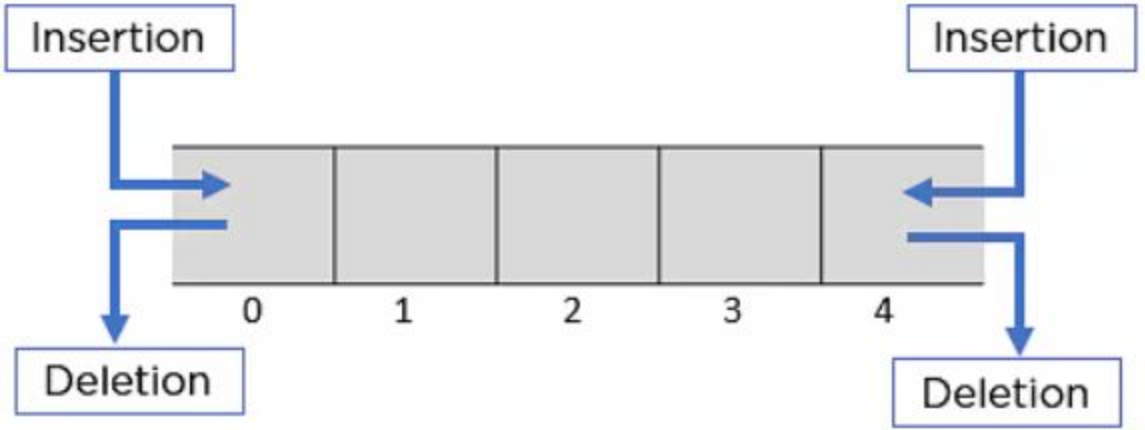
Double ended queue [Deque]

By Er. Kushal Ghimire

Deque ADT

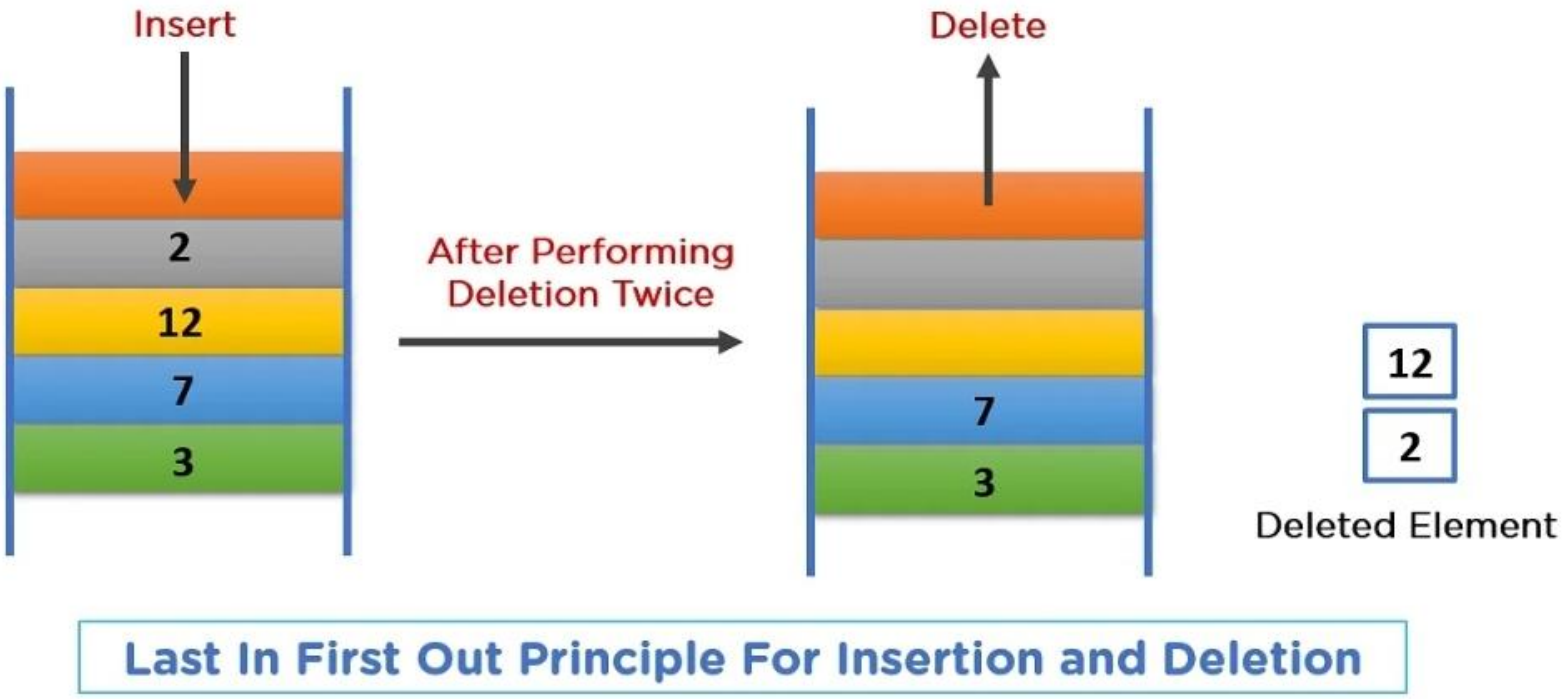
Deque is a more generalized version of a linear queue where, insertion and deletion is performed on both ends of the queue. Hence, the name Double Ended Queue (Deque).



