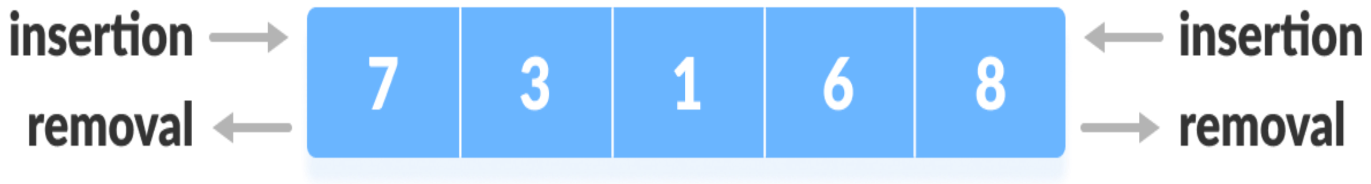
Types of Queues

Deque Data Structure

Deque or Double Ended Queue is a type of queue in which insertion and removal of elements can either be performed from the front or the rear. Thus, it does not follow FIFO rule (First In First Out).



Types of Deque

- **Input Restricted Deque**
  In this deque, input is restricted at a single end but allows deletion at both the ends.
- **Output Restricted Deque**
  In this deque, output is restricted at a single end but allows insertion at both the ends.

Applications of Deque Data Structure

- In undo operations on software.
- To store history in browsers.

Priority Queue

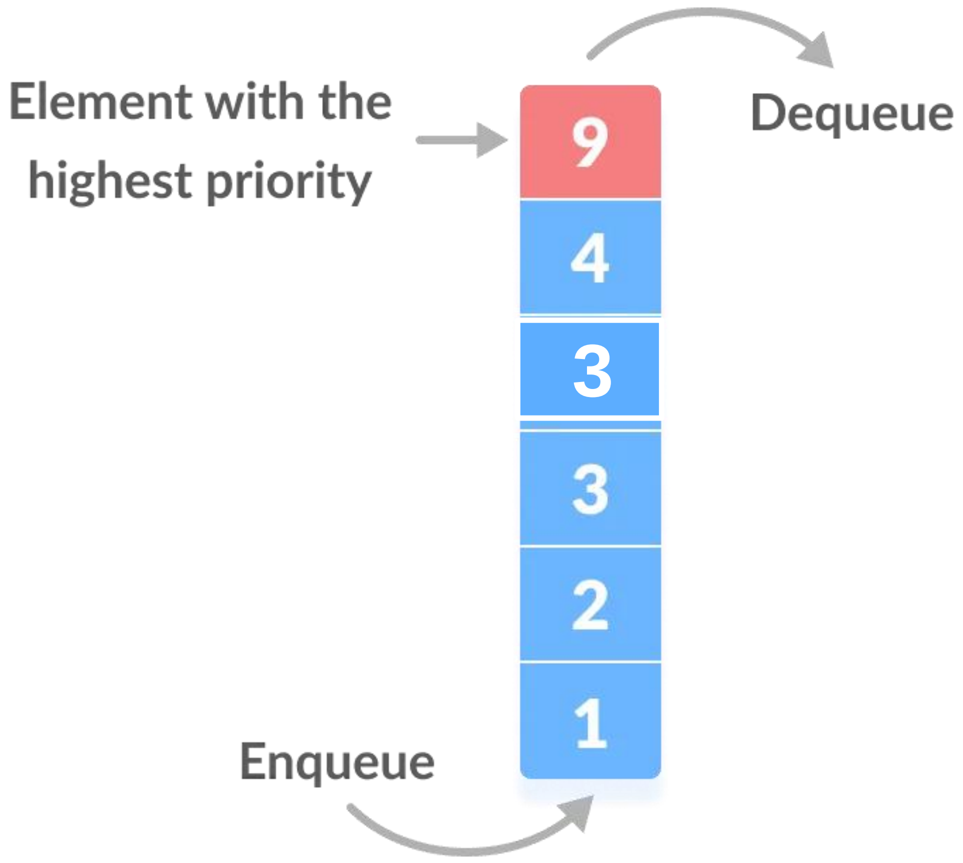
- A priority queue is a **special type of queue** in which each element is associated with a **priority value**.
- The elements are served on the basis of their priority. That is, higher priority elements are served first.
- However, if elements with the same priority occur, they are served according to their order in the queue.

