Circular Queue & Deque

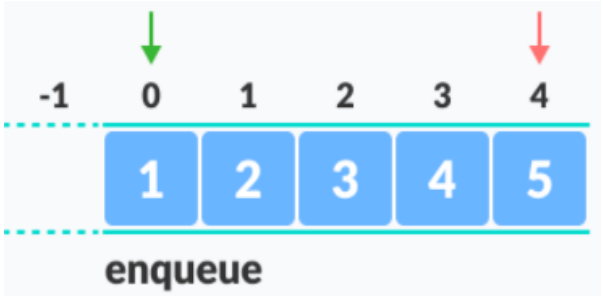
Anoop S Babu

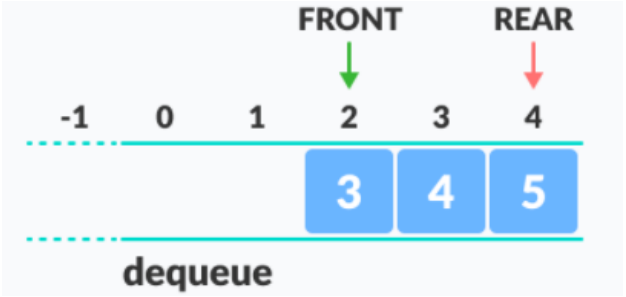
Faculty Associate

Dept. of Computer Science & Engineering

bsanoop@am.amrita.edu

Limitation of Queue





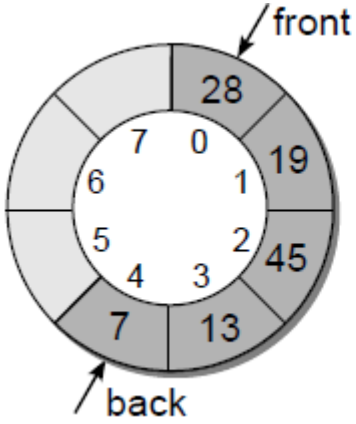
- **Wastage of space** in a regular queue implementation using arrays.
- Requires linear time for the enqueue and dequeue operations.

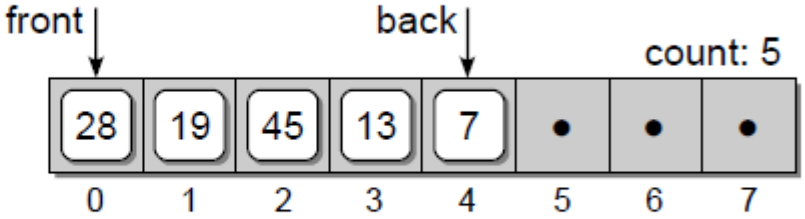
Types of Queues

- There are four different types of queues:
  - Simple Queue
  - Circular Queue
  - Double Ended Queue (Deque)
  - Priority Queue

Circular Queue

- A circular queue is a linear queue as it is based on the FIFO (First In First Out) principle except that **the last position is connected to the first position** in a circular queue that forms a circle.



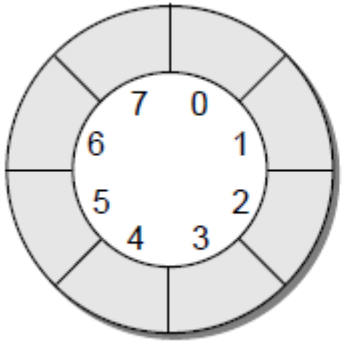


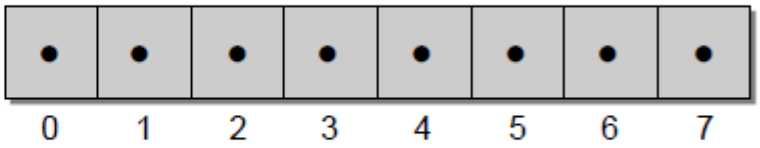
Circular Queue: Basic Operations

- **enqueue(*item*):** Adds the given **item to the back** of the queue.
- **dequeue():** Removes and returns the front item from the queue if the queue is not empty.
- **isEmpty():** Check if the queue is empty or not. Return a boolean value.
- **length():** Returns the number of items currently in the queue.
- **peek():** Get the value of the front of the queue without removing it.

