# Rajalakshmi Engineering College

Name: Riya Sharma.k
Email: 240701433@rajalakshmi.edu.in
Roll no: 240701433
Phone: 9790940806
Branch: REC
Department: I CSE FE
Batch: 2028
Degree: B.E - CSE

Scan to verify results

## NeoColab_REC_CS23231_DATA STRUCTURES

### REC_DS using C_Week 1_COD_Question 2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

### Section 1 : Coding

1. Problem Statement

Arun is learning about data structures and algorithms. He needs your help in solving a specific problem related to a singly linked list.

Your task is to implement a program to delete a node at a given position. If the position is valid, the program should perform the deletion; otherwise, it should display an appropriate message.

***Input Format***

The first line of input consists of an integer N, representing the number of elements in the linked list.

The second line consists of N space-separated elements of the linked list.

The third line consists of an integer x, representing the position to delete.

Position starts from 1.

***Output Format***

The output prints space-separated integers, representing the updated linked list after deleting the element at the given position.

If the position is not valid, print "Invalid position. Deletion not possible."

Refer to the sample output for formatting specifications.

***Sample Test Case***

Input: 5
8 2 3 1 7
2
Output: 8 3 1 7

***Answer***

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

void insert(int);
void display_List();
void deleteNode(int);

struct node {
   int data;
   struct node* next;
} *head = NULL, *tail = NULL;


typedef struct node node;
void insert(int value)
{
   node*newnode=(node*)malloc(sizeof(node));
   newnode->data=value;
   newnode->next=NULL;
   if(head==NULL)
   {
      head=tail=newnode;
```

```
    }
    else
    {
        tail->next=newnode;
        tail=newnode;
    }
}
int getlength()
{
    int count=0;
    node* temp=head;
    while(temp!=NULL)
    {
        count++;
        temp=temp->next;
    }
    return count;
}
void display_list()
{
    node*temp=head;
    while(temp!=NULL)
    {
        printf("%d ",temp->data);
        temp=temp->next;
    }
    printf("\n");
}
void delbeg()
{
    if(head!=NULL)
    {
        node*tempnode=head;
        head=head->next;
        free(tempnode);
    }
}
void delend()
{
    if(head==NULL)
    return;
    if(head->next==NULL)
```

```
    {
        free(head);
        head=NULL;
        return;
    }
    node*temp=head;
    while(temp->next->next!=NULL)
    temp=temp->next;
    free(temp->next);
    temp->next=NULL;
}
void delmid(int position)
{
    if(position==1)
    {
        delbeg();
        return;
    }
    node*temp=head;
    node*p=NULL;
    int count=1;
    while(temp!=NULL && count<position)
    {
        p=temp;
        temp=temp->next;
        count++;
    }
    p->next=temp->next;
    free(temp);
}
void deleteNode(int pos)
{
    int n=getlength();
    if(pos<1 || pos>n)
    {
        printf("Invalid position.Deletion not possible .");
    }
    else if(pos==1)
    {
        delbeg();
        display_list();
    }
```

```
    else if(pos==n)
    {
        delend();
        display_list();
    }
    else
    {
        delmid(pos);
        display_list();
    }
}

int main() {
    int num_elements, element, pos_to_delete;

    scanf("%d", &num_elements);

    for (int i = 0; i < num_elements; i++) {
        scanf("%d", &element);
        insert(element);
    }

    scanf("%d", &pos_to_delete);

    deleteNode(pos_to_delete);

    return 0;
}
```

***Status :*** Correct **Marks : 10/10**