

Problem List


876. Middle of the Linked List

Easy Topics Companies

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

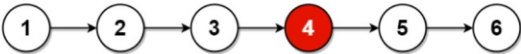
If there are two middle nodes, return the **second middle** node.

**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]`.

12.9K 241 94 Online

Code

```
1 /**
2  * Definition for singly-linked list.
3  * struct ListNode {
4  *     int val;
5  *     ListNode *next;
6  *     ListNode() : val(0), next(nullptr) {}
7  *     ListNode(int x) : val(x), next(nullptr) {}
8  *     ListNode(int x, ListNode *next) : val(x), next(next) {}
9  * };
10 */
11 class Solution {
12 public:
13     ListNode* middleNode(ListNode* head) {
14         int length = 0;
15         struct ListNode *temp = head;
16         while(temp != NULL){
17             length++;
18             temp = temp->next;
19         }
20         int mid = length/2;
21         temp = head;
22         for(int i=0; i<mid; i++){
23             temp = temp->next;
24         }
25         return temp;
26     }
27 };
28
29
```

SavedLn 1, Col 1

TestcaseTest Result

Problem List


876. Middle of the Linked List

Easy Topics Companies

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

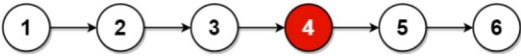
If there are two middle nodes, return the **second middle** node.

**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]`.

12.9K 241 97 Online

Code

```
6 struct ListNode {
7     int val;
8     ListNode *next;
9     ListNode() : val(0), next(nullptr) {}
10    ListNode(int x) : val(x), next(nullptr) {}
11    ListNode(int x, ListNode *next) : val(x), next(next) {}
12 };
13
14 class Solution {
15 public:
16     ListNode* middleNode(ListNode* head) {
17         int length = 0;
18         struct ListNode *temp = head;
19         while(temp != NULL){
20             length++;
21             temp = temp->next;
22         }
23         int mid = length/2;
24         temp = head;
25         for(int i=0; i<mid; i++){
26             temp = temp->next;
27         }
28         return temp;
29     }
30 };
31
```

SavedLn 1, Col 1

TestcaseTest Result

AcceptedRuntime: 0 ms

Case 1Case 2

Input

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

Output

[3,4,5]

Expected

[3,4,5]

Contribute a testcase

Problem List

Submit

Premium


DescriptionEditorialSolutionsSubmissions

876. Middle of the Linked List

EasyTopicsCompanies

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:



```
graph LR; 1((1)) --> 2((2)); 2 --> 3((3)); 3 --> 4((4)); 4 --> 5((5));
```

Input:

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


Output:

 [3,4,5]

Explanation:

 The middle node of the list is node 3.

Example 2:



```
graph LR; 1((1)) --> 2((2)); 2 --> 3((3)); 3 --> 4((4)); 4 --> 5((5)); 5 --> 6((6));
```

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].

12.9K24195 Online

Code

C++Auto

```
6 * ListNode() : val(0), next(nullptr) {}
7 * ListNode(int x) : val(x), next(nullptr) {}
8 * ListNode(int x, ListNode *next) : val(x), next(next) {}
9 * };
10 */
11 class Solution {
12 public:
```

SavedLn 1, Col 1

TestcaseTest Result

AcceptedRuntime: 0 ms

Case 1Case 2

Input

head =

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

Output

[4,5,6]

Expected

[4,5,6]

Contribute a testcase