Queue

Definition

Queue is a linear list where all additions are made at one end, called rear, and all deletions (accesses) are made from another end called front of the list. So, in a queue there must be two indicators or pointers. One is rear to add elements and another is front used to delete (access) the elements from the queue.

Queue is a FIFO (First In First Out) structure. That means the element that is added first will be deleted (accessed) first. As, stack. Queue can be implemented using array and linked list.

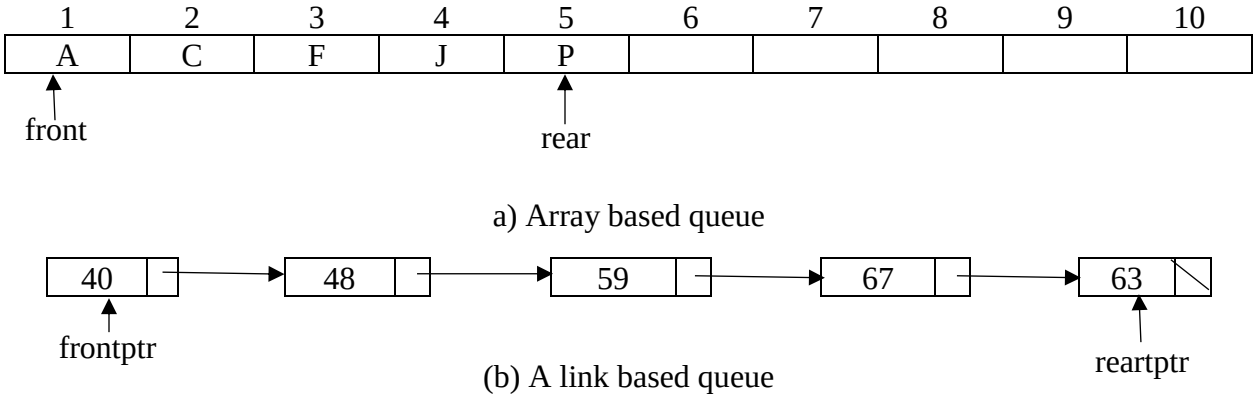


Fig. 1: Graphical representation of queue

Array based queue

The queue that will be created using an array is called array based queue. Here, we have to use two indicators (two index identifiers). One indicator will mark the front element and another will mark the rear element of the queue.

Addition of an element in an array based queue

We know in a queue element is added at the rear. So, to add an item at first we increase rear index and then place the element in the array based queue using the rear index.

Algorithm to add an element to queue

QINSERT(QUEUE, N, FRONT, REAR, ITEM)

This procedure inserts an element ITEM into a queue

1. [Queue already filled?]

if FRONT = 1 and REAR = N, or if FRONT = REAR+1, then:

Write: OVERFLOW, and Return.

2. [Find new value of REAR]

if FRONT = NULL, then: [Queue initially empty]

Set FRONT = 1 and REAR = 1

elese if REAR = N, then

Set REAR = 1

else

Set REAR = REAR+1

[End of if Structure]

3. Set QUEUE[REAR] = ITEM [This inserts new element]

4. Return

Deletion of an element from a queue

We know that, the element is deleted from the front of the queue. So, at first we access the element, and then we increase the front index.

Algorithm to delete an element from a queue

QDELETE(QUEUE, N, FRONT, REAR, ITEM)

This procedure delets an element from a queue and assign it to the variable ITEM

1. [Queue already empty]

if FRONT = NULL, then: UNDERFLOW, and Return

2. Set ITEM = QUEUE[FRONT]

3. [Find new value of FRONT]

if FRONT = REAR, then: [Queue has only one element to start]

Set FRONT = NULL and REAR = NULL

else if FRONT = N, then

Set FRONT = 1

else

Set FRONT = FRONT + 1

[End of if structure]

4. Return

What is a Deque (or double-ended queue)

The deque stands for Double Ended Queue. Deque is a linear data structure where the insertion and deletion operations are performed from both ends. We can say that deque is a generalized version of the queue.

