Queues

Definitions

- A queue is a linear abstract data type such that ***insertions are made at one end, called the rear***, and ***removals are made at the other end, called the front.***

- Queues are sometimes called **FIFOs**: first-in first-out.

  enqueue() → Queue → dequeue()

The two basic operations are:

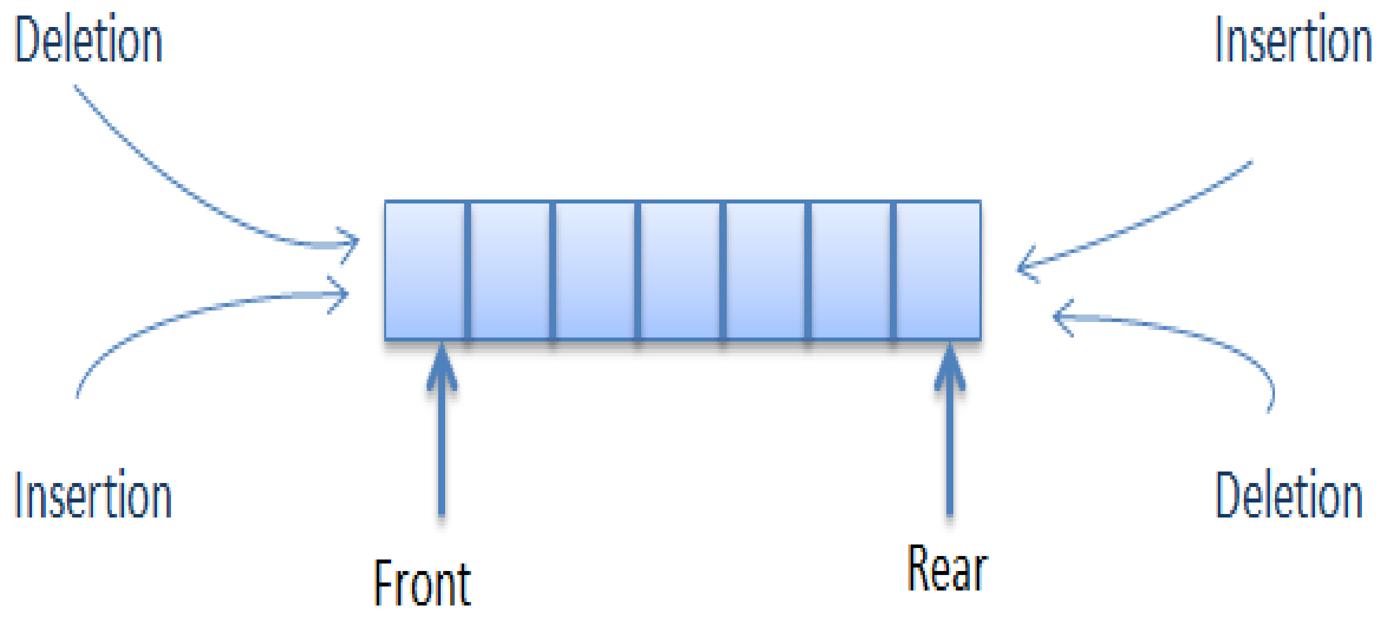
- **enqueue**: adds an element to the ***rear*** of the queue.
- **dequeue**: removes and returns the element at the ***front*** of the queue.

→Software queues are similar to physical ones: queuing at the supermarket, at the bank, at cinemas, etc.

q = new QueueImpl()	q → []
q.enqueue(a)	q → [a]
q.enqueue(b)	q → [a,b]
q.enqueue(c)	q → [a,b,c]
q.enqueue(d)	q → [a,b,c,d]
q.dequeue() → a	q → [b,c,d]
q.dequeue() → b	q → [c,d]
q.enqueue(e)	q → [c,d,e]
q.dequeue() → c	q → [d,e]
q.dequeue() → d	q → [e]

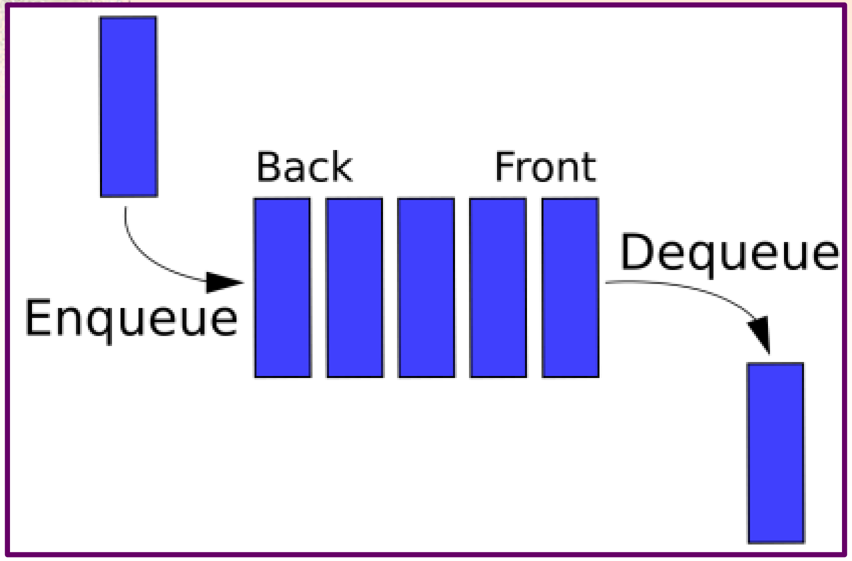
→ **Elements of a queue are processed in the same order as the they are inserted into the queue, here "*a" was the first element to join the queue and it was the first to leave the queue: first-come first-serve.*"**

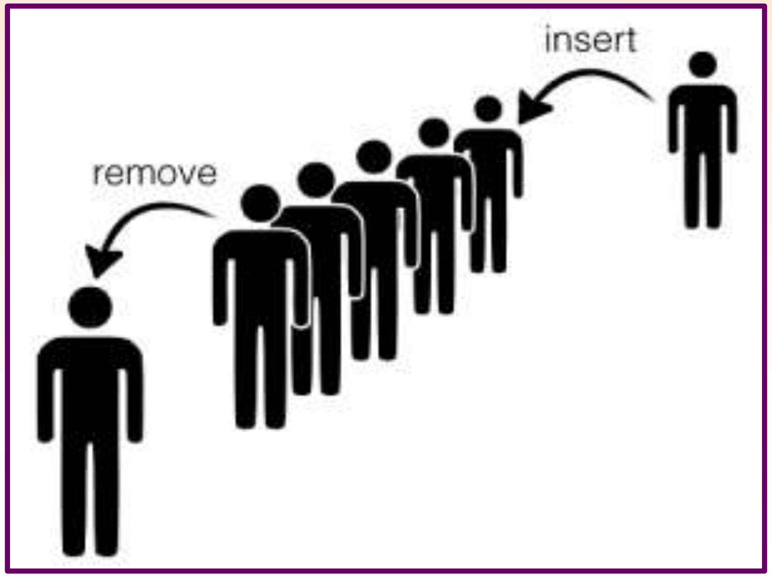
Ex.


Deletion
Insertion
Insertion
Deletion
Front
Rear

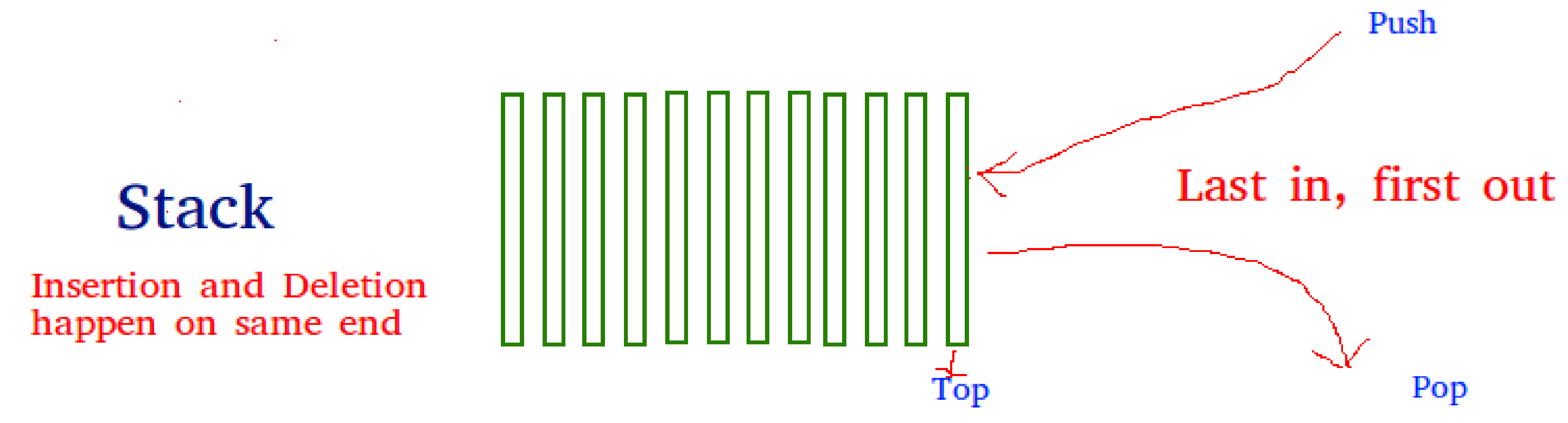
LAST IN LAST OUT
ONE WAY
FIRST IN FIRST OUT

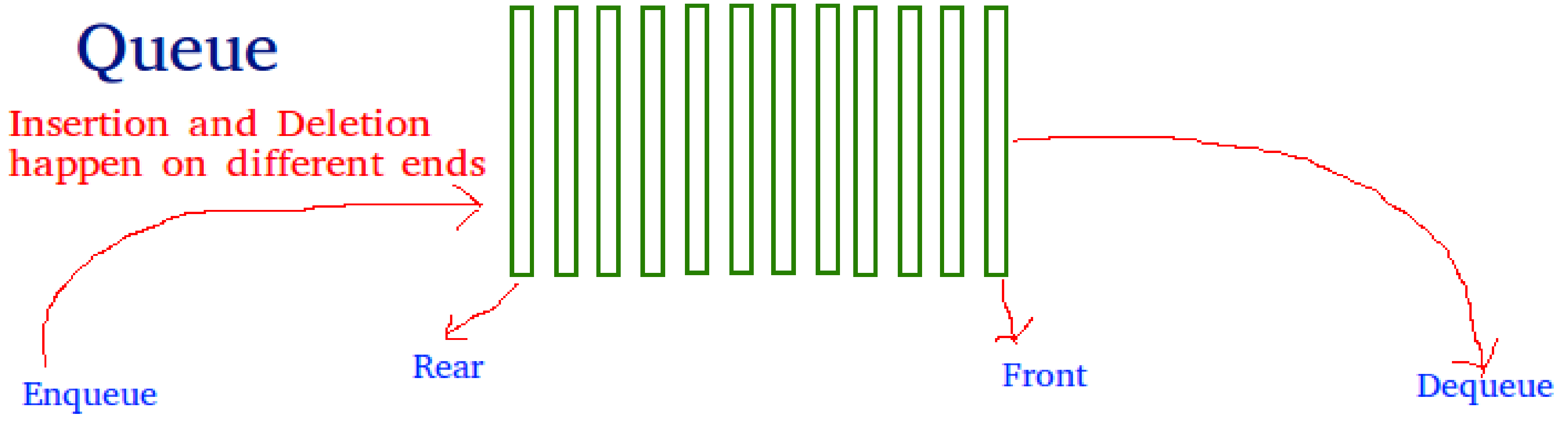
STARBUCKS COFFEE
E1


Back
Front
Enqueue
Dequeue


insert
remove

Different :→STACK /QUEUE





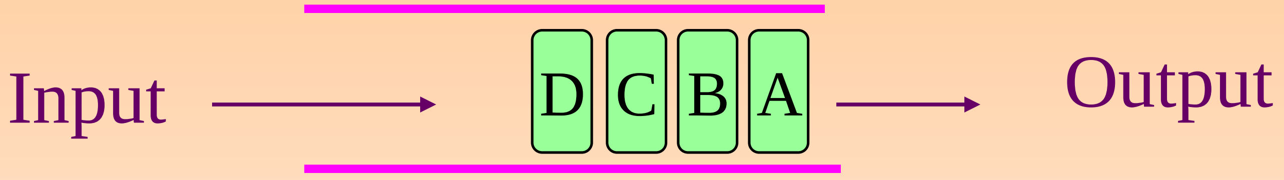
First in, first out

Topics

- Queue
  - Implementation of Queue
  - Usage of Queue

Queue

- Queue: *First In First Out (FIFO)*



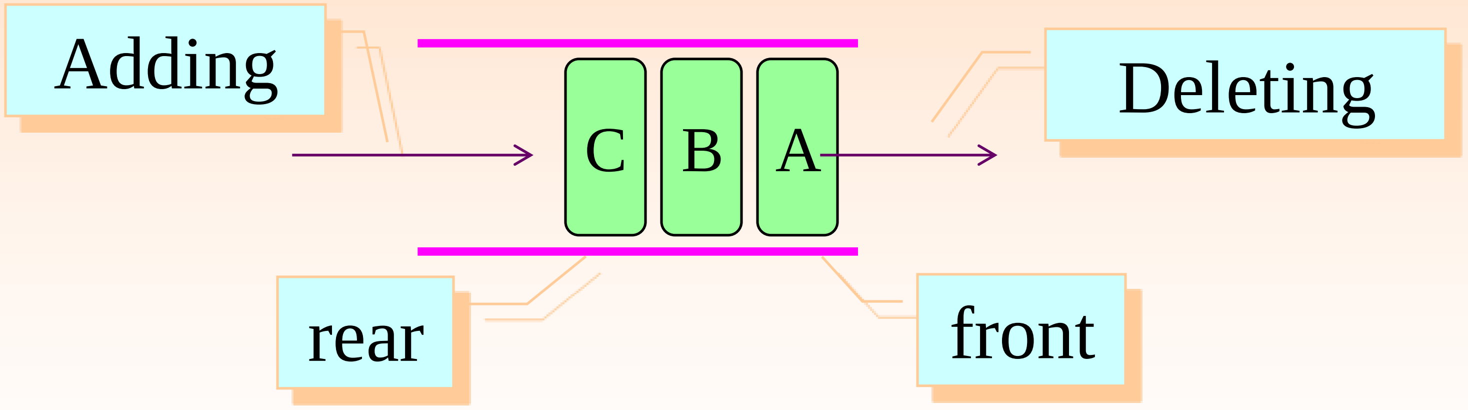
- Toll Station
  - Car comes, pays, leaves
- Check-out in Big Y market
  - Customer comes, checks out and leaves
