**Delete a Node in Single Linked List :-**

Easy Accuracy: 39.85% Submissions: 156K+ Points: 2

Given a singly linked list and an integer x.Delete xth node from the singly linked list.

**Example 1:**

**Input:** 1 -> 3 -> 4

  x = 3

**Output:** 1 -> 3

**Explanation:**

After deleting the node at 3rd

position (1-base indexing), the

linked list is as 1 -> 3.

**Example 2:**

**Input:** 1 -> 5 -> 2 -> 9

x = 2

**Output:** 1 -> 2 -> 9

**Explanation:**

After deleting the node at 2nd

position (1-based indexing), the

linked list is as 1 -> 2 -> 9.

**Your task:**Your task is to complete the method **deleteNode()** which takes two arguments: the address of the head of the linked list and an integer x. The function returns the head of the modified linked list.

**Constraints:**  
2 <= N <= 105  
1 <= x <= N

**Code :-**

//{ Driver Code Starts

// C program to find n'th Node in linked list

#include <stdio.h>

#include <stdlib.h>

#include<iostream>

using namespace std;

/\* Link list Node \*/

struct Node

{

int data;

struct Node\* next;

Node(int x){

data = x;

next = NULL;

}

};

void append(struct Node\*\* head\_ref, struct Node \*\*tail\_ref,

int new\_data)

{

struct Node\* new\_node = new Node(new\_data);

if (\*head\_ref == NULL)

\*head\_ref = new\_node;

else

(\*tail\_ref)->next = new\_node;

\*tail\_ref = new\_node;

}

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

struct Node\* deleteNode(struct Node \*head,int );

void printList(Node \*head)

{

while (head != NULL)

{

cout << head->data << " ";

head = head->next;

}

cout << "\n";

}

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

int main()

{

int T, i, n, l;

cin>>T;

while(T--){

struct Node \*head = NULL, \*tail = NULL;

cin>>n;

for(i=1;i<=n;i++)

{

cin>>l;

append(&head, &tail, l);

}

int kk;

cin>>kk;

head = deleteNode(head,kk);

printList(head);

}

return 0;

}

// } Driver Code Ends

/\* Link list Node

struct Node

{

int data;

struct Node\* next;

Node(int x){

data = x;

next = NULL;

}

};

\*/

/\*You are required to complete below method\*/

Node\* deleteNode(Node \*head,int x)

{

if(x==1)

return head->next;

int ind=1;

Node \*ptr=head, \*prev=NULL;

while(ptr){

if(ind==x and prev){

prev->next = ptr->next;

return head;

}

prev = ptr;

ptr = ptr->next;

++ind;

}

return head;

}

**T.C :- ϴ(N)**

**S.C :- O(1)**