Queue Implementation in C

Problem Statement: Implement a queue using an array in C and perform basic queue operations such as enqueue, dequeue, and display the queue.

Objective: - Learn the concept of queue data structure. - Implement queue operations using arrays. - Understand queue behavior (FIFO: First In First Out).

Algorithm:

1. **Enqueue Operation:**
    - Check if the queue is full (rear == SIZE-1).
    - If not full, increment `rear` and insert the element at `items[rear]`.
    - If `front == -1`, set `front = 0`.
    - If full, display “Queue is Full”.
2. **Dequeue Operation:**
    - Check if the queue is empty (front == -1).
    - If not empty, retrieve the element at `items[front]`.
    - Increment `front` to remove the element.
    - If `front > rear`, reset `front = rear = -1`.
    - If empty, display “Queue is Empty”.
3. **Display Operation:**
    - If the queue is empty, print “Queue is Empty”.
    - Else, print all elements from `front` to `rear`.

Program Code:

```
#include <stdio.h>
#define SIZE 5

void enQueue(int);
void deQueue();
void display();

int items[SIZE], front = -1, rear = -1;

int main() {
  deQueue(); // Cannot dequeue from empty queue

  enQueue(1);
  enQueue(2);
  enQueue(3);
  enQueue(4);
  enQueue(5);

  enQueue(6); // Queue is full
```

```c
  display();

  deQueue(); // Remove first element

  display();

  return 0;
}

void enQueue(int value) {
  if (rear == SIZE - 1)
    printf("\nQueue is Full!!");
  else {
    if (front == -1)
      front = 0;
    rear++;
    items[rear] = value;
    printf("\nInserted -> %d", value);
  }
}

void deQueue() {
  if (front == -1)
    printf("\nQueue is Empty!!");
  else {
    printf("\nDeleted : %d", items[front]);
    front++;
    if (front > rear)
      front = rear = -1;
  }
}

void display() {
  if (rear == -1)
    printf("\nQueue is Empty!!!");
  else {
    int i;
    printf("\nQueue elements are:\n");
    for (i = front; i <= rear; i++)
      printf("%d  ", items[i]);
  }
  printf("\n");
}
```

Sample Output:

```
Queue is Empty!!
Inserted -> 1
Inserted -> 2
Inserted -> 3
Inserted -> 4
Inserted -> 5
Queue is Full!!
Queue elements are:
1  2  3  4  5
Deleted : 1
Queue elements are:
2  3  4  5
```