Priority Queue

- **Assigning Priority Value**
  - Generally, the value of the element itself is considered for assigning the priority. For example,
    - The element with the highest value is considered the highest priority element. However, in other cases, we can assume the element with the lowest value as the highest priority element.
  - We can also set priorities according to our needs.

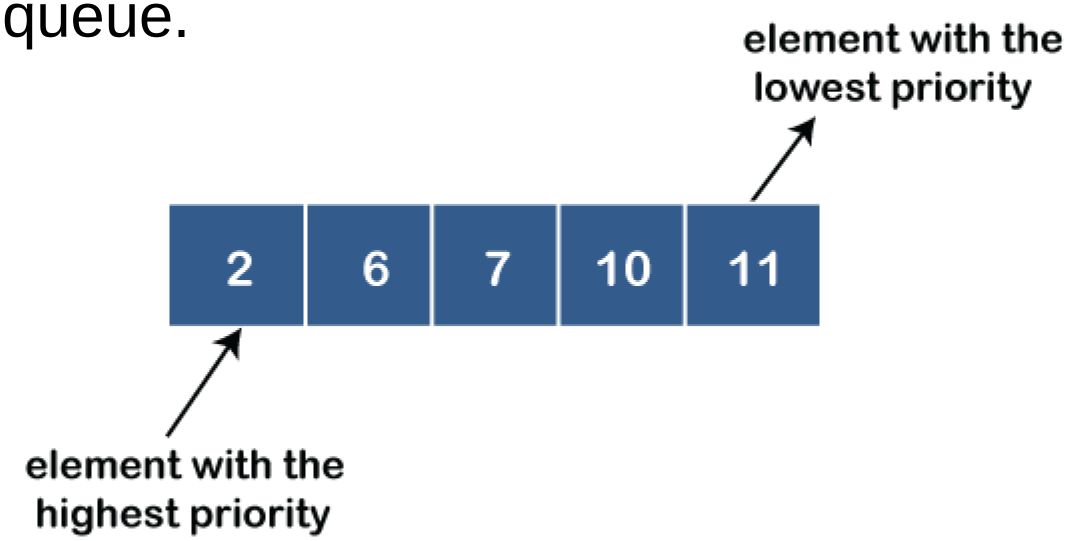
Priority Queue

- **Assigning Priority Value**



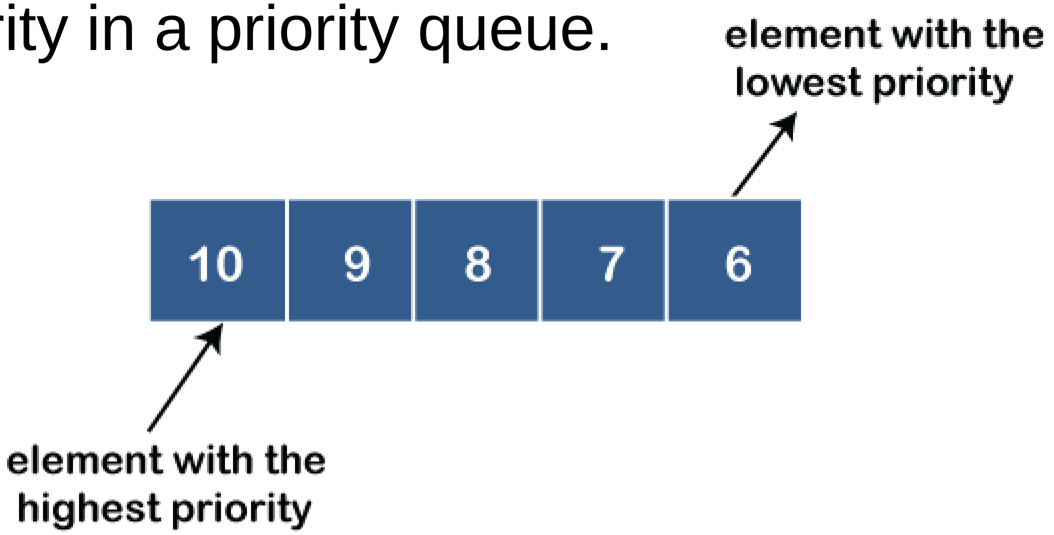
Types of Priority Queue

- **There are two types of priority queue:**
- **Ascending order priority queue:** In ascending order priority queue, a lower priority number is given as a higher priority in a priority.
- For example, we take the numbers from 1 to 5 arranged in an ascending order like 1,2,3,4,5; therefore, the smallest number, i.e., 1 is given as the highest priority in a priority queue.


element with the lowest priority
2
6
7
10
11
element with the highest priority

Types of Priority Queue

- **There are two types of priority queue:**
- **Descending order priority queue:** In descending order priority queue, a higher priority number is given as a higher priority in a priority.
- For example, we take the numbers from 1 to 5 arranged in descending order like 5, 4, 3, 2, 1; therefore, the largest number, i.e., 5 is given as the highest priority in a priority queue.


element with the lowest priority
10
9
8
7
6
element with the highest priority

Priority Queue

- **Difference between Priority Queue and Normal Queue**
