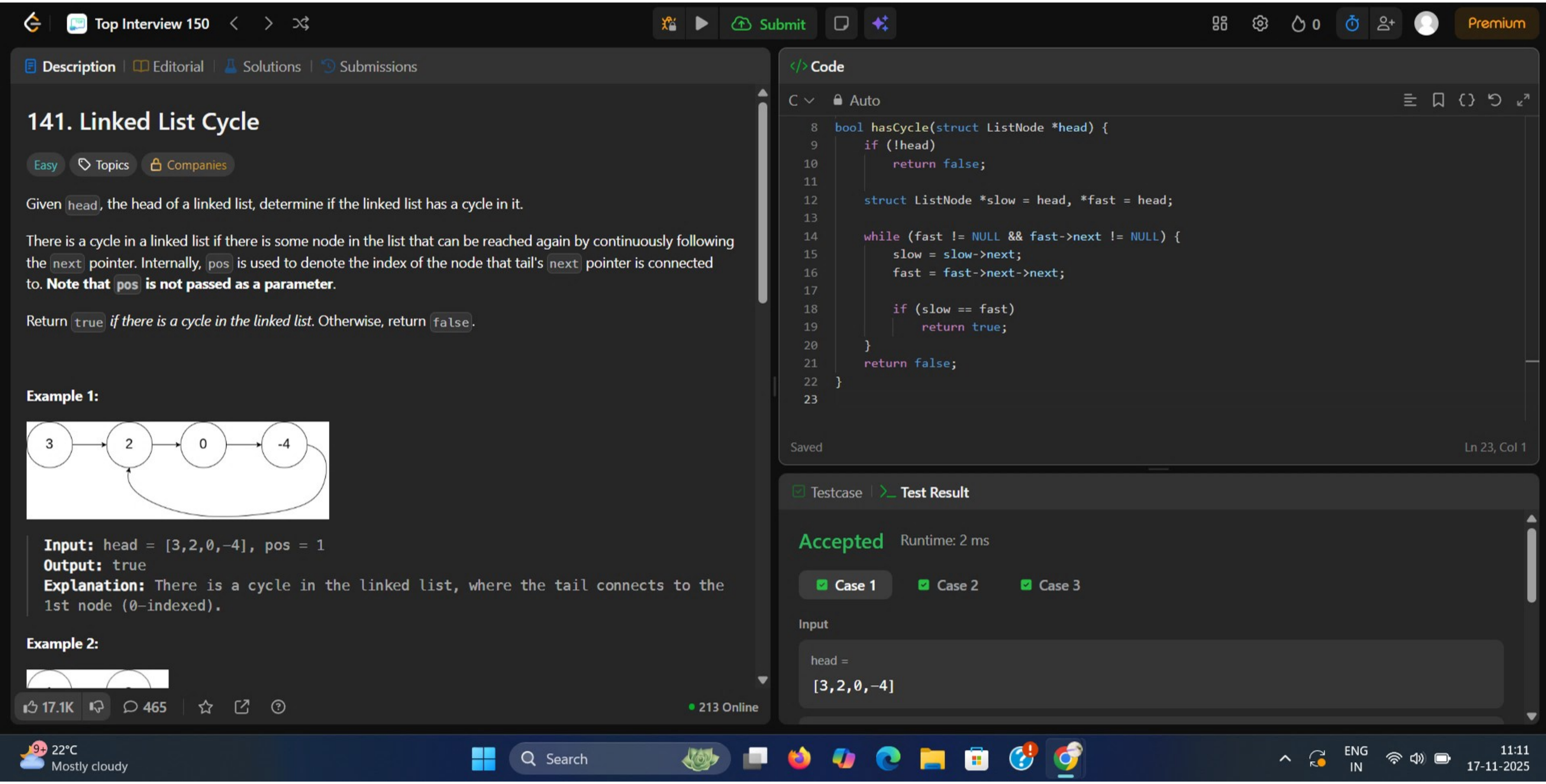


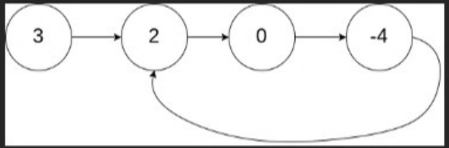
141. Linked List Cycle

Given `head`, the head of a linked list, determine if the linked list has a cycle in it.

