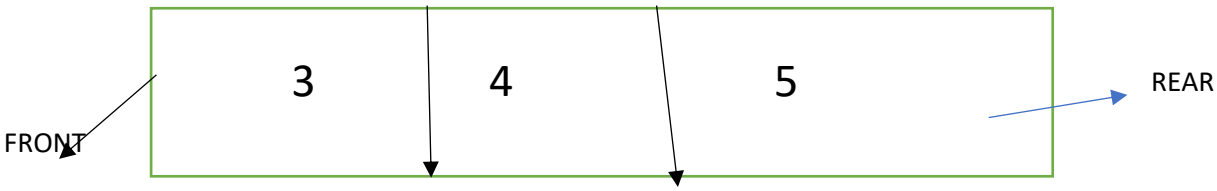
Queue is abstraction data type

First in First out.

We remove any element on first and if we add something we add to tail

A list or collection with the restriction that insertion can be performed at one end rear tail and deletion can be performed at other end.



Enqueue(5) # we first added 5 Then 4 and then 3

Enqueue(4)

Enqueue(3)

Dequeue()

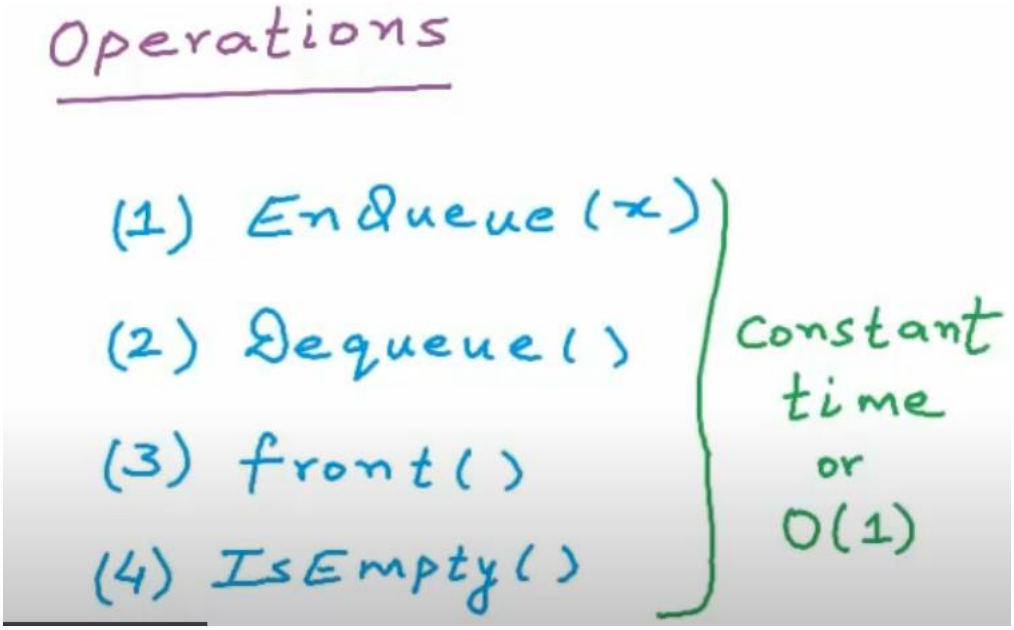
DEQUEUE operation ; deletes first element of the list we added so we need to remove 5 on this list.
FIRST IN FIRST OUT

On second dequeue() operation you remove 4 for example.

Where we uses queues actually?

On shared resources. Source can handle only one resource at a time that's why we use queues.

