Rajalakshmi Engineering College

Name: RAKSHITHA R
Email: 240701418@rajalakshmi.edu.in
Roll no: 240701418
Phone: 7305274265
Branch: REC
Department: I CSE FD
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

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

```
// You are using GCC
#include<stdio.h>
#include<stdlib.h>
typedef struct node{
   int data;
   struct node*prev;
   struct node*next;
}node;
node*createnode(int data){
   node*newnode=(node*)malloc(sizeof(node));
   newnode->data=data;
   newnode->prev=NULL;
   newnode->next=NULL;
   return newnode;
}
int main()
{
   int n,i,value;
   node*head=NULL,*tail=NULL,*temp;
   scanf("%d",&n);
   for(i=0;i<n;i++){
     scanf("%d ",&value);
   node*newnode = createnode(value);
   if(head==NULL){
     head=tail=newnode;
   }
   else{
     tail->next=newnode;
     newnode->prev=tail;
     tail=newnode;
   }
   }
   printf("List in original order:\n");
   temp=head;
   while(temp!=NULL)
   {
     printf("%d ",temp->data);
     temp=temp->next;
   }printf("\n");
   printf("List in reverse order:\n");
   temp=tail;
   while(temp!=NULL){
     printf("%d ",temp->data);
     temp=temp->prev;
   }printf("\n");
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

```
// You are using GCC
#include<stdio.h>
#include<stdlib.h>
struct node{
   int t;
   struct node*next;
   struct node*prev;
};
void push(struct node**top,int t){
   struct node*nnode=(struct node*)malloc(sizeof(struct node));
   nnode->t=t;
   nnode->next=*top;
   nnode->prev=NULL;
   if(*top!=NULL){
      (*top)->prev=nnode;
   }*top=nnode;
}
void rem(struct node**top){
   if(*top==NULL){
      return;
```

```
    }
    struct node*temp=*top;
    while(temp->next!=NULL){
        temp=temp->next;
    }
    if(temp->prev!=NULL){
        temp->prev->next=NULL;
    }else{
        *top=NULL;
    }free(temp);
}
void pstack(struct node*top){
    if(top==NULL){
        printf("\n");
        return;
    }
    struct node*temp=top;
    while(temp->next!=NULL){
        temp=temp->next;
    }
    while(temp!=NULL){
        printf("%d ",temp->t);
        temp=temp->prev;
    }printf("\n");
}
void fstack(struct node*top){
    struct node*temp;
    while(top!=NULL){
        temp=top;
        top=top->next;
        free(temp);
    }
}
int main(){
    struct node*ts=NULL;
    int n,t;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        scanf("%d",&t);
        push(&ts,t);
    }
    rem(&ts);
    pstack(ts);
    fstack(ts);
}
```

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

3. Problem Statement

Imagine Anu is tasked with finding the middle element of a doubly linked list. Given a doubly linked list where each node contains an integer value and is inserted at the end, implement a program to find the middle element of the list. If the number of nodes is even, return the middle element pair.

Input Format

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

The second line consists of N space-separated integers, representing the values of the nodes in the doubly linked list.

Output Format

The first line of output prints the space-separated elements of the doubly linked list.

The second line prints the middle element(s) of the doubly linked list, depending on whether the number of nodes is odd or even.

Refer to the sample outputs for the formatting specifications.

Sample Test Case

Input: 5
10 20 30 40 50

Output: 10 20 30 40 50
30

Answer

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

```
#include<stdio.h>
#include<stdlib.h>
struct node{
   int d;
   struct node*next;
   struct node*prev;
};
struct node*create(int d){
   struct node*nnode=(struct node*)malloc(sizeof(node));
   nnode->d=d;
   nnode->next=NULL;
   nnode->prev=NULL;
   return nnode;
}
void ins(struct node**head,int d){
   struct node*nnode=create(d);
   if(*head==NULL){
      *head=nnode;
      return;
   }
   struct node*temp=*head;
   while(temp->next!=NULL){
      temp=temp->next;
   }
   temp->next=nnode;
   nnode->prev=temp;
}
void dis(struct node*head){
   struct node*temp=head;
   while(temp!=NULL){
      printf("%d ",temp->d);
      temp=temp->next;
   }
      printf("\n");
}
void mid(struct node*head,int n){
   struct node*temp=head;
   for(int i=0;i<(n-1)/2;i++){
      temp=temp->next;
   }
   if(n%2==1){
      printf("%d\n",temp->d);
```

```
    }else{
        printf("%d %d\n",temp->d,temp->next->d);
    }
} int main(){
    int n,d;
    struct node*head=NULL;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        scanf("%d",&d);
        ins(&head,d);
    }
    dis(head);
    mid(head,n);
}
```

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