Data Structures

Prof. Anand Jain, anand.jain@pccoepune.org

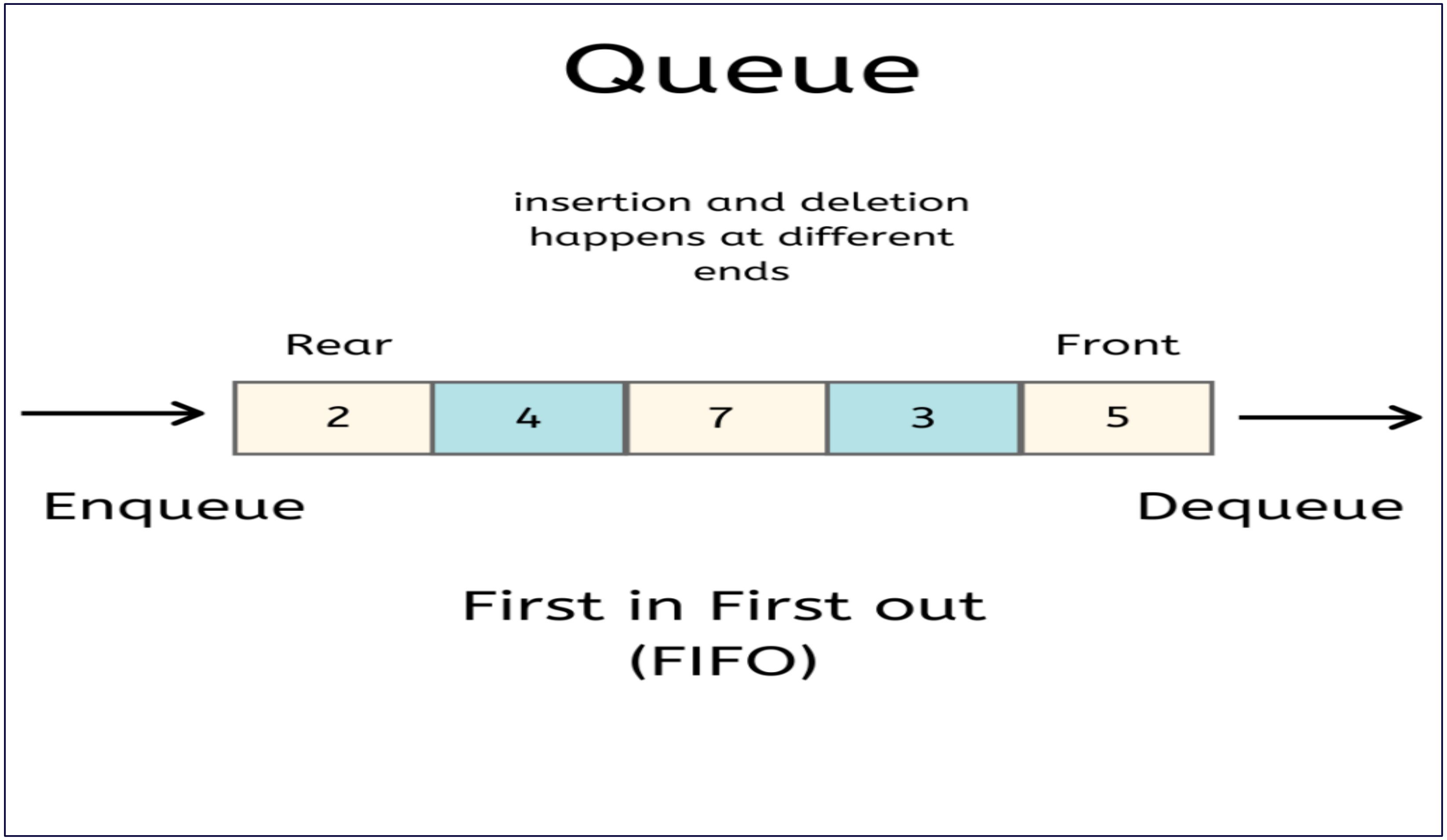
Unit 03 : Queue

Queue

- **Queue Introduction**
  - A linear list data structure which permits insertion of an element to occur at one end and deletion of an element to occur at other end is called as Queue.
  - The information in queue is processed in the same order as it was received, that is, on a first-in first-out (FIFO) or first-come first-serve (FCFS) basis.
  - Location in the queue where first element is present is called as Front (or Front End) of the queue and the location in the queue where last element is present is called as Rear (or Rear End) of the queue.
  - The difference between stacks and queues is in removing. In a stack we remove the item the most recently added; in a queue, we remove the item the least recently added.