We can give an example of printer.First document comes it has to be printed then we can print other documents. **1)** When a resource is shared among multiple consumers. Examples include CPU scheduling, Disk Scheduling.
2) When data is transferred asynchronously (data not necessarily received at same rate as sent) between two processes. Examples include IO Buffers, pipes, file IO, etc

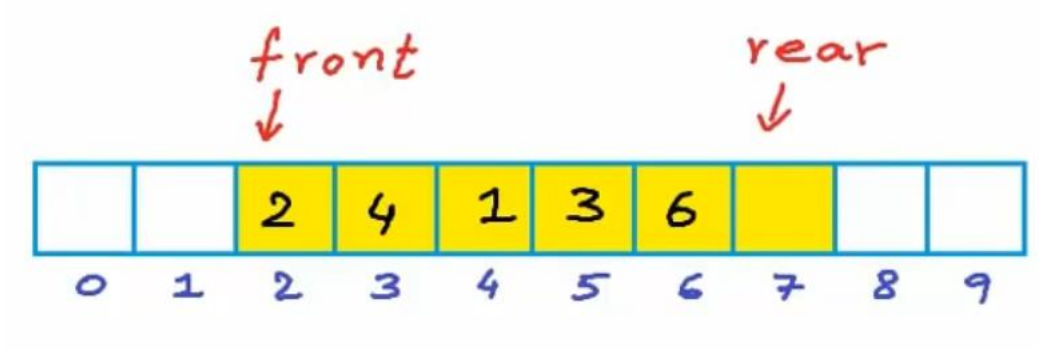


We can implemnt queues using

ARRAYS

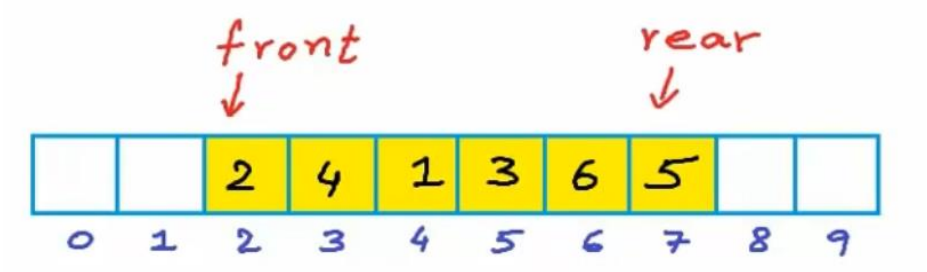
LINKED LISTS

ARRAY BASED IMPLEMENTATION: I am going to use this array to store my queue

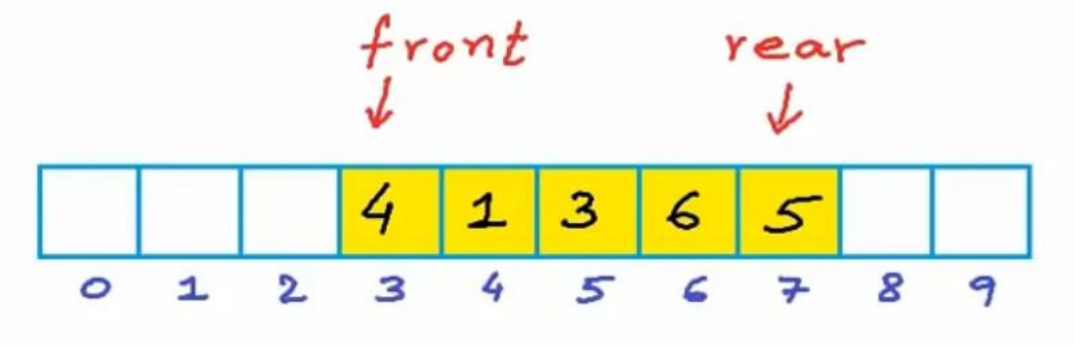


We must add to element rear side and always be removed on front.