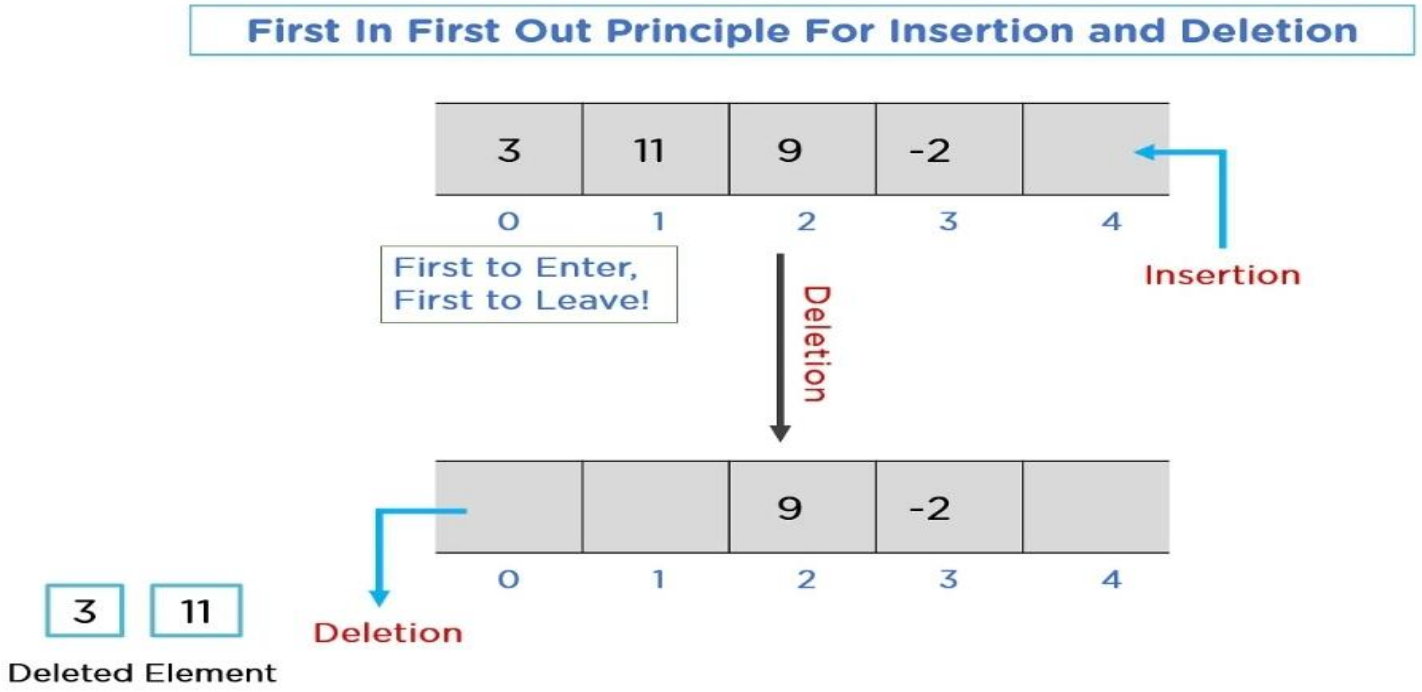
Properties of Deque

- A deque can perform both the insertion and deletion using the LIFO (Last In First Out) principle.
  - Limit the insertion and deletion at one end.
- A Deque can perform insertion and deletion operations using the FIFO (First In First Out) principle.
  - Limit insertion at one end and deletion at another end.

