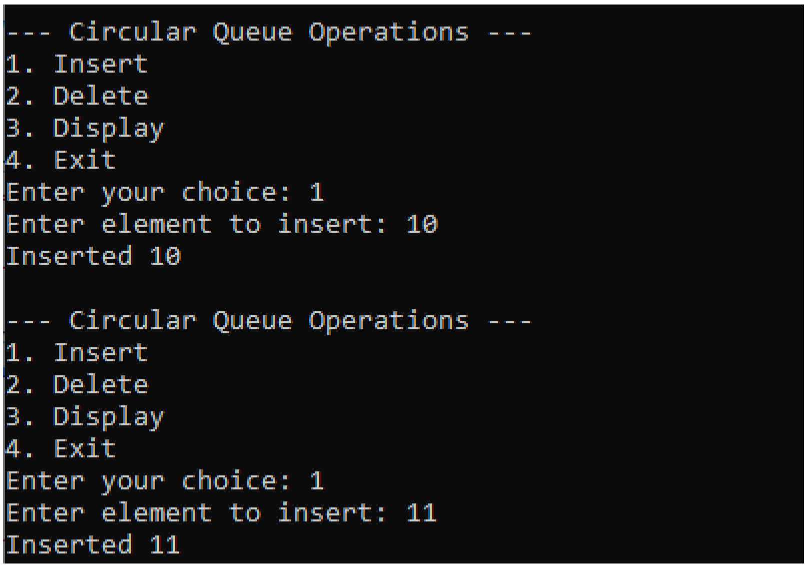
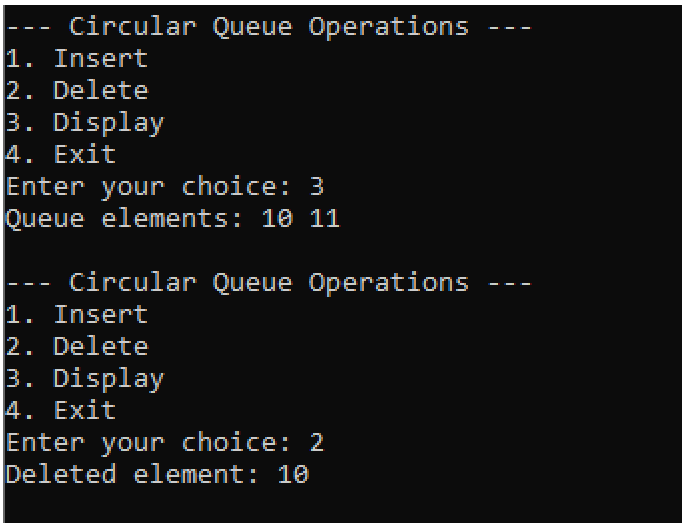
```c
#include <stdio.h>
#define MAX 5

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

void insert() {
    int item;
    if ((front == 0 && rear == MAX - 1) || (front == rear + 1)) {
        printf("Queue Overflow! Cannot insert.\n");
    } else {
        printf("Enter element to insert: ");
        scanf("%d", &item);

        if (front == -1)
            front = rear = 0;
        else if (rear == MAX - 1)
            rear = 0;
        else
            rear++;

        queue[rear] = item;
        printf("Inserted %d\n", item);
    }
}

void delete() {
    if (front == -1) {
        printf("Queue Underflow! Queue is empty.\n");
    } else {
        printf("Deleted element: %d\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");
    } else {
        printf("Queue elements: ");
        int i = front;
        if (front <= rear) {
            while (i <= rear) {
                printf("%d ", queue[i]);
                i++;
            }
        } else {
            while (i < MAX) {
                printf("%d ", queue[i]);
                i++;
            }
            i = 0;
            while (i <= rear) {
                printf("%d ", queue[i]);
                i++;
            }
        }
        printf("\n");
    }
}

int main() {
    int choice;

    while (1) {
        printf("\n--- Circular Queue Operations ---\n");
        printf("1. Insert\n");
        printf("2. Delete\n");
        printf("3. Display\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1: insert(); break;
            case 2: delete(); break;
            case 3: display(); break;
            case 4: return 0;
            default: printf("Invalid choice! Try again.\n");
        }
    }
}
```



```
--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter element to insert: 10
Inserted 10

--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter element to insert: 11
Inserted 11
```



```
--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 3
Queue elements: 10 11

--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 2
Deleted element: 10
```