Representation of deque

Types of deque

There are two types of deque -

- Input restricted queue
- Output restricted queue

Input restricted Queue

In input restricted queue, insertion operation can be performed at only one end, while deletion can be performed from both ends.

input restricted double ended queue

Output restricted Queue

In output restricted queue, deletion operation can be performed at only one end, while insertion can be performed from both ends.

Output restricted double ended queue

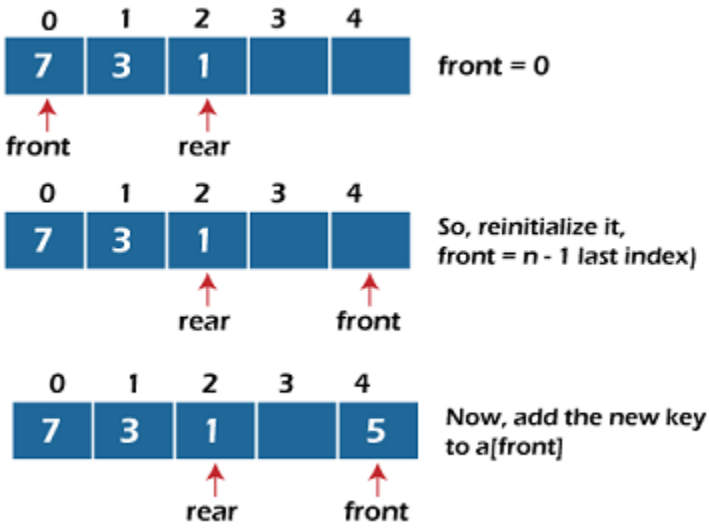
Operations performed on deque

There are the following operations that can be applied on a deque -

- Insertion at front
- Insertion at rear
- Deletion at front
- Deletion at rear

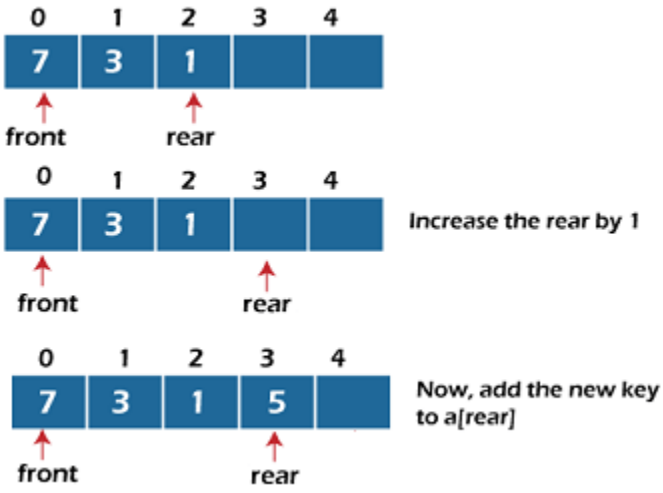
Insertion at the front end

- If the queue is empty, both rear and front are initialized with 0. Now, both will point to the first element.
- Otherwise, check the position of the front if the front is less than 1 (front < 1), then reinitialize it by **front = n - 1**, i.e., the last index of the array.



Insertion at the rear end

- If the queue is empty, both rear and front are initialized with 0. Now, both will point to the first element.
- Otherwise, increment the rear by 1. If the rear is at last index (or size - 1), then instead of increasing it by 1, we have to make it equal to 0.

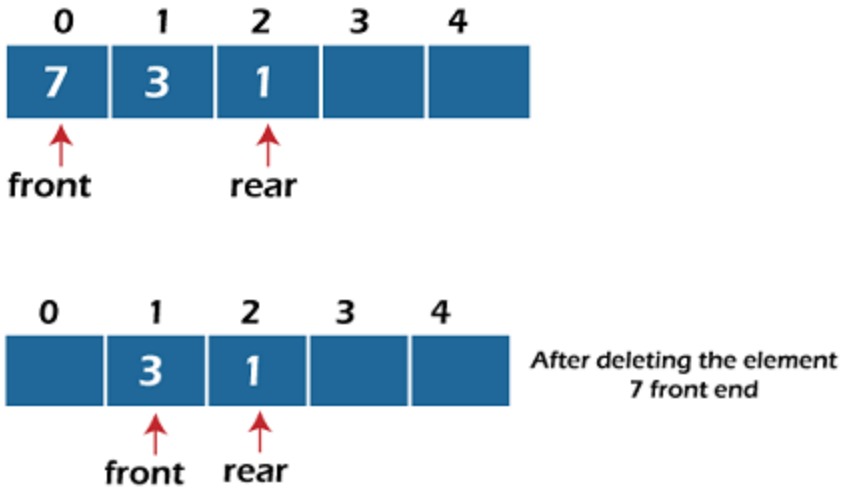


Deletion at the front end

If the deque has only one element, set rear = -1 and front = -1.

