CIRCULAR QUEUE

Suppose a queue Q has maximum size 5, say 5 elements pushed and 2 elements popped.

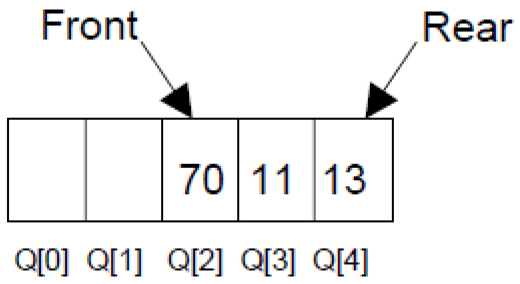

Fig. 4.10

Now if we attempt to add more elements, even though 2 queue cells are free, the elements cannot be pushed. Because in a queue, elements are always inserted at the rear end and hence rear points to last location of the queue array Q[4]. That is queue is full (overflow condition) though it is empty. This limitation can be overcome if we use **circular queue**.

In circular queues the elements Q[0],Q[1],Q[2] .... Q[n – 1] is represented in a circular fashion with Q[1] following Q[n]. A circular queue is one in which the insertion of a new element is done at the very first location of the queue if the last location at the queue is full.

Suppose Q is a queue array of 6 elements. Push and pop operation can be performed on circular. The following figures will illustrate the same.

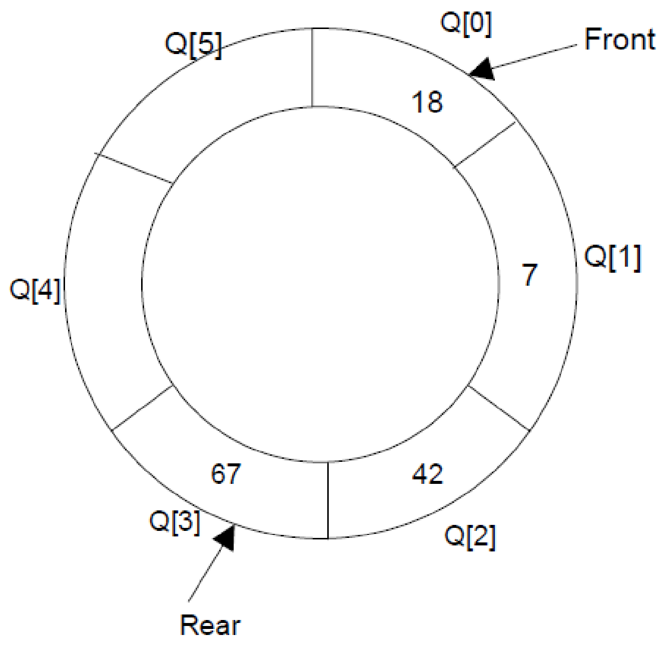

Fig. 4.11. A circular queue after inserting 18, 7, 42, 67.

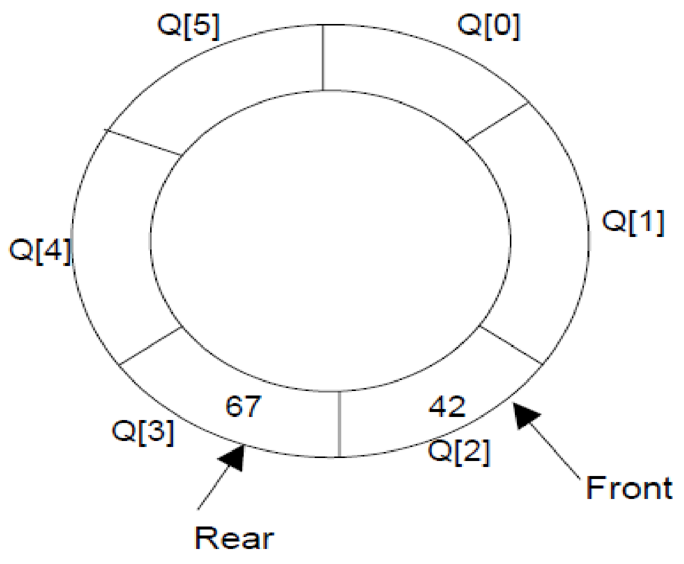

Fig. 4.12. A circular queue after popping 18, 7

After inserting an element at last location Q[5], the next element will be inserted at the very first location (i.e., Q[0]) that is circular queue is one in which the first element comes just after the last element.

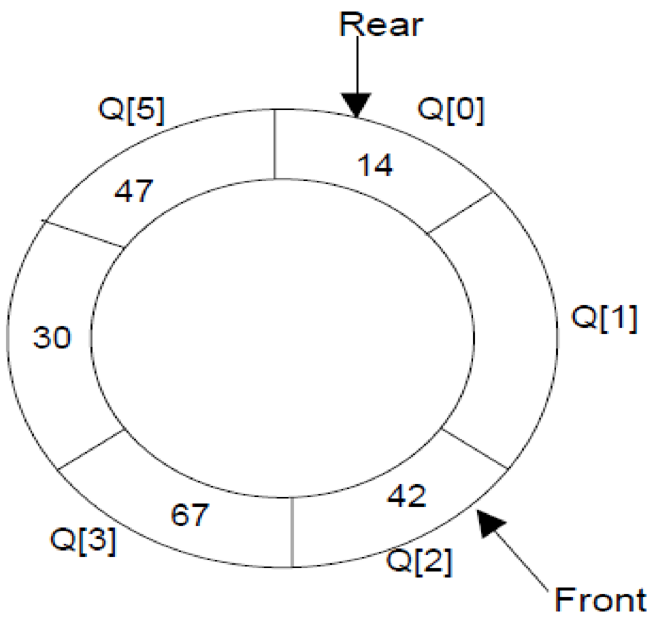

Fig. 4.13. A circular queue after pushing 30, 47, 14

At any time the position of the element to be inserted will be calculated by the relation Rear = (Rear + 1) % SIZE

After deleting an element from circular queue the position of the front end is calculated by the relation Front= (Front + 1) % SIZE

After locating the position of the new element to be inserted, rear, compare it with front. If (rear = front), the queue is full and cannot be inserted anymore.

CIRCULAR QUEUE ALGORITHMS

Let Q be the array of some specified size say SIZE. FRONT and REAR are two pointers where the elements are deleted and inserted at two ends of the circular queue. DATA is the element to be inserted.

Inserting an element to circular Queue

```
1. If [(rear+1) % Size = = front]
      (a) Display “Queue is full”
      (b) Exit
2. Input the value to be inserted and assign to variable “DATA”
3. If (front is equal to – 1)
      (a) front = 0
4. rear = (rear + 1) % Size
5. Q[rear] = DATA
6. Exit
```

Deleting an element from a circular queue

```
1. If (FRONT is equal to – 1)
      (a) Display “Queue is empty”
      (b) Exit
2. If (REAR is equal to FRONT)
      (a) FRONT = –1
      (b) REAR = –1
3. Else
      (a) DATA = Q[FRONT]
      (b) FRONT = (FRONT +1) % SIZE
4. Exit
```

(HW) Displaying the elements in a circular queue