Name: Mithlesh Yeole

Roll no. : B3-B3-59

Practical 3B

Code:

```
#include <stdio.h>
#define SIZE 5
typedef struct {
    int items[SIZE];
    int front, rear;
} CircularQueue;
void initialize(CircularQueue *q) {
    q->front = -1;
    q->rear = -1;
}
int isEmpty(CircularQueue *q) {
    return (q->front == -1);
}
int isFull(CircularQueue *q) {
    return ((q->rear + 1) % SIZE == q->front);
}
void enqueue(CircularQueue *q, int value) {
    if (isFull(q)) {
        printf("Queue is full %d\n", value);
        return;
    }
    if (isEmpty(q)) {
        q->front = 0;
        q->rear = 0;
    } else {
        q->rear = (q->rear + 1) % SIZE;
    }
```

```
    q->items[q->rear] = value;
    printf("Added %d\n", value);
}
void dequeue(CircularQueue *q) {
    if (isEmpty(q)) {
        printf("Queue is empty\n");
        return;
    }
    printf("Removed %d from the queue\n", q->items[q->front]);
    if (q->front == q->rear) {
        q->front = -1;
        q->rear = -1;
    } else {
        q->front = (q->front + 1) % SIZE;
    }
}
void display(CircularQueue *q) {
    if (isEmpty(q)) {
        printf("Queue is empty\n");
        return;
    }
    printf("Queue elements: ");
    int i = q->front;
    while (1) {
        printf("%d ", q->items[i]);
        if (i == q->rear) break;
        i = (i + 1) % SIZE;
    }
    printf("\n");
}
int main() {
    CircularQueue q;
```

```
 initialize(&q);
   enqueue(&q, 1);
   enqueue(&q, 2);
   enqueue(&q, 3);
   enqueue(&q, 4);
   enqueue(&q, 5);
   enqueue(&q, 6);
   display(&q);
   dequeue(&q);
   dequeue(&q);
   display(&q);
   enqueue(&q, 10);
   enqueue(&q, 20);
   display(&q);
   return 0;
}
```

Output: