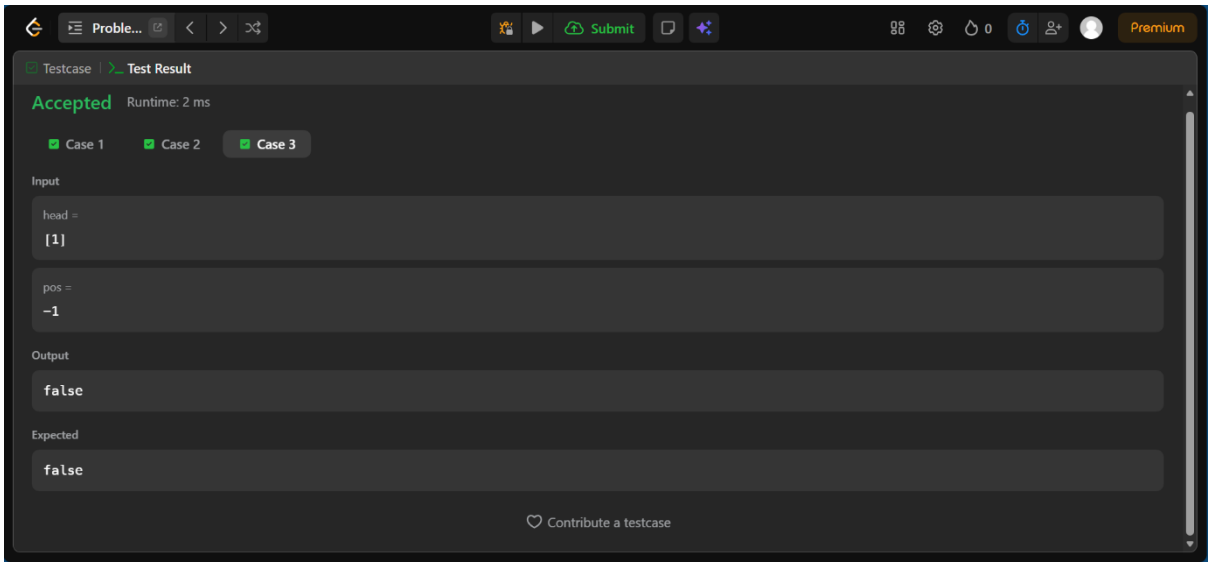
```
bool hasCycle(struct ListNode *head) {
    // If list is empty or has only one node, cycle is impossible
    if (head == NULL || head->next == NULL)
        return false;

    struct ListNode *slow = head;
    struct ListNode *fast = head;

    // Move fast pointer by 2 and slow pointer by 1
    while (fast != NULL && fast->next != NULL) {
        slow = slow->next;
        fast = fast->next->next;

        // If slow and fast meet, there is a cycle
        if (slow == fast)
            return true;
    }

    // Fast reached the end → no cycle
    return false;
}
```


Problem List
Submit
Premium
Code
C
Auto
bool hasCycle(struct ListNode *head) {
// If list is empty or has only one node, cycle is impossible
if (head == NULL || head->next == NULL)
return false;
struct ListNode *slow = head;
struct ListNode *fast = head;
// Move fast pointer by 2 and slow pointer by 1
while (fast != NULL && fast->next != NULL) {
slow = slow->next;
fast = fast->next->next;
// If slow and fast meet, there is a cycle
if (slow == fast)
return true;
}
// Fast reached the end → no cycle
return false;
}
Chaithanya J
Access all features with our Premium subscription!
My Lists
Notebook
Progress
Points
Try New Features
Orders
My Playgrounds
Settings
Saved
Ln 1, Col 1

OUTPUT:


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





OBSERVATION:

21/12/25

Page No:
Date.

Prg - 7b

Leet code:

Linked List Cycle.

Code:-

```
bool hasCycle (struct ListNode *head) {
    if (head == NULL || head->next == NULL)
        return false;

    struct ListNode * slow = head;
    struct ListNode * fast = head;

    while (fast != NULL && fast->next != NULL)
{
    slow = slow->next;
    fast = fast->next->next;

    if (slow == fast)
        return true;
    }
    return false;
}
```

O/p:

Ip => head = [3, 2, 0, -4]

pos = 1

O/p = true.