13. Write a C program to implement Queue operations such as ENQUEUE, DEQUEUE

and Display

**Aim:**To write a C program to implement a queue data structure using an array that supports the following operations:

1. ENQUEUE — insert an element at the rear of the queue
2. DEQUEUE — remove an element from the front of the queue
3. DISPLAY — show all elements present in the queue

**Algorithm**

1. Define constants:

MAX\_SIZE ← some integer (maximum number of elements)

2. Declare variables:

queue[0..MAX\_SIZE-1] // array to hold queue elements

front ← -1 // index of front element

rear ← -1 // index of rear element

3. Function isEmpty():

if front = -1 or front > rear then

return TRUE

else

return FALSE

4. Function isFull():

if rear = MAX\_SIZE-1 then

return TRUE

else

return FALSE

5. Procedure ENQUEUE(value):

if isFull() then

print “Queue Overflow”

else

if front = -1 then // first insertion

front ← 0

rear ← rear + 1

queue[rear] ← value

print value “enqueued”

6. Function DEQUEUE():

if isEmpty() then

print “Queue Underflow”

return some error value

else

value ← queue[front]

front ← front + 1

if front > rear then

// queue becomes empty after removal

front ← rear ← -1

return value

7. Procedure DISPLAY():

if isEmpty() then

print “Queue is empty”

else

for i from front to rear do

print queue[i]

8. Main program:

Initialize front ← rear ← -1

loop forever:

Print menu: 1. ENQUEUE 2. DEQUEUE 3. DISPLAY 4. EXIT

Read choice from user

If choice = 1 then

prompt for value; call ENQUEUE(value)

Else if choice = 2 then

call DEQUEUE(); show removed value

Else if choice = 3 then

call DISPLAY()