We want to insert number 5 We add to right side because its queue

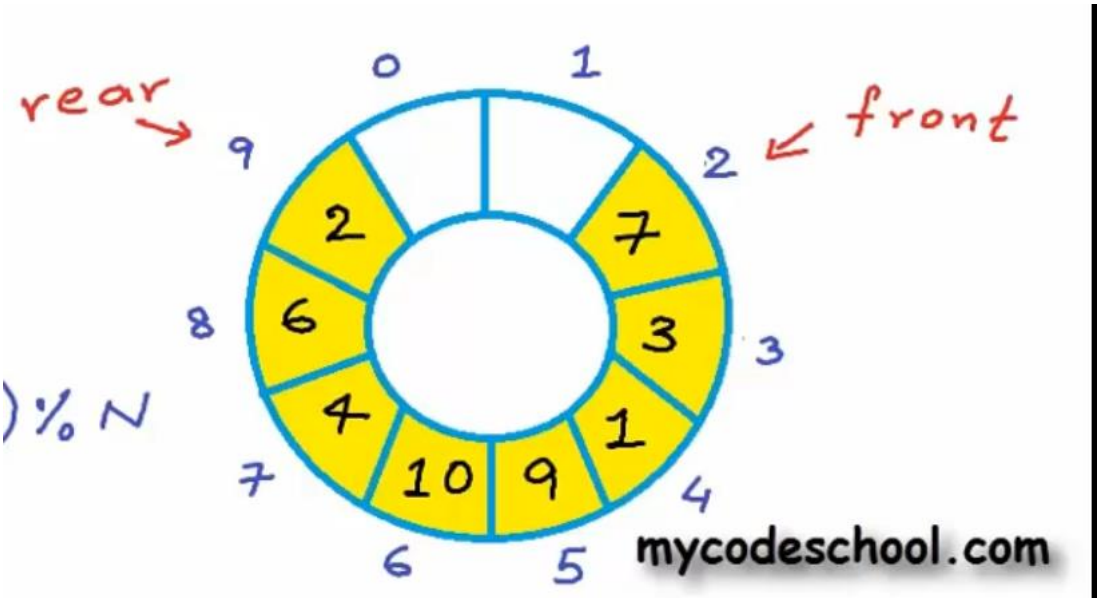


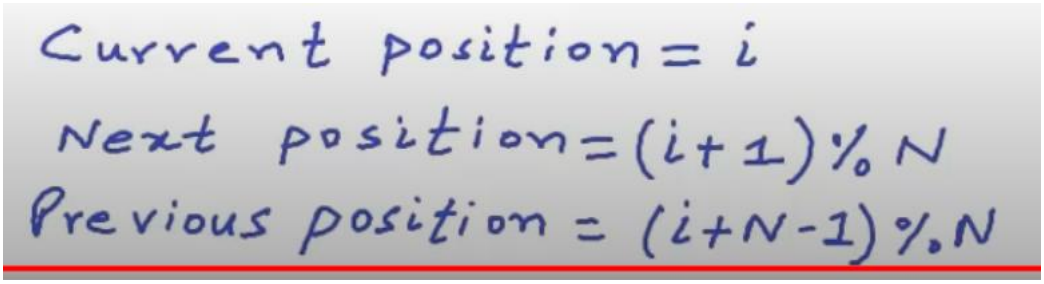
I remove 2 on my array



Those empty cells of the array we can not use them.So we can use Circular array

We say that there is no end in the array



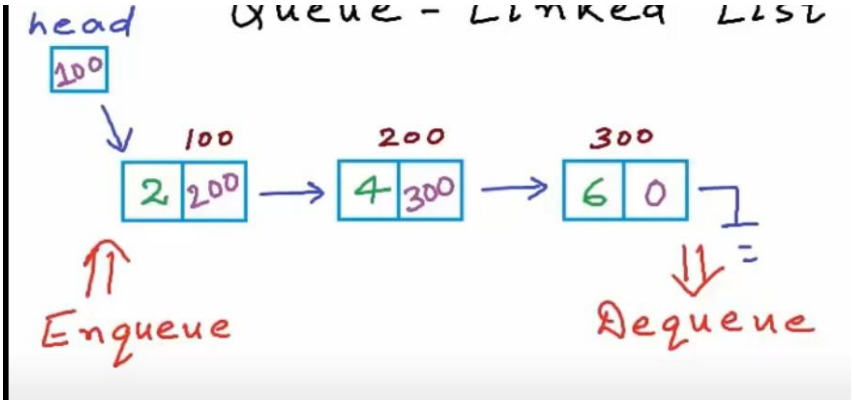


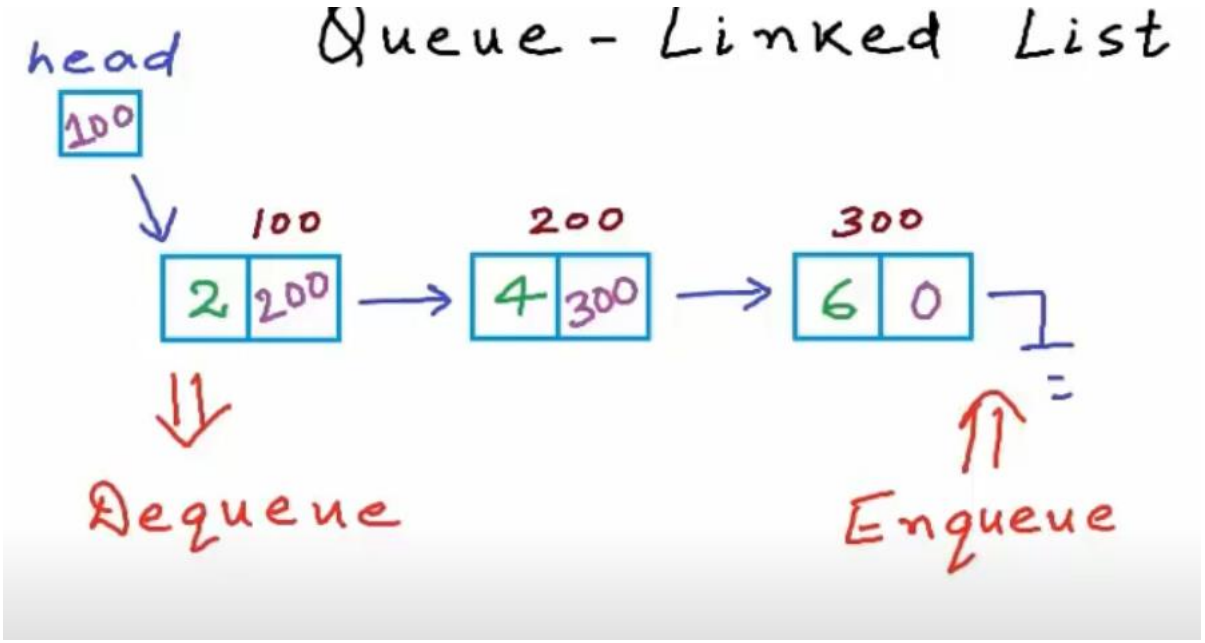
Linked List implementation of Queue

We may see that our array is full and this is a problem. So we need to create new array and and have to copy all of elements in array.so this is a problem actually

Also unused memory of arrays are problems so e s can use Linked lists

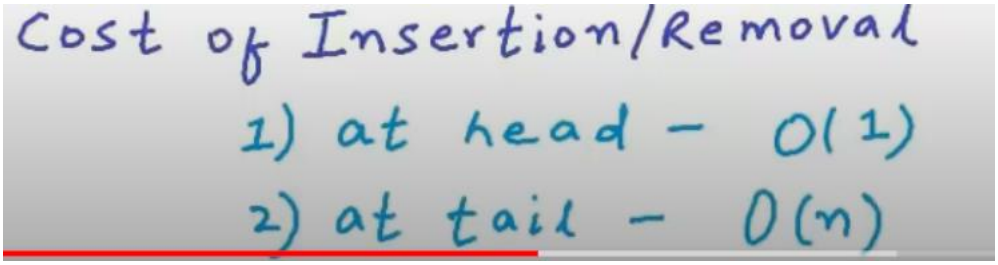
If we pick head side of queue then we will use dequeue