Working of Circular Queue

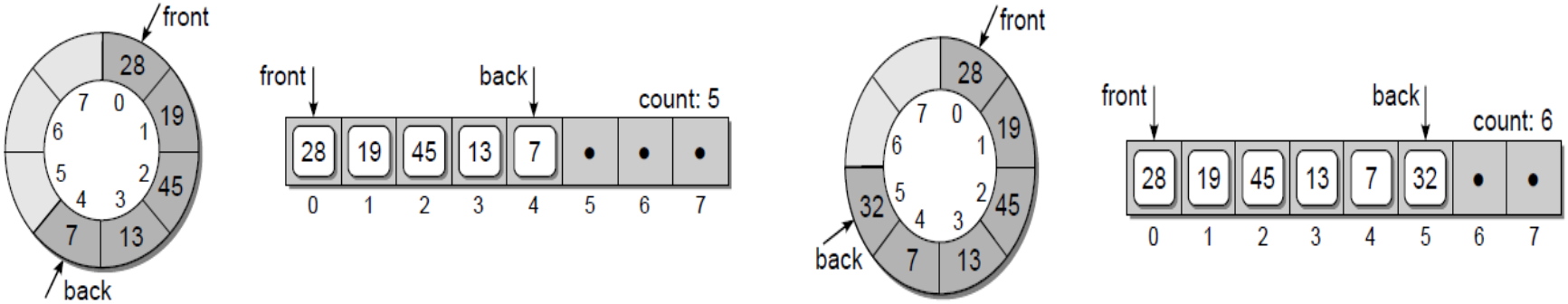
- The circular queue work as follows:
  - two pointers FRONT and REAR
  - FRONT track the first element of the queue
  - REAR track the last elements of the queue
  - initially, set value of FRONT and REAR to -1





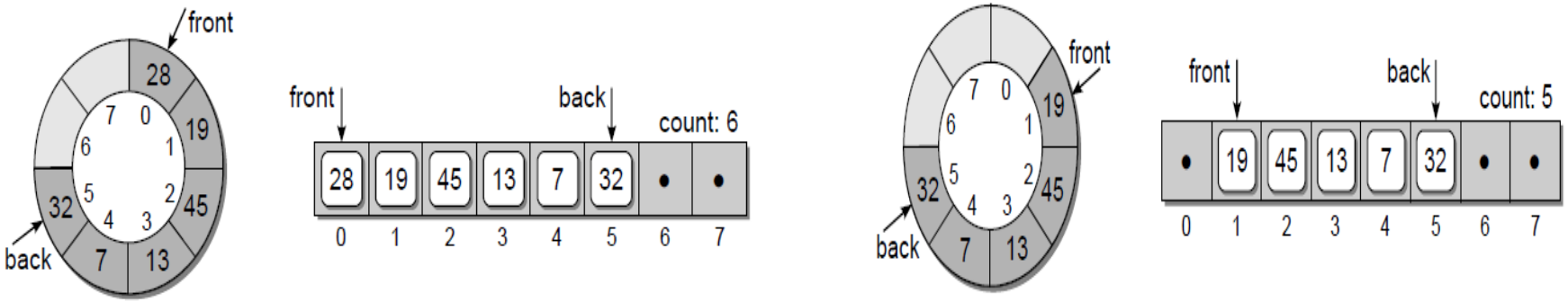
Working of Circular Queue

- **Enqueue Operation**
  - check if the queue is full
  - for the first element, set value of FRONT to 0
  - circularly increase the REAR index by 1 (i.e. if the rear reaches the end, next it would be at the start of the queue)
  - add the new element in the position pointed to by REAR

