1. **Macro Definition:**
   * Defines a macro named **SIZE** with a value of 3. It likely represents the size of the queue.
2. **Global Variables:**
   * Declares global variables:
     + **Queue**: An array to store elements of the queue.
     + **front**: Represents the front (start) of the queue.
     + **rear**: Represents the rear (end) of the queue.
3. **isEmpty Function:**
   * Checks if the queue is empty.
   * Returns 1 if empty, indicating true; otherwise, returns 0.
4. **Qfull Function:**
   * Checks if the queue is full.
   * Returns 1 if full, indicating true; otherwise, returns 0.
5. **enque Function:**
   * Enqueues (inserts) an element into the queue.
   * Checks if the queue is not full.
   * If the queue is initially empty, sets the front to 0.
   * Increments the rear and inserts the data at the rear.
   * Prints a message indicating the successful insertion.
6. **deque Function:**
   * Dequeues (removes) an element from the front of the queue.
   * Checks if the queue is not empty.
   * Retrieves the element at the front, increments the front, and returns the element.
   * Prints the updated front index.
7. **main Function:**
   * The **main** function demonstrates the usage of the queue functions.
   * Enqueues values (10, 20, 30, 40).
   * Dequeues elements and prints them.
   * Attempts to enqueue 40 (prints a message indicating the queue is full).
   * Dequeues elements until the queue is empty.

**Note:**

* The queue uses a circular array to manage the front and rear indices.
* The program prints messages to indicate whether the queue is full or empty during operations.

Algorithm:

1. \*Define Constants and Global Variables:\*

- Define the size of the circular queue (SIZE) using a macro.

- Declare an array Queue to store the elements.

- Initialize front and rear pointers to -1.

2. \*\*Define isEmpty Function:\*\*

- Check if the circular queue is empty by comparing front and rear.

- If front is one position ahead of rear, the queue is empty.

3. \*\*Define Qfull Function:\*\*

- Check if the circular queue is full using the circular condition.

- If (rear + 1) % SIZE is equal to front, the queue is full.

4. \*\*Define enque Function:\*\*

- Check if the queue is not full using Qfull function.

- If front is -1, set it to 0.

- Increment rear and insert the element at the rear index.

- Print the inserted element.

5. \*\*Define deque Function:\*\*

- Check if the queue is not empty using isEmpty function.

- Retrieve the element at the front index and increment front.

- Print the updated front index and return the retrieved element.

6. \*\*Define main Function:\*\*

- Demonstrate the usage of the circular queue by calling enque and deque functions.