Sairam INSTITUTIONS

Sri SAI RAM ENGINEERING COLLEGE
INSTITUTE OF TECHNOLOGY
West Tambaram, Chennai - 44

YEAR	SEM
II	III

CS8391

DATA STRUCTURES
(Common to CSE & IT)

UNIT - III

NON LINEAR DATA STRUCTURES - TREES

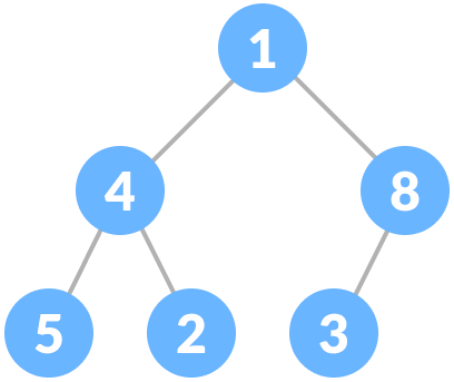
3.7 HEAP APPLICATION - MIN HEAP

4 QUALITY EDUCATION

NBA NATIONAL BOARD OF ACCREDITATION | NAAC (A+) | nirf

HEAP APPLICATIONS – MIN HEAP

Heap data structure is a complete binary tree that satisfies **the heap property**. It is also called as **a binary heap**. A complete binary tree is a special binary tree in which every level, except possibly the last, is filled all the nodes are as far left as possible.



The efficient way of implementing priority queue is Binary Heap (or) Heap.

Heap has two properties :

1. Structure Property.
2. Heap Order Property.

1. Structure Property:

The Heap should be a complete binary tree, which is a completely filled tree, which is a completely filled binary tree with the possible exception of the bottom level, which is filled from left to right. A Complete Binary tree of height H, has between 2h and (2h+1 - 1) nodes.

Sentinel Value : The zeroth element is called the sentinel value. It is not a node of the tree. This value is required because while addition of new node, certain operations are performed in a loop and to terminate the loop, sentinel value is used. Index 0 is the sentinel value. It stores irrelated value, inorder to terminate the program in case of complex codings. Structure Property : Always index 1 should be starting position.

2. Heap Order Property:

The property that allows operations to be performed quickly is a heap order property.

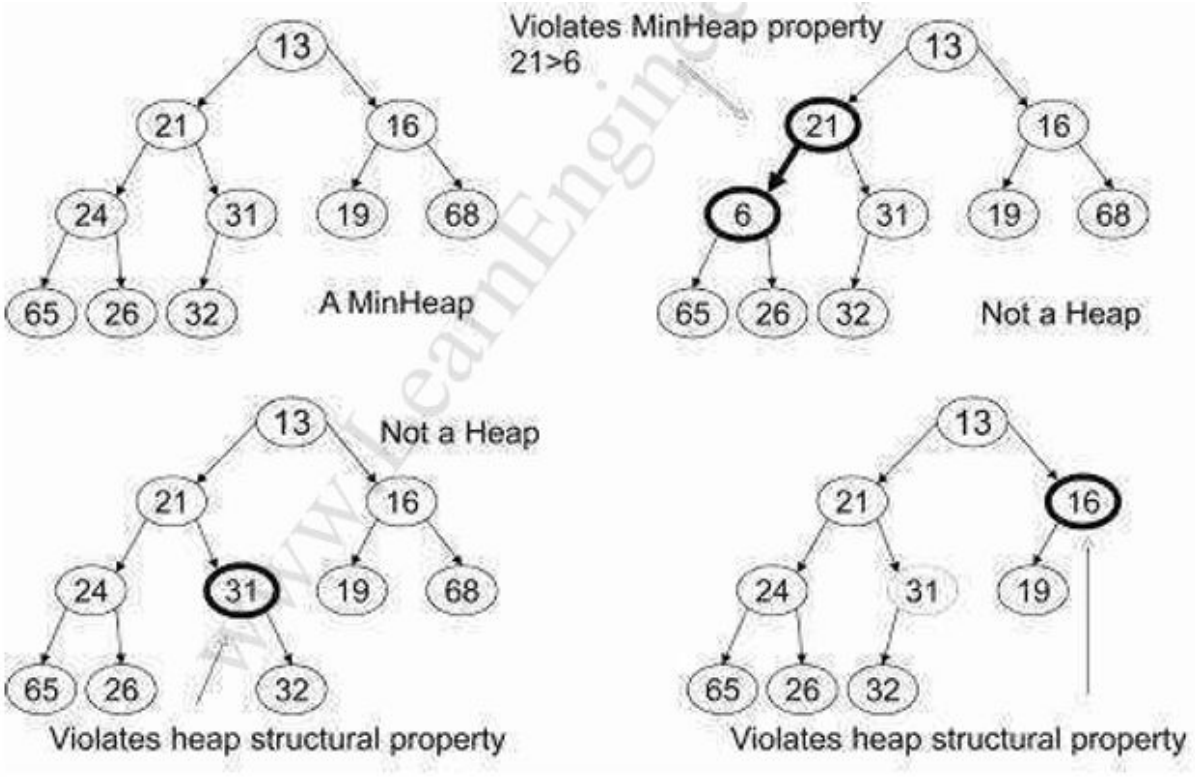
Mintree: Parent should have lesser value than children.