- More examples: Printer, Office Hours, …

More Examples of Queue

- In our daily life
  - Airport Security Check
  - Cinema Ticket Office
  - Bank, ATM
  - Anything else ?

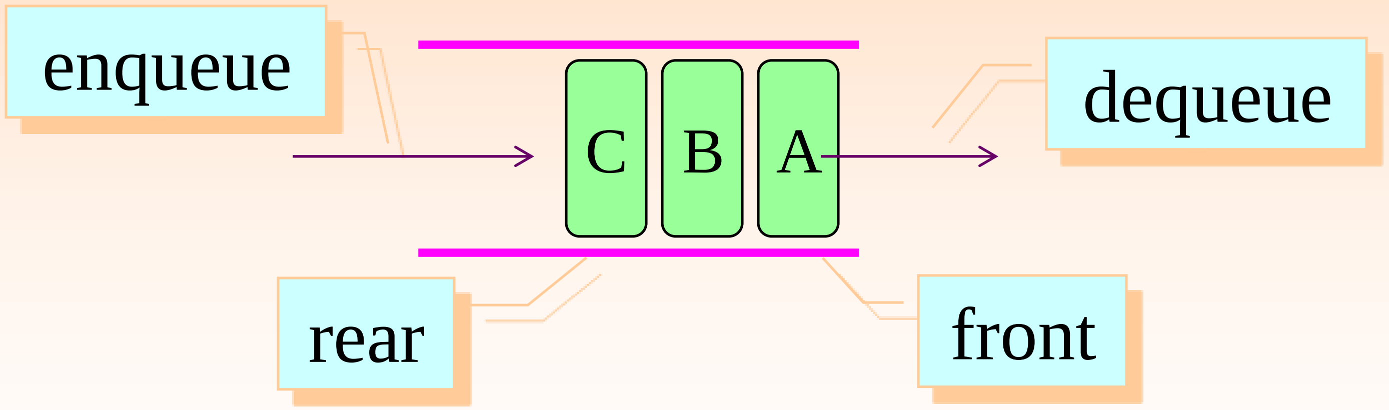
What Is Queue

- Queue is an abstract data type
- Adding an entry at the rear
- Deleting an entry at the front



Abstract Data Types

- Queue
  - Operating on both ends
  - Operations: ***EnQueue(in), DeQueue(out)***



