Types of Queue:-

There are four types of Queue:-

(i) Linear Queue.
(ii) Circular Queue
(iii) Priority Queue
(iv) Deque (Doube ended Queue).

(i) Linear Queue:-

- A linear Queue, follow the (first in first Out), much like a line in real life.
- In Linear Queue, an insertion takes place from one end while deletion occurs from another end.

Insertion ← FRONT
Deletion ← REAR

- Linear Queue can be implemented with the help of a linear array of one dimension, thats why it is called linear Queue.

Operations :- → Enqueue
→ Dequeue

Explanation :- (Enqueue).

Initial

Index	1	2	3	4	5
values					

FRONT = 0
Rear = 0

Insert "A"

Index	1	2	3	4	5
values	A				

Front = 1
Rear = 1

Insert "B", "C", "D"

Index	1	2	3	4	5
values	A	B	C	D	

Front = 1
rear = 4

Insert "E"

Index	1	2	3	4	5
Values	A	B	C	D	E

Front = 1
Rear = 5

Insert "F"

Index	1	2	3	4	5
values	A	B	C	D	E

Front = 1
Rear = 5

(print "Overflow" as queue is full now.

Explanation (Dequeue):-

Index	1	2	3	4	5
values	A	B	C	D	E

Front 1
Rear 5

Delete

item = A

Index	1	2	3	4	5
values		B	C	D	E

Front 2
Rear 5

Delete

Item B

Index	1	2	3	4	5
values			C	D	E

Front 3
Rear 5

Delete

Item C

Index	1	2	3	4	5
Values				D	E

Front 4
Rear 5

Delete

Item D

Index	1	2	3	4	5
values					E

Front 5
Rear 5

Delete

Item E

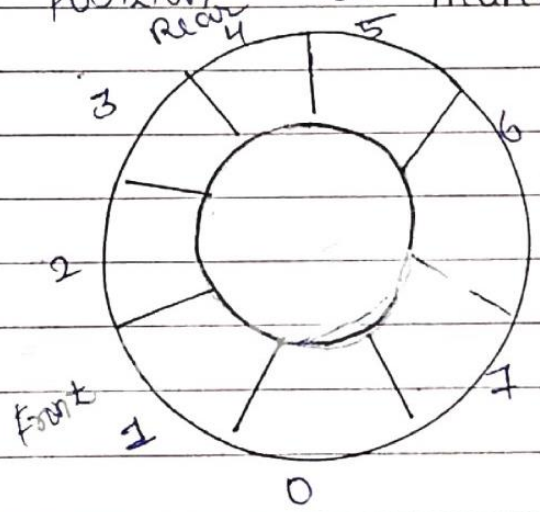
Index	1	2	3	4	5
values					

Front 0
Rear 0

STUDYBYNOTES

2. Circular Queue:-

- Circular Queue is a linear data structure.
- In this, operations are performed based on FIFO (first in first out) principle and last position is connected back to the first position to make a circle.



- In a normal Queue, we can insert elements until Queue becomes full. But once queue becomes full, we cannot insert the next element even if there is a space in front of queue.

FRONT = 0
REAR = 0

1	2	3	4

(a) Initially empty Queue

FRONT = 1
REAR = 2

1	A	2	B	3		4	

FRONT=1	A	B	C	D
REAR=4	1	2	3	4

(c) After insertion of C and D in queue

FRONT=2		B	C	D
REAR=4	1	2	3	4

(D) After ... of deletion from queue

FRONT=3			C	D
REAR=4	1	2	3	4

(e) After deletion from queue

FRONT=3	E		C	D
REAR=1	1	2	3	4

(f) After insertion of E

FRONT=4	E			D
REAR=1	1	2	3	4

(G1) After deletion from queue

FRONT=4	E			
REAR=2	1	2	3	4

(h) After insertion of G1

FRONT=1	E	G1		
REAR=2	1	2	3	4

(i) After deletion

FRONT=2		G1		
REAR=2	1	2	3	4

(j) After deletion from queue

FRONT=0				
REAR=0	1	2	3	4

(K) After deletion from queue

* Algorithm

CIR_QINSERT (Q, N, FRONT, REAR, ITEM).

Step 1: if (FRONT = 1 and REAR = N) or (FROIVT = REAR+1)
Then
Write "overflow" and Exit.

Step 2: IF FRONT = NULL Then
Set FRONT = 1 and REAR = 1

Else If REAR = N Then
Set REAR := 1
Else
Set REAR := REAR+1

Step3:- Set Q[REAR] := ITEM

Step 4:- Exit.

(... Circular Queue deletion)

Algorithm CIR_QDELETE(Q, N, FRONT, REAR, ITEM).

