Experiment 03\circular_queue.c

```c
#include <stdio.h>
#define MAX 5

int queue[MAX];
int front = -1, rear = -1;

void enqueue(int value) {
    if ((front == 0 && rear == MAX - 1) || (rear == (front - 1) % (MAX - 1))) {
        printf("Queue Overflow\n");
        return;
    }
    if (front == -1) {
        front = rear = 0;
        queue[rear] = value;
    } else if (rear == MAX - 1 && front != 0) {
        rear = 0;
        queue[rear] = value;
    } else {
        rear++;
        queue[rear] = value;
    }
    printf("%d enqueued\n", value);
}

void dequeue() {
    if (front == -1) {
        printf("Queue Underflow\n");
        return;
    }
    printf("%d dequeued\n", queue[front]);
    if (front == rear) {
        front = rear = -1;
    } else if (front == MAX -1) {
        front = 0;
    } else {
        front++;
    }
}

void display() {
    if (front == -1) {
        printf("Queue is empty\n");
        return;
    }
    printf("Queue: ");
    if (rear >= front) {
        for (int i = front; i <= rear; i++)
            printf("%d ", queue[i]);
    } else {
        for (int i = front; i < MAX; i++)
            printf("%d ", queue[i]);
        for (int i = 0; i <= rear; i++)
            printf("%d ", queue[i]);
    }
    printf("\n");
}

int main() {
    int choice, value;
    while (1) {
        printf("\n1. Enqueue  2. Dequeue  3. Display  4. Exit\n");
        printf("Enter choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                printf("Enter value: ");
                scanf("%d", &value);
                enqueue(value);
                break;
            case 2:
                dequeue();
                break;
            case 3:
                display();
                break;
            case 4:
                return 0;
            default:
                printf("Invalid choice\n");
        }
    }
}

```