Else if front is at end (that means front = size - 1), set front = 0.

Else increment the front by 1, (i.e., front = front + 1).



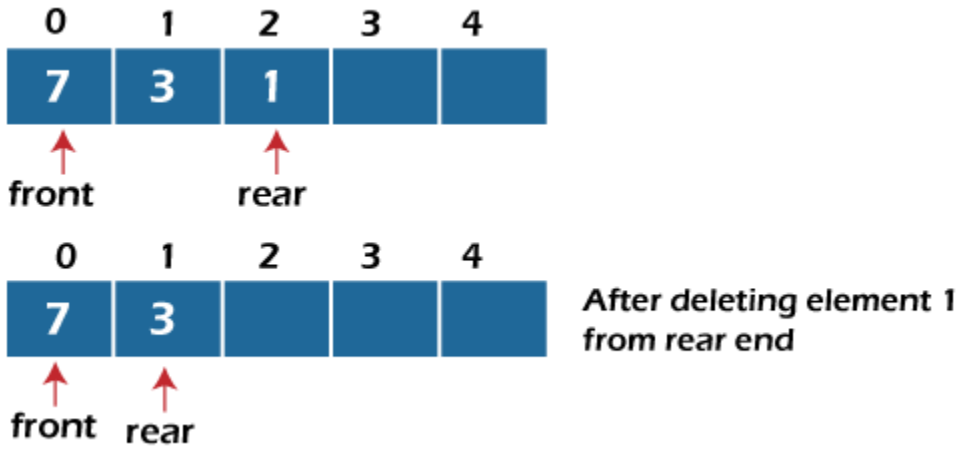
Deletion at the rear end

If the queue is empty, i.e., front = -1, it is the underflow condition, and we cannot perform the deletion.

If the deque has only one element, set rear = -1 and front = -1.

If rear = 0 (rear is at front), then set rear = n - 1.

Else, decrement the rear by 1 (or, rear = rear -1).



Characteristics of a Priority Queue

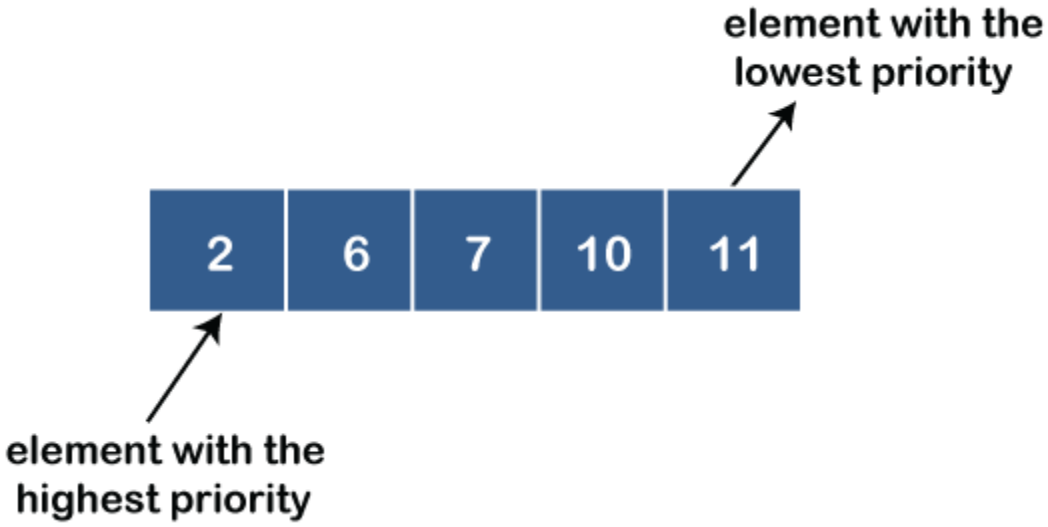
A queue is called a Priority Queue when it has the following characteristics:

- Each item in the queue must have a priority associated with it.
- Higher or lower priority elements must be dequeued before lower or higher priority elements respectively depending on priority order taken by user that is if user consider lower number as higher priority or higher number as higher priority.
- If two elements in the queue have the same priority value then the first in first out rule is followed for these two elements alone i.e. the element that entered the priority queue first will be the first to be removed.

