```
#include<stdio.h>

#include<conio.h>

#include<stdlib.h>


int CQ[10],f=-1,r=-1,MAX,c=0,elem; /* Global declarations */


int  CQfull()

{                  /* Function to Check Circular Queue Full */

   if( c==MAX)

          return 1;

   else

          return 0;

}


int CQempty()

{                 /* Function to Check Circular Queue Empty */

   if(c== 0)

           return 1;

   else

          return 0;

}


void CQinsert(int elem)

{                    /* Function for Insert operation */

       r=(r+1) % MAX;

       CQ[r]=elem;

       c++;


}


int CQdelete()
```

```
{                    /* Function for Delete operation */
       f=(f+1) % MAX;
       elem=CQ[f];
       c--;
       return(elem);

}



void display()
{       /* Function to display status of Circular Queue */
    int i,temp;

    if(CQempty())
        printf(" \n No elements in the Queue\n");
    else
    {
        printf(" \n Contents of the Queue are\n");
        temp=(f+1)%MAX;
        for(i=1;i<=c;i++)
           {
                printf("%d ",CQ[temp]);
                temp=(temp+1) % MAX;
           }
    }
}



void main()
{                      /* Main Program */
    int ch;
```

```
printf("\n Enter the size of the queue :");
scanf("%d",&MAX);
while(1)
 {
    printf("\n  1-Insert, 2-Delete,3-Display, Any other key to EXIT\n");
    printf("\n Eneter your choice ");
    scanf("%d",&ch);
    switch(ch)
    {
    case 1: if(CQfull())
                 printf("\n Queue Overflow");
              else
                {
                     printf("\nEnter the element to be Inserted ");
                      scanf("%d",&elem);
                      CQinsert(elem);
                }
            break;
    case 2: if(CQempty())
                     printf("\n Queue Underflow");
              else
                 {
                    elem=CQdelete();
                    printf("\nDeleted Element is %d ",elem);

                 }
          break;
    case 3: display();
          break;
    default:exit(0);
    }
 }
}
```