Write a function to get Nth node in a Linked List

Write a GetNth() function that takes a linked list and an integer index and returns the data value stored in the node at that index position.

Algorithm:

```
1. Initialize count = 0
2. Loop through the link list
     a. if count is equal to the passed index then return current
         node
     b. Increment count
     c. change current to point to next of the current.
```

Implementation:

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

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

/* Given a reference (pointer to pointer) to the head
    of a list and an int, push a new node on the front
    of the list. */
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;
}

/* Takes head pointer of the linked list and index
    as arguments and return data at index*/
int GetNth(struct node* head, int index)
{
    struct node* current = head;
    int count = 0; /* the index of the node we're currently
                    looking at */
    while (current != NULL)
    {
       if (count == index)
          return(current->data);
       count++;
       current = current->next;
    }

    /* if we get to this line, the caller was asking
       for a non-existent element so we assert fail */
    assert(0);
}

/* Drier program to test above function*/
```

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

    /* Use push() to construct below list
     1->12->1->4->1  */
    push(&head, 1);
    push(&head, 4);
    push(&head, 1);
    push(&head, 12);
    push(&head, 1);

    /* Check the count function */
    printf("Element at index 3 is %d", GetNth(head, 3));
    getchar();
}
```

Time Complexity: O(n)

Related Topics:

- Clone a linked list with next and random pointer | Set 2
- Given a linked list of line segments, remove middle points
- Construct a Maximum Sum Linked List out of two Sorted Linked Lists having some Common nodes
- 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

Tags: GetNth, Linked Lists

Writing code in comment? Please use **ideone.com** and share the link here.

Vineeth Reddy • 16 days ago

There is no use of creating an extra variable in GetNth function, which points to head. We can skip it and use head directly to traverse the linked list. to the given index

Kanika Gupta • 16 days ago

can't we do with single pointer in push function?why is it necessary to use double pointers??

• Reply • Share ›

Debasmit Das • 2 months ago

How about this code ? Any disadvantage ?

```
int GetNth(struct node **head_ref, int n)
{
struct node* temp=*head_ref;
int c=1;
while(c!=n)
{
temp=temp->next;
c++;
}
return temp->data;
}
```

1 • Reply • Share ›

Bharat ➚ Debasmit Das • a month ago

I think it will fail, in case if the number of nodes in the linked-list is smaller than the value of n. So you also need to include a check for the condition (temp!=null)

2 • Reply • Share ›

annymous ➚ Debasmit Das • 2 months ago

nops.. i dont find any disadvantage..

• Reply • Share ›

Adauta Garcia Ariel • 2 months ago

Hello, I want to share an implementention of this function in a recursive way. Sorry if it is already posted and it's just a waste of bytes my comment.

if you know a better way, please, let me know!

```
int getNthRecursive(struct node* node,int index){
 static int counter = 0 ;
if(node == NULL){
  counter = 0;
  assert(0);
 }else if(counter == index){
   counter = 0 ;
```

```
    return node->data;
  }else if (node != NULL){
   counter = counter +1;
   getNthRecursive(node->next,index);
  }
}
```

• Reply • Share ›

ANKIT SINGH ➚ Adauta Garcia Ariel • 2 months ago

How can you say that this implementation is better?

• Reply • Share ›

Adauta Garcia Ariel ➚ ANKIT SINGH • 2 months ago

Oh I'm sorry, I didn't want to say that. What I actually mean is that if someone wants, knows or finds a better way to do it, even adding improvements to the code, do it, but I would like to know their approach they use.

Reading code it's an amazing way of learning from others. I don't dare to say that this algorithm is better in terms of time complexity.

Probably it runs O(n), actually the same as the example of this post. It was ambiguos, thank you for the observation.

