Lecture - 25

Queues

- Is a linear list of elements
- Deletion take place only at one end called the front
- Insertion take place only at other end called the rear.
- Queues are also called first-in-first-out (FIFO)
- The order in which elements enter a queue is the order in which they leave.

eg → people waiting in a line at a Bank
→ Movie Ticket Counter
→ Railway Ticket Counter.
→ Time sharing system → in operating system in which programs with same priority form a queue while waiting to be executed. Called priority queue

Queues may be represented in computer by
- one-way list
- Linear Arrays.

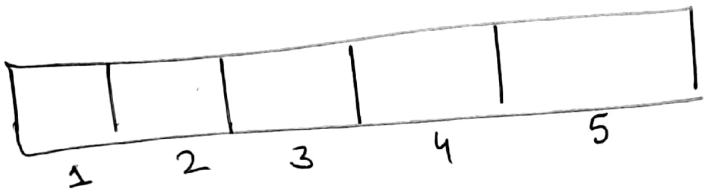
Array Representation of a Queue :-

- If FRONT = NULL || queue is empty
- When an element is added to the queue, the value of Rear increased by 1

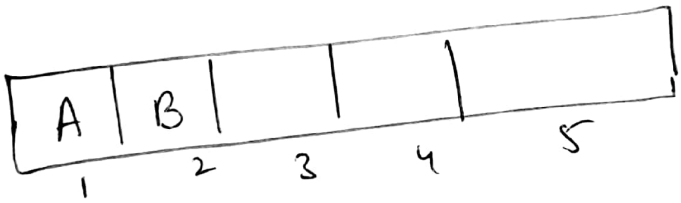
  REAR = REAR +1

- When an element is deleted from the queue, then the value of FRONT is increased by 1

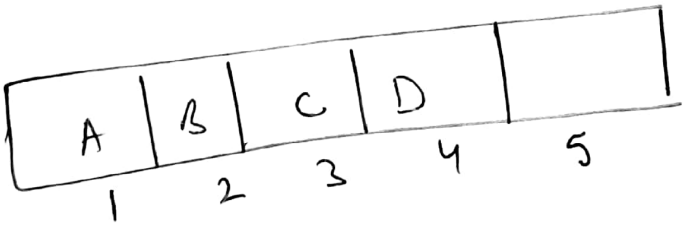
  FRONT = FRONT +1



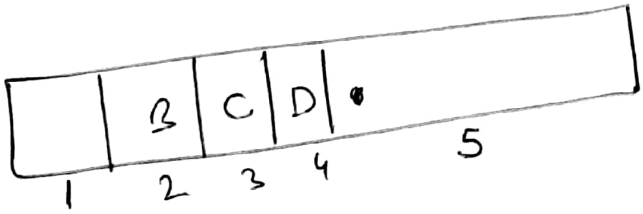
FRONT = NULL
REAR = NULL



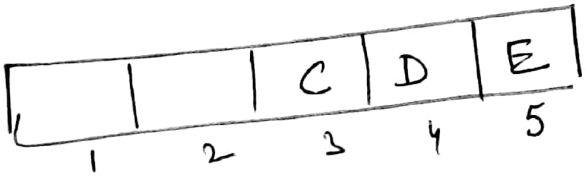
FRONT = 1
REAR = 2



FRONT = 1
REAR = 4



FRONT = 2
REAR = 4



FRONT = 3
REAR = 5

- Assume Array Queue as circular queue.

i.e if REAR = N
and we want to insert another item
then we reset REAR = 1 || since queue is circular.

- Similarly, if FRONT = N and an element of Queue is to be deleted, then we reset

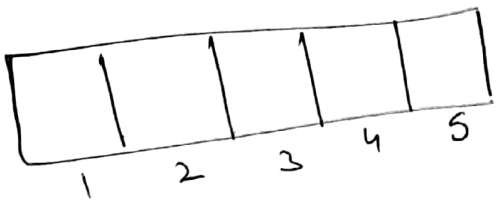
FRONT = 1 instead of FRONT to N+1.