Maxtree: Parent should have greater value than children.

These two properties are known as heap properties

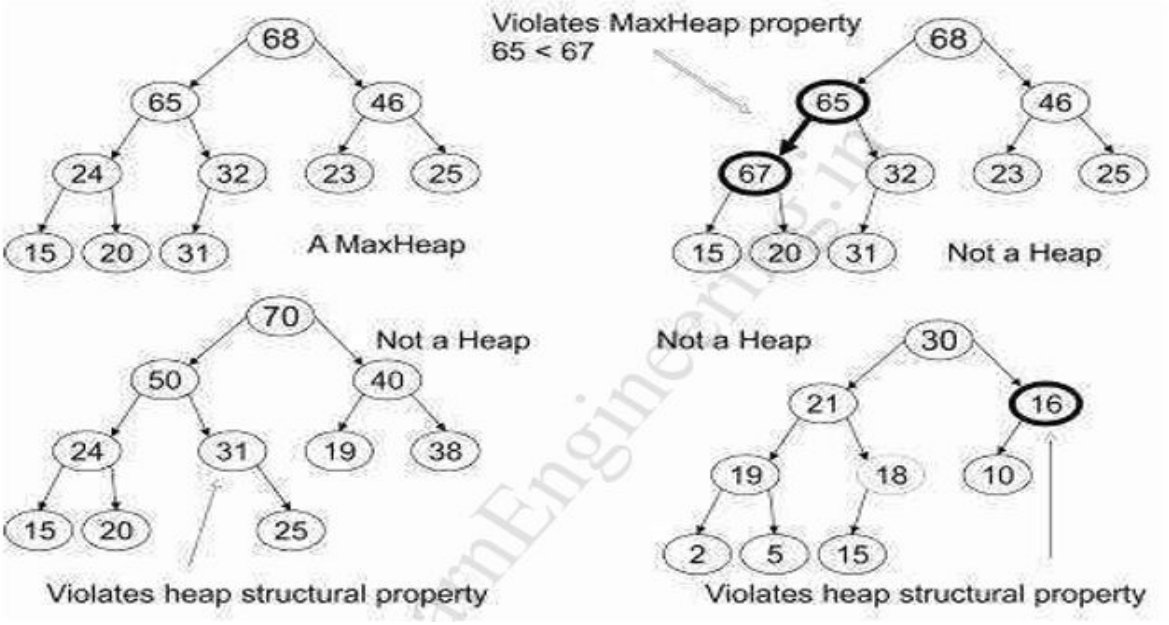
- Max-heap
- Min-heap

Min-heap: The smallest element is always in the root node. Each node must have a key that is less or equal to the key of each of its children. Examples



Max-Heap: The largest Element is always in the root node. Each node must have a key that is greater or equal to the key of each of its children.