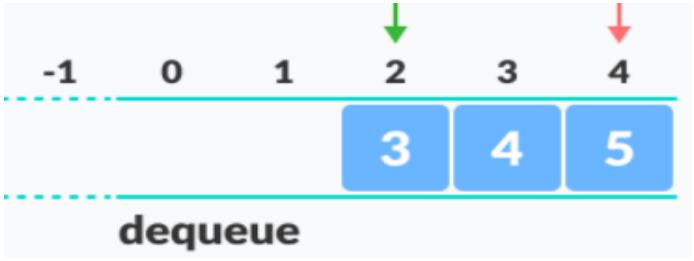


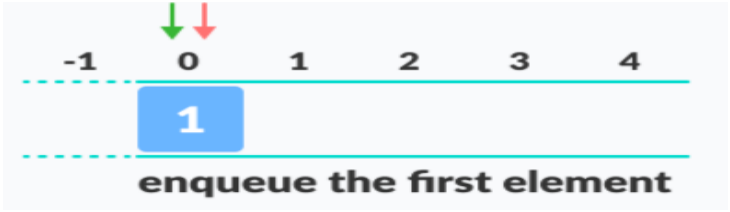
Working of Circular Queue

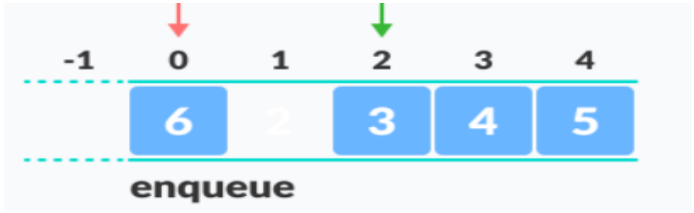
- **Dequeue Operation**
  - check if the queue is empty
  - return the value pointed by FRONT
  - circularly increase the FRONT index by 1
  - for the last element, reset the values of FRONT and REAR to -1

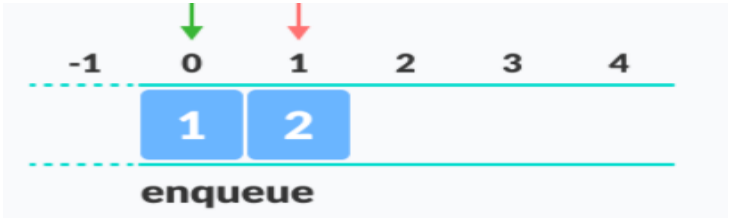


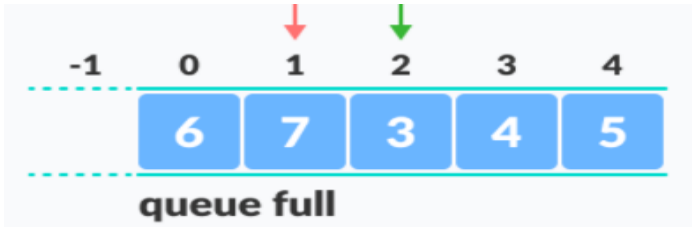
Working of Circular Queue

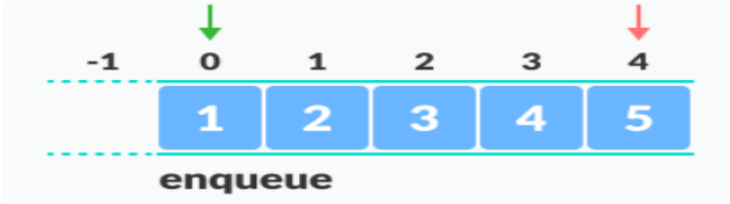










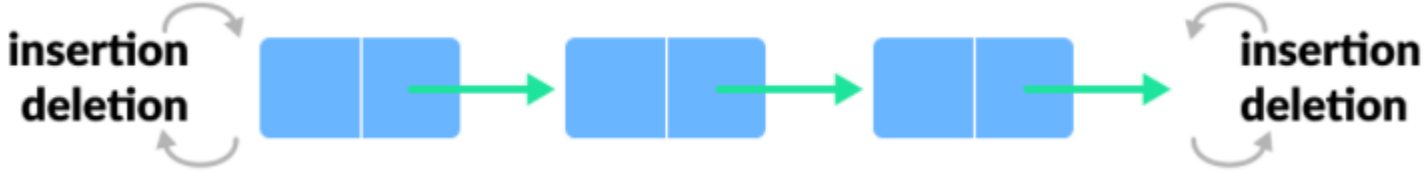


Check for full queue has a new additional case:

- Case 1: FRONT = 0 && REAR == SIZE - 1
- Case 2: FRONT = REAR + 1

Double Ended Queue (Deque)

- In a double ended queue, **insertion and removal** of elements can be performed **from either from the front or rear.**
- It does not follow the FIFO (First In First Out) rule.



Deque: Basic Operations

- **Insert at the Front**
- **Insert at the Rear**
- **Delete from the Front**
- **Delete from the Rear**
- **isEmpty():** Check if the queue is empty or not. Return a boolean value.
- **length():** Returns the number of items currently in the queue.
- **peek():** Get the front and the rear element of the deque.

Types of Deque

- **Input Restricted Deque**
  - In this deque, input is restricted at a single end but allows deletion at both the ends.
- **Output Restricted Deque**
  - In this deque, output is restricted at a single end but allows insertion at both the ends.