∵ queue is circular.

- Suppose an Queue Contain only 1 element

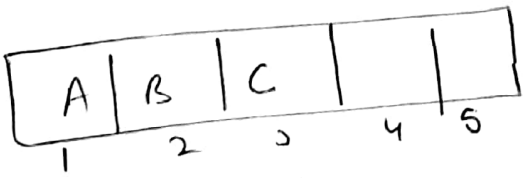
i.e FRONT = NULL
& REAR = NULL] If we delete it so we reset them as shown.

Example



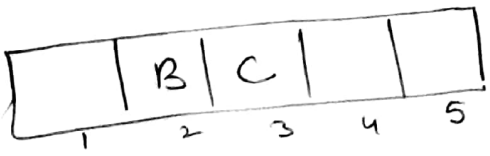
FRONT = 0
REAR = 0

Insert A, B, C



FRONT = 1
REAR = 3

Delete A



FRONT = 2
REAR = 3

Insert D and E

FRONT = 2
REAR = 5

	B	C	D	E
1	2	3	4	5

Delete B & C

FRONT = 4
REAR = 5

			D	E
1	2	3	4	5

Insert F

FRONT = 4
REAR = 1

see here our queue behave as circular queue

F			D	E
1	2	3	4	5

Delete D

FRONT = 5
REAR = 1

F				E
1	2	3	4	5

Delete E

FRONT = 1
REAR = 1

F				
1	2	3	4	5

Delete F

FRONT = 0
Rear = 0

or FRONT = NULL
REAR = NULL

1	2	3	4	5

This procedure will inserts an item into an queue.

QINSERT ( )

{

1. If FRONT = 1 and REAR = N then: // Queue is already filled.
   Write : overflow & return

2. If FRONT = NULL then: // Queue is intially empty.
   Set FRONT = 1 and Rear = 1

   Else if REAR = N then
   Set REAR = 1

   Else
   Set REAR = REAR + 1

3. Set Queue [REAR] = item // Insert new element
   name of array.

4. Return.

This procedure deletes an element from a queue and assigns it to the variable ITEM

1. If ~~front~~ FRONT = NULL then: || Queue already empty.
   Write Underflow & return

2. Set ITEM = Queue [FRONT] || Assign the value to the variable ITEM

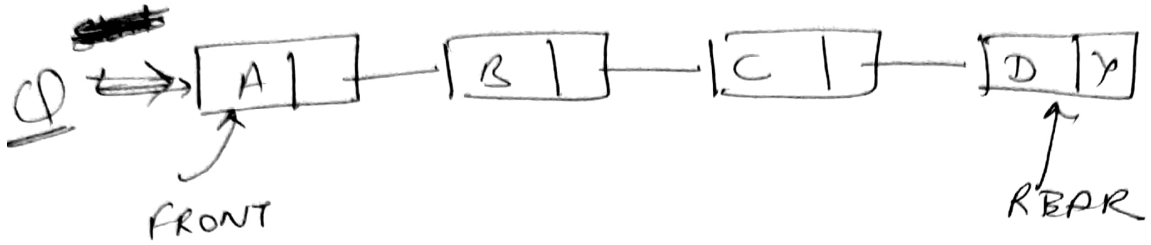
3. If FRONT = REAR then: || Queue has only one element to start.
   Set FRONT = NULL
   and REAR = NULL