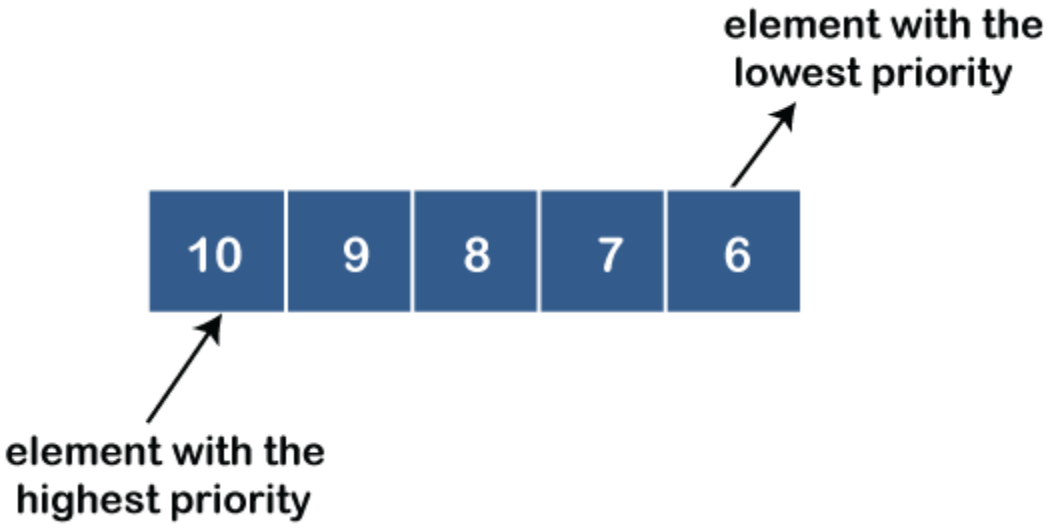
Types of Priority Queue

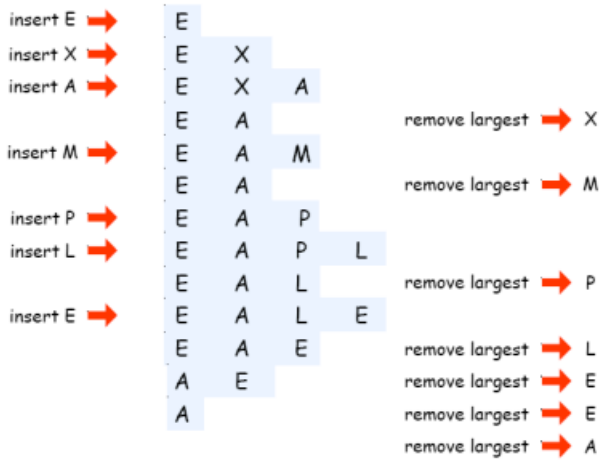
There are two types of priority queue:

- **Ascending order priority queue:** In ascending order priority queue, a lower priority number is given as a higher priority in a priority. For example, we take the numbers from 1 to 5 arranged in an ascending order like 1,2,3,4,5; therefore, the smallest number, i.e., 1 is given as the highest priority in a priority queue.



- **Descending order priority queue:** In descending order priority queue, a higher priority number is given as a higher priority in a priority. For example, we take the numbers from 1 to 5 arranged in descending order like 5, 4, 3, 2, 1; therefore, the largest number, i.e., 5 is given as the highest priority in a priority queue.




insert E
insert X
insert A
insert M
insert P
insert L
insert E
remove largest
remove largest
remove largest
remove largest
remove largest
remove largest