Step 1:- IF FRONT = NULL Then
Write "Underflow" and Exit
[End if]

Step 2:- Set ITEM = Q [FRONT]

Step 3:- IF FRONT = REAR Then
Set FRONT = REAR = NULL

Else if FRONT = N Then
Set FRONT := 1

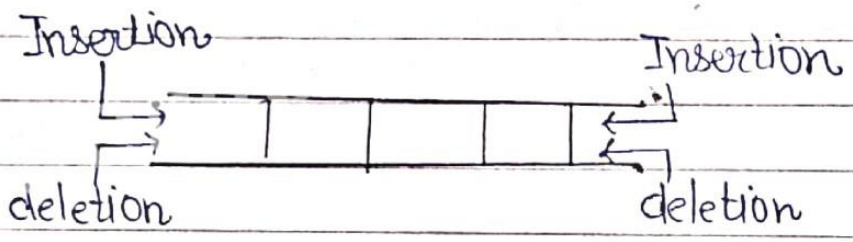
Else
Set FRONT = FRONT + 1
[End IF]

Step 4:- Exit

3. DEQUE (Double Ended Queue):-

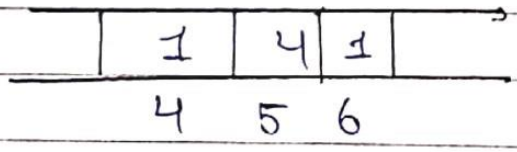
- In this insertion and deletion can be performed at both end i.e. front pointer can be used for insertion and rear pointer can be used for deletion.

- elements can be added and removed at both ends (front and rear), but not in middle.

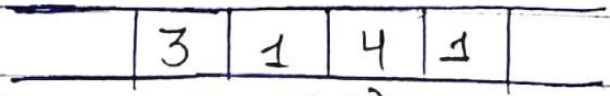


Example:

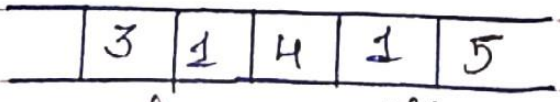
front = 4
Rear = 6



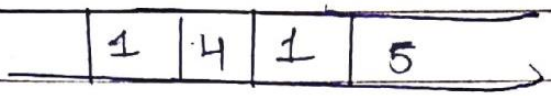
→ Perform Enque front(3)



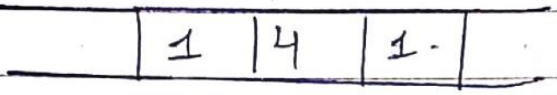
→ Perform Enque rear(5)



→ perform Deque front will delete 3



→ Performing Deque rear will delete 5.

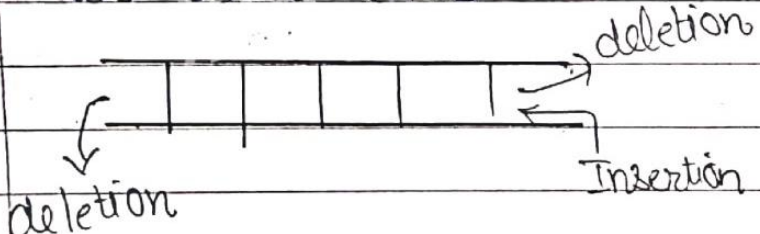


Variations

Input Restricted Queue

→ Insertion at one end (rear)

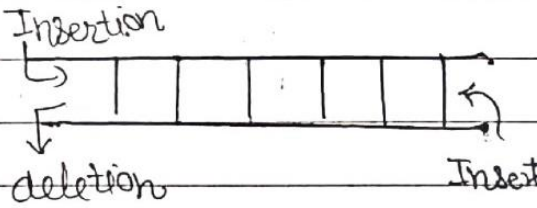
→ deletion can be performed at both ends.



Output Restricted Queue

→ deletion at one end (front)

→ Insertion can be Performed at both ends



(4) Priority Queue :-

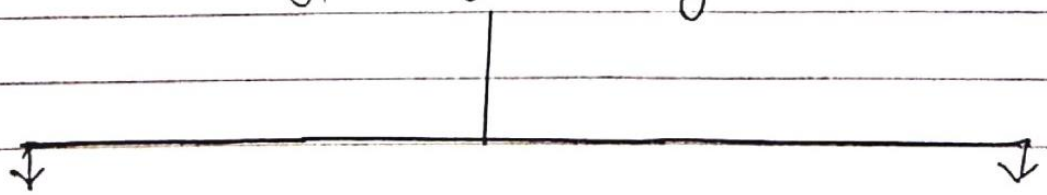
→ In this, Each Element is inserted or deleted on the basis of their priority.

→ High Priority > Lower Priority

→ Same Priority [FCFS basis]
↓
first come first Serve.

Definition :- A data structure that supports efficent insertions of new element and deletion of elements with the highest priority is known as Priority Queue.

Types of Priority Queue.



Ascending Order Priority Queue

[Lower Priority no. to high Priority).

1
2
3
4
5

Descending order Priority Queue

5
4
3
2
1

[Higher Priority no. to higher Priority)

* Representation of a Priority Queue :-

INFO	PRIORITY	LINK
222	2	6
444	4	4
555	4	9
333	1	1
111	2	3
666	5	8
777	4