

## Write a C function to print the middle of a given linked list

### Method 1:

Traverse the whole linked list and count the no. of nodes. Now traverse the list again till count/2 and return the node at count/2.

### Method 2:

Traverse linked list using two pointers. Move one pointer by one and other pointer by two. When the fast pointer reaches end slow pointer will reach middle of the linked list.

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

/* Link list node */
struct node
{
    int data;
    struct node* next;
};

/* Function to get the middle of the linked list*/
void printMiddle(struct node *head)
{
    struct node *slow_ptr = head;
    struct node *fast_ptr = head;

    if (head!=NULL)
    {
        while (fast_ptr != NULL && fast_ptr->next != NULL)
        {
            fast_ptr = fast_ptr->next->next;
            slow_ptr = slow_ptr->next;
        }
    }
}
```

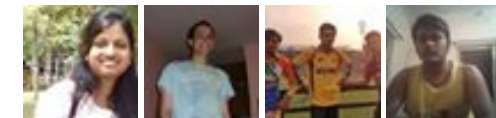
Google™ Custom Search



GeeksforGeeks



53,528 people like [GeeksforGeeks](#).



Interview Experiences

Advanced Data Structures

Dynamic Programming

Greedy Algorithms

Backtracking

Pattern Searching

Divide & Conquer

Mathematical Algorithms

Recursion

Geometric Algorithms

```

    }
    printf("The middle element is [%d]\n\n", slow_ptr->data);
}

```

```

void push(struct node** head_ref, int new_data)
{
    /* allocate node */
    struct node* new_node =
        (struct node*) malloc(sizeof(struct node));

    /* put in the data */
    new_node->data = new_data;

    /* link the old list off the new node */
    new_node->next = (*head_ref);

    /* move the head to point to the new node */
    (*head_ref) = new_node;
}

```

// A utility function to print a given linked list

```

void printList(struct node *ptr)
{
    while (ptr != NULL)
    {
        printf("%d->", ptr->data);
        ptr = ptr->next;
    }
    printf("NULL\n");
}

```

/\* Driver program to test above function\*/

```

int main()
{
    /* Start with the empty list */
    struct node* head = NULL;
    int i;

    for (i=5; i>0; i--)
    {
        push(&head, i);
        printList(head);
        printMiddle(head);
    }
}

```

return 0;



## Popular Posts

All permutations of a given string

Memory Layout of C Programs

Understanding "extern" keyword in C

Median of two sorted arrays

Tree traversal without recursion and without stack!

Structure Member Alignment, Padding and Data Packing

Intersection point of two Linked Lists

Lowest Common Ancestor in a BST.

Check if a binary tree is BST or not

Sorted Linked List to Balanced BST

```
}
```

Output:

5->NULL

The middle element is [5]

4->5->NULL

The middle element is [5]

3->4->5->NULL

The middle element is [4]

2->3->4->5->NULL

The middle element is [4]

1->2->3->4->5->NULL

The middle element is [3]

### Method 3:

Initialize mid element as head and initialize a counter as 0. Traverse the list from head, while traversing increment the counter and change mid to mid->next whenever the counter is odd. So the mid will move only half of the total length of the list.

Thanks to Narendra Kangralkar for suggesting this method.

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

```
/* Link list node */
struct node
{
    int data;
    struct node* next;
};
```

```
/* Function to get the middle of the linked list*/
void printMiddle(struct node *head)
{
    int count = 0;
    struct node *mid = head;
```

Market research  
that's fast and  
accurate.

Get \$75 off

 Google consumer surveys



## Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 47 minutes ago

**Aman** Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 1 hour ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 1 hour ago

**Sanjay Agarwal** bool

tree::Root\_to\_leaf\_path\_given\_sum(tree...

Root to leaf path sum equal to a given number · 1 hour ago

**GOPI GOPINATH** @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 1 hour ago

**newCoder3006** If the array contains negative numbers also. We...

Find subarray with given sum · 2 hours ago

AdChoices

[▶ Linked List](#)

[▶ JavaScript Array](#)

[▶ C++ Code](#)

AdChoices

[▶ Java Source Code](#)

```

while (head != NULL)
{
    /* update mid, when 'count' is odd number */
    if (count & 1)
        mid = mid->next;

    ++count;
    head = head->next;
}

/* if empty list is provided */
if (mid != NULL)
    printf("The middle element is [%d]\n\n", mid->data);
}

```

```

void push(struct node** head_ref, int new_data)
{
    /* allocate node */
    struct node* new_node =
        (struct node*) malloc(sizeof(struct node));

    /* put in the data */
    new_node->data = new_data;

    /* link the old list off the new node */
    new_node->next = (*head_ref);

    /* move the head to point to the new node */
    (*head_ref) = new_node;
}

// A utility function to print a given linked list
void printList(struct node *ptr)
{
    while (ptr != NULL)
    {
        printf("%d->", ptr->data);
        ptr = ptr->next;
    }
    printf("NULL\n");
}

/* Driver program to test above function*/
int main()
{

```

```

/* Start with the empty list */
struct node* head = NULL;
int i;

for (i=5; i>0; i--)
{
    push(&head, i);
    printList(head);
    printMiddle(head);
}

return 0;
}

```

Output:

