QUEUES

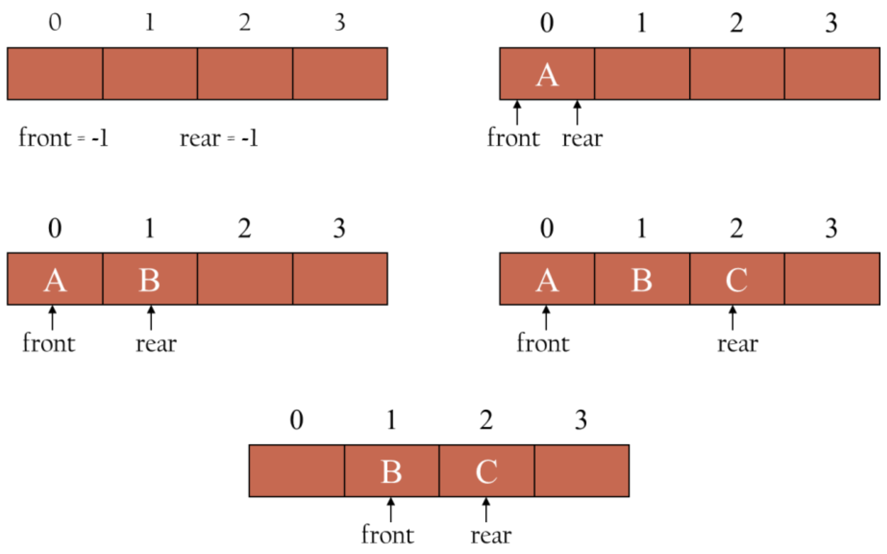
A queue is a

- Linear list in which data can only be inserted at one end, called the REAR
- And deleted from the other end called the FRONT
- These restrictions ensure that the data are processed through the queue in the order in which they are received
- The queue is a First-In-First-Out (FIFO) Structure

TYPES OF QUEUES

- Ordinary Queues
- Circular Queues
- Priority Queues
- Double Ended Queues

ORDINARY QUEUES

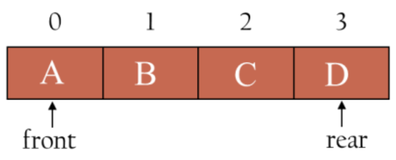


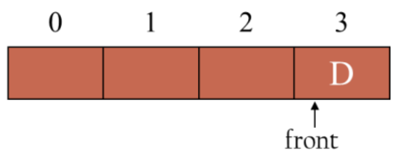
CIRCULAR QUEUES

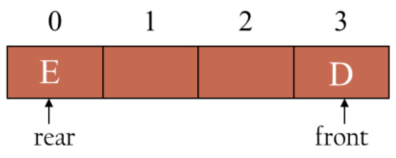
One of the major drawbacks of an ordinary queue is the unnecessary wastage of memory

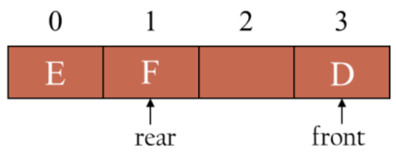
This problem can be solved by implementing the queue as a CIRCULAR queue

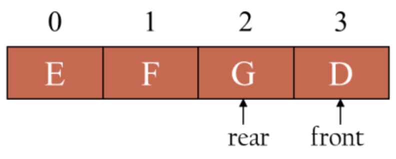
Meaning that whenever the rear pointer reaches MAX-1, it is set to 0 so that new elements can be added



