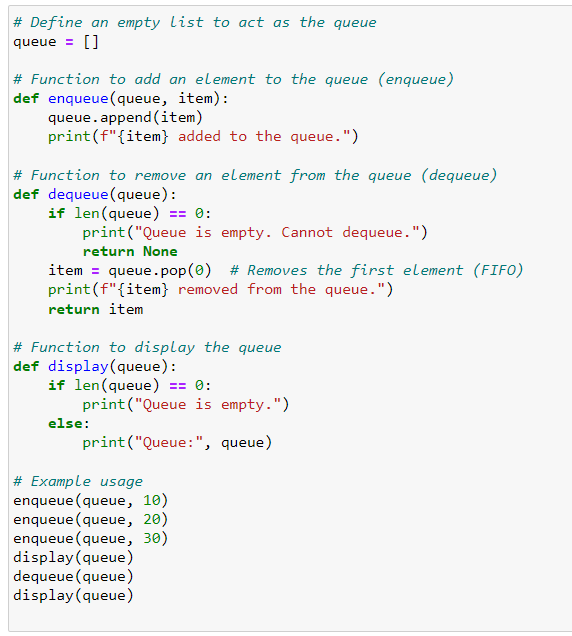
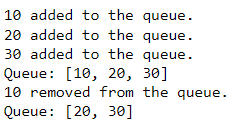
3. QUEUE

A **queue** is a linear data structure that follows the **First-In-First-Out (FIFO)** principle. The element added first will be removed first.



OUTPUT:



A queue supports two primary operations:

1. **Enqueue**: Add an element to the rear (end) of the queue.

**2. Dequeue**: Remove an element from the front (beginning) of the queue

**Queue Operations and Algorithm:**

1. **Initialize Queue**:
   * Create an empty queue Q.
2. **Enqueue Operation (Add an element to the queue)**:
   * **Input**: A queue Q [] and an element x to be added.
   * **Algorithm**:
     1. If the queue is not full (for fixed-size queues):
        + Add element x to the rear of the queue.
     2. Update the rear pointer.

**Dequeue Operation (Remove an element from the queue)**:

* + **Input**: A queue Q [].
  + **Algorithm**:
    1. If the queue is not empty:
       - Remove the element at the front of the queue.
       - Shift the front pointer to the next position.
    2. Return the removed element.
    3. If the queue is empty, return an underflow message.

1. **IsEmpty Operation (Check if the queue is empty)**:
   * **Input**: A queue Q [].
   * **Algorithm**:
     1. If the front pointer exceeds the rear pointer, return True (the queue is empty). Otherwise, return False.