Deque: LIFO

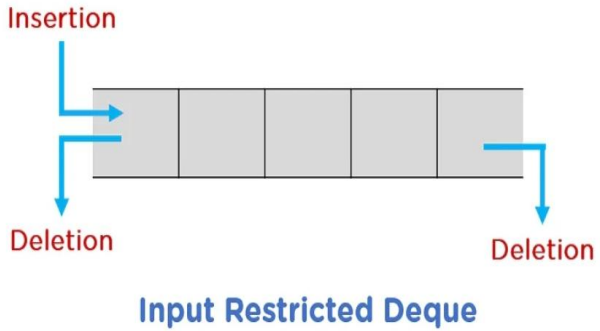

Insert
2
12
7
3
After Performing
Deletion Twice
Delete
7
3
12
2
Deleted Element
Last In First Out Principle For Insertion and Deletion

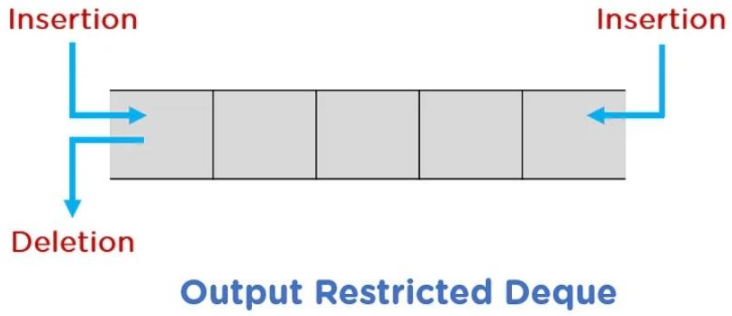
Deque: FIFO


First In First Out Principle For Insertion and Deletion
3
11
9
-2
0
1
2
3
4
First to Enter,
First to Leave!
Deletion
Insertion
9
-2
0
1
2
3
4
3
11
Deleted Element
Deletion

Types of Deque

- Input restricted deque:
  - It is a deque with some limitations while performing insertion operations. In the Input-Restricted Deque, it will perform the deletion at both ends, whereas it performs the insertion at only one end.
- Output restricted deque:
  - It is a deque with some limitations while performing deletion operations. In the Output-Restricted Deque, it will perform the insertion at both ends, whereas it performs the deletion of elements at only one end.





Operations of deque

The time required to implement all these functions must be constant, i.e., time-complexity = O(1).

- rear_enqueue(): insert an element from the rear end.
- rear_dequeue(): delete an element from the rear end.
- front_enqueue(): insert an element from the front end.
- front_dequeue(): delete an element from the front end.

Representation of deque

A double ended queue can be implemented using either of the following:

- Circular Queue
- Doubly Linked List

Implementation of Deque ADT using Circular Queue

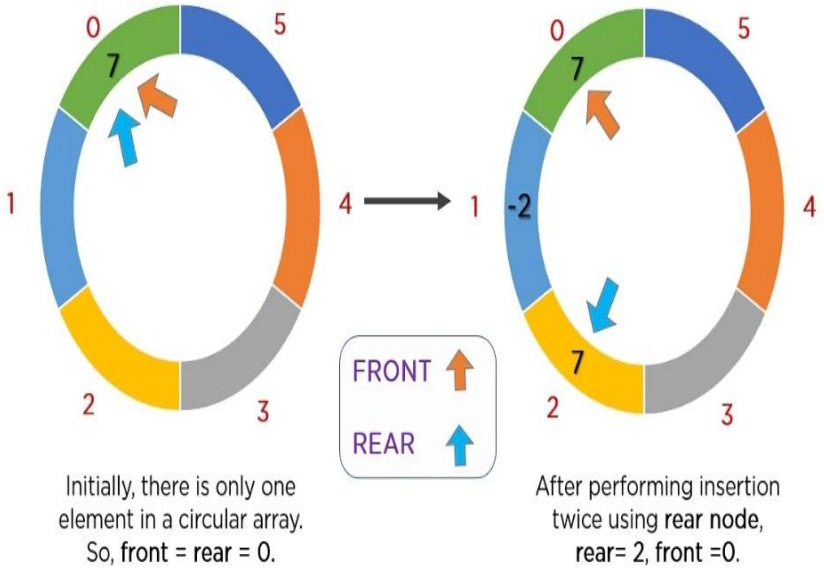
To implement deque using a circular queue, you should be able to implement the insertion and deletion at both ends of a queue.

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

Insertion at rear

If rear is not at last index then, increase rear and then insert an element in the space pointed by rear end.

Else set rear at index 0 then, insert an element in the space pointed by rear end.



Insertion at front

if front is at index 0 then front = MAX - 1 and insert an element at the space pointed by front.

Else front = front - 1 and insert an element at the space pointed by front.

Deletion at front

If front is at last index then delete an element pointed by front and set front to 0 index.

Else delete an element pointed by front and increase front as front = front + 1.

Deletion at rear

If rear is at index 0 then set rear to MAX - 1

Else rear = rear - 1