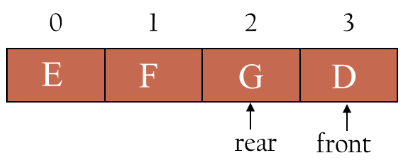


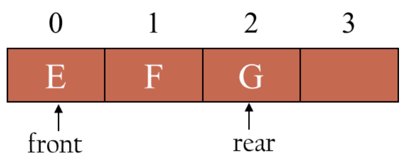


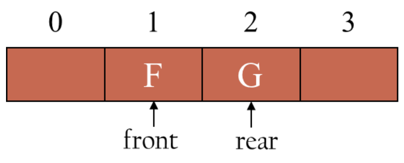


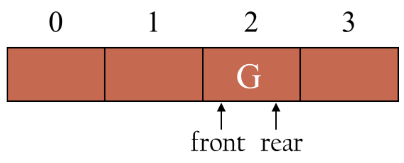


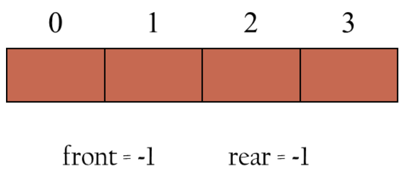
Queue
Overflow











Queue
Underflow

To delete an element in the queue

```
if(front == MAX-1)
{
     front = 0;
}
else
{
    front++;
}
```

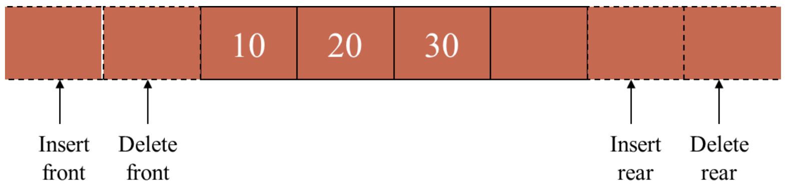
To insert an element in the queue

```
if( rear == MAX-1)
{
    rear = 0;
}
else
{
    rear++;
}
```

DOUBLE ENDED QUEUES

A double ended queue, also called as a deque, is a linear list in which insertions and deletions are done at both ends

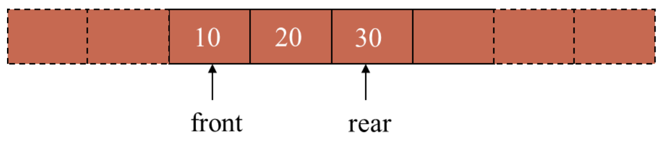
It is a special kind of queue where the queue policy may not appear to be strictly followed



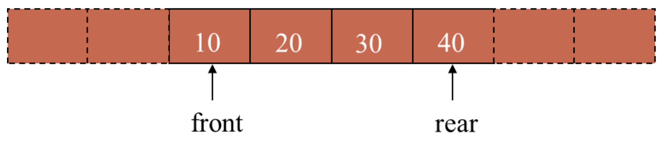
Generally elements are added at the rear end and removed front the front of the queue

But in a double ended queue we allow

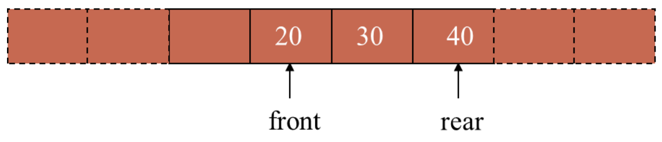
- Insertions at the front end also
- In addition to front deletion we allow rear deletion as well



