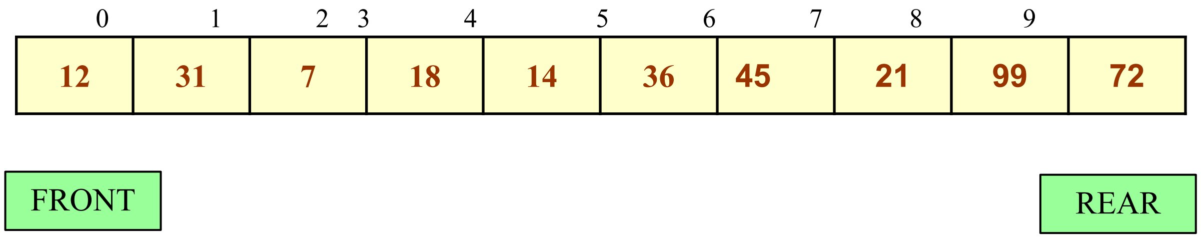
Types of Queues

- Circular Queue
- Priority Queue
- deQueue (Double Ended Queue)

Circular Queues

- LINEAR QUEUE

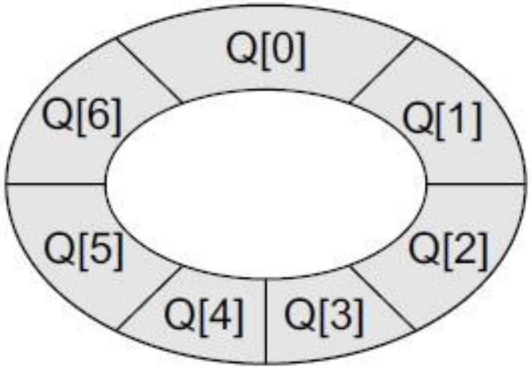


0	1	2	3	4	5	6	7	8	9
12	31	7	18	14	36	45	21	99	72

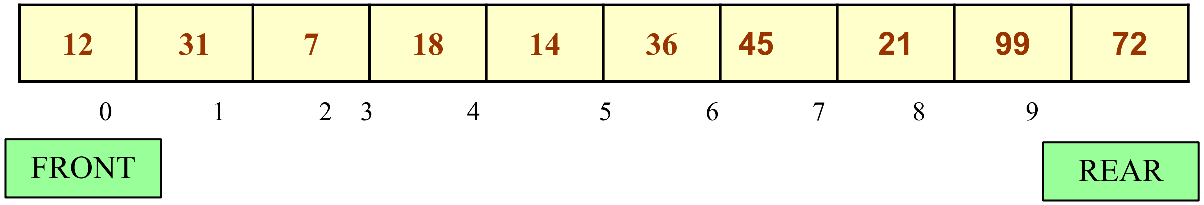
FRONT

REAR

- CIRCULAR QUEUE

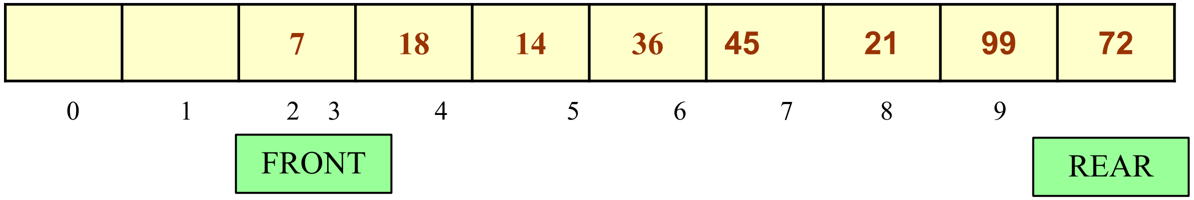


Circular Queues



12	31	7	18	14	36	45	21	99	72

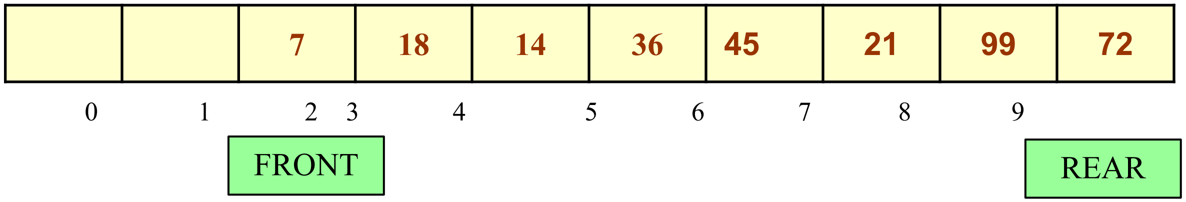
- In this queue, FRONT = 0 and REAR = 9.
- After delete two elements