1 • Reply • Share ›

ryan • 4 months ago

@GeeksforGeeks

initialize

count =1 not 0;

4 • Reply • Share ›

rajat • 4 months ago

i am making this program like this but it is not running in code block compiler

#include <stdio.h>

#include <stdlib.h>

struct node

{

int data ;

```
int data ;

struct node* next;

};

void push(struct node** head_ref,int value)

{

struct node* ptr=(struct node*)malloc(sizeof(struct node));
```

see more

• Reply • Share ›

ashish ➚ rajat • 3 months ago

In function 'getnode':
Line 35: error: expected expression before '=' token
In function 'main':
Line 71: warning: return type of 'main' is not 'int'

1 • Reply • Share ›

veena • 6 months ago

Just a suggestion: We can Avoid having a variable called counter which is 4 bytes(In Java) if we directly use index/position parameter passed to the function.

```
public SingleNode getNthNodeFromFront(int index, SingleNode head)
{
if(index < 1) return null;
while( index != 1){
head = head.link;
index--;
}
return head;
}
```

4 • Reply • Share ›

Vishal • 7 months ago

c++

```
Node* List::GetNthNode(int pos){
Node *t = m_head;
while(--pos)
t = t->next;
if(t)
{
cout<<"\n Value "<<t->info;
```

```
cout<< "\n Value "<<t->info;
return t;
}

return 0;
}
```

• Reply • Share ›

Anurag Nama • 7 months ago

getNth fun can also be written as :

```
int GetNth(struct node* head, int index)

{

struct node* current = head;

int count = 0;
/* the index of the node we're currently

looking at */

while (count!= index)

{
current = current->next;

count++;

}

return(current->data);
}
```

• Reply • Share ›

ashish ➦ Anurag Nama • 3 months ago

check for if current!=NULL and also index is greater than no of elements in linked lists

• Reply • Share ›

Balaji Nakkella ➦ Anurag Nama • 7 months ago

your prgm doesn't check boundary conditions like if list is empty,or index >no. of elements in list..

• Reply • Share ›

Jun • 8 months ago

http://codepad.org/wTQP12zP#ou...

• Reply • Share ›

ajorwal • 8 months ago

in the above question we have to find the node not the data par at tht value but the out shows the value at tht particular index. so how to find the node ??

• Reply • Share ›

ashish ajorwal • 3 months ago

I guess finding a node stands for finding the address of that node ....so for that simply return (node) instead of return (node->data).
Thanks

• Reply • Share ›

X ajorwal • 7 months ago

instead of return(current->data); just replace with return(current);

• Reply • Share ›

k X • 6 months ago

it will not work.

• Reply • Share ›

Pranav Kumar Jha ajorwal • 8 months ago

What do you mean by finding a node? Finding a node as far as i understand it, is retrieving the data it contains! :/

2 • Reply • Share ›

Abhishek Pasayat • 9 months ago

supposing i have a linked list comprising elemets 1->2->3->4->5 and i want to get the data stored at the Nth node from front where N=2,then what would be the Output?
It should be 2 or it should be 3??
Plz Help!!!

• Reply • Share ›

rajat Abhishek Pasayat • 4 months ago

it should be 2

• Reply • Share ›

GOPI GOPINATH Abhishek Pasayat • 9 months ago

It depends on the type of indexing u do in the list....here indexing starts from zero....so for n=2 , 3 will be the output

1 • Reply • Share ›

Abhishek Pasayat GOPI GOPINATH • 9 months ago

Suppose i created a linked list by pushing elements at the front and suppose the

created linked list is 5->4->3->2->1.
This means that the first element entered was 1,second element entered was 2 and so on.So if it is asked what would be the second element in the linked list,then what would be the answer??
It should be 4 or it should be 2??
Plz Help!!!

• Reply • Share ›

GOPI GOPINATH ➚ Abhishek Pasayat • 9 months ago

if the list is of the form 5->4->3->2->1
you have the head of the list pointing to '5', so start counting 'N' nodes from the head.
SO second node here will be '4'
It is for our simplicity we ve pushed the elements of the list in reverse order into the list, else we need to traverse the list each time when we need to add a new node.

1 • Reply • Share ›

The_Geek ➚ GOPI GOPINATH • 6 months ago

Why do we need to traverse? We will just keep a pointer at last inserted node, and add the new node in its next part. IMO, in this pushing type of code we will be able to access the nodes in reverse manner, which seems wrong to me.
Like in the above given example, we are pushing data, 1,4,1,12,1 in this order. While we will only be able to access it in reverse order, like, 1,12,1,4,1. So is it right? please explain me, if m wrong.

• Reply • Share ›

Abhishek Pasayat ➚ GOPI GOPINATH • 9 months ago

Thank You!!!

• Reply • Share ›

Abhishek Pasayat ➚ GOPI GOPINATH • 9 months ago

Thanks a lot!!!