Examples



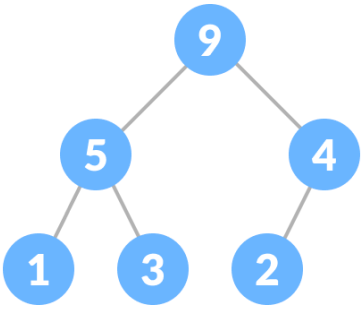
HEAP OPERATIONS:

There are 2 operations of heap

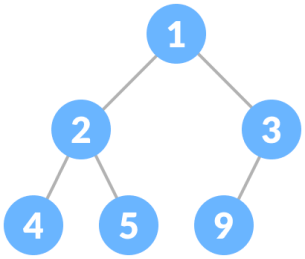
- Insertion - Adding a new key to the heap
- Deletion - Deleting a key from the heap.

Heap Property is the property of a node in which

- (for max heap) key of each node is always greater than its child node/s and the key of the root node is the largest among all other nodes;



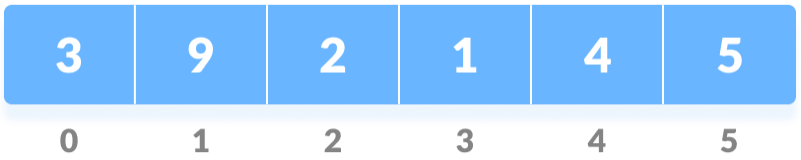
- (for min heap) key of each node is always smaller than the child node/s and the key of the root node is the smallest among all other nodes.



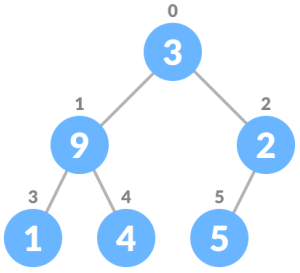
Heapify

Heapify is the process of creating a heap data structure from a binary tree. It is used to create a Min-Heap or a Max-Heap.

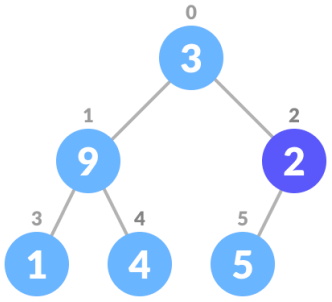
1. Let the input array be



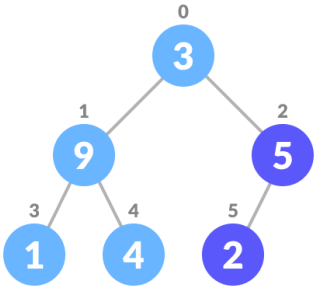
2. Create a complete binary tree from the array



3. Start from the first index of non-leaf node whose index is given by n/2 – 1



4. Set current element i as largest.
5. The index of left child is given by 2i + 1 and the right child is given by 2i + 2.
   If leftChild is greater than currentElement (i.e. element at ith index), set leftChildIndex as largest.
   If rightChild is greater than element in largest, set rightChildIndex as largest.
6. Swap largest with currentElement



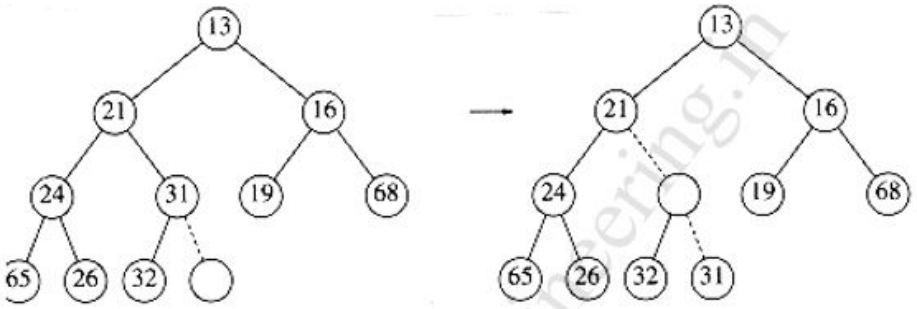
7. Repeat steps 3-7 until the sub trees are also heapified.

