Aim:

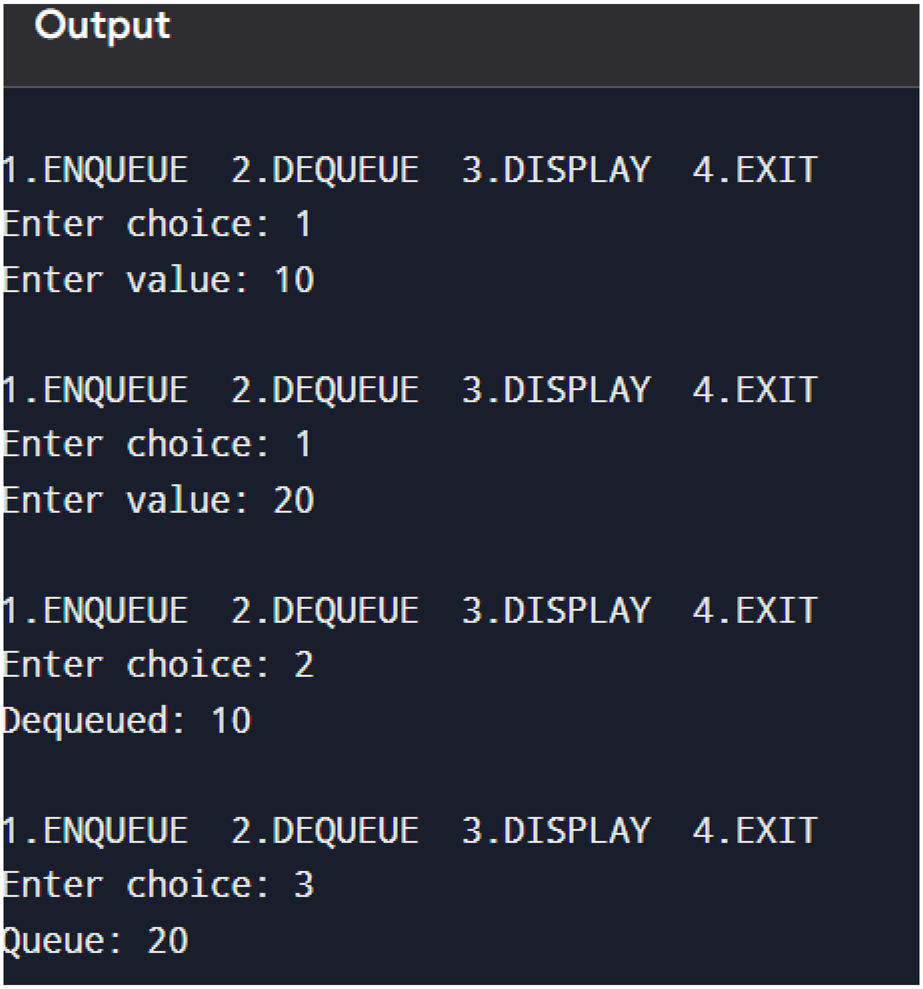
To implement a queue using an array with operations ENQUEUE, DEQUEUE, and DISPLAY.

Algorithm:

1. Start
2. Initialize front = -1 and rear = -1.
3. ENQUEUE(x):
    - If rear == MAX-1 → Overflow.
    - Else if front == -1 set front = 0.
    - Insert element at queue[++rear].
4. DEQUEUE():
    - If front == -1 or front > rear → Underflow.
    - Else remove element at queue[front++].
5. DISPLAY():
    - Print elements from front to rear.
6. Stop

CODE:

```
#include <stdio.h>
#define MAX 10

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

void enqueue(int val) {
    if (rear == MAX - 1)
        printf("Queue Overflow!\n");
    else {
        if (front == -1) front = 0;
        queue[++rear] = val;
    }
}

void dequeue() {
    if (front == -1 || front > rear)
        printf("Queue Underflow!\n");
    else
        printf("Dequeued: %d\n", queue[front++]);
}

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

int main() {
    int choice, val;
    while (1) {
        printf("\n1.ENQUEUE  2.DEQUEUE  3.DISPLAY  4.EXIT\n");
        printf("Enter choice: ");
        scanf("%d", &choice);

        if (choice == 1) {
            printf("Enter value: ");
            scanf("%d", &val);
            enqueue(val);
        }
        else if (choice == 2)
            dequeue();
        else if (choice == 3)
            display();
        else if (choice == 4)
            break;
        else
            printf("Invalid choice!\n");
    }
    return 0;
}
```



```
1.ENQUEUE  2.DEQUEUE  3.DISPLAY  4.EXIT
Enter choice: 1
Enter value: 10

1.ENQUEUE  2.DEQUEUE  3.DISPLAY  4.EXIT
Enter choice: 1
Enter value: 20

1.ENQUEUE  2.DEQUEUE  3.DISPLAY  4.EXIT
Enter choice: 2
Dequeued: 10

1.ENQUEUE  2.DEQUEUE  3.DISPLAY  4.EXIT
Enter choice: 3
Queue: 20
```

RESULT:

The program successfully executed and displayed the queue operations.