```

5->NULL
The middle element is [5]

4->5->NULL
The middle element is [5]

3->4->5->NULL
The middle element is [4]

2->3->4->5->NULL
The middle element is [4]

1->2->3->4->5->NULL
The middle element is [3]

```

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

► [Linked Data](#)

► [Programming C++](#)

AdChoices ►

► [Function Program](#)

► [Function Mid](#)

► [Function One](#)

# Free Online Database

 [zoho.com/creator/online-database](https://zoho.com/creator/online-database)

Create database apps in minutes.  
Just Drag-&-drop. Try NOW!



## Related Tpoics:

---

- Given a linked list, reverse alternate nodes and append at the end
- Pairwise swap elements of a given linked list by changing links
- Self Organizing List | Set 1 (Introduction)
- Merge a linked list into another linked list at alternate positions
- QuickSort on Singly Linked List
- Delete N nodes after M nodes of a linked list
- Design a stack with operations on middle element
- Swap Kth node from beginning with Kth node from end in a Linked List



10



Tweet

0



0

Writing code in comment? Please use [ideone.com](https://ideone.com) and share the link here.

39 Comments

GeeksforGeeks

Sort by Newest ▼





...on the discussion...



**Guest** • 6 days ago

hmmm....Can any one suggest the best method among this.

1 ^ | v • Reply • Share ›



**kinshuk chandra** • 10 days ago

Yes, method 2 and 3 are almost same, I agree with prakashyaji

Here is my code similar to counter method from <http://k2code.blogspot.in/2016/06/linked-list.html> :

```
// Function to get to the middle of the LL
mynode *getTheMiddle(mynode *head)
{
    mynode *middle = (mynode *)NULL;
    int i;

    for(i=1; head!=(mynode *)NULL; head=head->next,i++)
    {
        if(i==1)
            middle=head;
        else if ((i%2)==1)
            middle=middle->next;
    }

    return middle;
}
```

^ | v • Reply • Share ›



**prakashyaji** • a month ago

Method 2 and 3 are essentially the same. You are using two different pointers

(head) twice and slower pointer (mid) once.

5 ^ | v • Reply • Share ›



**Sonu Lamba** • 3 months ago

Function to get the middle of the linked list\*/

```
void printMiddle(struct node *head)
```

```
{
```

```
    struct node *slow_ptr = head;
```

```
    struct node *fast_ptr = head;
```

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

```
    {
```

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

```
        {
```

```
            fast_ptr = fast_ptr->next->next;
```

```
            slow_ptr = slow_ptr->next;
```

```
        }
```

```
        printf("The middle element is [%d]\n\n", slow_ptr->data);
```

```
    }
```

```
    /****While condition should be while (fast_ptr != NULL && fast_ptr->next->nex
```

^ | v • Reply • Share ›



**Lokesh** • 3 months ago

Is there any cpp compiler property that can be used in this problem? Please R

^ | v • Reply • Share ›



**Madhan Raj** • 7 months ago

all works on time complexity  $O(n)$  i prefer first method as the space complexit

.

^ | v • Reply • Share ›



**GeeksforGeeks** • 10 months ago





@All:

We have updated the programs so that the output of both methods matches in added more test cases in main().

Thanks for your comments and views.

^ | v · Reply · Share ›



darkpassenger · 11 months ago

plz correct the 3rd method its not working in case of 2 elements.....

1 ^ | v · Reply · Share ›



GeeksforGeeks → darkpassenger · 11 months ago

It works for a list with two elements, please see <http://ideone.com/gk8z>

^ | v · Reply · Share ›



sam → GeeksforGeeks · 6 months ago

The condition in while loop should be

```
while (fast_ptr.next != NULL && fast_ptr->next.next != NULL)
```

^ | v · Reply · Share ›



darkpassenger → GeeksforGeeks · 11 months ago

it should print 4 in your test case but its printing 20 so its wrong the same case its printing 4 so method 3 is wrong.

^ | v · Reply · Share ›



darkpassenger → darkpassenger · 11 months ago

since we have to print  $mid = (n/2)$  in case n is even so its

^ | v · Reply · Share ›



Srb · a year ago

3rd method fail when there are only 2 numbers in linked list.....it is missing corr

^ | v · Reply · Share ›



**GeeksforGeeks** → Srb · a year ago

It seems to be working. Please see <http://ideone.com/gk8zC6>

^ | v · Reply · Share ›



**Srb** → GeeksforGeeks · 11 months ago

u r storing linked list as 4->20->NULL

and so for this case output should be 4 not 20(as it is showing |  
correct me if i m wrong

^ | v · Reply · Share ›



**GeeksforGeeks** → Srb · 11 months ago

When linked list is 4->20->NULL, we consider it as a lis

^ | v · Reply · Share ›



**srb** → GeeksforGeeks · 11 months ago

plz elaborate this....according to me if their is 2 element  
middle element then answer should be first number..no  
list answer should be (4->20->NULL) 4.

or is their case that when we have even no. of element  
mid+1 both as middle element....reply soon asap

^ | v · Reply · Share ›



**Gaurav Kumar** · a year ago

```
void middle()
```

```
{
```

```
int t=1, f=1;
```

```
struct node *temp=head;.
```

```
struct node *temp1=head;.
```

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

```
{
```

```
temp=temp->next;
t++;
}
temp=head;

if(t%2!=0)
{
while(temp->next!=NULL)
{
temp=(temp->next)->next;
temp1=temp1->next;
```

[see more](#)

^ | v • Reply • Share ›



**Ankit Malhotra** • a year ago

Simpler implementation.