  - In a queue, the **first-in-first-out rule** is implemented whereas, in a **priority queue**, the values are inserted **on the basis of priority**.
  - The element with the highest priority is removed first in Priority Queue.

Operations on Priority Queue

- Poll() - This function will remove the highest priority element from the priority queue.
- Add(value) - This function will insert ‘value' element in a priority queue. (Each element is inserted into a priority queue, conceptually it is inserted *in order of* its priority)

Example

- A priority queue with the following values:

  1, 3, 4, 8, 14, 22

- All the values are arranged in ascending order.

Example

- A priority queue with the following values:

  1, 3, 4, 8, 14, 22

- Observe how the priority queue will look after performing the following operations:
- **poll():** In the above priority queue, the '1' element has the highest priority, so it will be removed from the priority queue.

  3, 4, 8, 14, 22

Example

- Priority queue:

  3, 4, 8, 14, 22

- **add(2):** This function will insert '2' element in a priority queue. As 2 is the smallest element among all the numbers so it will obtain the highest priority.

  2, 3, 4, 8, 14, 22

Example

- Priority queue:

  2, 3, 4, 8, 14, 22

- **poll():** It will remove '2' element from the priority queue as it has the highest priority queue.

  3, 4, 8, 14, 22

Example

- Priority queue:

  3, 4, 8, 14, 22

- **add(5):** It will insert 5 element after 4 as 5 is larger than 4 and lesser than 8, so it will obtain the third highest priority in a priority queue.

  3, 4, 5, 8, 14, 22

Example

- Priority queue:

  3, 4, 5, 8, 14, 22

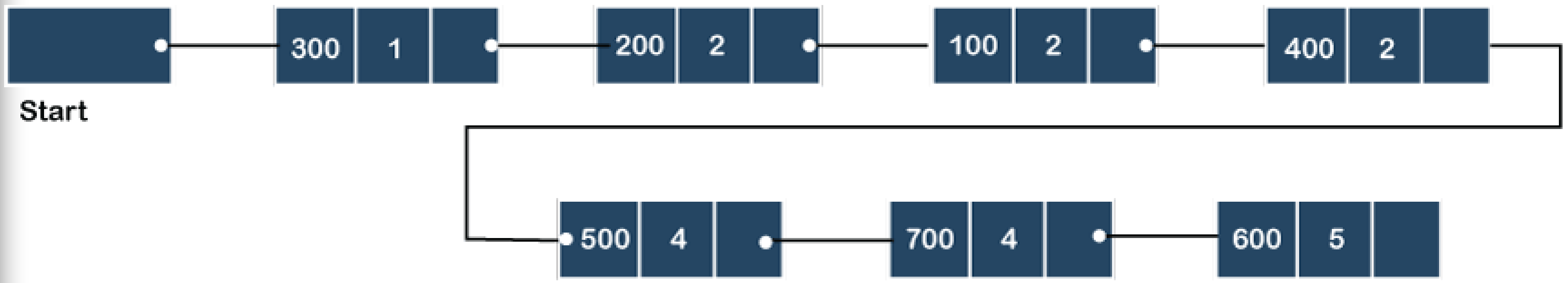
- **add(8):** It will insert 8 element after 8 and before 14 as 8 is already existing in the queue. In such a case First come First Serve strategy will be followed and duplicate priority will be inserted after all the original duplicate value(s)

  3, 4, 5, 8, 8, 14, 22

Implementation of Priority Queue

Implementation of Priority Queue

- Using Linked List for **priority queue, where lower priority number is considered the higher priority,** **i.e.,** lower priority number = higher priority



Implementation of Priority Queue

- Using Heap Data Structure for **priority queue**.

Implementation of Priority Queue

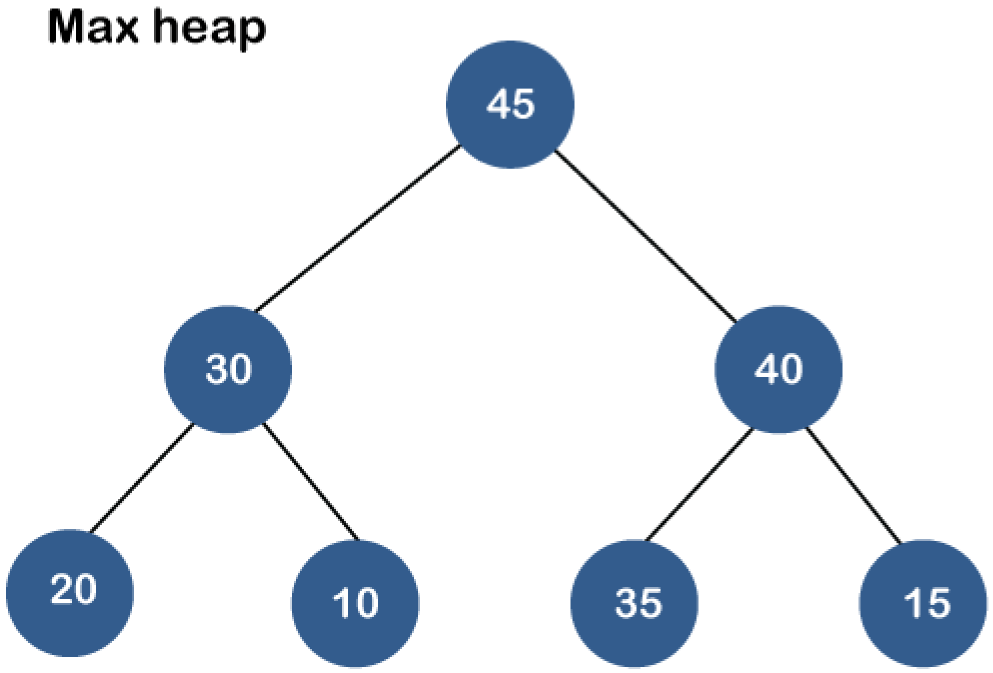
- **What is Heap?**
- A heap is a tree-based data structure that forms a complete binary tree, and satisfies the heap property.
- If A is a parent node of B, then A is ordered with respect to the node B for all nodes A and B in a heap.
- It means that the value of the parent node could be more than or equal to the value of the child node, **or** the value of the parent node could be less than or equal to the value of the child node.