		7	18	14	36	45	21	99	72

- Now, if you want to insert a new element, it cannot be done because the space is available only at the left of the queue.
- If REAR = MAX – 1, then OVERFLOW condition exists.
- **This is the major drawback of a linear queue. Even if space is available, no insertions can be done once REAR is equal to MAX – 1.**

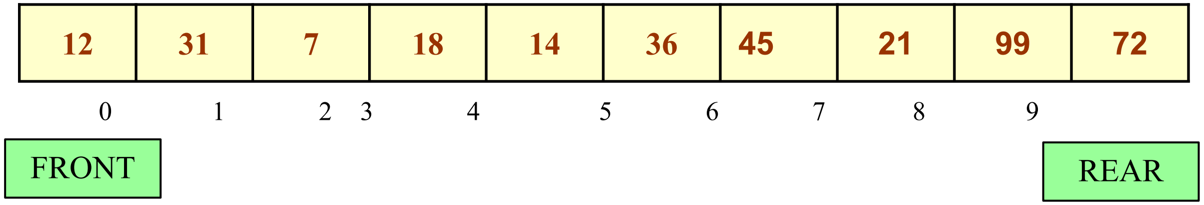
Circular Queues



- This leads to wastage of space. **In order to overcome this problem, we use circular queues.**

- In a circular queue, the first index comes right after the last index.

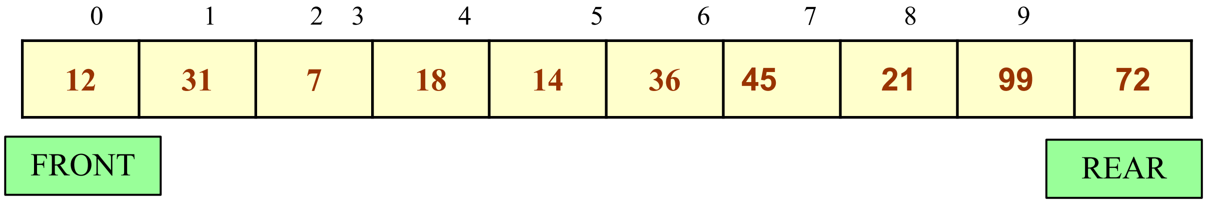
- A circular queue is full, only when **FRONT=0 and REAR = MAX – 1.**



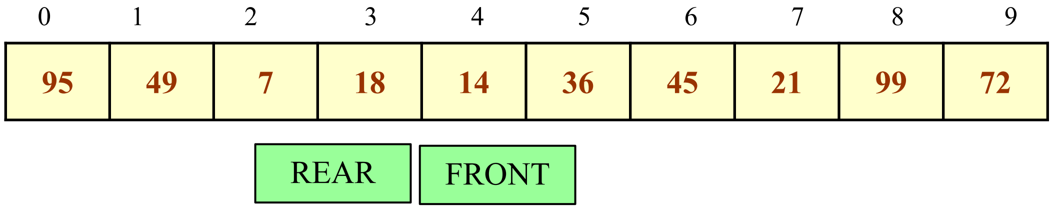
Inserting an Element in a Circular Queue

- For insertion we check for three conditions which are as follows:

- If **FRONT=0 and REAR= MAX – 1** OR **REAR = FRONT – 1** then the circular queue is full.

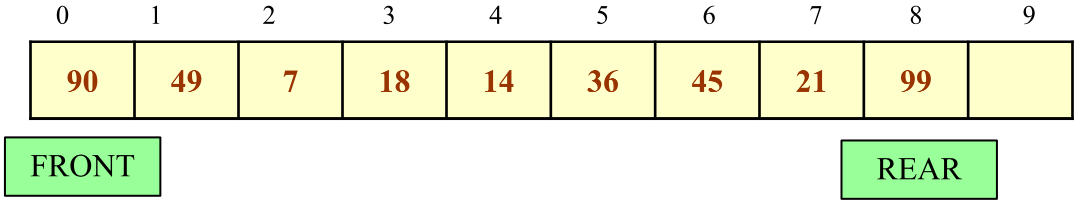


0	1	2	3	4	5	6	7	8	9
12	31	7	18	14	36	45	21	99	72
FRONT									REAR



0	1	2	3	4	5	6	7	8	9
95	49	7	18	14	36	45	21	99	72
			REAR	FRONT					

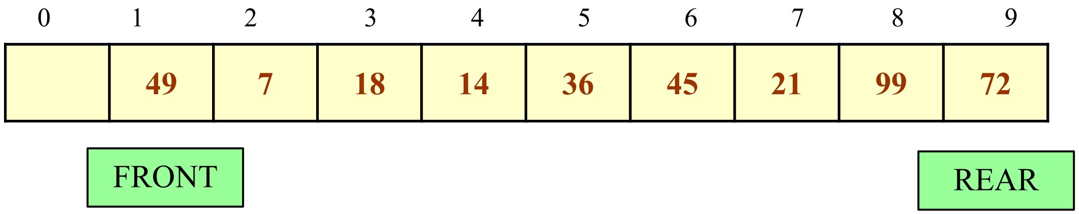
- If **REAR != MAX – 1**, then the rear will be incremented and value will be inserted