   Else if FRONT = N then:
   Set FRONT = 1

   Else
   Set FRONT = FRONT + 1

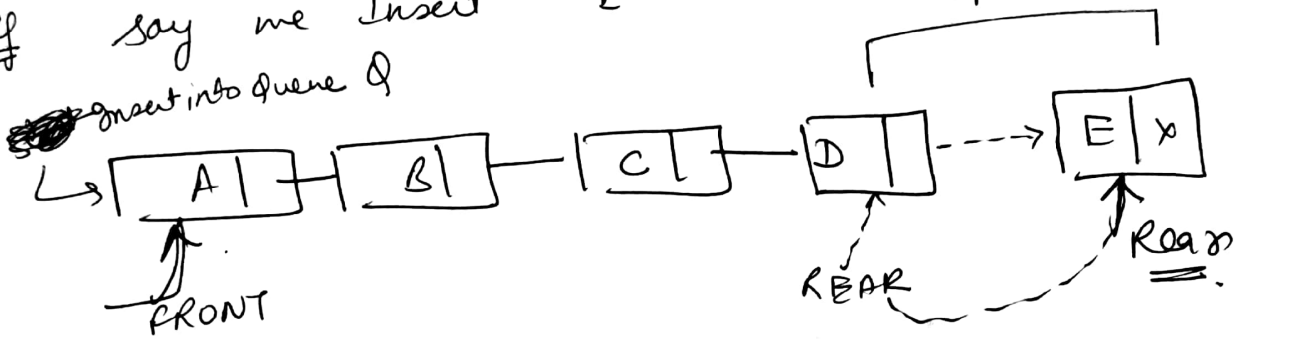
4. Return.

Linked Representation of Queues :-

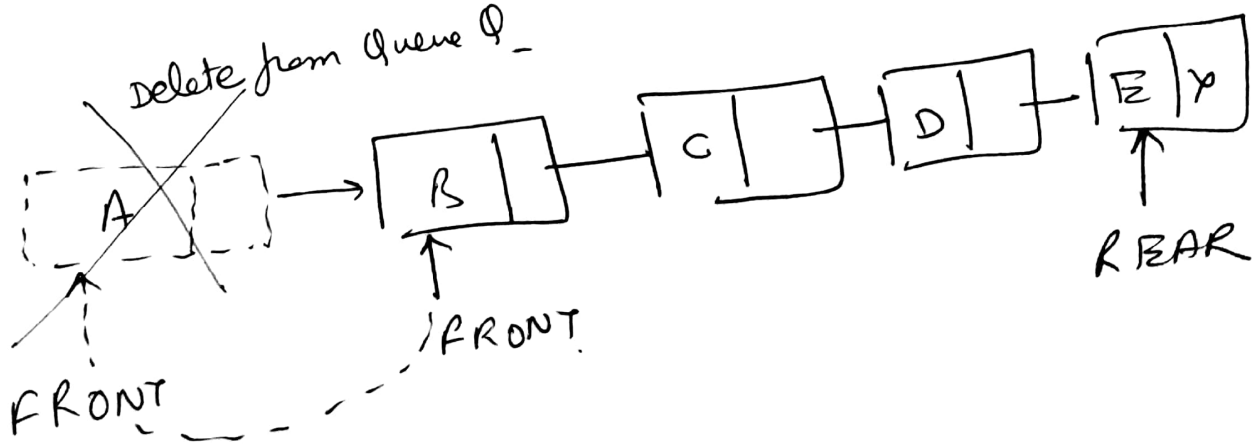


- Item to be added as the last node of the linked list. The REAR pointer is updated to point to the last node just added to the list.

eg say we Insert "E" insert into Queue Q



- In case of deletion, the first node of the list pointed by FRONT is deleted. & FRONT pointer is updated to point to the next node.



The procedure insert an ITEM in a linked queue

1. Create a new node with help of pointer say Temp either using malloc() or new function.

2. Read the item into the new node
   i.e INFO[Temp] = ITEM
   and LINK[Temp] = NULL

3. If FRONT = NULL then: // Queue is empty then insert Temp containing item as first node of the Queue Q.
   FRONT = REAR = Temp

   Else set LINK[REAR] = Temp // Rear points to new node appended to the end of list.
   and REAR = Temp.

4. Exit

This procedure deletes ~~from~~ the front element of the linked queue & stores it in ITEM.

1. If FRONT = NULL then || linked queue is empty.
Write Underflow & Exit

2. Set Temp = FRONT || assign a pointer at the first node.

3. ITEM = INFO [Temp] || saves the value of first node

4. FRONT = LINK [Temp] || Reset FRONT to point to next node
(or) FRONT = LINK [FRONT]

5. Free (temp) ;

6. Exit