QUEUE

Neha Shah

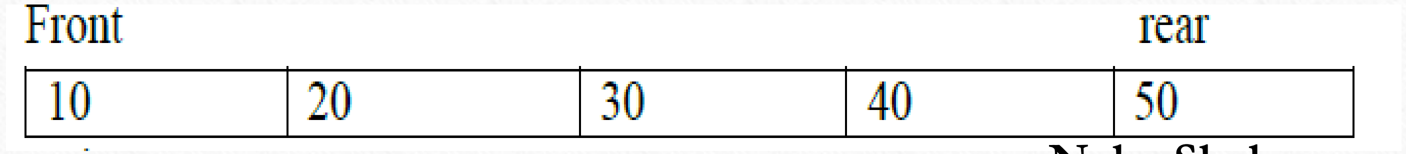
Queue Overview

- Queue Definition
- Example of Queue
- Front and rear in Queue

Neha Shah

The Queue

- It is linear Data structure.
- It is an ordered collection of an elements in which insertion takes place at one end called as rear and deletion takes place at another end called as front of the queue, queue follows FIFO(First in First Out)principle.
- So the elements that are inserted at first will be the first that will get deleted from it.



Front | | | | rear

10	20	30	40	50

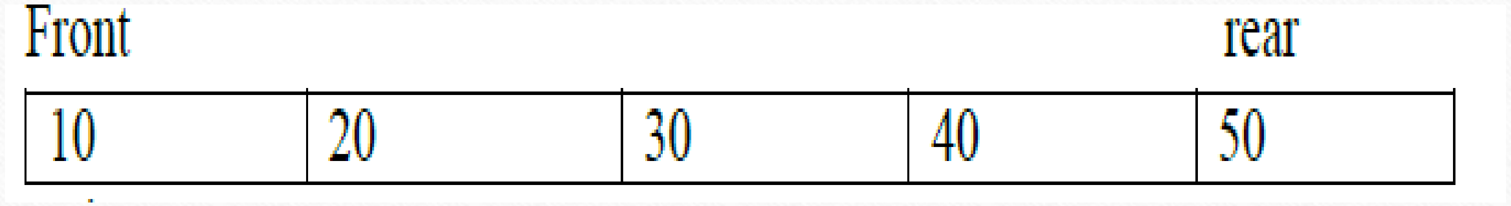
Working Principle

Queue works on FIFO(First in First Out)principle

That is the first element inserted will be the first element that will be removed from queue.

Example

- Consider a queue of 5 elements
- You want to insert 10,20,30,40,50 into the queue.
- First element to get inserted into the queue will be 10
- Second will be 20 then 30, 40, and last element will be 50.