Queue

❖ **Array Representation of Queue**


Queue
insertion and deletion happens at different ends
Rear
Front
2
4
7
3
5
Enqueue
Dequeue
First in First out (FIFO)

Queue

- **Linked List Representation of Queue**

Queue

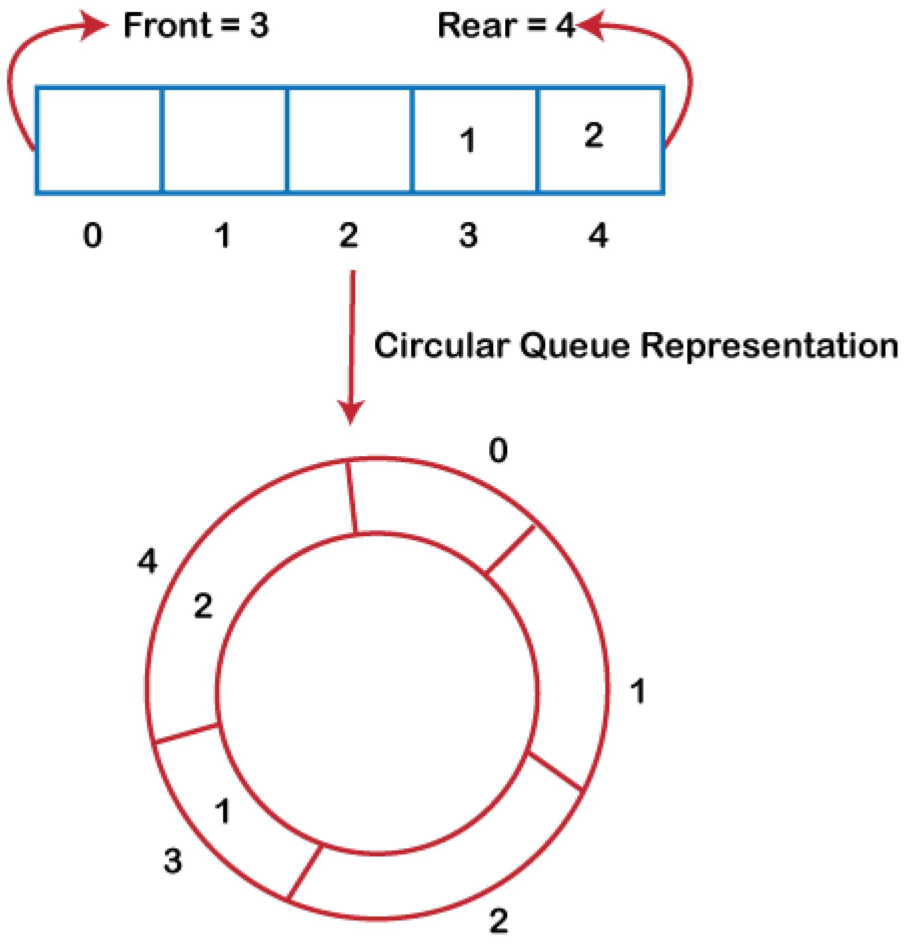
- **Queue Operations**
  - enqueue: To add a new data element at the rear of the queue.
  - dequeue: To remove an element from the front of the queue.
  - isEmpty: To return True if the queue is empty, else return False.
  - size: To check the size of the queue, in other words count the number of elements in the queue and return it.
  - show: To print all the queue elements.

Queue

- Points to Note
  - At any time number of elements in queue are (rear-front+1), except initially empty queue.
  - When front = rear, there is only one element in the queue, except initially empty queue.
  - When front = -1 OR front=rear+1, queue is empty.
  - When rear=MAX-1, queue is full.

- When rear is at the last position and front is at other than zero position, then queue is space available in queue to insert the element but we can not insert it because rear is at the last position. To overcome this problem, we use circular queue.

Circular Queue

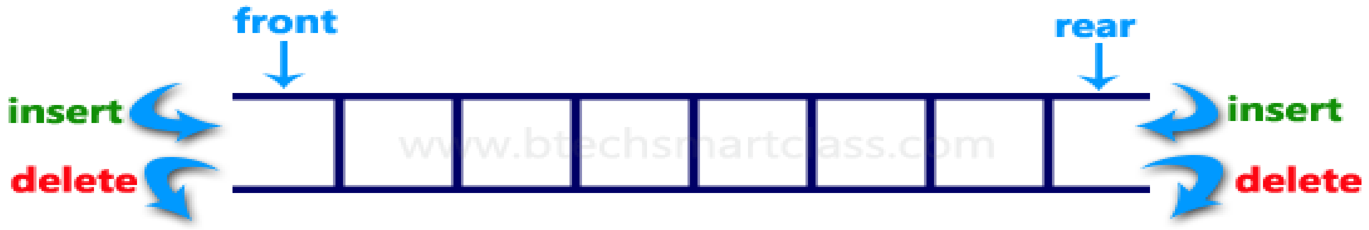


Circular Queue

- Points to Note
  - Queue is full if (rear+1)%max == front
  - Next position to insert in queue is calculated as (rear+1)%max
  - If front and rear are -1, queue is empty
  - If last element is deleted then set f =r = -1
  - Next position to delete is calculated as (front+1)%max
  - While displaying, start from front till you reached to rear. Next position to print is calculated as (position+1)%max