Rules for the insertion: To insert an element X, into the heap, do the following:

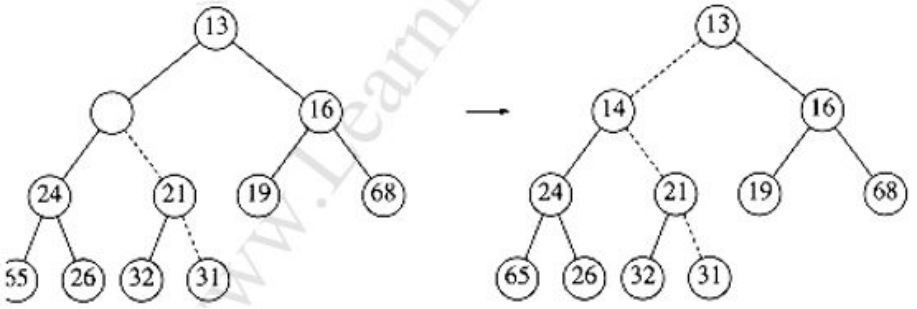
Step1: Create a hole in the next available location , since otherwise the tree will not be complete.

Step2: If X can be placed in the hole, without violating heap order, then do insertion, otherwise slide the element that is in the holes parent node, into the hole, thus, bubbling the hole up towards the root.

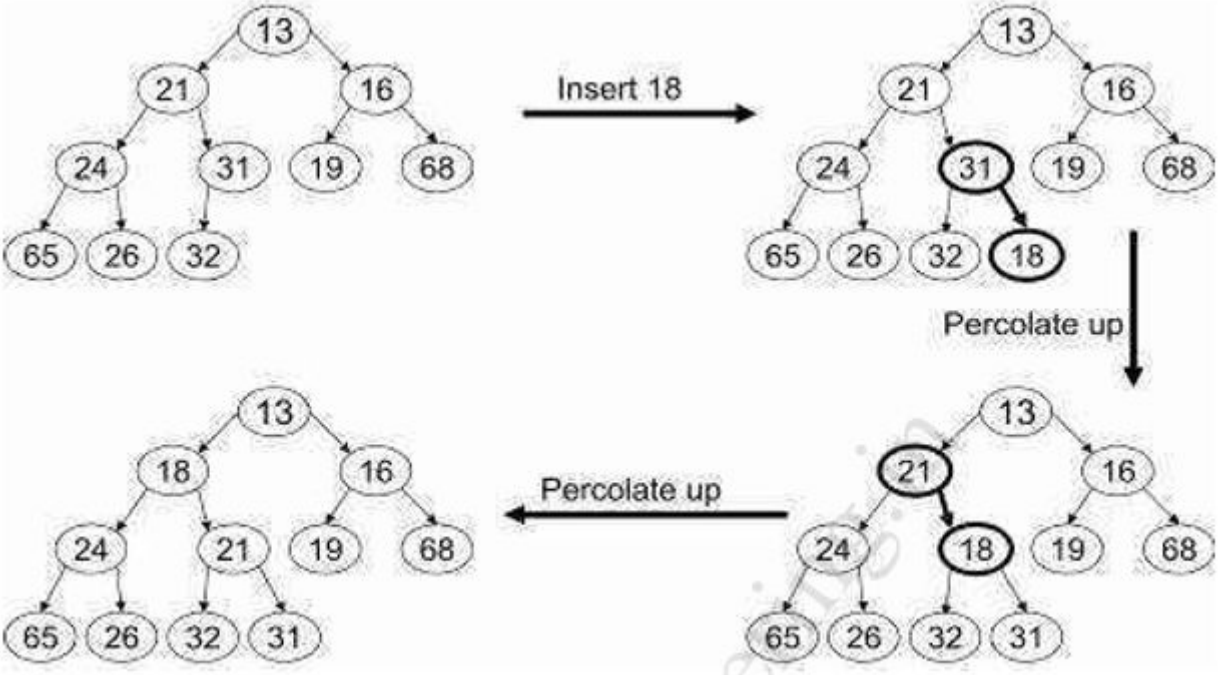
Step3: Continue this process until X can be placed in the hole.