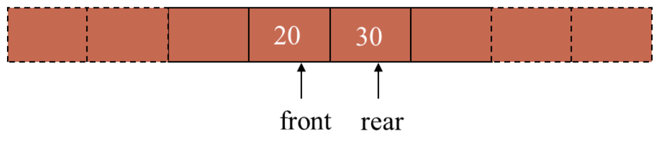
Initial configuration



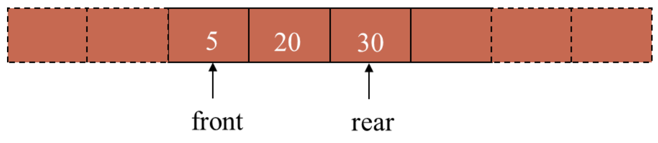
Element 40 is inserted at rear



Front Deletion



Rear deletion



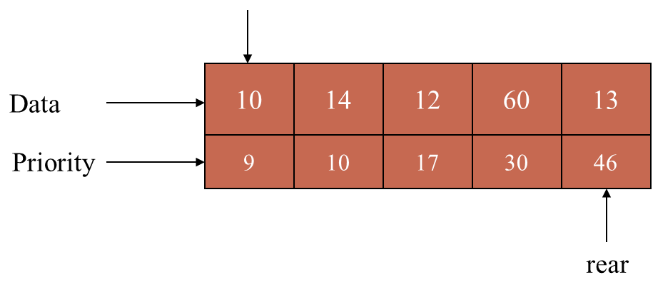
Element 5 in inserted at the front

PRIORITY QUEUES

In a priority queue the order of deletion depends on the priority of the element

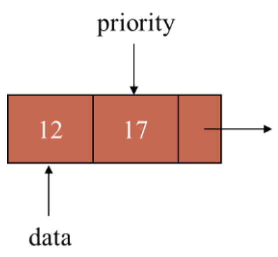
Elements are deleted either in increasing or decreasing order of priority rather than the order in which they arrive in the queue

This also implies that insertion in the queue will happen in order of their priority

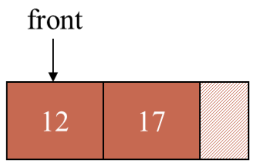


Priority Queues are implemented using linked lists

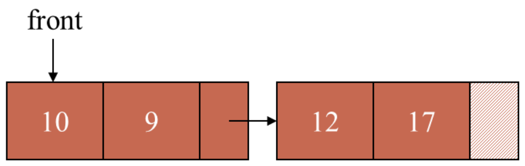
A node in the priority queue will contain data, priority and the next pointer



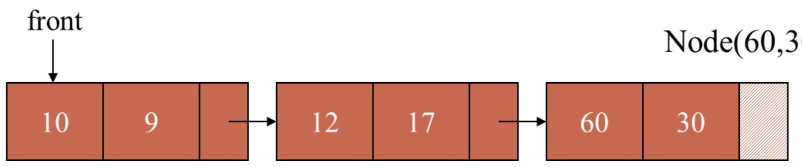
WORKING



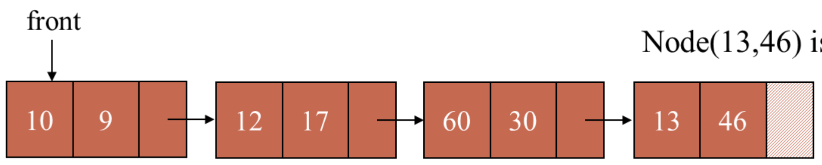
Initial configuration



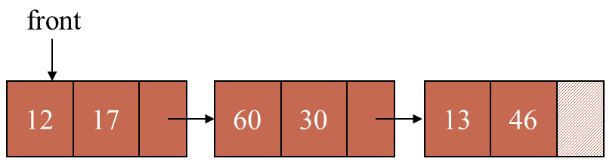
Node(10,9) is inserted



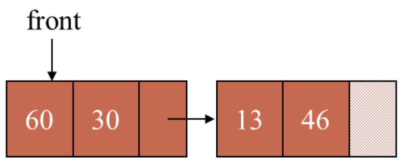
Node(60,30) is inserted



Node(13,46) is inserted



Delete



Delete

APPLICATIONS OF QUEUES

ROUND ROBIN SCHEDULING:

1. In a multitasking operating system, the CPU time is shared between multiple processes.
    - At a given time, only one process is running, all the others are ‘sleeping’.
    - The CPU time is administered by the scheduler. The scheduler keeps all current processes in a queue with the active process at the front of the queue.
    - Every process is granted a specific amount of CPU time, its ‘quantum’.
    - If the process is still running after its quantum run out, it is suspended and put towards the end of the queue.
2. Access to shared resources