Rajalakshmi Engineering College

Name: tharunika R
Email: 241801296@rajalakshmi.edu.in
Roll no: 241801296
Phone: 6369646218
Branch: REC
Department: I AI & DS FD
Batch: 2028
Degree: B.E - AI & DS

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 : 0

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;

#include<stdio.h>
#include<stdlib.h>
struct Node {
   int data;
   struct Node* next;
};
struct Node* createNode(int data) {
   struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
   newNode->data = data;
   newNode->next = NULL;
```

```
    return newNode;
}
void append(struct Node** head,int data) {
    struct Node* newNode = createNode(data);
    if(*head == NULL) {
        *head = newNode;
        return;
    }
    struct Node* temp = *head;
    while(temp->next != NULL) {
        temp = temp->next;
    }
    temp->next = newNode;
}
void deleteAtPosition(struct Node** head, int position) {
    if(*head == NULL) {
        printf("Invalid position. Deletion not possible\n");
        return;
    }
    struct Node* temp = *head;
    if(position == 1) {
        *head = temp->next;
        free(temp);
        return;
    }
    for(int i = 1; temp != NULL && i < position - 1; i++){
        temp = temp->next;
    }
    if(temp == NULL || temp->next == NULL) {
        printf("Invalid podition. Deletion not possible\n");
        return;
    }
    struct Node* next = temp->next->next;
    free(temp->next);
    temp->next = next;
}
void printList(struct Node* head) {
    if (head == NULL) {
        printf("Invalid position. Deletion not possible\n");
        return;
    }
    struct Node* temp = head;
    while(temp != NULL) {
        printf("%d ", temp->data);
        temp = temp->next;
    }
    printf("\n");
}
int main() {
    int N,x;
    scanf("%d",&N);
    struct Node* head = NULL;
    for (int i = 0; i < N; i++) {
        int data;
        scanf("%d", &data);
        append(&head, data);
    }
    scanf("%d", &x);
    deleteAtPosition(&head, x);
    printList(head);
    return 0;
}


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 : Wrong   ***Marks : 0/10***