Double Ended Queue

- Introduction
  - Double ended queue (deque) is a more generalized form of queue data structure which allows insertion and removal of elements from both the ends, i.e , front and back.
  - It provides all the capabilities of stacks and queues in a single data structure.
  - Deque is represented using circular array or doubly linked list.

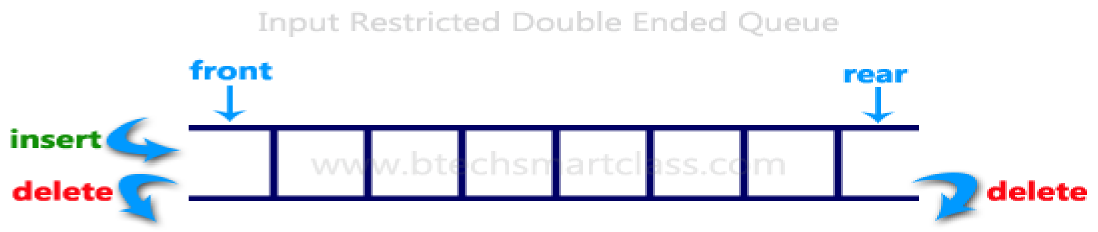


Double Ended Queue

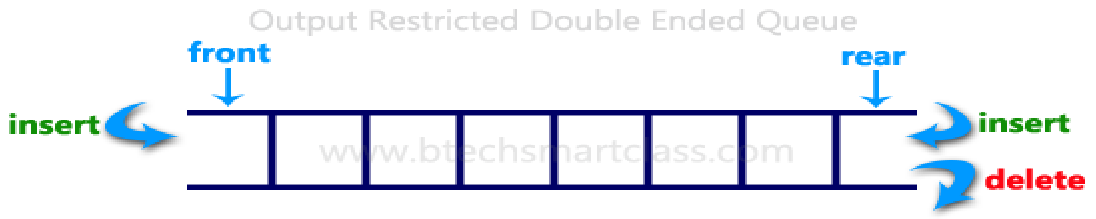
- Operations on Deque
  - add_front(item) adds a new item to the front of the deque. It needs the item and returns nothing.
  - add_rear(item) adds a new item to the rear of the deque. It needs the item and returns nothing.
  - remove_front() removes the front item from the deque. It needs no parameters and returns the item. The deque is modified.
  - remove_rear() removes the rear item from the deque. It needs no parameters and returns the item. The deque is modified.
  - is_empty() tests to see whether the deque is empty. It needs no parameters and returns a boolean value.
  - size() returns the number of items in the deque. It needs no parameters and returns an integer.

Double Ended Queue

- Types of Deque
  - Input Restricted Queue : In input restricted double-ended queue, the insertion operation is performed at only one end and deletion operation is performed at both the ends.

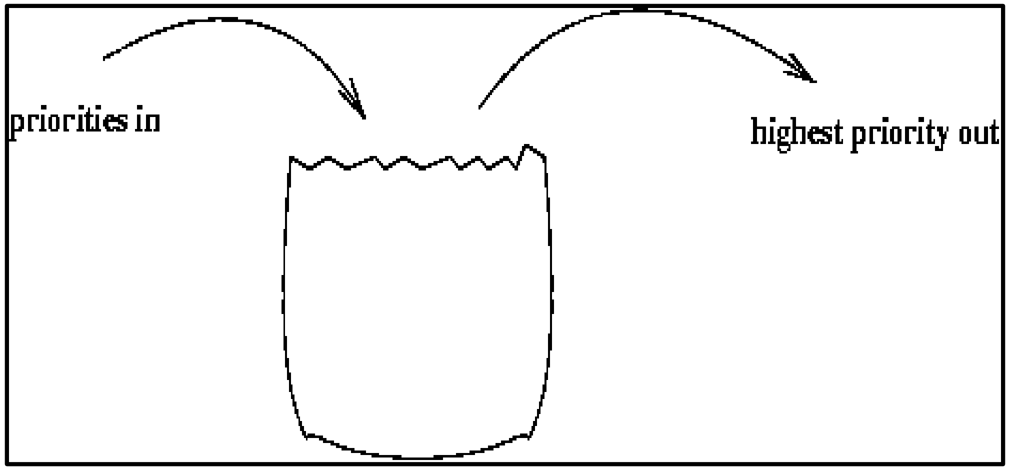


  - Output Restricted Queue : In output restricted double ended queue, the deletion operation is performed at only one end and insertion operation is performed at both the ends.