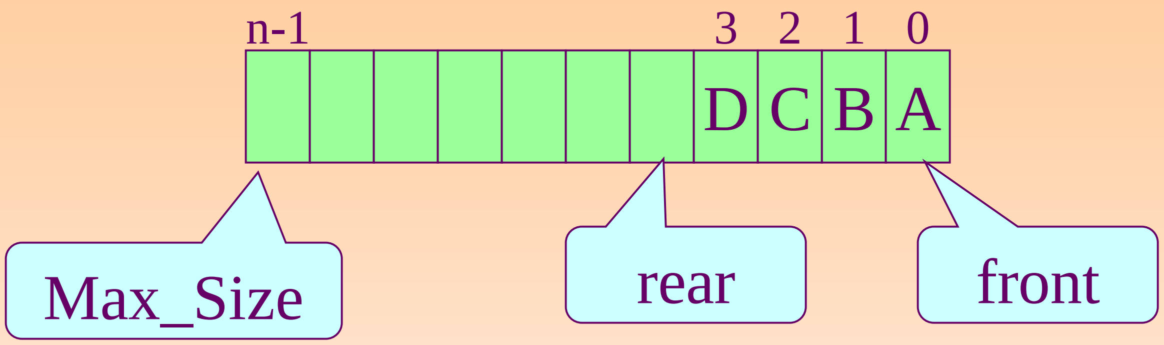
Queue

Queue is FIFO (First-In First-Out)

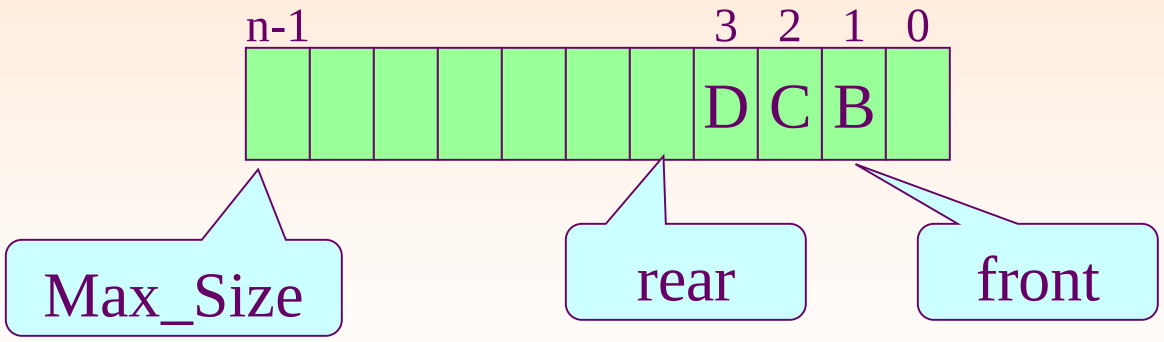
A queue is open at two ends. You can only add entry (enqueue) at the rear , and delete entry (dequeue) at the front.

Note that you cannot add/extract entry in the ***middle*** of the queue.

Array Implementation of Queue



After A leaves,



Operations

- enqueue
  - add a new item at the rear
- dequeue
  - remove a item from the front
- isEmpty
  - check whether the queue is empty or not
- isFull
  - check whether the queue is full or not
- size
  - return the number of items in the queue
- peek
  - return the front item

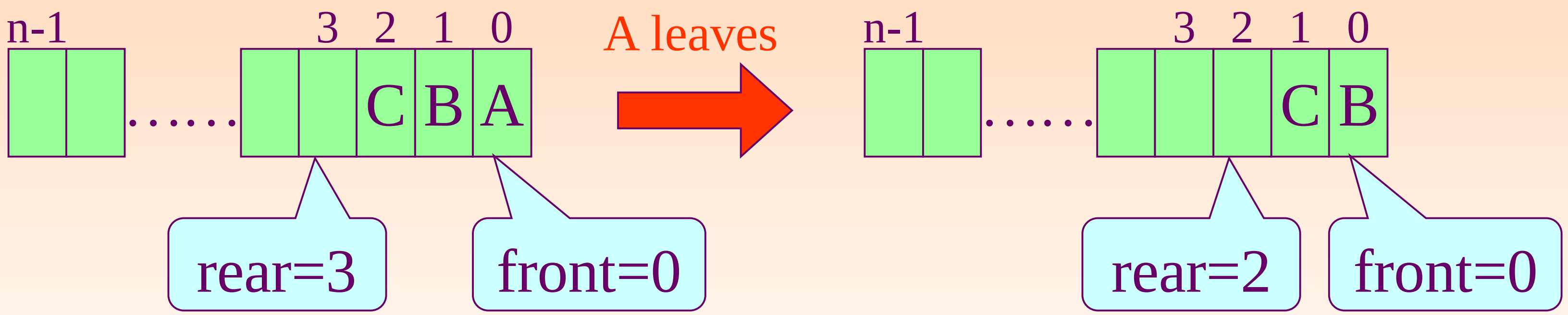
Circular Array

what is a circular queue ?

