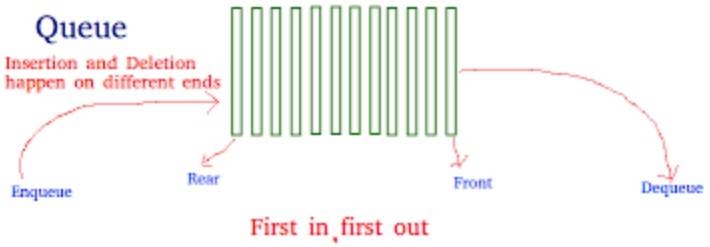
Queue

Queue : Queue is also an abstract data type or a linear data structure, just like stack data structure, in which the first element is inserted from one end called the REAR(also called tail), and the removal of existing element takes place from the other end called as FRONT(also called head).



Operations on Queue

a. **enqueue()** − This function defines the operation for adding an element into queue.

b. **dequeue()-** This function defines the operation for removing an element from queue.

c. **init()-** This function is used for initializing the queue.

d. **peek()** − Gets the element at the front of the queue without removing it.

e. **isfull()** − Checks if the queue is full.

f. **isempty()** − Checks if the queue is empty.

Types of Queue

1. **Simple queue** : Simple queue defines the simple operation of queue in which insertion occurs at the rear of the list and deletion occurs at the front of the list.

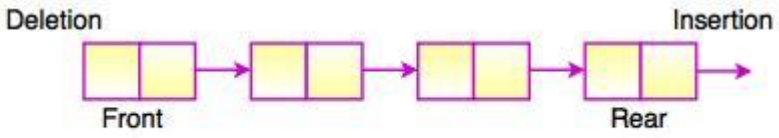


Fig. Simple Queue

2. **Circular Queue** : Circular Queue is a linear data structure in which the operations are performed based on FIFO (First In First Out) principle and the last position is connected back to the first position to make a circle. It is also called 'Ring Buffer'.

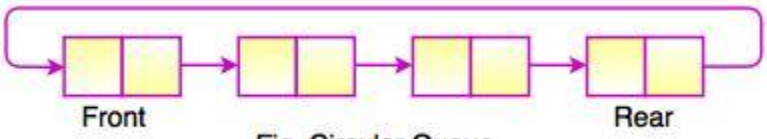


Fig. Circular Queue

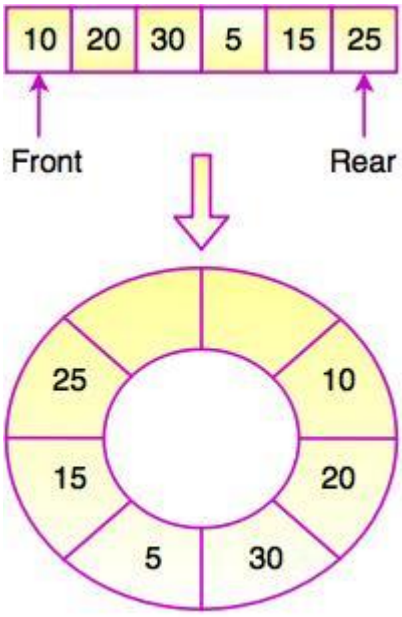


Fig. Circular Queue

3. **Priority Queue**: Priority queue is a collection of elements such that each element has been assigned a priority and the order in which elements are deleted and processed comes from following rules:
   - An element of higher priority is processed before any elements of lower priority
   - Two elements with the same priority are processed according to the order in which they were added to the queue.
   - An example of priority queue in computer science occurs in timesharing system in which the processes of higher priority are executed before any process of lower priority.
4. **Double Ended Queue**: It is also a homogeneous list of elements in which insertion and deletion operations are performed from both the ends. That is, we can insert elements

from the rear end or from the front end. Hence, it is called Double-Ended Queue. There are two types of deques.

These two types are due to the restrictions put to perform either the insertions or deletions only at one end.

- **Input restricted deques** : Input restricted deques allows insertions at only one end of the array or list but deletions allows at both ends.
- **Output restricted deques** : Output restricted deques allows deletions at only one end of the array or list but insertions allow at both ends.

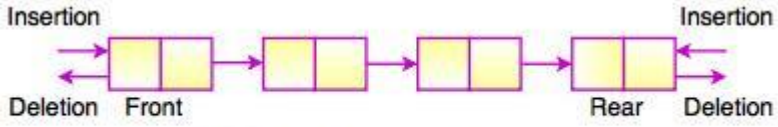


Fig. Double Ended Queue (Dequeue)

Applications of Queue

1. When a resource is shared among multiple consumers. Examples include CPU scheduling, Disk Scheduling.

2. Serving requests on a single shared resource, like a printer, CPU task scheduling etc.

3. Handling of interrupts in real-time systems. The interrupts are handled in the same order as they arrive i.e First come first served.

4. In real life scenario, Call Center phone systems uses Queues to hold people calling them in an order, until a service representative is free.

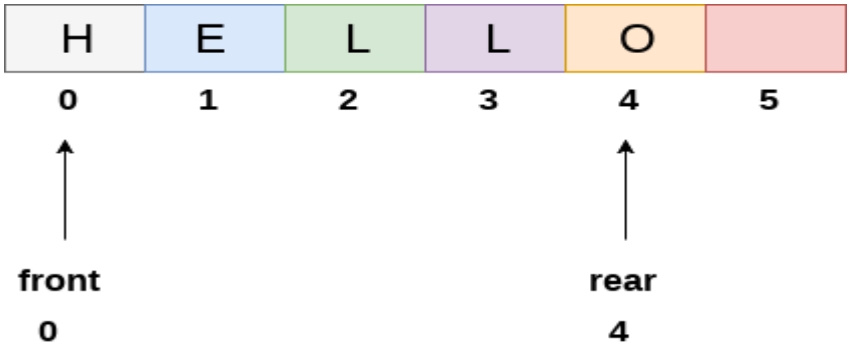
Representation of Queue:

1. **Array representation of Queue:**

We can easily represent queue by using linear arrays. There are two variables i.e. front and rear that are implemented in the case of every queue. Front and rear variables point to the position

from where insertions and deletions are performed in a queue. Initially, the value of front and queue is -1 which represents an empty queue.

Array representation of a queue containing 5 elements along with the respective values of front and rear, is shown in the following figure.



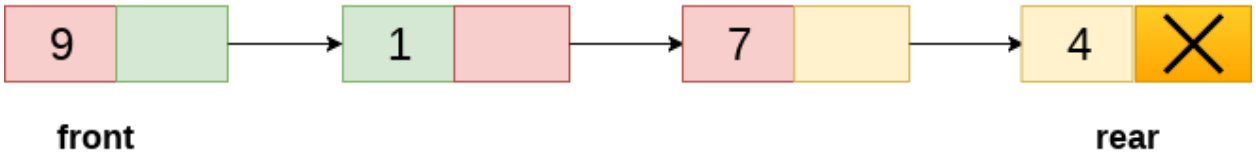
Queue

2. **Linked list Representation**

In the linked queue, there are two pointers maintained in the memory i.e. front pointer and rear pointer. The front pointer contains the address of the starting element of the queue while the rear pointer contains the address of the last element of the queue.

Insertion and deletions are performed at rear and front end respectively. If front and rear both are NULL, it indicates that the queue is empty.

The linked representation of queue is shown in the following figure.



Linked Queue