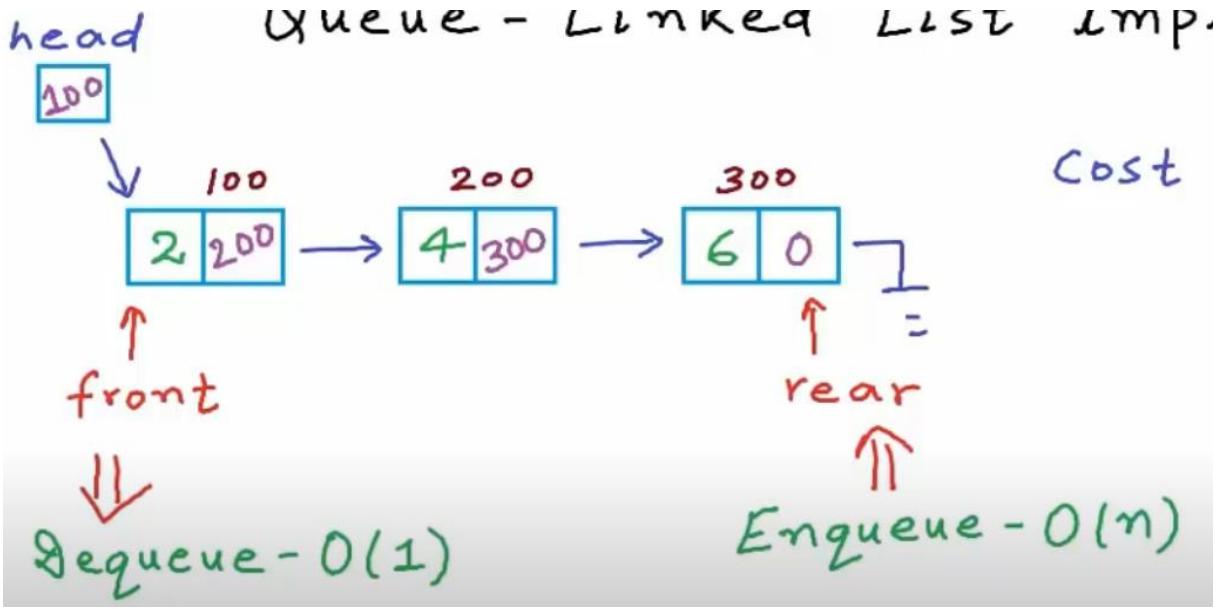


If we pick tail queue operation then dequeue always happen on head

[Eger head tarafiyle calisirsan dequeue tail tarafinda oluyor ama eger tail tarafiyla basliyorsan dequeue head tarafinda oluyor]



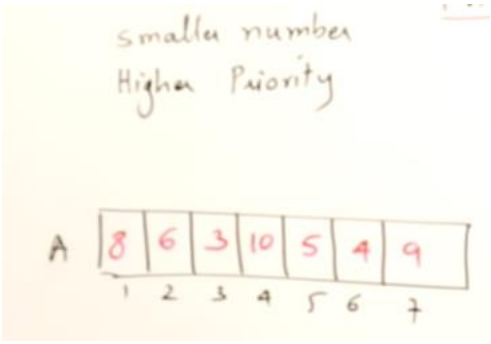
If you insert or remove something on the head it takes constant time but for tail it takes lineer time because linked lists begins to left to right that's why it takes lineer time****


Queue - Linked List Imp.
head
100
100
200
300
2
200
4
300
6
0
Cost
front
rear
Dequeue - O(1)
Enqueue - O(n)

Priority Queue

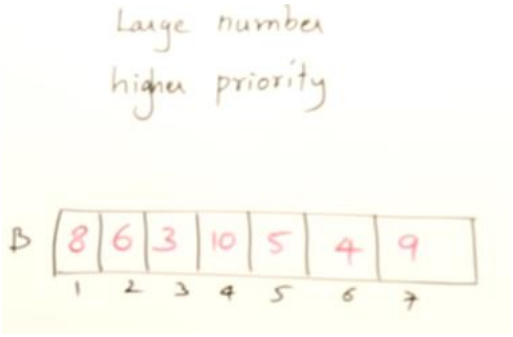
Queue means fifo first in first out.

But priority queue does not means fifo. The elements will have priority and they are inserted and deleted based on prioprity.If we delete something we always looking for high priority elements.