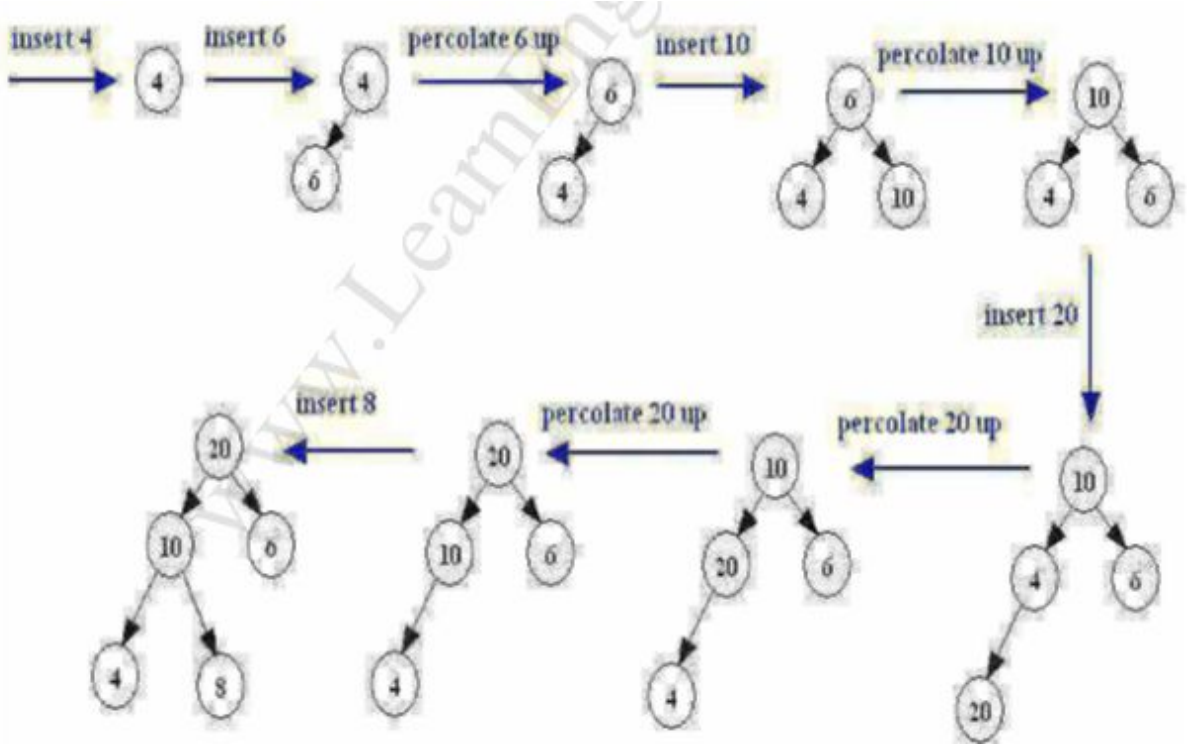
Attempt to insert 14: creating the hole, and bubbling the hole up



Example Insert – 18 in a min heap.



1. Insert the keys 4, 6, 10, 20, and 8 in this order in an originally empty max- heap



Removing the root node of a max- or min-heap, respectively

Procedure for Deletemin :

* Deletemin operation is deleting the minimum element from the loop.

