Delete a given node in Linked List under given constraints

Given a Singly Linked List, write a function to delete a given node. Your function must follow following constraints:
1) It must accept pointer to the start node as first parameter and node to be deleted as second parameter i.e., pointer to head node is not global.
2) It should not return pointer to the head node.
3) It should not accept pointer to pointer to head node.

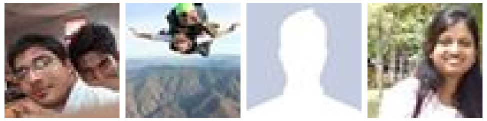
You may assume that the Linked List never becomes empty.

Let the function name be deleteNode(). In a straightforward implementation, the function needs to modify head pointer when the node to be deleted is first node. As discussed in previous post, when a function modifies the head pointer, the function must use one of the given approaches, we can't use any of those approaches here.

Solution
We explicitly handle the case when node to be deleted is first node, we copy the data of next node to head and delete the next node. The cases when deleted node is not the head node can be handled normally by finding the previous node and changing next of previous node. Following is C implementation.

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

/* structure of a linked list node */
struct node
{
    int data;
```



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

```
    struct node *next;
};
```

```
void deleteNode(struct node *head, struct node *n)
{
    // When node to be deleted is head node
    if(head == n)
    {
        if(head->next == NULL)
        {
            printf("There is only one node. The list can't be made emp
            return;
        }

        /* Copy the data of next node to head */
        head->data = head->next->data;

        // store address of next node
        n = head->next;

        // Remove the link of next node
        head->next = head->next->next;

        // free memory
        free(n);

        return;
    }


    // When not first node, follow the normal deletion process

    // find the previous node
    struct node *prev = head;
    while(prev->next != NULL && prev->next != n)
        prev = prev->next;

    // Check if node really exists in Linked List
    if(prev->next == NULL)
    {
        printf("\n Given node is not present in Linked List");
        return;
    }

    // Remove node from Linked List
    prev->next = prev->next->next;
```

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

```
    // Free memory
    free(n);

    return;
}

/* Utility function to insert a node at the begining */
void push(struct node **head_ref, int new_data)
{
    struct node *new_node =
        (struct node *)malloc(sizeof(struct node));
    new_node->data = new_data;
    new_node->next = *head_ref;
    *head_ref = new_node;
}

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

/* Driver program to test above functions */
int main()
{
    struct node *head = NULL;

    /* Create following linked list
      12->15->10->11->5->6->2->3 */
    push(&head,3);
    push(&head,2);
    push(&head,6);
    push(&head,5);
    push(&head,11);
    push(&head,10);
    push(&head,15);
    push(&head,12);

    printf("Given Linked List: ");
    printList(head);

    /* Let us delete the node with value 10 */
    printf("\nDeleting node %d: ", head->next->next->data);
    deleteNode(head, head->next->next);

    printf("\nModified Linked List: ");
    printList(head);

    /* Let us delete the the first node */
    printf("\nDeleting first node ");
    deleteNode(head, head);

    printf("\nModified Linked List: ");
    printList(head);

    getchar();
    return 0;
}
```

Output:

```
Given Linked List: 12 15 10 11 5 6 2 3

Deleting node 10:
Modified Linked List: 12 15 11 5 6 2 3

Deleting first node
Modified Linked List: 15 11 5 6 2 3
```

Please write comments if you find the above codes/algorithms incorrect, or find other ways to solve the same problem.

Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 43 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...

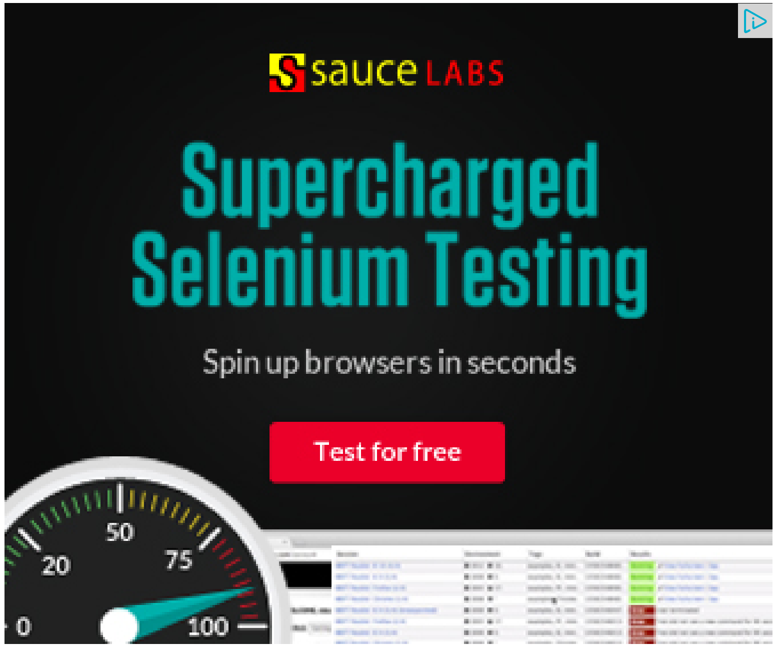
Root to leaf path sum equal to a given number · 1 hour ago

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



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

7 Tweet 0 0

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

13 Comments GeeksforGeeks

Sort by Newest ▾

Join the discussion

Join the discussion…

kavi • 7 months ago

I understand that if the list contains only a SINGLE NODE, and we wish to dele
program.

I can already see that statement , LIS "The list can't be made empty" , But still
IMPOSSIBLE if the list contains single node?

Am I right? Please let me know if any other way if that single node can be dele
Thanks.

^ | ˅ • Reply • Share ›

ashatm • 11 months ago

when the first node is to be deleted, you have copied the data of the second n
second node, but that actually does not delete the given node; also the constr
how does that mean we can't update head in the delete function(in case we d
head to the second node)?

^ | ˅ • Reply • Share ›

ashatm ➦ ashatm • 11 months ago

i get it... sorry for the silly doubt..

^ | ˅ • Reply • Share ›

Nidhi ➦ ashatm • 10 months ago

How will the changes made in head pointer be maintained whe
neither are e returning head?

^ | ˅ • Reply • Share ›

Rakesh • a year ago

n = prev->next;
This line is missing before

prev->next = prev->next->next;

⌃ | ⌄ • Reply • Share ›

abhishek • 2 years ago

```
 package sam;

import java.util.ArrayList;
import java.util.List;

public class am {

        public static void main(String[] args) {
                //Generate linked list with 19->20->3->70->4->5
                List<Integer> n = new ArrayList<Integer>();
                n.add(19);
                n.add(20);
                n.add(3);
                n.add(70);
                n.add(4);
                n.add(5);
                LinkedList formed = generateLinkedList(n);
                deleteNode(formed.next.next.next);
```

see more

⌃ | ⌄ • Reply • Share ›

jogi • 2 years ago

this question was asked in Oravle interview.

⌃ | ⌄ • Reply • Share ›

ahmet alp balkan • 2 years ago

I think you should state that this linked list is singly-linked.

^ | ⌄ • Reply • Share ›

Chummi ➦ ahmet alp balkan • 2 years ago

interviewers generally do not mention it. If they say linked list they meai

^ | ⌄ • Reply • Share ›

GeeksforGeeks ➦ ahmet alp balkan • 2 years ago

@ahmet alp balkan: Thaks for the suggestion. We have added Singly t

^ | ⌄ • Reply • Share ›

trying_to_learn ➦ GeeksforGeeks • 2 years ago

Hi,

I understand the algorithm and tried to run it in Visual Studio ani
I do not understand this line in the deleteNode function.

// store address of next node
n = head->next;

Followed by free(n).

This means that n points to the next node which is 15 in this ca
means free the memory to which n is pointing at i.e 15. The prc
which is the head node but I did not understand how it deleted

Any help would be highly appreciated.

Thank you,

Best
A

```
/* Paste your code here (You may delete these lines if
```

^ | ⌄ • Reply • Share ›

amitp49 ➦ trying_to_learn • 2 years ago

@trying_to_learn

Here the main logic is we can not change the head poin
to next node. Rather what it does is it copy the data of n
second both node have the same data as of second no

So in fact we reach to the state where we have deleted
y it delete the second node to reach to final answer stat

Hope it helps u...

```
/* Copy the data of next node to head */
       head->data = head->next->data;

       // store address of next node
       n = head->next;

       // Remove the link of next node
       head->next = head->next->next;

       // free memory
       free(n);
```

^ | ⌄ • Reply • Share ›

Manish ➦ amitp49 • 2 years ago

Need one help. If I copy the head node to some temp nc
to head->next, then my head is pointing to the second n
have to set the temp->next = null to delete the previous
then since there is no ref to the prev node, memory will

then since there is no ref to the prev node, memory will
it is required.

Node temp = head->next;
head = head->next;
temp->next = null;

Can some one tell me what is wrong with the above line

```
/* Paste your code here (You may delete these li
```

^ | v • Reply • Share ›

Subscribe Add Disqus to your site