```
node * mid (node * ptr)
{
node * mptr = ptr;
while (ptr && ptr->next)
{
mid = mid->next;
ptr = (ptr->next)->next;
}
return mid;
}
```

^ | v • Reply • Share ›



**Ankit Malhotra** → Ankit Malhotra • a year ago



Replace mptr with mid above to avoid compile error

^ | v • Reply • Share ›



**neham** • a year ago

Method2 and Method3 seems similar to me. In both the methods, we r keeping while other actually points to middle element when other reaches end.

If I am wrong..can you please point out the exact difference between both metl

2 ^ | v • Reply • Share ›



**anonymous** • a year ago

Even though in case of even length linked list,you can print either of the two mi maintain consistency in all the methods provided.

In case of 1->2->3->4

2nd method would output 2

and the third method would output 3

We should use ( head->next!=NULL )in 3rd method.

^ | v • Reply • Share ›



**Tanuj Makkar** • a year ago

good one

^ | v • Reply • Share ›



**Pavan Kulkarni** • a year ago

baap method....hats off to Narendra Kangralkar..

^ | v • Reply • Share ›



**Baboo Lal Kushwah** • a year ago

good job dear..

^ | v • Reply • Share ›



**Tony** • a year ago



Thank you for sharing your info. I really appreciate your efforts and I will be waiting for your next post thank you once again.

^ | v • Reply • Share ›



**pradeep gupta** • a year ago

If a linked list has even number of nodes, which element we should consider a

^ | v • Reply • Share ›



**Kartik** → pradeep gupta • a year ago

Any of the middle two elements can be considered as middle. In case | programs can easily modified to print both.

^ | v • Reply • Share ›



**novice** • 2 years ago

/\* this code will **print** the middle elements/element of **link** list whe

```
void disp_mid(struct node *q)
{
    struct node *first , *second;
    first = q;
    second = q;
    int flag=0;
    while(first->link!=NULL)
    {
        first=first->link;
        if(first->link==NULL)
        {
            printf("\t%d\t%d", second->data, second->link->data);
            flag=1;
            break;
        }
        second=second->link;
    }
```

[see more](#)

^ | v • Reply • Share ›



**nikoo28** → novice • 2 years ago

this is the same approach as above, you are incrementing pointer 'first' there any advantage using the function this way?

```
/* Paste your code here (You may delete these lines if not wr
```

^ | v • Reply • Share ›



**Prashant Tiwari** • 2 years ago

```
Type& findMiddleNode()
{
    int check = 0;
    nodeType *current;
    nodeType *mid;

    current = first;
    mid = first;

    while (current != NULL)
    {
        current = current->link;
        check = (check + 1) % 2;

        if (check == 0)
            mid = mid->link;
    }

    return mid->info;
```

```
}
```

^ | v • Reply • Share ›



**Narendra Kangralkar** • 3 years ago

Find the middle of linked list using single pointer.

```
void
print_middle(node_t *head)
{
    node_t *middle = NULL;
    int i;

    for (i = 1; head != NULL; head = head -> next, ++i) {
        if (i == 1)
            middle = head;
        else if (i & 1)
            middle = middle -> next;
    }
    printf("middledle of list: %d\n", middle -> data);
}
```

^ | v • Reply • Share ›



**mahesh** → Narendra Kangralkar • 11 months ago


Isn't it you are using head also a pointer ? i mean to say both head and achieving here

```
/* Paste your code here (You may delete these lines if not wri
```

^ | v • Reply • Share ›



**rocky** • 3 years ago



```
slow = head;
fast = head->next->next;

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

cout <data << endl;

}
```

^ | v • Reply • Share ›



**Gowrishankar** • 3 years ago

I am not sure if the fast method is actually fast as it is generally portrayed to be traverse the entire list once and the slower one, half of it... thus the total number of nodes v

^ | v • Reply • Share ›



**SSR** → Gowrishankar • 3 years ago

Isn't the fast pointer travelling at the rate of two nodes at a time skipping nodes, so it's actually faster??

^ | v • Reply • Share ›



**GeeksforGeeks** → Gowrishankar • 3 years ago

@Gowrishankar: Thanks for pointing this out. We have updated the po

^ | v • Reply • Share ›



**satya482** • 4 years ago

What is the middle of even length linkedlist?

^ | v • Reply • Share ›





**Sandeep** → satya482 · 4 years ago

@satya482

The above implementation prints the first middle element in that case.  
>3->4->5->6, then printMiddle() prints 3. You can modify it as per your

^ | v · Reply · Share ›



Subscribe



Add Disqus to your site