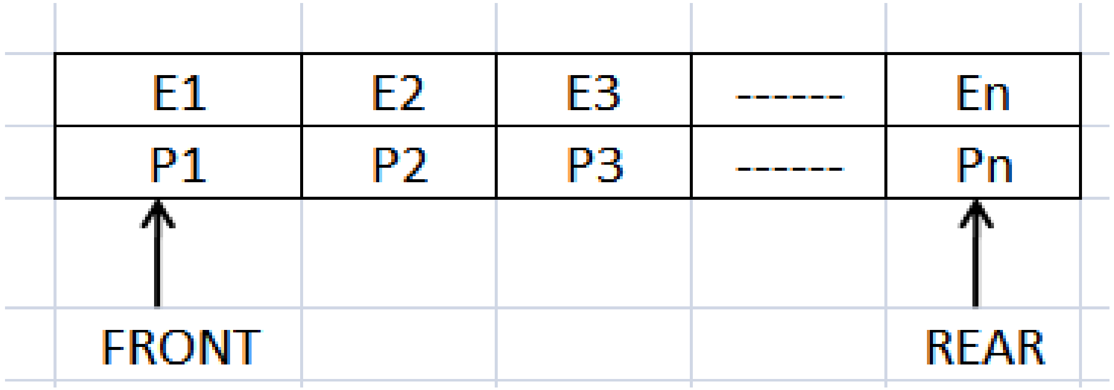
Priority Queue

- Introduction
  - Priority Queue is a Queue where each element is assigned a priority and elements com out in order by priority.
  - A priority queue does not follow FIFO rule.
  - Various ways to implement:
    - Simple/Circular Array
    - Multi-Queue
    - Double Linked List
    - Heap Tree



Priority Queue

❖ Array Implementation of Priority Queue

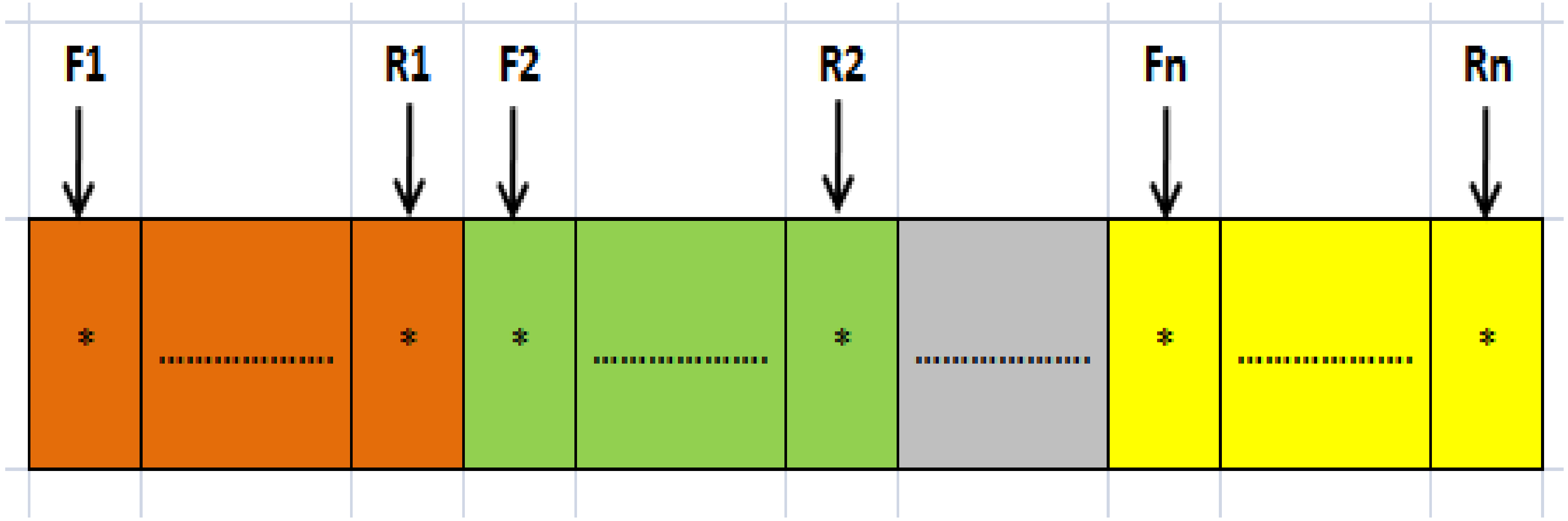


- Method 1: Find the element with highest priority and process it. If it is not front-most element, shift all the trailing elements

- Method 2: Add element, sort the queue so that highest priority element is at the front, and process it.

