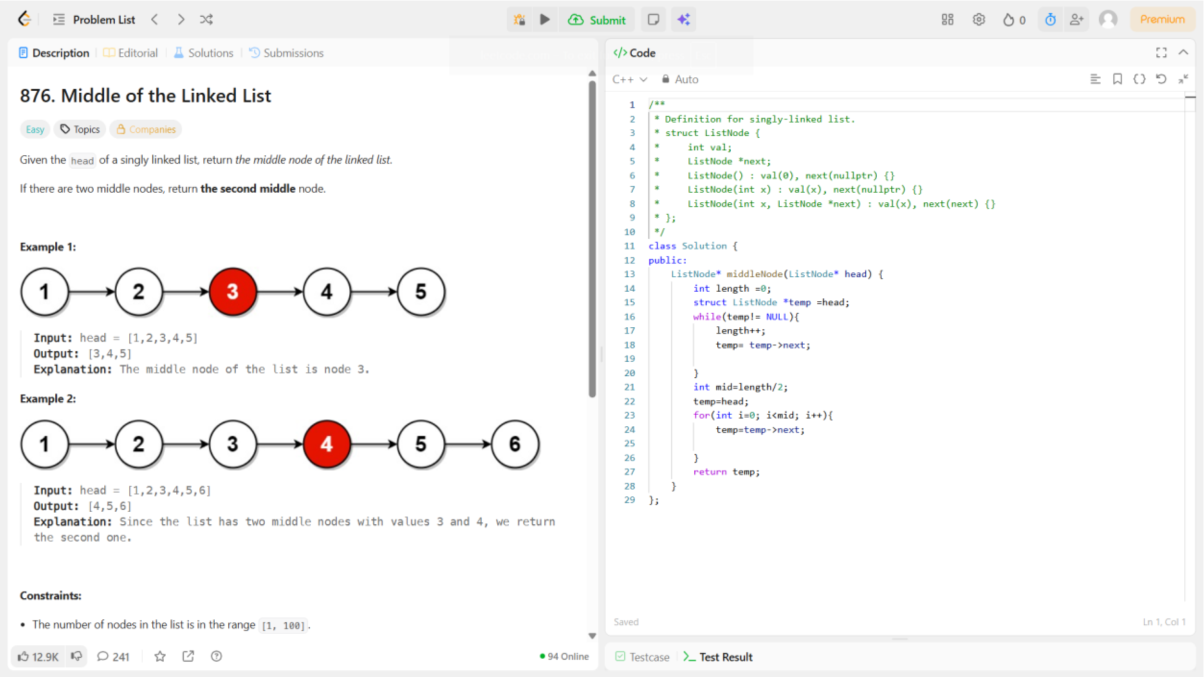


876. Middle of the Linked List

Given the `head` of a singly linked list, return *the middle node of the linked list*.

If there are two middle nodes, return **the second middle** node.

Example 1:

```
Input: head = [1,2,3,4,5]
Output: [3,4,5]
Explanation: The middle node of the list is node 3.
```

Example 2:

```
Input: head = [1,2,3,4,5,6]
Output: [4,5,6]
Explanation: Since the list has two middle nodes with values 3 and 4, we return
the second one.
```

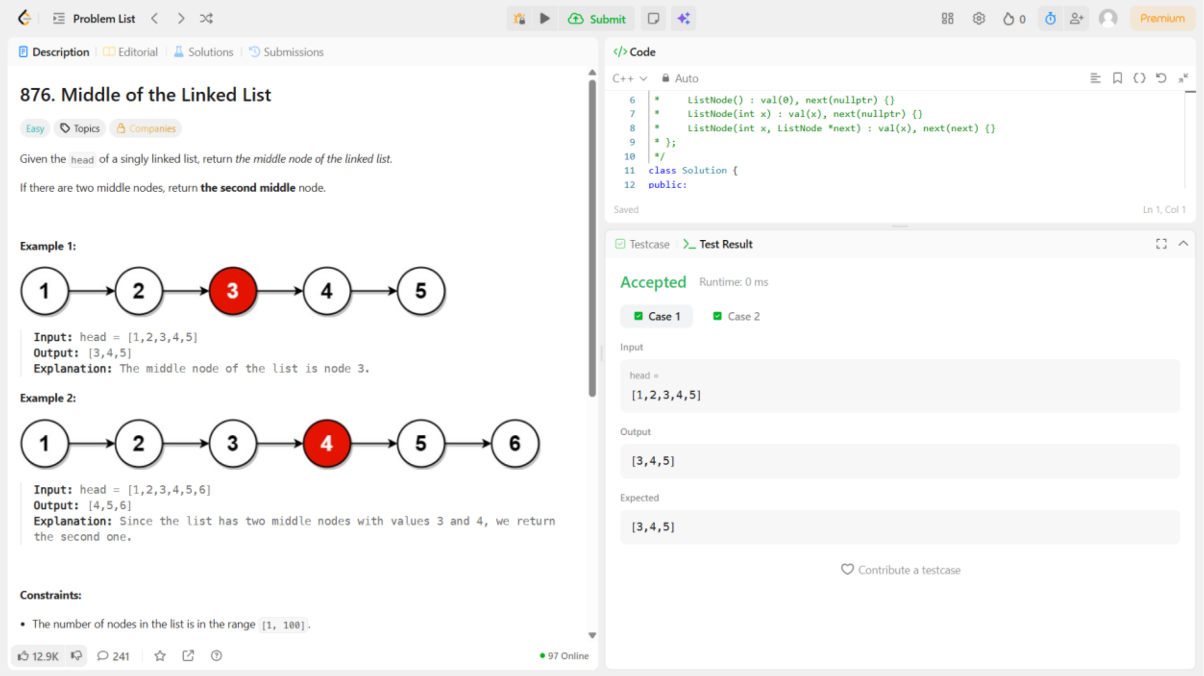
Constraints:

- The number of nodes in the list is in the range `[1, 100]`.

```cpp
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode() : val(0), next(nullptr) {}
 *     ListNode(int x) : val(x), next(nullptr) {}
 *     ListNode(int x, ListNode *next) : val(x), next(next) {}
 * };
 */
class Solution {
public:
    ListNode* middleNode(ListNode* head) {
        int length =0;
        struct ListNode *temp =head;
        while(temp!= NULL){
            length++;
            temp= temp->next;

        }
        int mid=length/2;
        temp=head;
        for(int i=0; i<mid; i++){
            temp=temp->next;

        }
        return temp;
    }
};
```



876. Middle of the Linked List

Given the `head` of a singly linked list, return *the middle node of the linked list*.

If there are two middle nodes, return **the second middle** node.

Example 1:

Input: head = [1,2,3,4,5]
Output: [3,4,5]
Explanation: The middle node of the list is node 3.

Example 2:

Input: head = [1,2,3,4,5,6]
Output: [4,5,6]
Explanation: Since the list has two middle nodes with values 3 and 4, we return the second one.

Constraints:

The number of nodes in the list is in the range [1, 100].

 *   ListNode() : val(0), next(nullptr) {}
 *   ListNode(int x) : val(x), next(nullptr) {}
 *   ListNode(int x, ListNode *next) : val(x), next(next) {}
 * };
 */
class Solution {
public:

Accepted Runtime: 0 ms

☑ Case 1 ☑ Case 2

Input

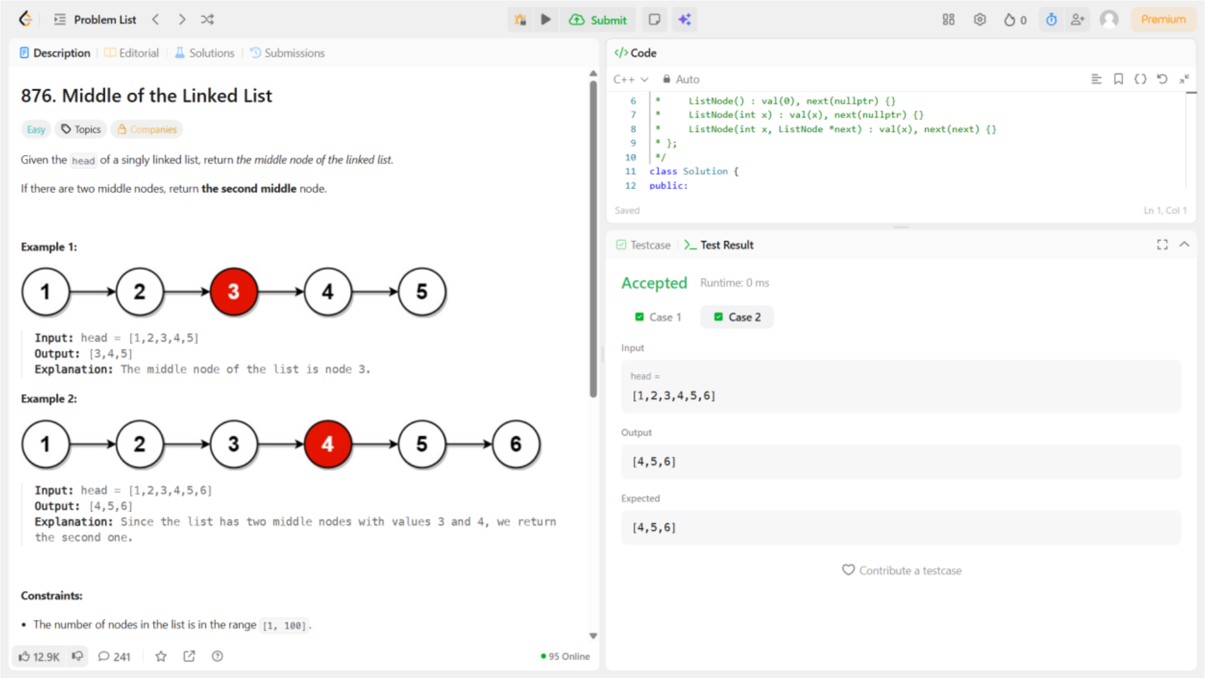
```
head =
[1,2,3,4,5]
```

Output

```
[3,4,5]
```

Expected

```
[3,4,5]
```



876. Middle of the Linked List

Given the `head` of a singly linked list, return *the middle node of the linked list*.

If there are two middle nodes, return **the second middle** node.

Example 1:

```
Input: head = [1,2,3,4,5]
Output: [3,4,5]
Explanation: The middle node of the list is node 3.
```

Example 2:

```
Input: head = [1,2,3,4,5,6]
Output: [4,5,6]
Explanation: Since the list has two middle nodes with values 3 and 4, we return
the second one.
```

Constraints:

- The number of nodes in the list is in the range `[1, 100]`.

```
 *     ListNode() : val(0), next(nullptr) {}
 *     ListNode(int x) : val(x), next(nullptr) {}
 *     ListNode(int x, ListNode *next) : val(x), next(next) {}
 * };
 */
class Solution {
public:
```

Accepted Runtime: 0 ms

☑ Case 1 ☑ Case 2

Input

head =

```
[1,2,3,4,5,6]
```

Output

```
[4,5,6]
```

Expected

```
[4,5,6]
```