• Reply • Share ›

learner • 9 months ago

http://ideone.com/CI1uib

• Reply • Share ›

Deepesh Panjabi • 9 months ago

http://ideone.com/0vvbWt

• Reply • Share ›

elena • 9 months ago

hey I want to know that how we store value in a linked list,and link list is returned by function and then printed by main

• Reply • Share ›

DS+Algo=Placement elena • 8 months ago

A node is created using malloc() and then node->data is assigned the value.
The linked list is returned by returning only the pointer of its head.
I hope you understood or maybe I couldn't understand what you are asking.
Please clearly ask your doubt.

• Reply • Share ›

vishal • 10 months ago

```
struct node *getnthNode(struct node * start, int n)
{
struct node *p=start;
int count=1;

while(count<=n)
{
count++;
p=p->link;
}

printf("%d th node data part = %d",n,p->data);

return start;
}
```

• Reply • Share ›

Ansuraj Khadanga vishal • a month ago

What if the list is empty.. or "n=1" ? Take care of boundary conditions.

• Reply • Share ›

ashish vishal • 3 months ago

take care of the boundary cases....please

• Reply • Share ›

ANA vishal • 10 months ago

I think it will be while (count < n)

• Reply • Share ›

Abhi • 10 months ago

#include<stdio.h>

```
#include<stdlib.h>

typedef struct Node

{

int data;

struct Node* next;

}Node;

void InsertNode(Node** head,int new_data)

{

Node* new_node=(Node*)malloc(sizeof(Node));

new_node->data=new_data;
```

see more

• Reply • Share ›

praveen • 10 months ago

@neelabhsingh

• Reply • Share ›

neelabhsingh • 10 months ago

@GeeksforGeeks, here you are counting the number of node, but suppose that is number of node in the list is less than N then above code will not work.

• Reply • Share ›

Hitesh Lalwani neelabhsingh • 2 months ago

```
//get nth node from list

NODE* GetNthNode(LinkedList list, int n)

{

NODE* cur = list.head;

while(cur!=NULL && n>0){

cur = cur->next;

n--;

}
```

```
if(n==0) printf("Element found.");

return cur;

}
```

• Reply • Share ›

praveen neelabhsingh • 10 months ago

That is the reason assert(0) is written at the end of that function.

• Reply • Share ›

DS+Algo=Placement praveen • 8 months ago

hey what is assert()?

• Reply • Share ›

gautham • 10 months ago

here after each push the head node doesnot update hence here instead of linkedList single nodes are created.. correct me if i am wrong..

• Reply • Share ›

nithin gautham • 10 months ago

head is updated, see the last line of push function

• Reply • Share ›

Himanshu Dagar • a year ago

http://ideone.com/U646FI

can refer to this code

• Reply • Share ›

Himanshu Dagar • a year ago

(y)

• Reply • Share ›

Himanshu Dagar • a year ago

liked

great job!!!

• Reply • Share ›

kuldeep tripathi • a year ago

#include<iostream>

#include <algorithm>

```
#include <cstdio>

using namespace std;

struct node

{

int info;

struct node *link;

};

class LinkedList

{
```

see more

3 • Reply • Share ›

Load more comments

- Interview Experiences
- Advanced Data Structures
- Dynamic Programming
- Greedy Algorithms
- Backtracking
- Pattern Searching
- Divide & Conquer
- Mathematical Algorithms
- Recursion
- Geometric Algorithms

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

Recent Comments

- lebron

  since the array size is 5, it takes constant...

  K’th Smallest/Largest Element in Unsorted Array | Set 3 (Worst Case Linear Time) · 2 hours ago

- lebron

  merge sort

  K’th Smallest/Largest Element in Unsorted Array | Set 3 (Worst Case Linear Time) · 2 hours ago

- Shubham Sharma

  You saved my time :)

  Searching for Patterns | Set 2 (KMP Algorithm) · 3 hours ago

- Prakhar

  Why so many LOCs, if I'm not wrong (please...

  Largest Sum Contiguous Subarray · 3 hours ago

- Aayush Gupta

  For R4 Q3, Another solution would be to use a...

  Amazon Interview Experience | Set 168 · 4 hours ago

- EigenHarsha

  For Power Of 2, We Simply Doing.. var1 =

  Practo Interview Experience | Set 2 (Off-Campus) · 4 hours ago