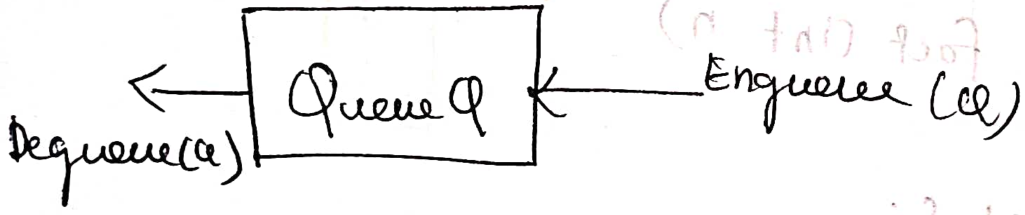
The Queue ADT

Queue is a linear data structure which follows First In First out Principle in which insertion is performed at rear end and deletion is performed at front end.

Model of a queue



operations on Queue

Enqueue :-

which inserts an element at the end of the list (called the rear).

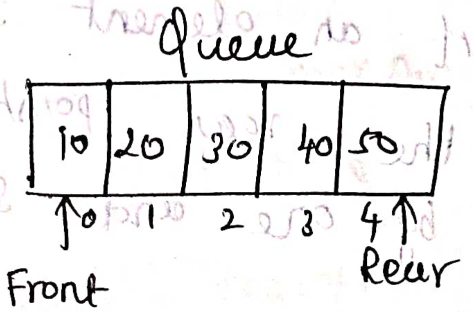
Dequeue :-

which deletes the element at the start of the list (known as front)

Exceptional Conditions

* overflow:-

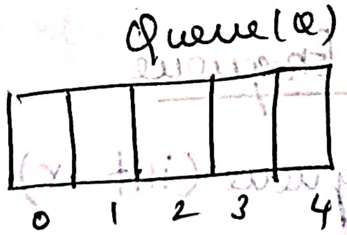
Attempt to insert an element when the queue is full it said to be overflow



Enqueue (60) - it causes overflow

* Underflow

Attempt to delete an element when the queue is empty is said to be under flow.



Front = -1
Rear = -1

Dequeue(Q) causes underflow

Queue Implementation

There are two ways for implementing the queue

* Array Implementation
* Linked list implementation

Array Implementation

There are two pointers used for implementing. one is Rear and other one is Front pointer

* To insert an element 'x' on to the queue (Q). the rear pointer is incremented by one and said

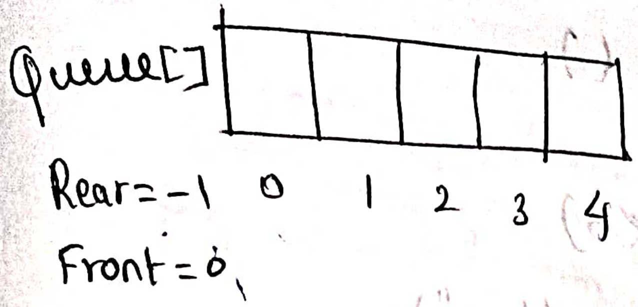
Queue [rear] = x;

* To delete an element Q [Front] is returned and the front pointer is incremented

Routine to Enqueue

```
Void enqueue (int x)
{
    if (rear > max_Arraysize)
    Printf ("Queue overflow");
    else
    {
        rear = rear + 1;
        Queue [rear] = x;
    }
}
```

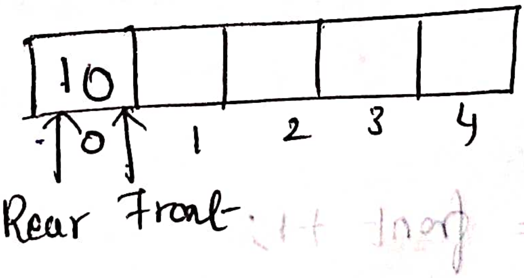
Example :-



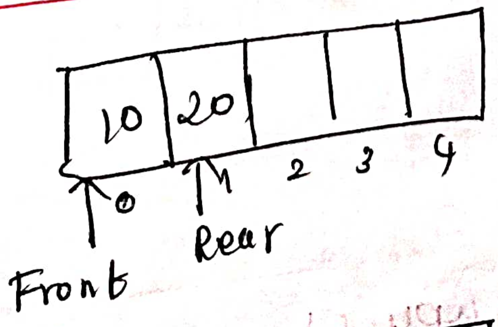
Enqueue (10)

```
if (rear > max Array size)
        -1 > 5 ✗
else
    rear = rear+1
    rear = -1+1 = 0
```

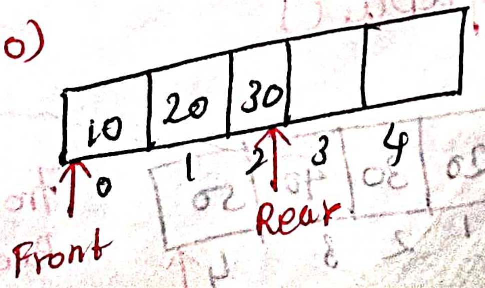
$$\boxed{\text{Queue}[0] = 10}$$



Enque (20)



Enqueue(30)



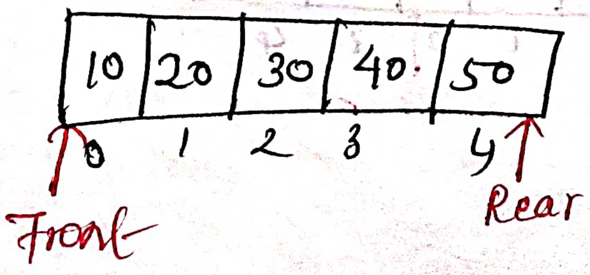
Dequeue (Q)

```
Void delete ()
{
  if (front <0)
  printf ("underflow");
  else
  {
   x = queue [front];
   if (front == rear)
   {
     front =0;
     rear = -1;

   }
  else
  front = front +1;
 }
 return x;
}
```

Example

① Dequeue ()

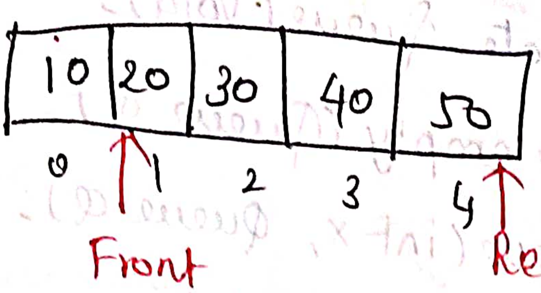


Front = 0
Rear = 4

$x = 10$

front = front + 1

$\boxed{\text{Front} = 0 + 1 = 1}$



(2) Dequeue ( )

Front = 1

Rear = 4

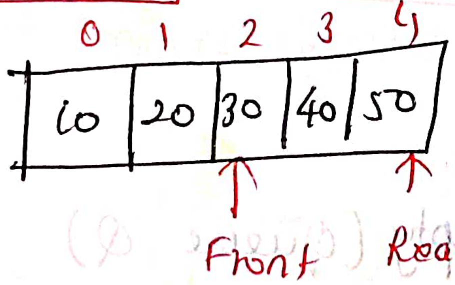
$x = \text{Queue}[1]$

$x = 20$

front = front + 1

= 1+1

$\boxed{\text{Front} = 2}$



Front = 2

Rear = 4

Linked List Implementation of Queue

Enqueue operation is performed at end of the list. Dequeue operation is performed at front of the list.