

# Rajalakshmi Engineering College

Name: karthieswaran L

Email: 241801117@rajalakshmi.edu.in

Roll no: 241801117

Phone: 9840695531

Branch: REC

Department: I AI & DS FB

Batch: 2028

Degree: B.E - AI & DS

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>
```

```

struct Node{
    int data;
    struct Node*next;
    struct Node*prev;
};

void insertAtend(struct Node** head,int data){
    struct Node*newNode=(struct Node*)malloc(sizeof(struct Node));
    newNode->data=data;
    newNode->next=NULL;
    if(*head==NULL){
        newNode->prev=NULL;
        *head=newNode;
        return;
    }
    struct Node*last=*head;
    while(last->next!=NULL)
    {
        last=last->next;
    }
    last->next=newNode;
    newNode->prev=last;
}

void deleteAtposition(struct Node**head,int position){
    if(*head==NULL)
    {
        return;
    }
    struct Node*temp=*head;
    if(position==1){
        *head=temp->next;
        if(*head!=NULL){
            (*head)->prev=NULL;
        }
        free(temp);
        return;
    }
    for(int i=1;temp!=NULL && i<position;i++){
        temp=temp->next;
    }
    if(temp==NULL)
    {
        printf("Invalid position.Try again.\n");
    }
}

```

```

        return;
    }
    if(temp->prev!=NULL){
        temp->prev->next=temp->next;
    }
    if(temp->next!=NULL){
        temp->next->prev=temp->prev;
    }
    free(temp);
}

void displayList(struct Node*node){
    int count=1;
    while(node!=NULL){
        printf(" node%d:%d\n",count,node->data);
        node=node->next;
        count++;
    }
}

void freeList(struct Node*head){
    struct Node*temp;
    while(head!=NULL){
        temp=head;
        head=head->next;
        free(temp);
    }
}

int main()
{
    struct Node*head=NULL;
    int n,p,item;
    scanf("%d",&n);
    for(int i = 0;i<n;i++)
    {
        scanf("%d",&item);
        insertAtend(&head,item);
    }
    scanf("%d",&p);
    printf("Data entered in the list:\n");
    displayList(head);
    if(p<1||p>n){
        printf("Invalid position.Try again.\n");
    }
}

```

```
else{
    deleteAtposition(&head,p);
    printf("\nAfter deletion the new list:\n");
    displayList(head);
}
freeList(head);
return 0;
}
```

**Status :** Correct

**Marks : 10/10**