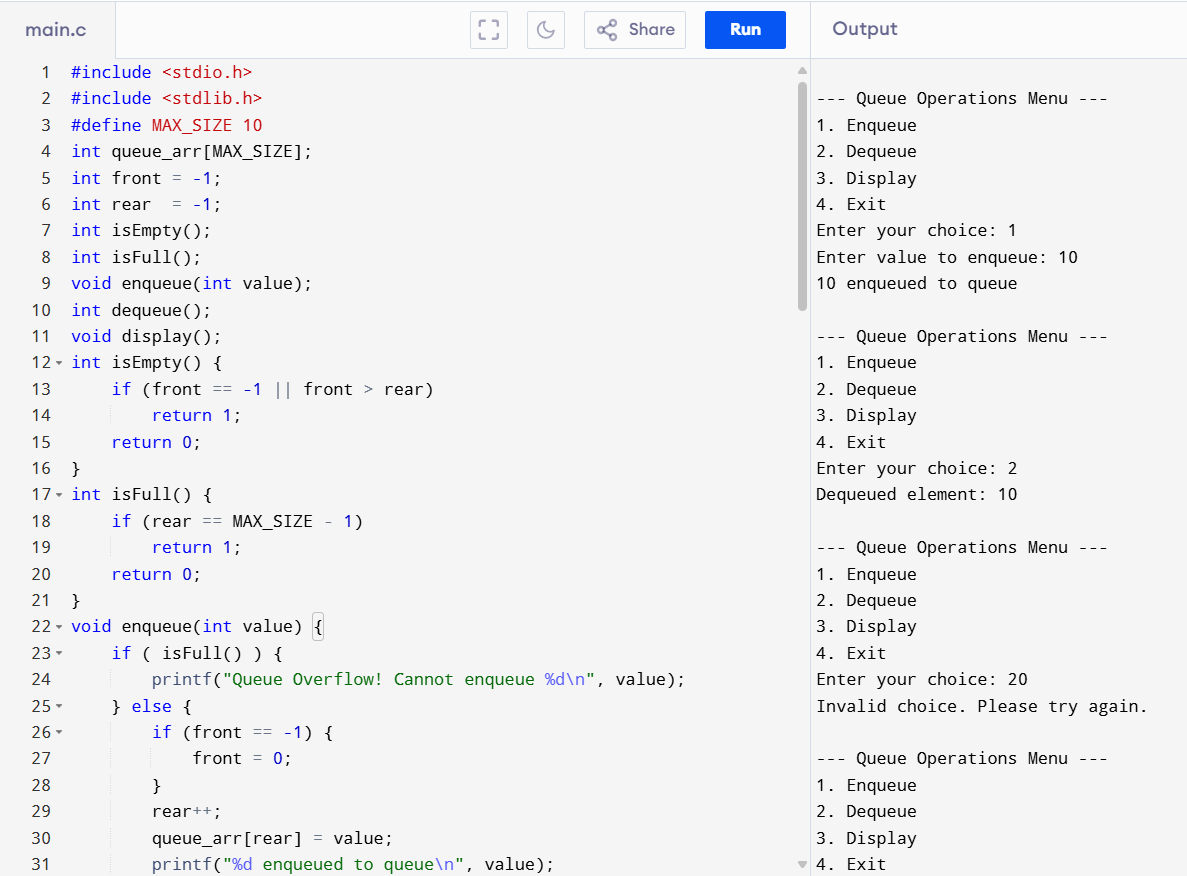
Else if choice = 4 then

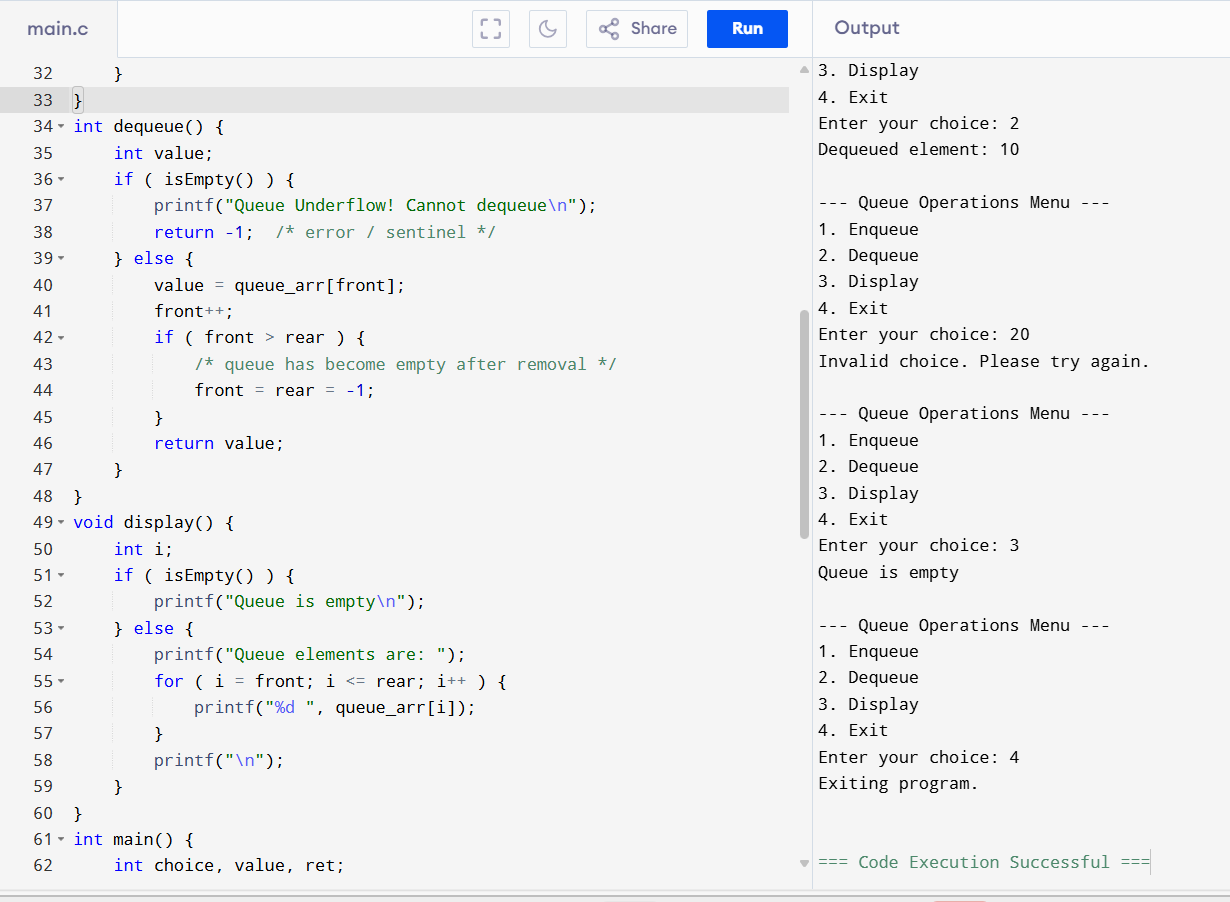
exit

Else

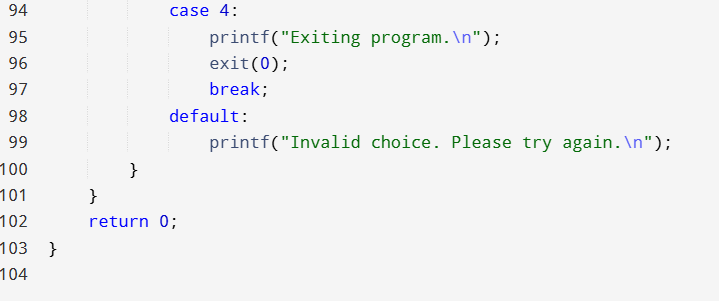
print “Invalid choice”

**Program**

****

****

****

****

**Result:** The program has executed successfully and output is verified.