```c
#include<stdio.h>
struct Node{
    int data;
    struct Node *link;
};
typedef struct Node node;
node *start=NULL,*temp,*new1,*pre,*next,*curr;
int ch,pos,ele;
char c;
void insert_beg();
void insert_end();
void insert_at_specific_position();
void delete_first();
void delete_last();
void delete_specific_element();
void display();
void main(){
    while(1){
        printf("Circular linked list operation");
        printf("\n1.Create\n2.Insert at beginning\n3.Insert at end\n4.Insert at specific position\n5.Delete first element\n6.dslete last element\n7.Delete specific element\n8.Display\n9.Exit\n");
        printf("Enter your choice:");
        scanf("%d",&ch);
        switch(ch){
            case 1:create();
                   break;
            case 2:insert_beg();
                   break;
            case 3:insert_end();
                   break;
            case 4:insert_at_specific_position();
                   break;
            case 5:delete_first();
                   break;
            case 6:delete_last();
                   break;
            case 7:delete_specific_element();
                   break;
            case 8:display();
                   break;
            case 9:exit(0);
                   break;
        }
    }
}
void create(){
    do{
    new1=(node*)malloc(sizeof(node));
    printf("Enter element:");
    scanf("%d",&new1->data);
    if(start==NULL){
        start=new1;
        curr=new1;
    }
    else{
```

```
            curr->link=newl;
            curr=newl;
        }
        printf("Do you want to add another element(Y/N):");
        scanf("%s",&c);
        }while(c=='y' || c=='Y');
        curr->link=start;
}
void insert_beg(){
        newl=(node*)malloc(sizeof(node));
        printf("Enter element:");
        scanf("%d",&newl->data);
        if(start==NULL){
            start=newl;
            newl->link=start;
            return;
        }
        temp=start;
        while(temp->link!=start){
            temp=temp->link;
        }
        temp->link=newl;
        newl->link=start;
        start=newl;
}
void insert_end(){
        newl=(node*)malloc(sizeof(node));
        printf("Enter element:");
        scanf("%d",&newl->data);
        if(start==NULL){
            start=newl;
            newl->link=start;
            return;
        }
        temp=start;
        while(temp->link!=start){
            temp=temp->link;
        }
        temp->link=newl;
        newl->link=start;
}
void insert_at_specific_position(){
        newl=(node*)malloc(sizeof(node));
        printf("Enter element:");
        scanf("%d",&newl->data);
        printf("Enter position:");
        scanf("%d",&pos);
        if(pos==1){
            temp=start;
            while(temp->link!=start){
            temp=temp->link;
            }
            temp->link=newl;
            newl->link=start;
```

```
        temp=temp->link;
        i++;
    }
    if(temp->link==start && i!=pos-1){
        printf("Entered position is greater than number of elements");
        return;
    }
    if(temp->link==start){
        temp->link=new1;
        new1->link=start;
        return;
    }
    new1->link=temp->link;
    temp->link=new1;
}
void delete_first(){
    if(start==NULL){
        printf("Circular linked list is empty");
        return;
    }
    if(start->link==start){
        free(start);
        start=NULL;
        return;
    }
    temp=start;
    while(temp->link!=start){
        temp=temp->link;
    }
    temp->link=start->link;
    free(start);
    start=temp->link;
}
void delete_last(){
    if(start==NULL){
        printf("Circular linked list is empty");
        return;
    }
    if(start->link==start){
        free(start);
        start=NULL;
        return;
    }
    temp=start;
    while(temp->link->link!=start){
        temp=temp->link;
    }
    free(temp->link);
    temp->link=start;
}
void delete_specific_element(){
    if(start==NULL){
        printf("Circular linked list is empty");
        return;
```

```
    if(start==NULL){
        printf("Circular linked list is empty");
        return;
    }
    printf("Enter element to be deleted:");
    scanf("%d",&ele);
    if(start->data==ele && start->link==start){
        free(start);
        start=NULL;
        return;
    }
    if(start->data==ele){
        temp=start;
        while(temp->link!=start){
            temp=temp->link;
        }
        temp->link=start->link;
        free(start);
        start=temp->link;
        return;
    }
    pre=NULL;
    next=start;
    while(next->data!=ele && next->link!=start){
        pre=next;
        next=next->link;
    }
    if(next->data==ele && next->link!=start){
        pre->link=next->link;
        free(next);
        return;
    }
    if(next->data==ele && next->link==start){
        pre->link=start;
        free(next);
        return;
    }
    printf("Element not found");
}
void display(){
    if(start==NULL){
        printf("Circular linked list is empty");
        return;
    }
    temp=start;
    printf("Circular linked list contains:\n");
    while(temp->link!=start){
        printf("%d\t",temp->data);
        temp=temp->link;
    }
    printf("%d",temp->data);
}
```

```
Circular linked list operation
1.Create
2.Insert at beginning
3.Insert at end
4.Insert at specific position
5.Delete first element
6.delete last element
7.Delete specific element
8.Display
9.Exit
Enter your choice:1
Enter element:12
Do you want to add another element(Y/N):y
Enter element:23
Do you want to add another element(Y/N):Y
Enter element:43
Do you want to add another element(Y/N):N
Circular linked list operation
1.Create
2.Insert at beginning
3.Insert at end
4.Insert at specific position
5.Delete first element
6.delete last element
7.Delete specific element
8.Display
9.Exit
Enter your choice:3
Enter element:45
Circular linked list operation
1.Create
2.Insert at beginning
3.Insert at end
4.Insert at specific position
5.Delete first element
6.delete last element
7.Delete specific element
8.Display
9.Exit
Enter your choice:8
Circular linked list contains:
12      23      43      45Circular linked list
1.Create
2.Insert at beginning
3.Insert at end
4.Insert at specific position
5.Delete first element
6.delete last element
7.Delete specific element
8.Display
9.Exit
```

```
Enter your choice:8
Circular linked list contains:
12      23      43      45Circular linked list operati
1.Create
2.Insert at beginning
3.Insert at end
4.Insert at specific position
5.Delete first element
6.delete last element
7.Delete specific element
8.Display
9.Exit
Enter your choice:6
Circular linked list operation
1.Create
2.Insert at beginning
3.Insert at end
4.Insert at specific position
5.Delete first element
6.delete last element
7.Delete specific element
8.Display
9.Exit
Enter your choice:5
Circular linked list operation
1.Create
2.Insert at beginning
3.Insert at end
4.Insert at specific position
5.Delete first element
6.delete last element
7.Delete specific element
8.Display
9.Exit
Enter your choice:8
Circular linked list contains:
23      43Circular linked list operation
1.Create
2.Insert at beginning
3.Insert at end
4.Insert at specific position
5.Delete first element
6.delete last element
7.Delete specific element
8.Display
9.Exit
Enter your choice:9

Process returned 0 (0x0)   execution time : 51.782 s
Press any key to continue.
```