Front				rear
10	20	30	40	50

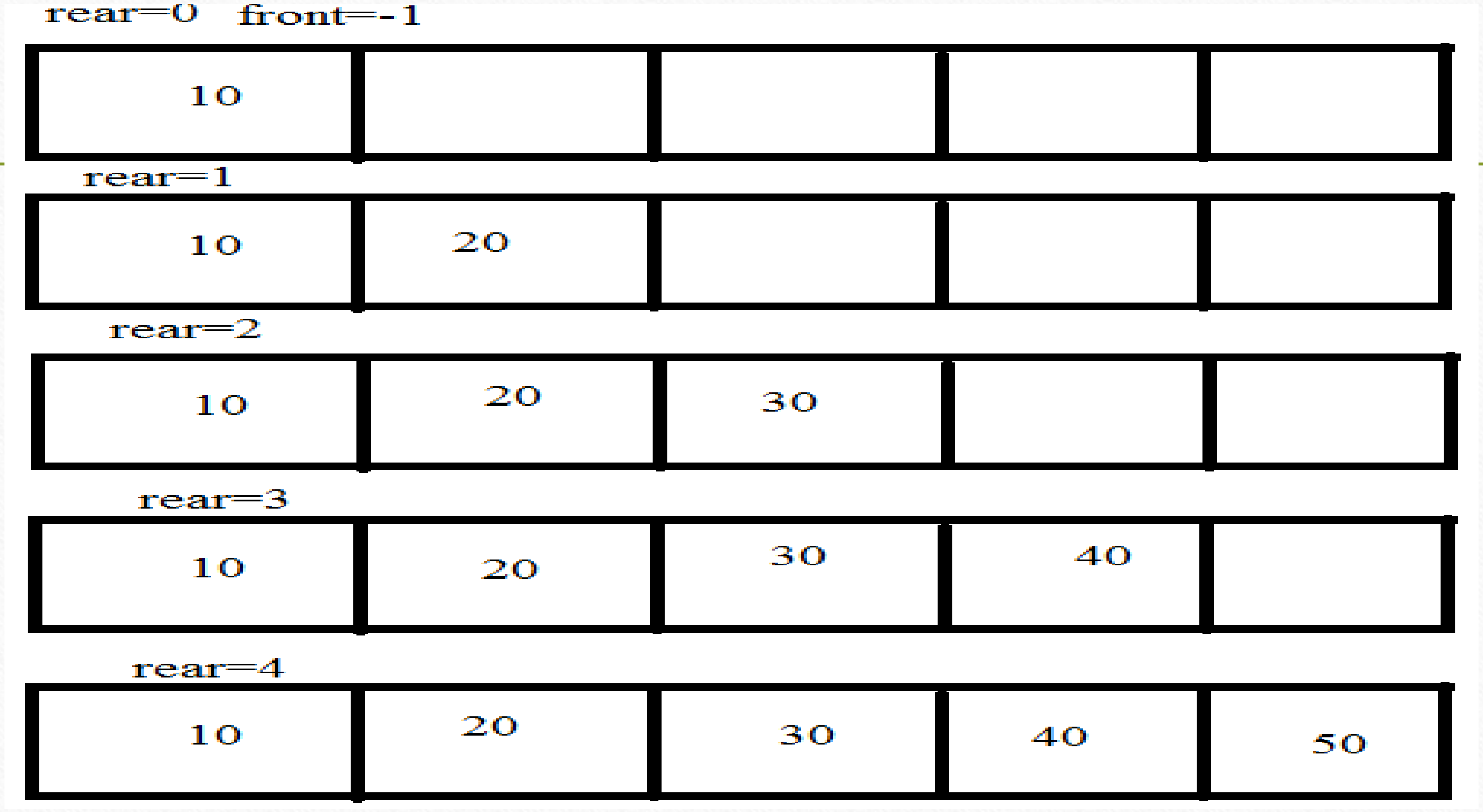
Neha Shah

Front and Rear in the Queue

- Rear: It is a position/end from which elements can be inserted into queue.
- If there are no elements in the queue then rear will be equal to -1 that is rear==-1
- Front: It is a position/end from which elements can be deleted from queue.
- If there are no elements in the queue then front will be equal to -1 that is front==-1

