```
/*INSERTION AND DELETION IN circular LINKED LIST*/
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
void Insertion_at_beg();
void Insertion_at_end();
void Insertion_at_specific();
void Deletion_from_beg();
void Deletion_from_end();
void Deletion_from_specific();
void display();
struct node
{
int data;
struct node *next;
}
struct node *start=NULL;
void main()
{
int ch;
clrscr();
printf("1.Insertion at beg \n
        2.Insertion at end \n
        3.Insert before data\n
        4.Insert after data\n
```

```
            5.Deletion from beg\n
            6.Deletion from end\n
            7.Deletion from specific \n
            8.Display\n
            9.sort list
            10.Exit");
do
{
printf("\nEnter ur choice: ");
scanf("%d",&ch);
switch(ch)
{
case 1:Insertion_at_beg();break;
case 2:Insertion_at_end();break;
case 3:Insert_Before_data();break;
case 4:Insert_After_data();break;
case 5:Deletion_from_beg();break;
case 6:Deletion_from_end();break;
case 7:delete_Specific();break;
case 8:display_node();break;
case 9:Sort_list();break;
case 10:printf("Program Exited");break;
default:printf("Invalid choice");
}
}while(ch!=10);
```

```
getch();
}

void Insertion_at_beg()
{
struct node *ptr,*new_node;
printf("Enter the item");
scanf("%d",&item);
new_node=(struct node *)malloc(sizeof(struct node));
new_node->data=item;
new_node->next=NULL;
ptr=start;
while(ptr->next!=start)
ptr=ptr->next;
ptr -> next = new_node;
new_node -> next = start;
start = new_node
printf("Inserted Element is %d",item);
}

void Insertion_at_end()
{
struct node *ptr, *new_node;
int num;
```

```
printf("\n Enter the data : ");
scanf("%d", &num);
new_node = (struct node *)malloc(sizeof(struct node));
new_node -> data = num;
ptr = start;
while(ptr -> next != start)
ptr = ptr -> next;
ptr -> next = new_node;
new_node -> next = start;
}
```

```
Void Insert_Before_data()
{
struct node *new_node, *ptr, *preptr;
int num, val;
printf("\n Enter the data : ");
scanf("%d", &num);
printf("\n Enter the value before which the data has to be inserted : ");
scanf("%d", &val);
new_node = (struct node *)malloc(sizeof(struct node));
new_node -> data = num;
ptr = start;
while(ptr -> data != val)
```

```
{
preptr = ptr;
ptr = ptr -> next;
}
preptr -> next = new_node;
new_node -> next = ptr;
}



Void Insert_After_data()
{
struct node *new_node, *ptr, *preptr;
int num, val;
printf("\n Enter the data : ");
scanf("%d", &num);
printf("\n Enter the value after which the data has to be inserted : ");
scanf("%d", &val);
new_node = (struct node *)malloc(sizeof(struct node));
new_node -> data = num;
ptr = start;
new_node = ptr;
while (ptr-> != NULL)
        {
        if (ptr->data == n)
        {
```

```
                    new_node->next=ptr->next;
                    ptr->next=new_node;
            }

else
ptr=ptr->next;
}
}

void delete_beg()
{
 struct node *ptr;
ptr = start;
while(ptr -> next != start)
ptr = ptr -> next;
ptr -> next = start -> next;
free(start);
start = ptr -> next;
}

Void delete_Specific()
{
struct node *ptr, *preptr;
```

```
int val;
printf("\n Enter the value of the node which has to be deleted : ");
scanf("%d", &val);
ptr = start;
 if(ptr -> data == val)
            {
            ptr = start;
            while(ptr -> next != start)
            ptr = ptr -> next;
            ptr -> next = start -> next;
            free(start);
            start = ptr -> next;
            }
 else
            {
            while(ptr -> data != val)
            {
            preptr = ptr;
            ptr = ptr -> next;
            }
            preptr -> next = ptr -> next;
            free(ptr);
            }
}
```

```
void delete_end()
{
int data1
ptr = start;
while(ptr -> next != start)
{
preptr = ptr;
ptr = ptr -> next;
}
preptr -> next = ptr -> next;
data1=ptr->data;
free(ptr);
}

printf("Deleted element %d",data1);
free(ptr);
}

void Sort_list()
{
struct node *ptr1, *ptr2;
int temp;
ptr1 = start;
while(ptr1 -> next != NULL)
{
```

```
ptr2 = ptr1 -> next;
while(ptr2 != NULL)
{
if(ptr1 -> data > ptr2 -> data)
{
temp = ptr1 -> data;
ptr1 -> data = ptr2 -> data;
ptr2 -> data = temp;
}
ptr2 = ptr2 -> next;
}
ptr1 = ptr1 -> next;
}
}

void display_node()
{
struct node *ptr;
ptr = start;
while(ptr != NULL)
{
printf("\t %d", ptr -> data);
ptr = ptr -> next;
}
}
```