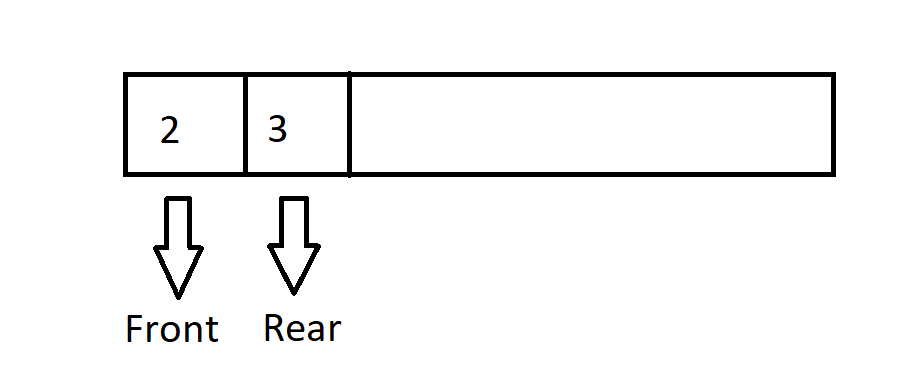
**QUEUE:**

* This is a linear data structure that stores a list of items in which an item can be inserted at one end that is a rear end and removed from the other end that is front end only.
* It follows FIFO technique that is First in First Out.



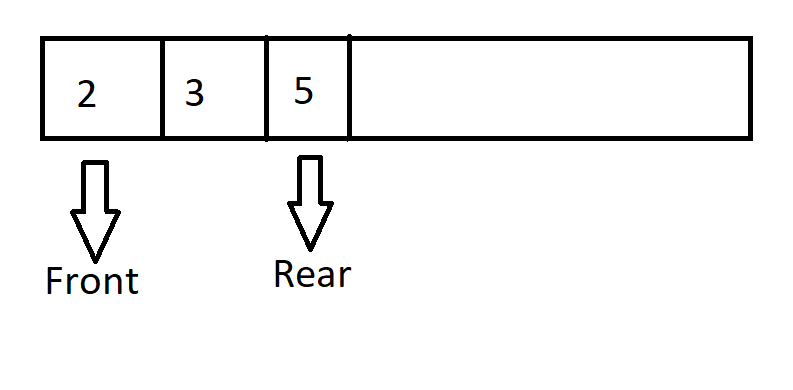
* Note that Front and Rear here are pointers, front point at the first element while rear points at the last element. Initially, the value of front and rear would be -1.



There are different Operations that we can apply on Queue.

1. **Enqueue**

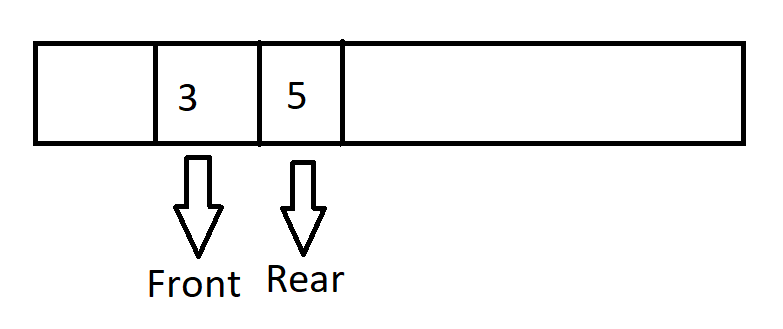
* It is similar to the push operation that we have studied in stack. It will insert a value in our queue from the back.
* For insertion, we will increase the front and rear value to 0 so that they can points to first index after filling that index, we will increase the value of rear only and fill the place where rear is pointing while front remains same at 0th index.
* For example, enqueue (5);



After adding 5, now the pointer is pointing towards 5.

1. **Dequeue**

* It is similar to the pop operation that we have studied in stack. In this operation one value would be removed from front.
* For example, dequeue (2)



Now the font is pointing towards 3 and 2 got removed.

1. **peek ()**

Using this function, you can find out the value that is going to be removed next. It will give us the value of front for example in our previous diagram if we will use peek value then it will return 3 because 3 is in front right now.

1. **isEmpty()**

It checks if the queue is empty or not.

1. **isFull()**

It checks whether a queue is full or not.