There is a cycle in a linked list if there is some node in the list that can be reached again by continuously following the `next` pointer. Internally, `pos` is used to denote the index of the node that tail's `next` pointer is connected to. **Note that `pos` is not passed as a parameter.**

Return `true` *if there is a cycle in the linked list*. Otherwise, return `false`.

Example 1:



```
Input: head = [3,2,0,-4], pos = 1
Output: true
Explanation: There is a cycle in the linked list, where the tail connects to the
1st node (0-indexed).
```

Example 2:

```c
bool hasCycle(struct ListNode *head) {
    if (!head)
        return false;

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

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

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

Saved

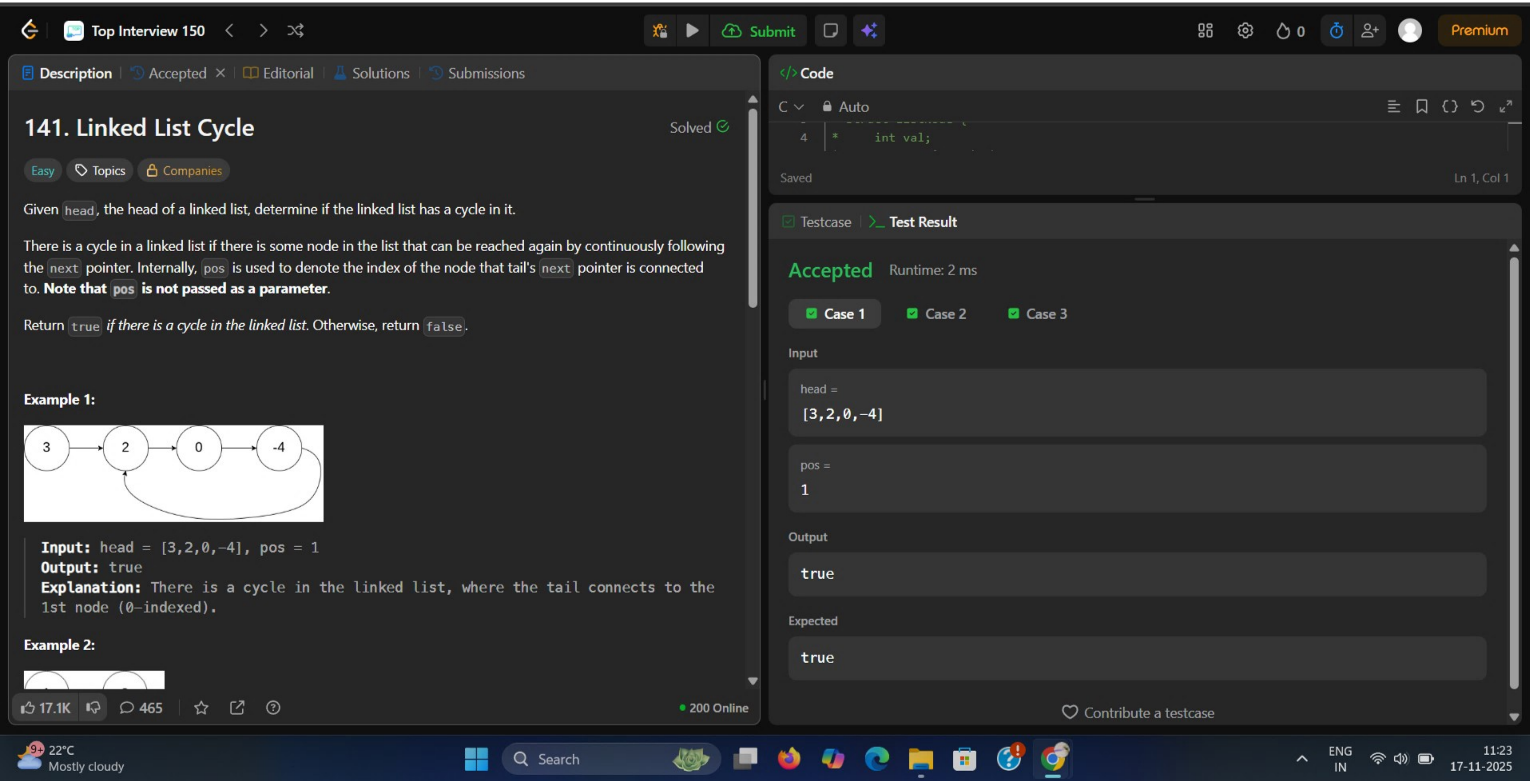
Ln 23, Col 1

Accepted Runtime: 2 ms

Input

head =

[3,2,0,-4]



141. Linked List Cycle

Solved

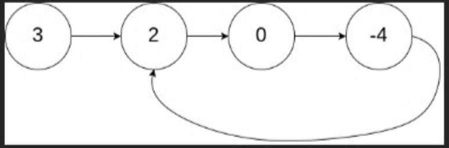
Easy · Topics · Companies

Given `head`, the head of a linked list, determine if the linked list has a cycle in it.

There is a cycle in a linked list if there is some node in the list that can be reached again by continuously following the `next` pointer. Internally, `pos` is used to denote the index of the node that tail's `next` pointer is connected to. **Note that `pos` is not passed as a parameter.**

Return `true` *if there is a cycle in the linked list*. Otherwise, return `false`.

Example 1:



```
Input: head = [3,2,0,-4], pos = 1
Output: true
Explanation: There is a cycle in the linked list, where the tail connects to the
1st node (0-indexed).
```

Example 2:

17.1K · 465 · 200 Online

C · Auto

```
*     int val;
```

Saved · Ln 1, Col 1

Accepted Runtime: 2 ms

Case 1 · Case 2 · Case 3

Input

head =
```
[3,2,0,-4]
```

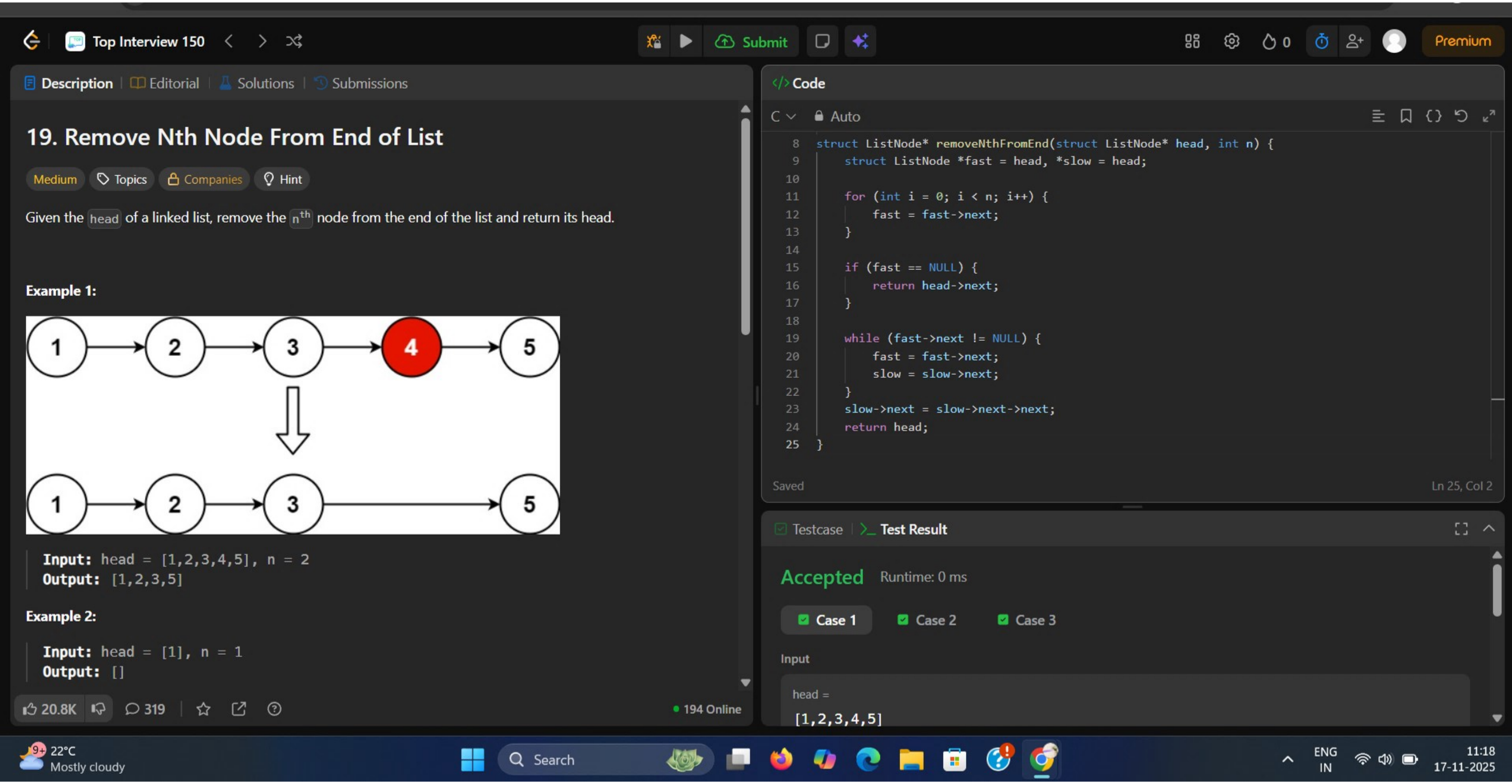
pos =
```
1
```

Output
```
true
```

Expected
```
true
```

Contribute a testcase

22°C Mostly cloudy · Search · ENG IN · 11:23 17-11-2025

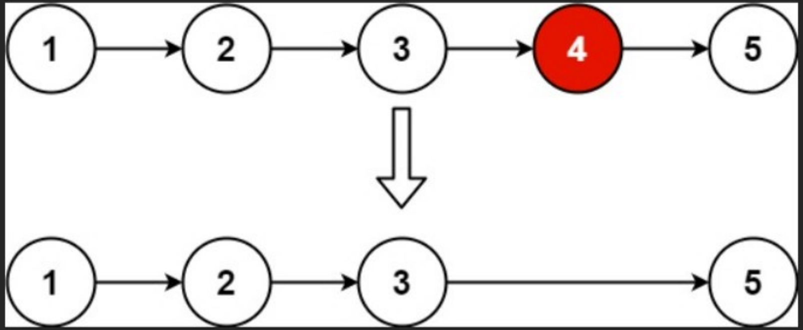


19. Remove Nth Node From End of List

Medium · Topics · Companies · Hint

Given the `head` of a linked list, remove the `n`th node from the end of the list and return its head.

Example 1:



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

Example 2:

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

20.8K · 319 · 194 Online

