Aim:

Write a program that uses functions to perform the following **operations on Circular linked list**
i)Creation ii)insertion iii)deletion iv) Traversal

Source Code:

AlloperationsinCLL.c

```c
#include<stdio.h>
#include<stdlib.h>
struct node {
    int data;
    struct node*next;
};
void insert();
void deletion();
void find();
void print();
struct node*head=NULL;
int main()
{
    int choice;
    printf("CIRCULAR LINKED LIST IMPLEMENTATION OF LIST ADT\n");
    while(1)
    {
        printf("1.INSERT");
        printf(" 2.DELETE");
        printf(" 3.FIND");
        printf(" 4.PRINT");
        printf(" 5.QUIT\n");
        printf("Enter the choice: ");
        scanf("%d",&choice);
        switch(choice) {
            case 1:insert();break;
            case 2:deletion();break;
            case 3:find();break;
            case 4:print();break;
            case 5:exit(0);
        }
    }
}
void insert()
{
    int x,n;
    struct node*newnode,*temp=head,*prev;
    newnode=(struct node*)malloc(sizeof(struct node));
    printf("Enter the element to be inserted: ");
    scanf("%d",&x);
    printf("Enter the position of the element: ");
    scanf("%d",&n);
    newnode->data=x;
    newnode->next=NULL;
    if(head==NULL)
```

```c
    {
        head=newnode;
        newnode->next=newnode;
    }
    else if(n==1)
    {
        temp=head;
        newnode->next=temp;
        while(temp->next!=head)
        temp=temp->next;
        temp->next=newnode;
        head=newnode;
    }
    else
    {
        for(int i=1;i<n-1;i++)
        {
            temp=temp->next;
        }
        newnode->next=temp->next;
        temp->next=newnode;
    }
}
void deletion()
{
    struct node*temp=head,*prev,*temp1=head;
    int key,count=0;
    printf("Enter the element to be deleted: ");
    scanf("%d",&key);
    if(temp->data==key)
    {
        prev=temp->next;
        while(temp->next!=head)
        {
            temp=temp->next;
        }
        temp->next=prev;
        free(head);
        head=prev;
        printf("Element deleted\n");
    }
    else
    {
        while(temp->next!=head)
        {
            if(temp->data==key)
            {
                count+=1;
                break;
            }
            prev=temp;
            temp=temp->next;
        }
        if(temp->data==key)
        {
            prev->next=temp->next;
```

```c
            free(temp);
            printf("Element deleted\n");
        }
        else
        {
            printf("Element does not exist...!\n");
        }
    }
}
void find()
{
    struct node*temp=head;
    int key,count=0;
    printf("Enter the element to be searched: ");
    scanf("%d",&key);
    while(temp->next!=head)
    {
        if(temp->data==key)
        {
            count=1;
            break;
        }
        temp=temp->next;
    }
    if(count==1)
    printf("Element exist...!\n");
    else {
        if(temp->data==key)
        printf("Element exist...!\n");
        else
        printf("Element does not exist...!\n");
    }
}
void print() {
    struct node*temp=head;
    printf("The list element are: ");
    while(temp->next!=head) {
        printf("%d -> ",temp->data);
        temp=temp->next;
    }
    printf("%d -> ",temp->data);
    printf("\n");
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
CIRCULAR LINKED LIST IMPLEMENTATION OF LIST ADT 1
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 1
Enter the choice: 1
Enter the element to be inserted: 12
Enter the position of the element: 1
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 1

| Enter the choice: 1 |
| Enter the element to be inserted: 14 |
| Enter the position of the element: 2 |
| 1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 1 |
| Enter the choice: 1 |
| Enter the element to be inserted: 15 |
| Enter the position of the element: 3 |
| 1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 4 |
| Enter the choice: 4 |
| The list element are: 12 -> 14 -> 15 -> 2 |
| 1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 2 |
| Enter the choice: 2 |
| Enter the element to be deleted: 14 |
| Element deleted 4 |
| 1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 4 |
| Enter the choice: 4 |
| The list element are: 12 -> 15 -> 3 |
| 1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 3 |
| Enter the choice: 3 |
| Enter the element to be searched: 12 |
| Element exist...! 5 |
| 1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 5 |
| Enter the choice: 5 |

| Test Case - 2 |
| User Output |
| CIRCULAR LINKED LIST IMPLEMENTATION OF LIST ADT 1 |
| 1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 1 |
| Enter the choice: 1 |
| Enter the element to be inserted: 54 |
| Enter the position of the element: 1 |
| 1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 2 |
| Enter the choice: 2 |
| Enter the element to be deleted: 1 |
| Element does not exist...! 4 |
| 1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 4 |
| Enter the choice: 4 |
| The list element are: 54 -> 1 |
| 1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 1 |
| Enter the choice: 1 |
| Enter the element to be inserted: 65 |
| Enter the position of the element: 2 |
| 1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 4 |
| Enter the choice: 4 |
| The list element are: 54 -> 65 -> 5 |
| 1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT 5 |
| Enter the choice: 5 |