What is the priority of the number? Number itself it priority. SMALLER NUMBER HAS HIGHER PRIORITY 3 and 4 6 etc…so I I delete something I should remove 3

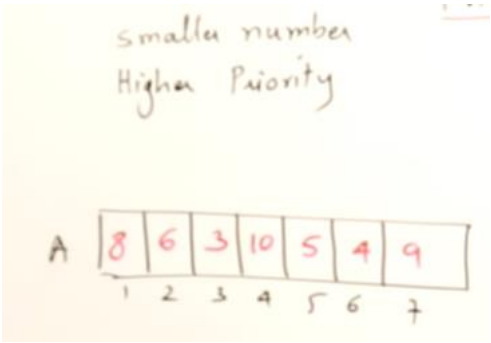
Then other version



I can also say that Higher number has the priority. So who has the priorty? FIRST 10 then 9 then 8 etc..

There are two methos of giving priority smaller number or higher number priority.

DELETE AND INSERT



IF I want to add this array I can add to near of the 9.

When I want to remove something in this array I can remove smaller number 3 then 4 etc.

What is the time complexity ? O(N) means lineer time.

Why ? Because as we know we have to shift all of elements in array.

If you implement Priority queue just implement normal array insert or delete is O(n)

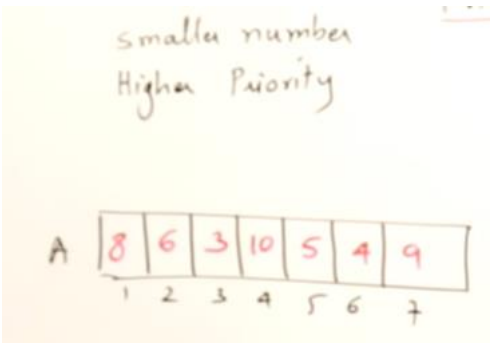
What is the best method? HEAP

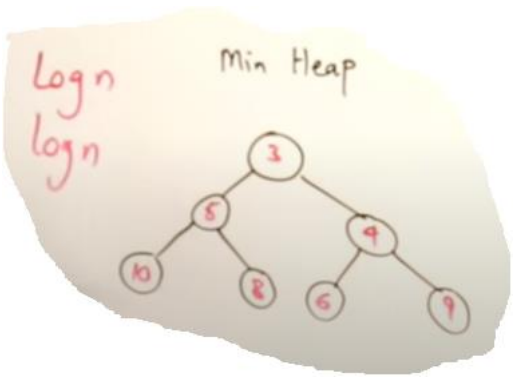
Takes O(logn) time.

For insertion and deletion.

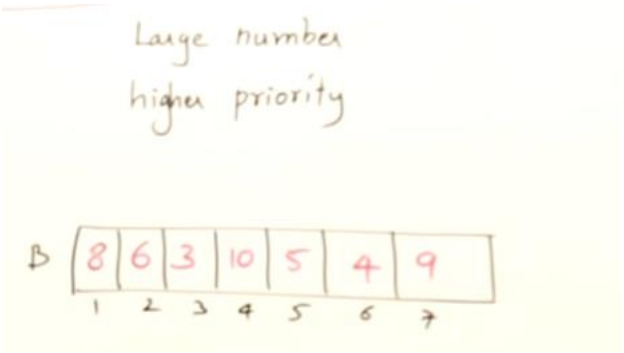
Heap is a best data structures for implementing priority queue.

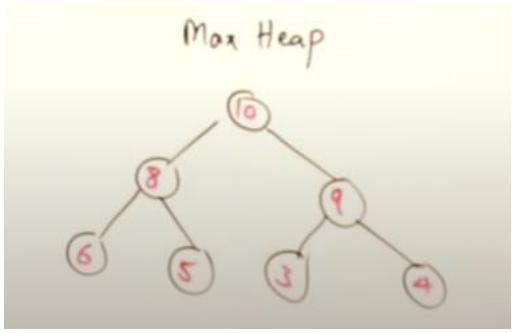
If you have a smaller number priority then create a Min heap





If you have a higher number priority then create max heap.





HEAP IS A FASTER DATA STRUCTURE FOR IMPLEMENTING PRIORITY QUEUE