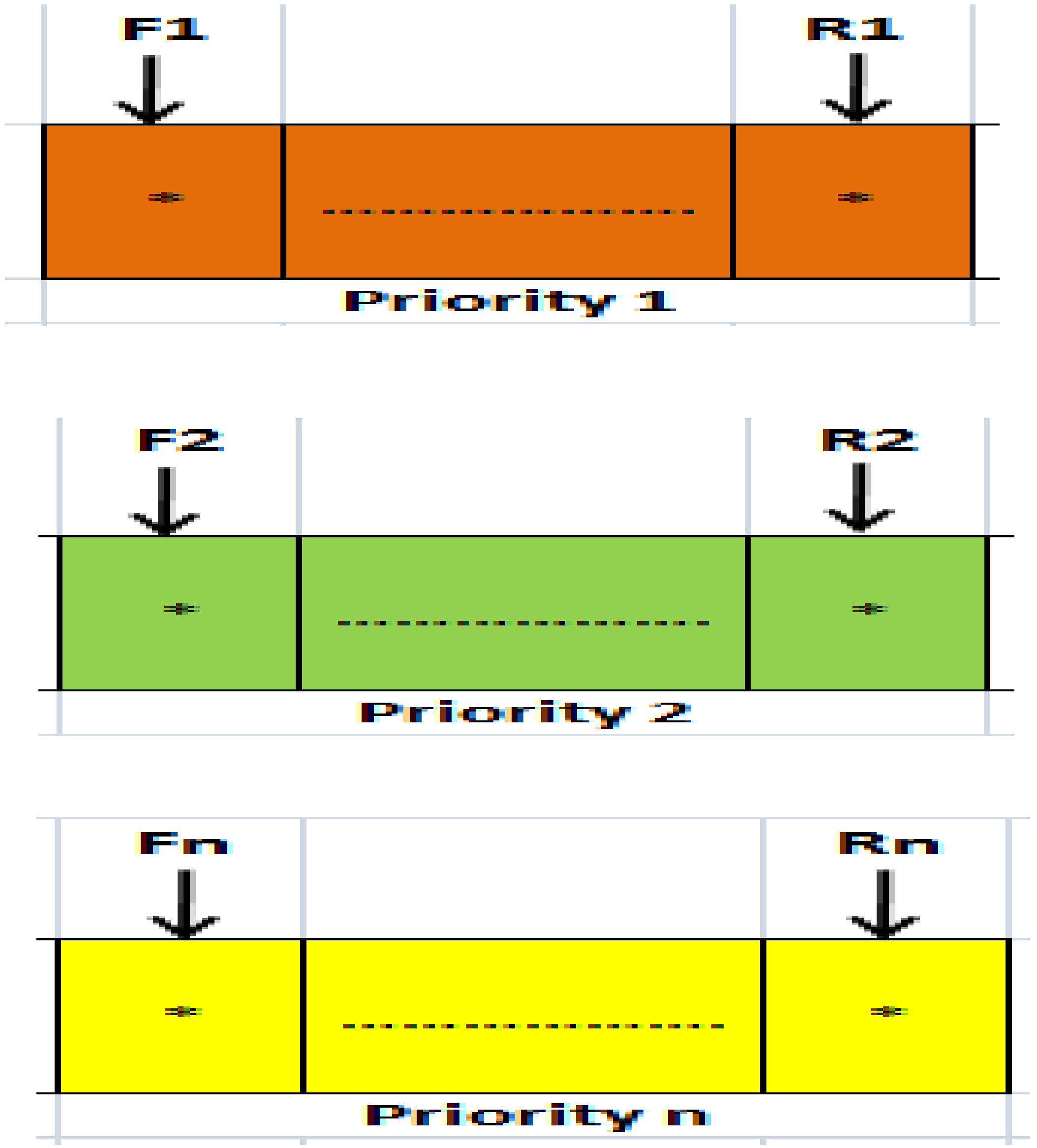
Priority Queue

Multi-Queue Representation of Priority Queue



Priority Queue

- Multi-Queue Representation of Priority Queue → **Multi-Queue with Simple Queues**



Thank You !