

GeeksforGeeks

A computer science portal for geeks

[Android App](#) [GeeksQuiz](#)

[Login/Register](#)

- [Home](#)
- [Algorithms](#)
- [DS](#)
- [GATE](#)
- [Interview Corner](#)
- [Q&A](#)
- [C](#)
- [C++](#)
- [Java](#)
- [Books](#)
- [Contribute](#)
- [Ask a Q](#)
- [About](#)

[Array](#)

[Bit Magic](#)

[C/C++](#)

[Articles](#)

[GFacts](#)

[Linked List](#)

[MCQ](#)

[Misc](#)

[Output](#)

[String](#)

[Tree](#)

[Graph](#)

Given only a pointer to a node to be deleted in a singly linked list, how do you delete it?

A **simple solution** is to traverse the linked list until you find the node you want to delete. But this solution requires pointer to the head node which contradicts the problem statement.

Fast solution is to copy the data from the next node to the node to be deleted and delete the next node. Something like following.

```
struct node *temp = node_ptr->next;
node_ptr->data = temp->data;
node_ptr->next = temp->next;
```

```
free(temp);
```

Program:

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

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

void deleteNode(struct node *node_ptr)
{
    struct node *temp = node_ptr->next;
    node_ptr->data = temp->data;
    node_ptr->next = temp->next;
    free(temp);
}

/* 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);

    printf("\n Before deleting \n");
    printList(head);

    /* I m deleting the head itself.
    You can check for more cases */
    deleteNode(head);

    printf("\n After deleting \n");
    printList(head);
    getchar();
    return 0;
}

```

This solution doesn't work if the node to be deleted is the last node of the list. To make this solution work we can mark the end node as a dummy node. But the programs/functions that are using this function should also be modified.

You can try this problem for doubly linked list.

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: [Linked Lists](#), [loop](#)

Like < 13

Tweet < 2

 < 1

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



GeeksforGeeks

Like

93,232 people like GeeksforGeeks.



Feedback, social media

- [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](#)

Follow @GeeksforGeeks

[Subscribe](#)

• 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](#)

•

@geeksforgeeks, [Some rights reserved](#) [Contact Us!](#)

Powered by [WordPress](#) & [MooTools](#), customized by geeksforgeeks team