

## 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.

Solution:

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
struct ListNode
```

```
{
```

```
    int val;
```

```
    struct ListNode *next;
```

```
};
```

```
struct ListNode* middleNode(struct ListNode* head)
```

```
{
```

```
    struct ListNode *tort = head, *hare = head;
```

```
    while (hare != NULL && hare->next != NULL)
```

```
    {
```

```
        tort = tort->next;
```

```
        hare = hare->next->next;
```

```
    }
```

```
    return tort;
```

```
}
```

```
struct ListNode* insertAtEnd(struct ListNode* head, int val)
```

```
{
```

```
    struct ListNode* newNode = (struct ListNode*)malloc(sizeof(struct ListNode));
```

```
    newNode->val = val;
```

```
    newNode->next = NULL;
```

```
    if (head == NULL)
```

```
        return newNode;
```

```
    struct ListNode* temp = head;
```

```
    while (temp->next != NULL)
```

```
        temp = temp->next;
```

```

temp->next = newNode;
return head;
}
int main()
{
    struct ListNode* head = NULL;
    int n, x;

    printf("Enter number of nodes: ");
    scanf("%d", &n);
    printf("Enter values: ");
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &x);
        head = insertAtEnd(head, x);
    }
    struct ListNode* mid = middleNode(head);
    printf("Middle element: %d\n", mid->val);
    return 0;
}

```

## OUTPUT:

```
Microsoft Windows [Version 10.0.26200.7171]
(c) Microsoft Corporation. All rights reserved.

c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\C Tutorial>cd "c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\C Tutorial\" && gcc 876.c -o 876 && "c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\C Tutorial\876"
Enter number of nodes: 6
Enter values: 1 2 3 4 5 6
Middle element: 4

c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\C Tutorial>cd "c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\C Tutorial\" && gcc 876.c -o 876 && "c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\C Tutorial\876"
Enter number of nodes: 3
Enter values: 1 2 3
Middle element: 2

c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\C Tutorial>
```

Description

Accepted

Editorial

Solutions

Submissions

All Submissions

Accepted 36 / 36 testcases passed

Javeed submitted at Nov 30, 2025 22:02

Editorial

Solution

BLACK FRIDAY SALE

LIMITED TIME OFFER - \$40 off Annual Subscription

LeetCode's Thanksgiving Sale IS NOW LIVE! Get \$40 OFF - Your Final Offer of the Year!

Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

8.45 MB | Beat: 57.28%

150%

100%

50%

0%

1ms

2ms

3ms

Code

C

```
1 /**
2  * Definition for singly-linked list.
3  * struct ListNode {
```

Code

Auto

```
7  /**
8  * struct ListNode* middleNode(struct ListNode* head) {
9  *     struct ListNode *tort = head, *hare = head;
10 *
11 *     while (hare != NULL && hare->next != NULL) {
12 *         tort = tort->next; // moves 1 step
13 *         hare = hare->next->next; // moves 2 steps
14 *     }
15 *     return tort;
16 * }
```

Testcase

Test Result

Accepted

Runtime: 0 ms

Case 1

Case 2

Input

head =

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

Output

[4,5,6]

Expected

[4,5,6]

Contribute a testcase

DescriptionAccepted xEditorialSolutionsSubmissions

All Submissions

Accepted36 / 36 testcases passed  
Javeed submitted at Nov 30, 2025 22:02

EditorialSolution

BLACK FRIDAY SALE

LIMITED TIME OFFER - \$40 off Annual Subscription  
LeetCode's Thanksgiving Sale IS NOW LIVE! Get \$40 OFF - Your Final Offer of the Year!

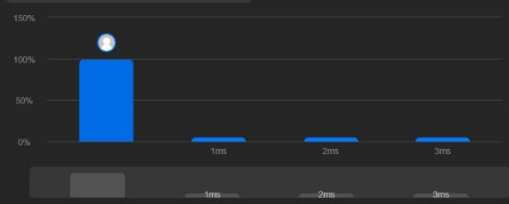
Runtime

0 msBeats 100.00%

Analyze Complexity

Memory

8.45 MBBeats 57.28%



CodeC

```
1 /**
2  * Definition for singly-linked list.
3  * struct ListNode {
4  *     int val;
5  *     struct ListNode *next;
6  * };
7
8  struct ListNode* middleNode(struct ListNode* head) {
9      struct ListNode *tort = head, *hare = head;
10
11     while (hare != NULL && hare->next != NULL) {
12         tort = tort->next; // moves 1 step
13         hare = hare->next->next; // moves 2 steps
14     }
15     return tort;
16 }
```

Code

Auto

```
7 /**
8  struct ListNode* middleNode(struct ListNode* head) {
9      struct ListNode *tort = head, *hare = head;
10
11     while (hare != NULL && hare->next != NULL) {
12         tort = tort->next; // moves 1 step
13         hare = hare->next->next; // moves 2 steps
14     }
15     return tort;
16 }
```

SavedLn 11, Col 16

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