What is Heap?

- Therefore, we can say that there are two types of heaps:
  - Max Heap
  - Min Heap
- Both the heaps are the binary heap, as each has exactly two child nodes.

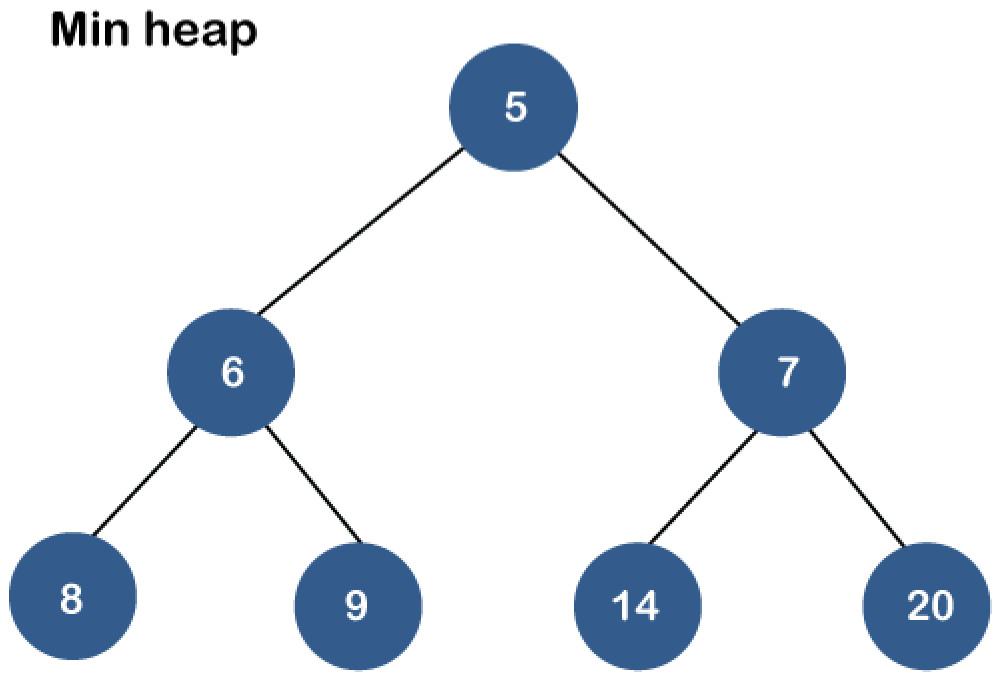
Max Heap

- The max heap is a heap in which the value of the parent node is greater than the value of the child nodes.



Min Heap

- The min heap is a heap in which the value of the parent node is less than the value of the child nodes.



Priority Queue Operations

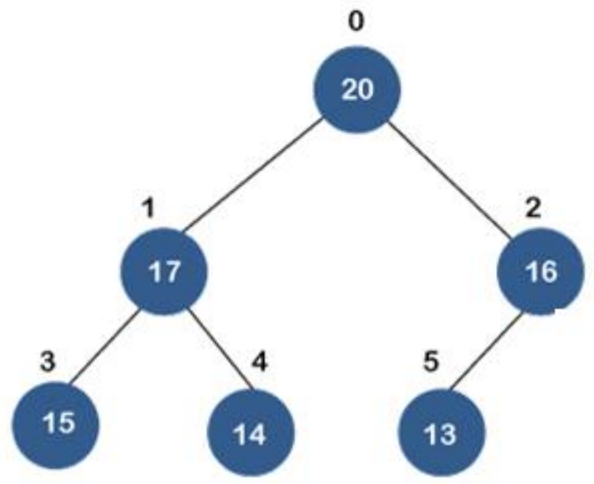
- The common operations that we can perform on a priority queue are insertion, deletion and peek.

Inserting the element in a priority queue (max heap)

- To insert an element in a priority queue, it will move to the empty slot by looking from top to bottom and left to right.
- If the element is not in a correct place then it is compared with the parent node; if it is found out of order, elements are swapped. This process continues until the element is placed in a correct position.

