

# Rajalakshmi Engineering College

Name: Sneha Raju R  
Email: 240701519@rajalakshmi.edu.in  
Roll no: 240701519  
Phone: 7550004064  
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\_CY

Attempt : 1  
Total Mark : 30  
Marks Obtained : 30

#### Section 1 : Coding

##### 1. Problem Statement

Vanessa is learning about the doubly linked list data structure and is eager to play around with it. She decides to find out how the elements are inserted at the beginning and end of the list.

Help her implement a program for the same.

##### ***Input Format***

The first line of input contains an integer N, representing the size of the doubly linked list.

The next line contains N space-separated integers, each representing the values to be inserted into the doubly linked list.

##### ***Output Format***

The first line of output prints the integers, after inserting them at the beginning, separated by space.

The second line prints the integers, after inserting at the end, separated by space.

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 5

1 2 3 4 5

Output: 5 4 3 2 1

1 2 3 4 5

### **Answer**

```
#include <stdio.h>
#include <stdlib.h>
struct node{
    int ele;
    struct node *prev;
    struct node *next;
};
typedef struct node Node;

void insertend(Node **head,int ele){
    Node *newnode=(Node *)malloc(sizeof(Node));
    newnode->ele=ele;
    newnode->next=NULL;
    if(*head==NULL){
        newnode->prev=NULL;
        *head=newnode;
    }else{
        Node *temp=*head;
        while(temp->next!=NULL){
            temp=temp->next;
        }
        temp->next=newnode;
        newnode->prev=temp;
    }
}
```

```

}

void displayf(Node *head){
    while(head!=NULL){
        printf("%d ",head->ele);
        head=head->next;
    }
    printf("\n");
}

```

```

void displayb(Node *head){
    Node *tail=head;
    while(tail->next!=NULL){
        tail=tail->next;
    }
    while(tail!=NULL){
        printf("%d ",tail->ele);
        tail=tail->prev;
    }
    printf("\n");
}

```

```

int main(){
    Node *head=NULL;
    int n,ele;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        scanf("%d",&ele);
        insertend(&head,ele);
    }
    displayb(head);
    displayf(head);
    return 0;
}

```

**Status :** Correct

**Marks : 10/10**

## 2. Problem Statement

Ashiq is developing a ticketing system for a small amusement park. The park issues tickets to visitors in the order they arrive. However, due to a

system change, the oldest ticket (first inserted) must be revoked instead of the last one.

To manage this, Ashiq decided to use a doubly linked list-based stack, where:

Pushing adds a new ticket to the top of the stack. Removing the first inserted ticket (removing from the bottom of the stack). Printing the remaining tickets from bottom to top.

### ***Input Format***

The first line consists of an integer  $n$ , representing the number of tickets issued.

The second line consists of  $n$  space-separated integers, each representing a ticket number in the order they were issued.

### ***Output Format***

The output prints space-separated integers, representing the remaining ticket numbers in the order from bottom to top.

Refer to the sample output for formatting specifications.

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

Input: 7

24 96 41 85 97 91 13

Output: 96 41 85 97 91 13

### ***Answer***

```
#include <stdio.h>
#include <stdlib.h>
struct node{
    int ele;
    struct node *prev;
    struct node *next;
};
typedef struct node Node;

void insertend(Node **head,int ele){
```

```

Node *newnode=(Node *)malloc(sizeof(Node));
newnode->ele=ele;
newnode->next=NULL;
if(*head==NULL){
    newnode->prev=NULL;
    *head=newnode;
}else{
    Node *temp=*head;
    while(temp->next!=NULL){
        temp=temp->next;
    }
    temp->next=newnode;
    newnode->prev=temp;
}
}

void displayf(Node *head){
    head=head->next;
    while(head!=NULL){
        printf("%d ",head->ele);
        head=head->next;
    }
    printf("\n");
}

```

```

int main(){
    Node *head=NULL;
    int n,ele;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        scanf("%d",&ele);
        insertend(&head,ele);
    }
    displayf(head);
    return 0;
}

```

**Status :** Correct

**Marks :** 10/10

3. Problem Statement

Sam is learning about two-way linked lists. He came across a problem where he had to populate a two-way linked list and print the original as well as the reverse order of the list. Assist him with a suitable program.

### ***Input Format***

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

The second line consists of  $n$  space-separated integers, representing the elements.

### ***Output Format***

The first line displays the message: "List in original order:"

The second line displays the elements of the doubly linked list in the original order.

The third line displays the message: "List in reverse order:"

The fourth line displays the elements of the doubly linked list in reverse order.

Refer to the sample output for the formatting specifications.

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

Input: 5

1 2 3 4 5

Output: List in original order:

1 2 3 4 5

List in reverse order:

5 4 3 2 1

### ***Answer***

```
#include <stdio.h>
#include <stdlib.h>
struct node{
    int ele;
    struct node *next;
```

```

    struct node *prev;
};
typedef struct node Node;

void insertend(Node **head,int ele){
    Node *newnode=(Node *)malloc(sizeof(Node));
    newnode->next=NULL;
    newnode->ele=ele;
    if(*head==NULL){
        newnode->prev=NULL;
        *head=newnode;
    }else{
        Node *temp=*head;
        while(temp->next!=NULL){
            temp=temp->next;
        }
        temp->next=newnode;
        newnode->prev=temp;
    }
}

```

```

void displayf(Node *head){
    while(head!=NULL){
        printf("%d ",head->ele);
        head=head->next;
    }
    printf("\n");
}

```

```

void displayb(Node *head){
    Node *tail=head;
    while(tail->next!=NULL){
        tail=tail->next;
    }
    while(tail!=NULL){
        printf("%d ",tail->ele);
        tail=tail->prev;
    }
    printf("\n");
}

```

```

int main(){

```

```
Node *head=NULL;
int n,ele;
scanf("%d",&n);
for(int i=0;i<n;i++){
    scanf("%d",&ele);
    insertend(&head,ele);
}
printf("List in original order:\n");
displayf(head);
printf("List in reverse order:\n");
displayb(head);
return 0;
}
```

**Status :** Correct

**Marks : 10/10**