6) Develop a menu driven Program in C for the following operations on Circular QUEUE of Characters (Array Implementation of Queue with maximum size MAX)

a. Insert an Element on to Circular QUEUE
b. Delete an Element from Circular QUEUE
c. Demonstrate Overflow and Underflow situations on Circular QUEUE
d. Display the status of Circular QUEUE
e. Exit Support the program with appropriate functions for each of the above operations.

```
#include<stdio.h>
#include<stdlib.h>

#define MAX 5

char circular_queue[MAX];
int front = -1, rear = -1;

int isEmpty()
{
   if (front == -1 && rear == -1)
      return 1;
   else
      return 0;
}

int isFull()
{
   if ((rear + 1) % MAX == front)
      return 1;
   else
      return 0;
}

void insertElement(char element)
{
   if (isFull())
   {
      printf("Circular Queue Overflow\n");
      return;
   }
   else if (isEmpty())
   {
      front = rear = 0;
```

```
    }
    else
    {
        rear = (rear + 1) % MAX;
    }
    circular_queue[rear] = element;
}

void deleteElement()
{
    if (isEmpty())
    {
        printf("Circular Queue Underflow\n");
        return;
    }
    else if (front == rear)
    {
        front = rear = -1;
    }
    else
    {
        front = (front + 1) % MAX;
    }
}

void display()
{
    int i;
    if (isEmpty())
    {
        printf("Circular Queue is empty\n");
        return;
    }
    printf("Circular Queue elements: ");
    i = front;
    do
    {
        printf("%c ", circular_queue[i]);
        i = (i + 1) % MAX;
    }
    while (i != (rear + 1) % MAX);
    printf("\n");
}
```

```
int main()
{
   int choice;
   char element;
   do
   {
      printf("\n\n---- Circular Queue Menu ----\n");
      printf("1. Insert an Element\n");
      printf("2. Delete an Element\n");
      printf("3. Display Circular Queue\n");
      printf("4. Exit\n");
      printf("Enter your choice: ");
      scanf("%d", &choice);

      switch(choice)
      {
      case 1:
         printf("Enter element to be inserted: ");
         scanf(" %c", &element);
         insertElement(element);
         break;
      case 2:
         deleteElement();
         break;
      case 3:
         display();
         break;
      case 4:
         printf("Exiting...\n");
         break;
      default:
         printf("Invalid choice! Please enter a valid option.\n");
      }
   }
   while(choice != 4);

   return 0;
}
```

OUTPUT

```
---- Circular Queue Menu ----
1. Insert an Element
```

2. Delete an Element
3. Display Circular Queue
4. Exit
Enter your choice: 1
Enter element to be inserted: A

---- Circular Queue Menu ----
1. Insert an Element
2. Delete an Element
3. Display Circular Queue
4. Exit
Enter your choice: 1
Enter element to be inserted: B

---- Circular Queue Menu ----
1. Insert an Element
2. Delete an Element
3. Display Circular Queue
4. Exit
Enter your choice: 1
Enter element to be inserted: C

---- Circular Queue Menu ----
1. Insert an Element
2. Delete an Element
3. Display Circular Queue
4. Exit
Enter your choice: 3
Circular Queue elements: A B C

---- Circular Queue Menu ----
1. Insert an Element
2. Delete an Element
3. Display Circular Queue
4. Exit
Enter your choice: 2

---- Circular Queue Menu ----
1. Insert an Element

2. Delete an Element
3. Display Circular Queue
4. Exit
Enter your choice: 3
Circular Queue elements: B C

---- Circular Queue Menu ----
1. Insert an Element
2. Delete an Element
3. Display Circular Queue
4. Exit
Enter your choice: 4
Exiting...