GeeksforGeeks
A computer science portal for geeks

Custom Search

Courses

Login

Suggest an Article

Delete a Linked List node at a given position

Given a singly linked list and a position, delete a linked list node at the given position.

Example:

```
Input: position = 1, Linked List = 8->2->3->1->7
Output: Linked List =  8->3->1->7

Input: position = 0, Linked List = 8->2->3->1->7
Output: Linked List = 2->3->1->7
```

Recommended: Please solve it on "*PRACTICE*" first, before moving on to the solution.

1 الإرهابي يندم فينتقم viu.com محمد رمضان يفضح الإرهاب بأقوى أفلامه

2 Join as a Senior Engineer Challenge Yourself, Build Something Bi
Andela

If node to be deleted is root, simply delete it. To delete a middle node, we must have pointer to the node previous to the node to be deleted. So if positions is not zero, we run a loop position-1 times and get pointer to the previous node.

Below is the implementation of above idea.

C/C++

```c
// A complete working C program to delete a node in a linked list
// at a given position
#include <stdio.h>
#include <stdlib.h>

// A linked list node
struct Node
{
    int data;
    struct Node *next;
};

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

/* Given a reference (pointer to pointer) to the head of a list
   and a position, deletes the node at the given position */
void deleteNode(struct Node **head_ref, int position)
{
   // If linked list is empty
   if (*head_ref == NULL)
      return;

   // Store head node
   struct Node* temp = *head_ref;

    // If head needs to be removed
    if (position == 0)
    {
        *head_ref = temp->next;   // Change head
        free(temp);               // free old head
```

```
            return;
        }

        // Find previous node of the node to be deleted
        for (int i=0; temp!=NULL && i<position-1; i++)
             temp = temp->next;

        // If position is more than number of ndoes
        if (temp == NULL || temp->next == NULL)
             return;

        // Node temp->next is the node to be deleted
        // Store pointer to the next of node to be deleted
        struct Node *next = temp->next->next;

        // Unlink the node from linked list
        free(temp->next);  // Free memory

        temp->next = next;  // Unlink the deleted node from list
    }

    // This function prints contents of linked list starting from
    // the given node
    void printList(struct Node *node)
    {
        while (node != NULL)
        {
            printf(" %d ", node->data);
            node = node->next;
        }
    }

    /* Drier program to test above functions*/
    int main()
    {
        /* Start with the empty list */
        struct Node* head = NULL;

        push(&head, 7);
        push(&head, 1);
        push(&head, 3);
        push(&head, 2);
    push(&head, 8);

    puts("Created Linked List: ");
    printList(head);
    deleteNode(&head, 4);
    puts("\nLinked List after Deletion at position 4: ");
    printList(head);
    return 0;
}
```

Java

```
// A complete working Java program to delete a node in a linked list
// at a given position
class LinkedList
{
    Node head;  // head of list

    /* Linked list Node*/
    class Node
    {
        int data;
        Node next;
        Node(int d)
        {
            data = d;
            next = null;
        }
    }

    /* Inserts a new Node at front of the list. */
    public void push(int new_data)
    {
        /* 1 & 2: Allocate the Node &
                  Put in the data*/
        Node new_node = new Node(new_data);

        /* 3. Make next of new Node as head */
        new_node.next = head;
```

```
            /* 4. Move the head to point to new Node */
            head = new_node;
        }

        /* Given a reference (pointer to pointer) to the head of a list
           and a position, deletes the node at the given position */
        void deleteNode(int position)
        {
            // If linked list is empty
            if (head == null)
                return;

            // Store head node
            Node temp = head;

            // If head needs to be removed
            if (position == 0)
            {
                head = temp.next;   // Change head
                return;
            }

            // Find previous node of the node to be deleted
            for (int i=0; temp!=null && i<position-1; i++)
                temp = temp.next;

            // If position is more than number of ndoes
            if (temp == null || temp.next == null)
                return;

            // Node temp->next is the node to be deleted
            // Store pointer to the next of node to be deleted
            Node next = temp.next.next;

            temp.next = next;  // Unlink the deleted node from list
        }

        /* This function prints contents of linked list starting from
            the given node */
        public void printList()
        {
```

```
        Node tnode = head;
        while (tnode != null)
        {
            System.out.print(tnode.data+" ");
            tnode = tnode.next;
        }
    }

    /* Drier program to test above functions. Ideally this function
       should be in a separate user class.  It is kept here to keep
       code compact */
    public static void main(String[] args)
    {
        /* Start with the empty list */
        LinkedList llist = new LinkedList();

        llist.push(7);
        llist.push(1);
        llist.push(3);
        llist.push(2);
        llist.push(8);

        System.out.println("\nCreated Linked list is: ");
        llist.printList();

        llist.deleteNode(4);  // Delete node at position 4

        System.out.println("\nLinked List after Deletion at position 4: ");
        llist.printList();
    }
}
```

