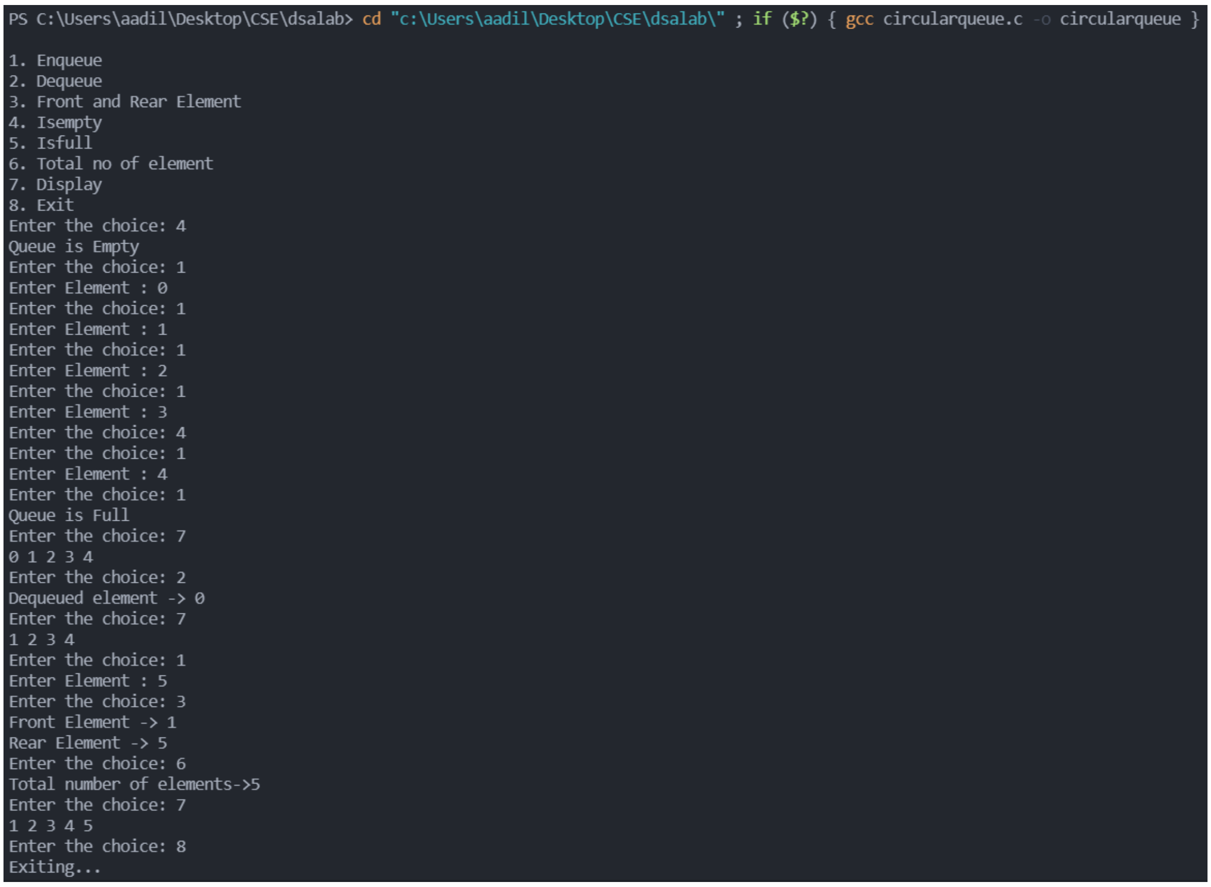
20BCS042 MOHD ADIL

PROGRAM 8: CIRCULAR QUEUE USING ARRAY

```c
#include <stdio.h>
#include <stdlib.h>
struct Queue
{
    int size;
    int front;
    int rear;
    int *Q;
};
void create(struct Queue *q, int size)
{
    q->size = size;
    q->front = q->rear = 0;
    q->Q = (int *)malloc(q->size * sizeof(int));
}
void enqueue(struct Queue *q)
{
    if ((q->rear + 1) % q->size == q->front)
        printf("Queue is Full\n");
    else
    {
        q->rear = (q->rear + 1) % q->size;
        printf("Enter Element : ");
        int n;
        scanf("%d", &n);
        q->Q[q->rear] = n;
    }
}
int dequeue(struct Queue *q)
{
    int x = -1;
    if (q->front == q->rear)
        printf("Queue is Empty\n");
    else
    {
        q->front = (q->front + 1) % q->size;
        x = q->Q[q->front];
    }
    return x;
}
int isEmpty(struct Queue *q)
{
    if (q->front == q->rear)
    {
        printf("Queue is Empty\n");
        return 1;
    }
    return 0;
}

int isFull(struct Queue *q)
{
    if ((q->rear + 1) % q->size == q->front)
    {
        printf("Queue is Full\n");
        return 1;
    }

    return 0;
}
void Display(struct Queue q)
{
    if (q.front == q.rear)
    {
        printf("Queue is Empty\n");
    }
    else
    {
        int i = (q.front + 1) % q.size;
        do
        {
            printf("%d ", q.Q[i]);
            i = (i + 1) % q.size;
        } while (i != (q.rear + 1) % q.size);
    }

    printf("\n");
}
int main()
{
    struct Queue q;
    create(&q, 6);
    int no_of_elements;
    int choice;
    printf("\n1. Enqueue\n2. Dequeue\n3. Front and Rear Element
\n4. Isempty\n5. Isfull\n6. Total no of element\n7. Display\n8.
Exit\n");
    while (1)
    {
        printf("Enter the choice: ");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            enqueue(&q);
            break;
        case 2:
            printf("Dequeued element->%d\n", dequeue(&q));
            break;
        case 3:
            printf("Front Element->%d\n", q.Q[q.front + 1]);
            printf("Rear Element->%d\n", q.Q[q.rear]);
        case 4:
            isEmpty(&q);
            break;
        case 5:
            isFull(&q);
            break;
        case 6:
            no_of_elements = q.front > q.rear ? (q.size - q.front
+ q.rear) : (q.rear - q.front);
            printf("Total number of elements->%d\n",
no_of_elements);
            break;
        case 7:
            Display(q);
            break;
        case 8:
            printf("Exiting...");
            exit(0);
            break;
        }
    }

    return 0;
}
```

OUTPUT:



```
PS C:\Users\aadil\Desktop\CSE\dsalab> cd "c:\Users\aadil\Desktop\CSE\dsalab\" ; if ($?) { gcc circularqueue.c -o circularqueue }

1. Enqueue
2. Dequeue
3. Front and Rear Element
4. Isempty
5. Isfull
6. Total no of element
7. Display
8. Exit
Enter the choice: 4
Queue is Empty
Enter the choice: 1
Enter Element : 0
Enter the choice: 1
Enter Element : 1
Enter the choice: 1
Enter Element : 2
Enter the choice: 1
Enter Element : 3
Enter the choice: 4
Enter the choice: 1
Enter Element : 4
Enter the choice: 1
Queue is Full
Enter the choice: 7
0 1 2 3 4
Enter the choice: 2
Dequeued element -> 0
Enter the choice: 7
1 2 3 4
Enter the choice: 1
Enter Element : 5
Enter the choice: 3
Front Element -> 1
Rear Element -> 5
Enter the choice: 6
Total number of elements->5
Enter the choice: 7
1 2 3 4 5
Enter the choice: 8
Exiting...
```