0	1	2	3	4	5	6	7	8	9
90	49	7	18	14	36	45	21	99	
FRONT								REAR	

Inserting an Element in a Circular Queue

- If **FRONT!=0 and REAR=MAX -1**, then it means that the queue is not full. So, set REAR = 0 and insert the new element.



0	1	2	3	4	5	6	7	8	9
	49	7	18	14	36	45	21	99	72

FRONT

REAR

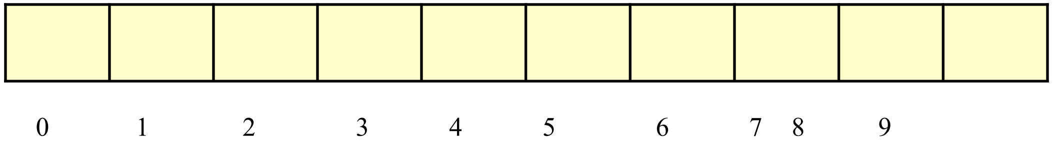
Algorithm to Insert an Element in a Circular Queue

```
Step 1: IF FRONT = 0 and Rear = MAX – 1 OR REAR = FRONT – 1,
        then
         Write “OVERFLOW”
                 Goto Step 4
        [END OF IF]
Step 2: IF FRONT = -1 and REAR = -1, then;
         SET FRONT = REAR = 0
    ELSE IF REAR = MAX – 1 and FRONT != 0
         SET REAR = 0
    ELSE
         SET REAR = REAR + 1
        [END OF IF]
Step 3: SET QUEUE[REAR] = VAL
Step 4: Exit
```

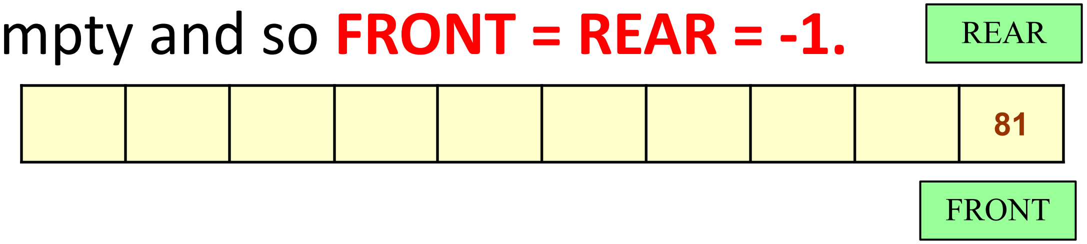
Deleting an Element from a Circular Queue

- To delete an element again we will check for three conditions:

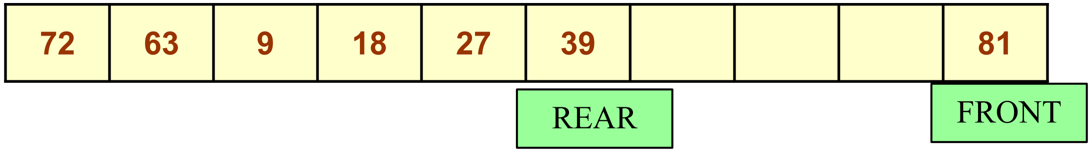
- If **FRONT = -1**, then it means there are **no elements in the queue**. So an underflow condition will be reported.



- If the queue is not empty and after returning the value on FRONT, if **FRONT = REAR**, then it means now the queue has become empty and so **FRONT = REAR = -1.**



- If the queue is not empty and after returning the value on FRONT, if **FRONT = MAX -1**, then **FRONT is set to 0.**



Algorithm to Delete an Element from a Circular Queue

```
Step 1: IF FRONT = -1, then
          Write "Underflow"
          Goto Step 4
        [END OF IF]
Step 2: SET VAL = QUEUE[FRONT]
Step 3: IF FRONT = REAR
          SET FRONT = REAR = -1
        ELSE
          IF FRONT = MAX -1
               SET FRONT = 0
          ELSE
               SET FRONT = FRONT + 1
          [END OF IF]
        [END OF IF]
Step 4: EXIT
```