Python

```
# Python program to delete a node in a linked list
# at a given position

# Node class
class Node:
        # Constructor to initialize the node object
        def __init__(self, data):
            self.data = data
            self.next = None

    class LinkedList:

        # Constructor to initialize head
        def __init__(self):
            self.head = None

        # Function to insert a new node at the beginning
        def push(self, new_data):
            new_node = Node(new_data)
            new_node.next = self.head
            self.head = new_node

        # Given a reference to the head of a list
        # and a position, delete the node at a given position
        def deleteNode(self, position):

            # If linked list is empty
            if self.head == None:
                return

            # Store head node
            temp = self.head

            # If head needs to be removed
            if position == 0:
                self.head = temp.next
                temp = None
                return

            # Find previous node of the node to be deleted
            for i in range(position -1 ):
                temp = temp.next
                if temp is None:
                    break

            # If position is more than number of nodes
        if temp is None:
            return
        if temp.next is None:
            return

        # Node temp.next is the node to be deleted
        # store pointer to the next of node to be deleted
        next = temp.next.next

        # Unlink the node from linked list
        temp.next = None

        temp.next = next


    # Utility function to print the linked LinkedList
    def printList(self):
        temp = self.head
        while(temp):
            print " %d " %(temp.data),
            temp = temp.next


# Driver program to test above function
llist = LinkedList()
llist.push(7)
llist.push(1)
llist.push(3)
llist.push(2)
llist.push(8)

print "Created Linked List: "
llist.printList()
llist.deleteNode(4)
print "\nLinked List after Deletion at position 4: "
llist.printList()

# This code is contributed by Nikhil Kumar Singh(nickzuck_007)
```

Output:

```
Created Linked List:
 8  2  3  1  7
Linked List after Deletion at position 4:
 8  2  3  1
```

Delete a Linked List node at a given position | GeeksforGeeks

Thanks to **Hemanth Kumar** for suggesting initial solution. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

Recommended Posts:

Find the middle of a given linked list in C and Java

Program for n'th node from the end of a Linked List

Write a function to get Nth node in a Linked List

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

Detect loop in a linked list

Write a function to delete a Linked List

Write a function that counts the number of times a given int occurs in a Linked List

Reverse a linked list

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

Write a function to get the intersection point of two Linked Lists.

Function to check if a singly linked list is palindrome

The Great Tree-List Recursion Problem.

Clone a linked list with next and random pointer | Set 1

Memory efficient doubly linked list

Given a linked list which is sorted, how will you insert in sorted way

ActiveState Wanna Fork Python?
• Cu
• Ad
• Tes

Article Tags : Linked List

Practice Tags : Linked List

10

☐ To-do ☐ Done

1.8

Based on **153** vote(s)

Feedback/ Suggest Improvement Add Notes Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments Share this post!

A computer science portal for geeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org

COMPANY

About Us
Careers
Privacy Policy
Contact Us

LEARN

Algorithms
Data Structures
Languages
CS Subjects
Video Tutorials

PRACTICE

Company-wise
Topic-wise
Contests
Subjective Questions

CONTRIBUTE

Write an Article
Write Interview Experience
Internships
Videos

@geeksforgeeks, Some rights reserved