

# Rajalakshmi Engineering College

Name: SANJAYRAM M

Email: 240701469@rajalakshmi.edu.in

Roll no: 240701469

Phone: 9360732156

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 2\_COD\_Question 5

Attempt : 1

Total Mark : 10

Marks Obtained : 10

### Section 1 : Coding

#### 1. Problem Statement

Ashwin is tasked with developing a simple application to manage a list of items in a shop inventory using a doubly linked list. Each item in the inventory has a unique identification number. The application should allow users to perform the following operations:

Create a List of Items: Initialize the inventory with a given number of items. Each item will be assigned a unique number provided by the user and insert the elements at end of the list.

Delete an Item: Remove an item from the inventory at a specific position.

Display the Inventory: Show the list of items before and after deletion.

If the position provided for deletion is invalid (e.g., out of range), it should

display an error message.

### ***Input Format***

The first line contains an integer  $n$ , representing the number of items to be initially entered into the inventory.

The second line contains  $n$  integers, each representing the unique identification number of an item separated by spaces.

The third line contains an integer  $p$ , representing the position of the item to be deleted from the inventory.

### ***Output Format***

The first line of output prints "Data entered in the list:" followed by the data values of each node in the doubly linked list before deletion.

If  $p$  is an invalid position, the output prints "Invalid position. Try again."

If  $p$  is a valid position, the output prints "After deletion the new list:" followed by the data values of each node in the doubly linked list after deletion.

Refer to the sample output for the formatting specifications.

### ***Sample Test Case***

Input: 4

1 2 3 4

5

Output: Data entered in the list:

node 1 : 1

node 2 : 2

node 3 : 3

node 4 : 4

Invalid position. Try again.

### ***Answer***

```
// You are using GCC
```

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

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

```
typedef struct Node{
    int element;
    struct Node *next;
    struct Node *prev;
}node;
```

```
void insertend(node*list,int e){
    node *newnode=(node*)malloc(sizeof(node));
    node *position;
    newnode->next=NULL;
    newnode->element=e;
    if(list->next==NULL){
        newnode->prev=list;
        list->next=newnode;
    }
    else{
        position=list;
        while(position->next!=NULL){
            position=position->next;
        }
        newnode->prev=position;
        position->next=newnode;
    }
}
```

```
void traverse(node *list){
    node *position=list->next;
    int i=1;
    while(position!=NULL){
        printf("node %d : %d\n",i,position->element);
        position=position->next;
        i++;
    }
}
```

```
node *find(node *list,int p){
    node *position=list->next;
    int i=1;
    while(position!=NULL && i!=p){
        position=position->next;
        i++;
    }
    return position;
```

```

}
void delet(node *list,int p){
    if(list->next!=NULL){
        node *temp;
        node *position=find(list,p);
        if(position!=NULL){
            temp=position;
            if(position->next!=NULL){
                temp=position;
                position->prev->next=position->next;
                position->next->prev=position->prev;
            }
            else{
                position->prev->next=NULL;
            }
            free(temp);
        }
    }
}

```

```

int main(){
    node *list=(node*)malloc(sizeof(node));
    list->next=NULL;
    int n,e,p;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        scanf("%d",&e);
        insertend(list,e);
    }
    scanf("%d",&p);
    printf("Data entered in the list:\n");
    traverse(list);
    if(p>n || p<1){
        printf("Invalid position. Try again.\n");
        return 0;
    }
    else{
        printf("After deletion the new list:\n");
        delet(list,p);
        traverse(list);
    }
    return 0;
}

```

}

**Status :** Correct

**Marks : 10/10**