```c
struct ListNode* removeNthFromEnd(struct ListNode* head, int n) {
    struct ListNode *fast = head, *slow = head;

    for (int i = 0; i < n; i++) {
        fast = fast->next;
    }

    if (fast == NULL) {
        return head->next;
    }

    while (fast->next != NULL) {
        fast = fast->next;
        slow = slow->next;
    }
    slow->next = slow->next->next;
    return head;
}
```

Saved · Ln 25, Col 2

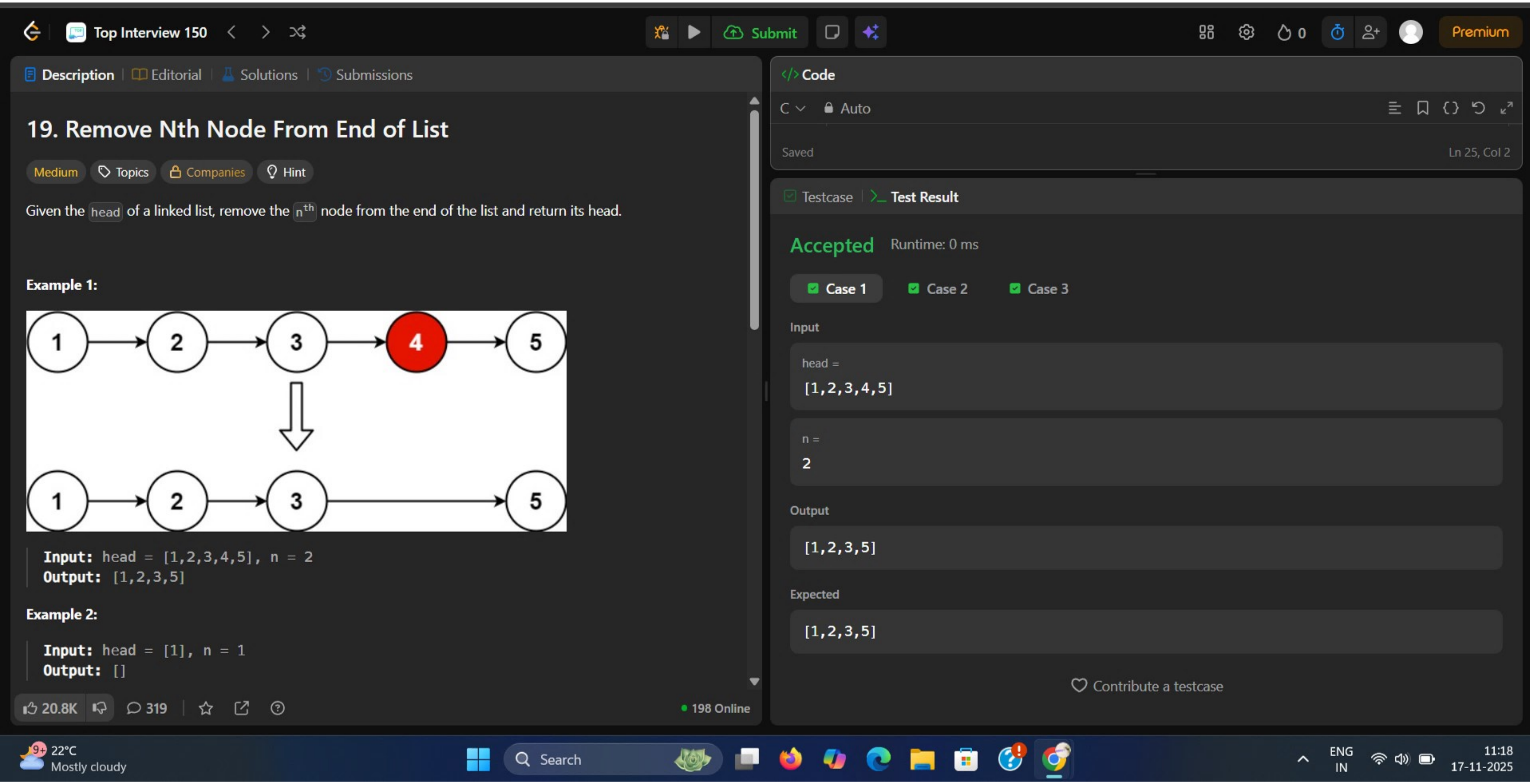
Accepted Runtime: 0 ms

Case 1 · Case 2 · Case 3

Input

head =

```
[1,2,3,4,5]
```


Top Interview 150

Submit

Premium

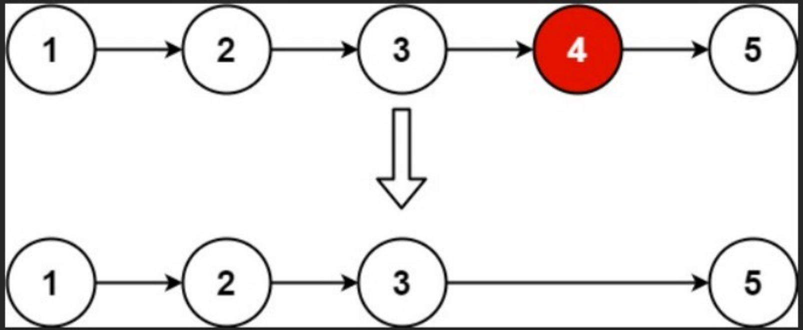
Description | Editorial | Solutions | Submissions

19. Remove Nth Node From End of List

Medium · Topics · Companies · Hint

Given the `head` of a linked list, remove the `n`th node from the end of the list and return its head.

Example 1:



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

Example 2:

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

20.8K · 319 · 198 Online

Code

C · Auto

Saved · Ln 25, Col 2

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 · Case 2 · Case 3

Input

```
head =
[1,2,3,4,5]
```

```
n =
2
```

Output

```
[1,2,3,5]
```

Expected

```
[1,2,3,5]
```

Contribute a testcase

22°C Mostly cloudy · Search · ENG IN · 11:18 17-11-2025