- A Circular Queue is a special version of queue where the last element of the queue is connected to the first element of the queue forming a circle.
- The operations are performed based on FIFO (First In First Out) principle. It is also called 'Ring Buffer'.

Two Solutions

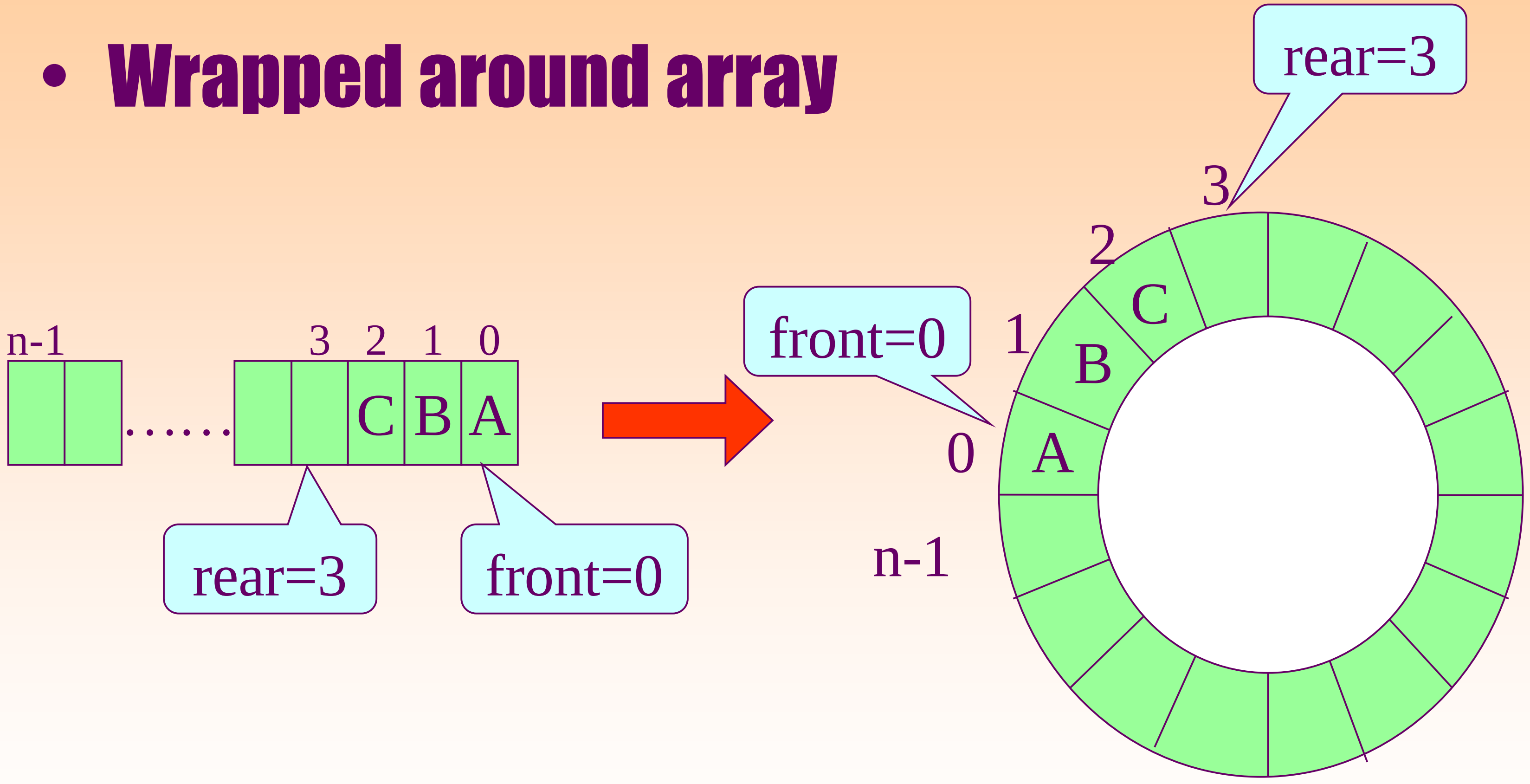
- Shifting all items to front in the array when dequeue operation. (Too Costly...)



