**Lab Task 11**



**Superior University Gold Campus**

|  |  |
| --- | --- |
| **Submitted to** | ****Mr. Rasikh Ali**** |
| **Submitted by** | **Javaid Ali** |
| **Roll No** | **SU92-BSSEM-S24-029 (Section – 3A)** |
| **Subject** | **Data Structures and Algorithms (Lab)** |
| **Class** | **BS – Software Engineering** |

# **Lab 11: Queue with Array and LinkedList**

**1-Queue With Array; Enqueue, Dequeue, Display**

**Key Features:**

* **FIFO (First-In-First-Out)** structure.
* front tracks the **start** (next element to dequeue).
* rear tracks the **end** (last enqueued element).

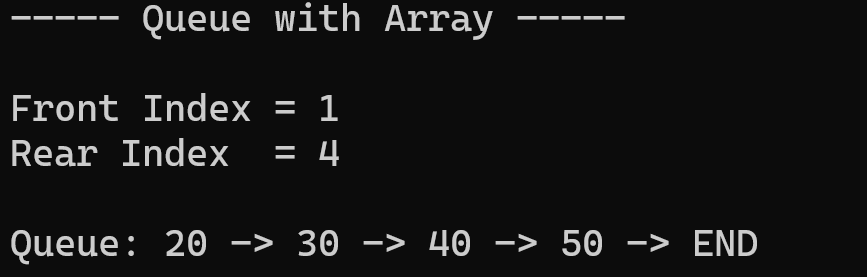
**Functions:**

1. **enqueue(d)** → Inserts d at the **rear** (if space is available).
   * If the queue is full (rear == 9), prints "Queue is Full".
   * If the queue is empty (front == -1), sets front = 0.
2. **dequeue()** → Removes the **front** element.
   * If the queue is empty (front == -1), prints "Queue is Empty!".
   * Otherwise, sets arr[front] = 0 (optional) and increments front.
3. **display()** → Prints the queue from **front to rear**.

**Why & How?**

* **Standard queue operations** (enqueue at rear, dequeue at front).
* **No circular buffer** → Once rear reaches 9, no more inserts (even if space is freed by dequeue).
* **Memory-efficient** (array-based) but **fixed capacity**.

**Outputs:**



**2- Queue With LinkedList; Enqueue, Dequeue, Display**

**Key Differences from Array-Based Queue:**

* Uses **nodes (**Queue**objects)** connected via pointers (next).
* No fixed size limit (unlike the array version).
* front and rear are **global pointers** tracking the start and end of the queue.

**Functions:**

1. **enqueue(d)**→ Adds d at the **rear** (FIFO).
2. **dequeue()** → Removes the **front** element.
3. **display()** → Prints the queue from front to rear.

**Why & How?**

* **Linked list** allows **dynamic resizing** (no overflow unless memory runs out).
* enqueue() adds a new node at the **rear**.
* dequeue() removes the **front** node and updates front to the next node.
* display() traverses the list from front to rear.

**Outputs:**

