EXERCISE -13

Write a C program to implement Queue operations such as ENQUEUE, DEQUEUE and Display

AIM:

To write a C program to implement Queue operations: Enqueue, Dequeue, and Display.

ALGORITHM:

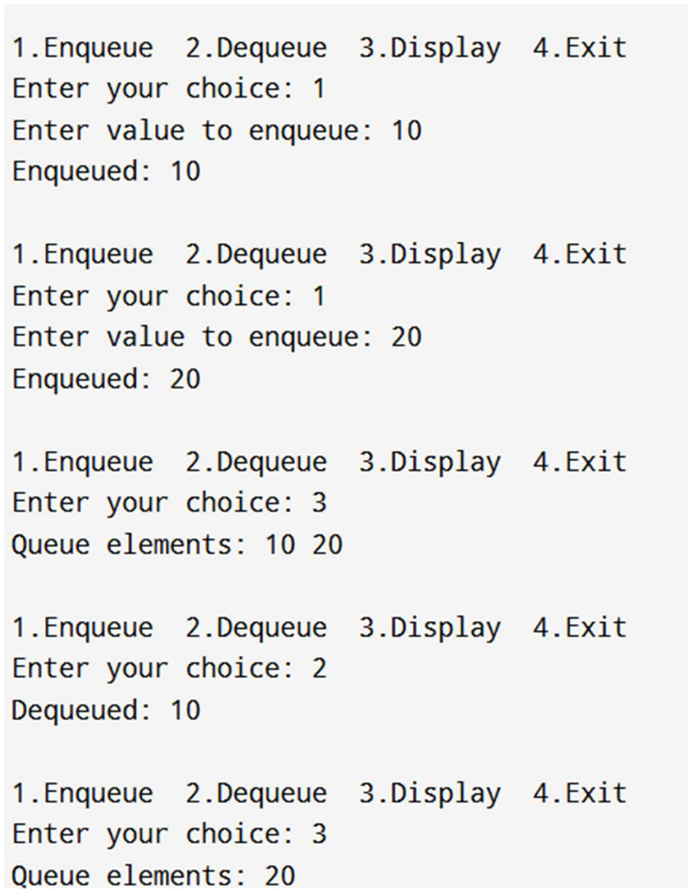
1. Initialize front = -1 and rear = -1.
2. ENQUEUE:
   - Check if the queue is full.
   - If not, insert the element at rear+1, then update rear.
   - If front == -1, set front = 0.
3. DEQUEUE:
   - Check if the queue is empty.
   - If not, remove the element at front, then update front.
   - If front > rear, reset both to -1.
4. DISPLAY:
   - Print all elements from front to rear.

PROGRAM:

```
#include <stdio.h>

#define SIZE 100

int queue[SIZE];

int front = -1, rear = -1;

void enqueue(int value) {

  if (rear == SIZE - 1) {

    printf("Queue is full!\n");

  } else {
        if (front == -1)
            front = 0;
        rear++;
        queue[rear] = value;
        printf("Enqueued: %d\n", value);
    }
}
void dequeue() {
    if (front == -1 || front > rear) {
        printf("Queue is empty!\n");
    } else {
        printf("Dequeued: %d\n", queue[front]);
        front++;
    }
}
void display() {
    if (front == -1 || front > rear) {
        printf("Queue is empty!\n");
    } else {
        printf("Queue elements: ");
        for (int i = front; 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 your choice: ");
        scanf("%d", &choice);
        switch (choice) {
        case 1:
            printf("Enter value to enqueue: ");
            scanf("%d", &value);
            enqueue(value);
            break;
        case 2:
            dequeue();
            break;
        case 3:
            display();
            break;
        case 4:
            return 0;
        default:
            printf("Invalid choice!\n");
        }
    }
}
```

Input & output:



```
1.Enqueue  2.Dequeue  3.Display  4.Exit
Enter your choice: 1
Enter value to enqueue: 10
Enqueued: 10

1.Enqueue  2.Dequeue  3.Display  4.Exit
Enter your choice: 1
Enter value to enqueue: 20
Enqueued: 20

1.Enqueue  2.Dequeue  3.Display  4.Exit
Enter your choice: 3
Queue elements: 10 20

1.Enqueue  2.Dequeue  3.Display  4.Exit
Enter your choice: 2
Dequeued: 10

1.Enqueue  2.Dequeue  3.Display  4.Exit
Enter your choice: 3
Queue elements: 20
```

RESULT :

The C program successfully implements Queue operations using an array.