```
#include<stdio.h>
#include<conio.h>
#include<alloc.h>
#define NULL 0

struct listelement
{
int item;
struct listelement *next;
};
typedef struct listelement node;

int menu()
{
int choice;
do
{
printf("\n\n MAIN MENU");
printf("\n ---------");
printf("\n 1.CREATE \n 2.INSERT \n 3.DELETE \n 4.Exit");
printf("\n Enter your choice:");
scanf("%d",&choice);
if(choice<1||choice>4)
printf("\n Wrong choice");
}while(choice<1||choice>4);
return(choice);
}

node *create(node **lastnode)
{
node *temp,*firstnode;
int info;
*lastnode=NULL;
firstnode=NULL;
printf("\n Enter the data:");
scanf("%d",&info);
while(info!=-999)
{
temp=(node *)malloc(sizeof(node));
temp->item=info;
temp->next=NULL;
```

```
if(firstnode==NULL)
firstnode=temp;
else
(*lastnode)->next=temp;
(*lastnode)=temp;
scanf("%d",&info);
}
if(firstnode!=NULL)
temp->next=firstnode;
return(firstnode);
}

void display(node *first,node *last)
{
do
{
printf("\t %d",first->item);
first=first->next;
}while(last->next!=first);
return;
}

void insert(node **first,node **last)
{
node *newnode;
node *temp;
int newitem,pos,i;
printf("\n Enter the new item:");
scanf("%d",&newitem);
printf("\n Position of insertion:");
scanf("%d",&pos);
if(((*first)==NULL)||(pos==1))
{
newnode=(node *)malloc(sizeof(node));
newnode->item=newitem;
newnode->next=*first;
*first=newnode;
if((*last)!=NULL)
(*last)->next=*first;
}
else
{
i=1;
```

```
temp=*first;
while((i<(pos-1)) && ((temp->next)!=(*first)))
{
i++;
temp=temp->next;
}
newnode=(node *)malloc(sizeof(node));
if(temp->next==(*first))
*last=newnode;
newnode->item=newitem;
newnode->next=temp->next;
temp->next=newnode;
}
}

void delet(node **first,node **last)
{
node *temp;
node *prev;
int target;
printf("\n Data to be deleted:");
scanf("%d",&target);
if(*first==NULL)
printf("\n List is empty");
else if((*first)->item==target)
{
if((*first)->next==*first)
*first=*last=NULL;
else
{
*first=(*first)->next;
(*last)->next=*first;
printf("\n Circular list\n");
display(*first,*last);
}
}
else
{
temp=*first;
prev=NULL;
while((temp->next!=(*first))&&((temp->item)!=target))
{
prev=temp;
```

```
temp=temp->next;
}
if(temp->item!=target)
{
printf("\n Element not found");
}
else
{
if(temp==*last)
*last=prev;
prev->next=temp->next;
printf("\n CIRCULAR LIST");
display(*first,*last);
}
}
}

void main()
{
node *start,*end;
int choice;
clrscr();
printf("\n CIRCULAR LINKED LIST");
printf("\n --------------------");
do
{
choice=menu();
switch(choice)
{
case 1:
printf("\n Type -999 to stop");
start=create(&end);
printf("\n Circular list\n");
display(start,end);
continue;

case 2:
insert(&start,&end);
printf("\n Circular  list \n");
display(start,end);
continue;

case 3:
```

```
delet(&start,&end);
continue;
default:
printf("\n End");
}
}while(choice!=4);
}
```

-->>Sample Input and Output

```
        CIRCULAR LINKED LIST
         --------------------------------

MIAN NENU

1.CREATE
2.INSERT
3.DELETE
4.EXIT
Enter your choice:1

Type -999 to stop
Enter the data:10
20
30
-999
Circular list

10  20  30

MIAN NENU

1.CREATE
2.INSERT
3.DELETE
4.EXIT
Enter your choice:2

Enter the new item:40

Position of insertion:2

Circular list
```

10 40 20 30

MIAN NENU

1.CREATE
2.INSERT
3.DELETE
4.EXIT
Enter your choice:3

Data to be deleted:20

Circular List
10 40 30

MIAN NENU

1.CREATE
2.INSERT
3.DELETE
4.EXIT
Enter your choice:3

Data to be deleted:60

Element not found