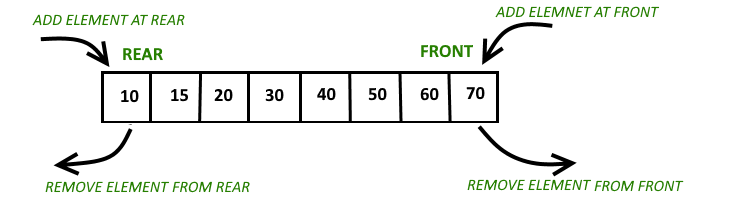
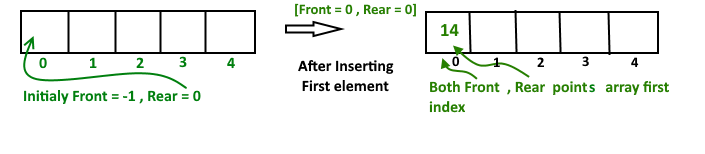
mplementation of Deque using circular array

[Deque or Double Ended Queue](http://quiz.geeksforgeeks.org/deque-set-1-introduction-applications/) is a generalized version of [Queue data structure](http://quiz.geeksforgeeks.org/queue-set-1introduction-and-array-implementation/) that allows insert and delete at both ends.In previous post we had discussed introduction of deque. Now in this post we see how we implement deque Using circular array.  
**Operations on Deque:**  
Mainly the following four basic operations are performed on queue:  
**insetFront()**: Adds an item at the front of Deque.  
**insertRear()**: Adds an item at the rear of Deque.  
**deleteFront()**: Deletes an item from front of Deque.  
**deleteRear()**: Deletes an item from rear of Deque.

In addition to above operations, following operations are also supported  
**getFront()**: Gets the front item from queue.  
**getRear()**: Gets the last item from queue.  
**isEmpty()**: Checks whether Deque is empty or not.  
**isFull()**: Checks whether Deque is full or not.  
[](http://cdncontribute.geeksforgeeks.org/wp-content/uploads/anod.png)

[**Recommended: Please try your approach on *{IDE}* first, before moving on to the solution.**](https://ide.geeksforgeeks.org/)

**Circular array implementation deque**  
For implementing deque, we need to keep track of two indices, front and rear. We enqueue(push) an item at the rear or the front end of qedue and dequeue(pop) an item from both rear and front end.  
**Working**  
1. Create an empty array ‘arr’ of size ‘n’  
initialize **front = -1** , **rear = 0**  
Inserting First element in deque, at either front or rear will lead to the same result.  
[](http://cdncontribute.geeksforgeeks.org/wp-content/uploads/deque-Copy-2.png)  
After insert **Front** Points = 0 and **Rear** points = 0  
**Insert Elements at Rear end**

a). First we check deque if Full or Not

b). IF Rear == Size-1

then reinitialize Rear = 0 ;

Else increment Rear by '1'

and push current key into Arr[ rear ] = key

Front remain same.

**Insert Elements at Front end**

a). First we check deque if Full or Not

b). IF Front == 0 || initial position, move Front

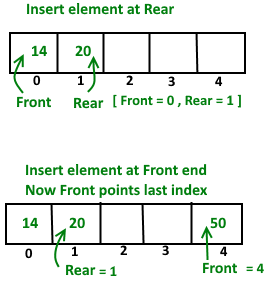
to points last index of array

front = size - 1

Else decremented front by '1' and push

current key into Arr[ Front] = key

Rear remain same.

[](https://www.geeksforgeeks.org/wp-content/uploads/deque.png)

**Delete Element From Rear end**

a). first Check deque is Empty or Not

b). If deque has only one element

front = -1 ; rear =-1 ;

Else IF Rear points to the first index of array

it's means we have to move rear to points

last index [ now first inserted element at

front end become rear end ]

rear = size-1 ;

Else || decrease rear by '1'

rear = rear-1;

**Delete Element From Front end**

a). first Check deque is Empty or Not

b). If deque has only one element

front = -1 ; rear =-1 ;

Else IF front points to the last index of the array

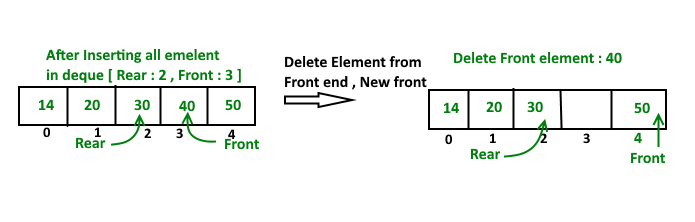
it's means we have no more elements in array so

we move front to points first index of array

front = 0 ;

Else || increment Front by '1'

front = front+1;

[](http://cdncontribute.geeksforgeeks.org/wp-content/uploads/deque-Copy.png)

Below is the implementation of above idea.