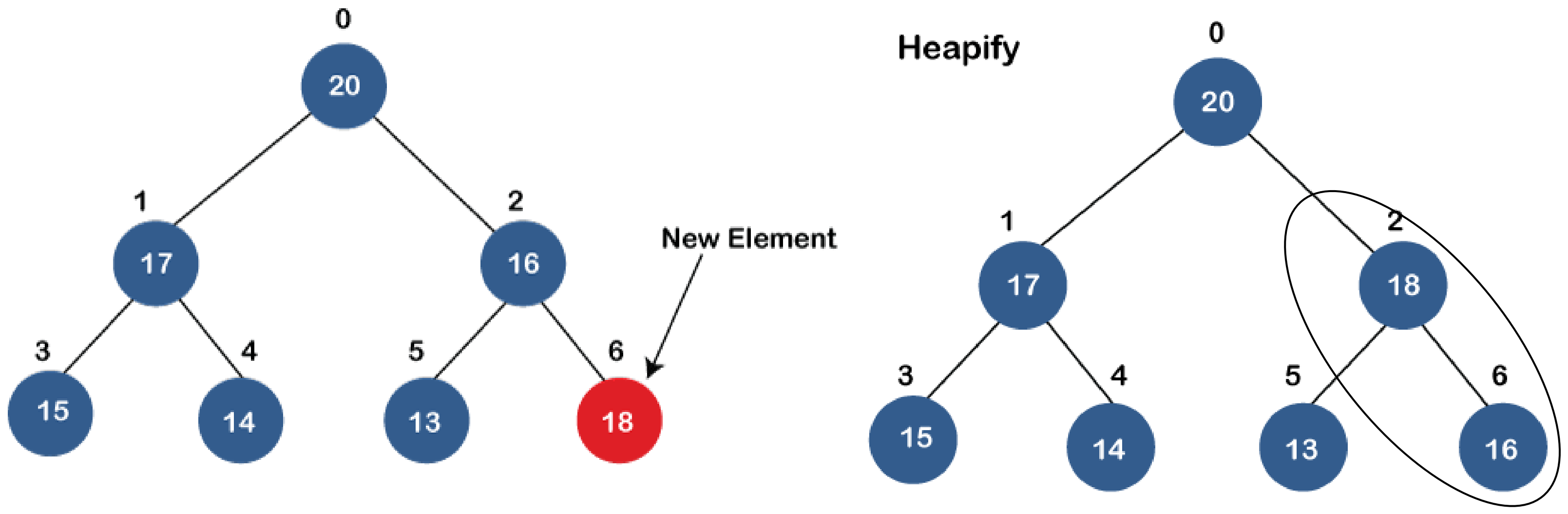
Inserting the element in a priority queue (max heap)

- To insert an element in a priority queue, it will move to the empty slot by looking from top to bottom and left to right.
- If the element is not in a correct place then it is compared with the parent node; if it is found out of order, elements are swapped. This process is known as **HEAPIFICAION** and **heapify** continues until the element is placed in a correct position.



Inserting the element in a priority queue (max heap)

- To insert an element in a priority queue, it will move to the empty slot by looking from top to bottom and left to right.
- If the element is not in a correct place then it is compared with the parent node; if it is found out of order, elements are swapped. This process continues until the element is placed in a correct position.



Removing the element from the priority queue (max heap)

- In a max heap, the maximum element is the root node.
- When we remove the root node, it creates an empty slot.
- The last inserted element will be added in this empty slot.
- Then, this element is compared with the child nodes, i.e., left-child and right child, and swap it if it is smaller than any of the two.
- It keeps moving down the tree until the heap property is restored.

Analysis of complexities using different implementations

Implementation	Add	Remove	Peek
Linked list	O(n)	O(1)	O(n)
Binary heap	O(logn)	O(logn)	O(1)
Binary search tree	O(logn)	O(logn)	O(1)