- Wrapped around array ---- Circular Array

Circular Array

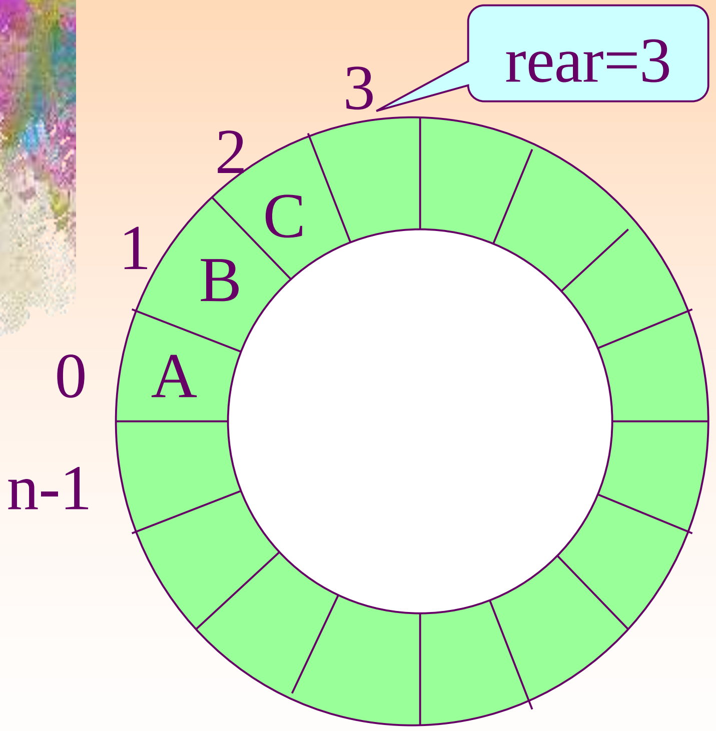
- Wrapped around array

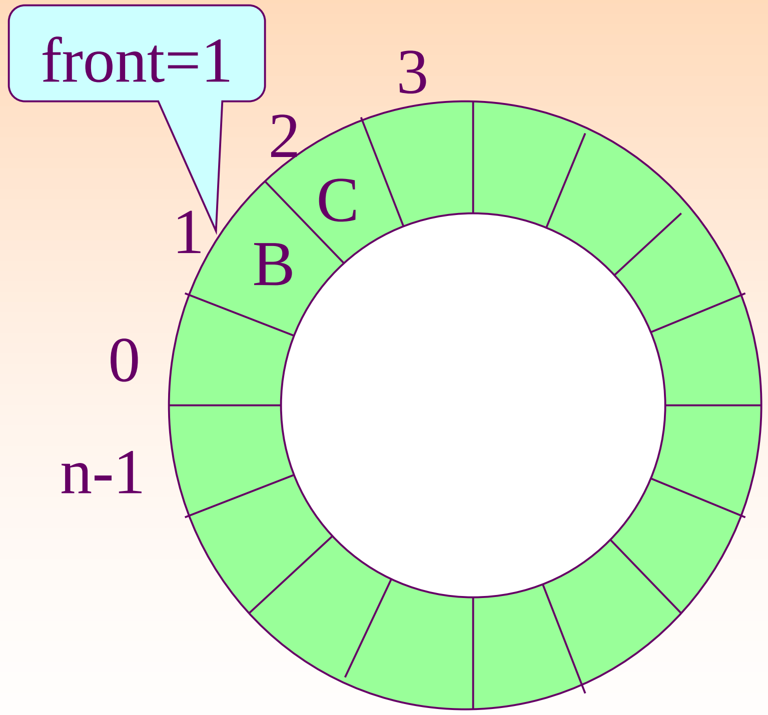


EnQueue & DeQueue In Circular Array

- EnQueue
  - rear = (rear + 1) MOD n

- DeQueue
  - front = (front + 1) MOD n





Empty/Full In Circular Array

- When rear equals front, Queue is empty
- When (rear + 1) MOD n equals front, Queue is full
- Circular array with capacity *n* at most can hold *n-1* items.