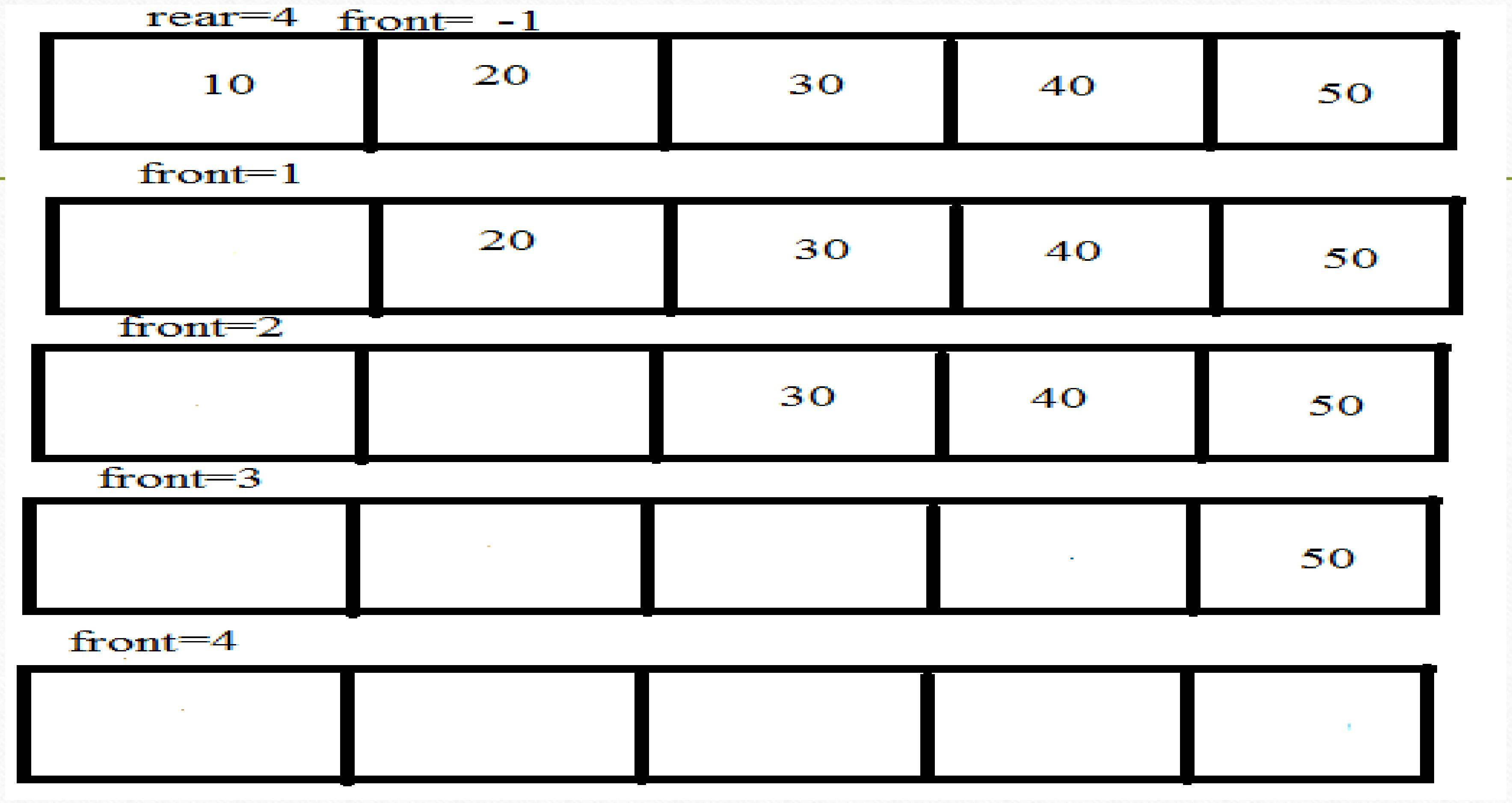
Neha Shah

Rear: The Insertion



Neha Shah

Front: The Deletion



Neha Shah

Operations on Queue

- Insertion :Inserting an element into the queue
- Deletion: Deleting an element from the queue
- Insert operation: Before inserting new element queue full condition must be checked. Element can be added at rear position only.
- Delete operation: Before deleting any element queue

Neha Shah

Insertion Algorithm

- Before inserting an element into the queue we must check for squeue full condition that is if there is space for element then only we can insert new element
- As we are implementing stack using an array
- Algorithm:
  1. Check whether stack is full or not [rear==Maxsize-1]
  2. If queue is full print message queue is full
  3. Otherwise, increment rear by one position[rear++]
  4. Take an element from the user and insert in at the queue[rear]=element

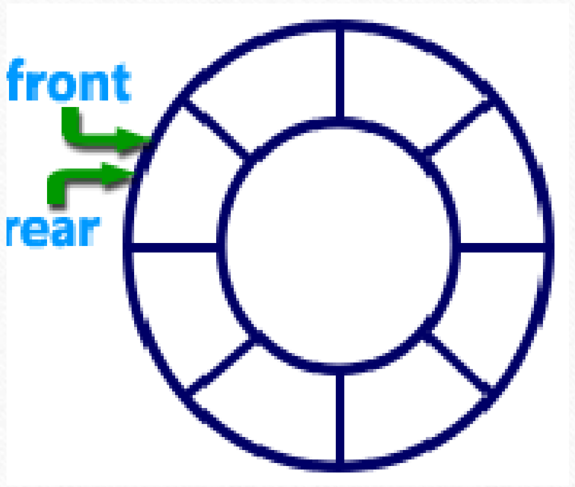
Neha Shah

Deletion Algorithm

- Before deleting an element from the queue we must check for queue empty condition that is if there is an element in the queue then only we can deletion operation is possible
- As we are implementing stack using an array
- Algorithm:
  1. Check whether queue is empty or not [front==-1]
  2. If queue is empty print message queue is empty
  3. Otherwise, delete an element from the queue. [element=queue[front]]
  4. Increment front by one position [front++]

Neha Shah

Circular Queue



Neha Shah

Primitive operation on queue

1. Insert an element into queue
2. Delete an element from queue

Insert operation:

- a. In insertion operation elements inserted into queue.
- b. Before inserting new element queue full condition must be checked
- c. Element can be added at rear position only

Neha Shah

Algorithm: Insert

- **Step 1:** If (FRONT = = (rear+1) % MAXSIZE) then Write "Queue is Overflow"
- **Step 2:** Else REAR = (REAR + 1)%MAXSIZE
- **Step 3:** QUEUE [REAR] = Element
- **Step 4:** If FRONT = -1 then FRONT = 0 and REAR=0

Neha Shah

Algorithm: Delete

Step 1: If FRONT==-1 then Write “Queue is Underflow”

Step 2: Else Element=QUEUE [FRONT]

Step 3: If FRONT = REAR then

FRONT = -1

REAR = -1

Else FRONT = (FRONT + 1) % MAXSIZE

Neha Shah

Priority Queue

- It works on basis of priority
- There are two types of priority queue
  - Ascending order priority queue
  - Descending order priority queue

Ascending order priority queue

- All data are arranged in ascending order .
- Insertion and deletion takes place on ascending order.

Neha Shah

Descending order priority queue

- All data are arranged in ascending order .
- Insertion and deletion takes place in descending order.

Neha Shah