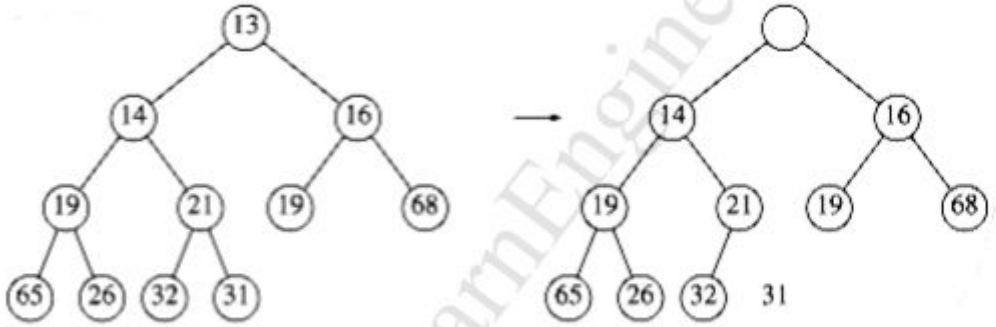
* In Binary heap | min heap the minimum element is found in the root.

* When this minimum element is removed, a hole is created at the root.

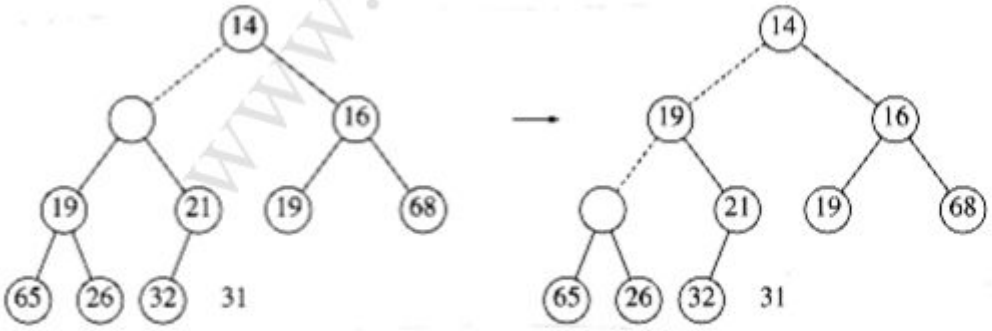
* Since the heap becomes one smaller, make the last element X in the heap to move somewhere in the heap.

* If X can be placed in the hole, without violating heap order property, place it , otherwise slide the smaller of the holes children into the hole, thus , pushing the hole down one level.

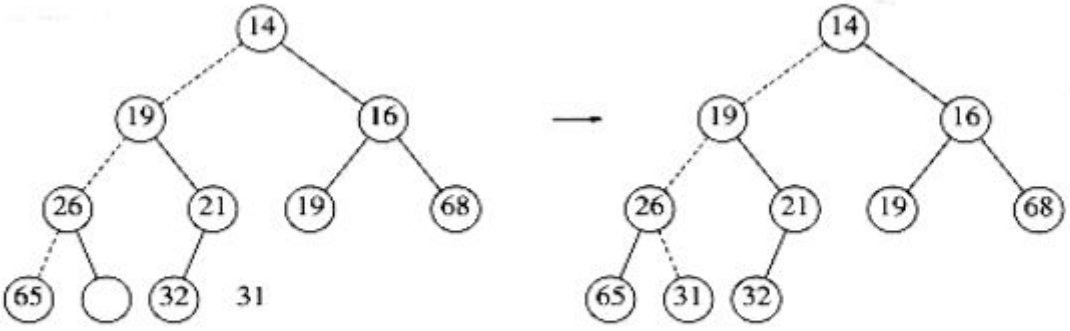
* Repeat this process until X can be placed in the hole. This general strategy is known as Percolate Down.



Creation of the hole at the root

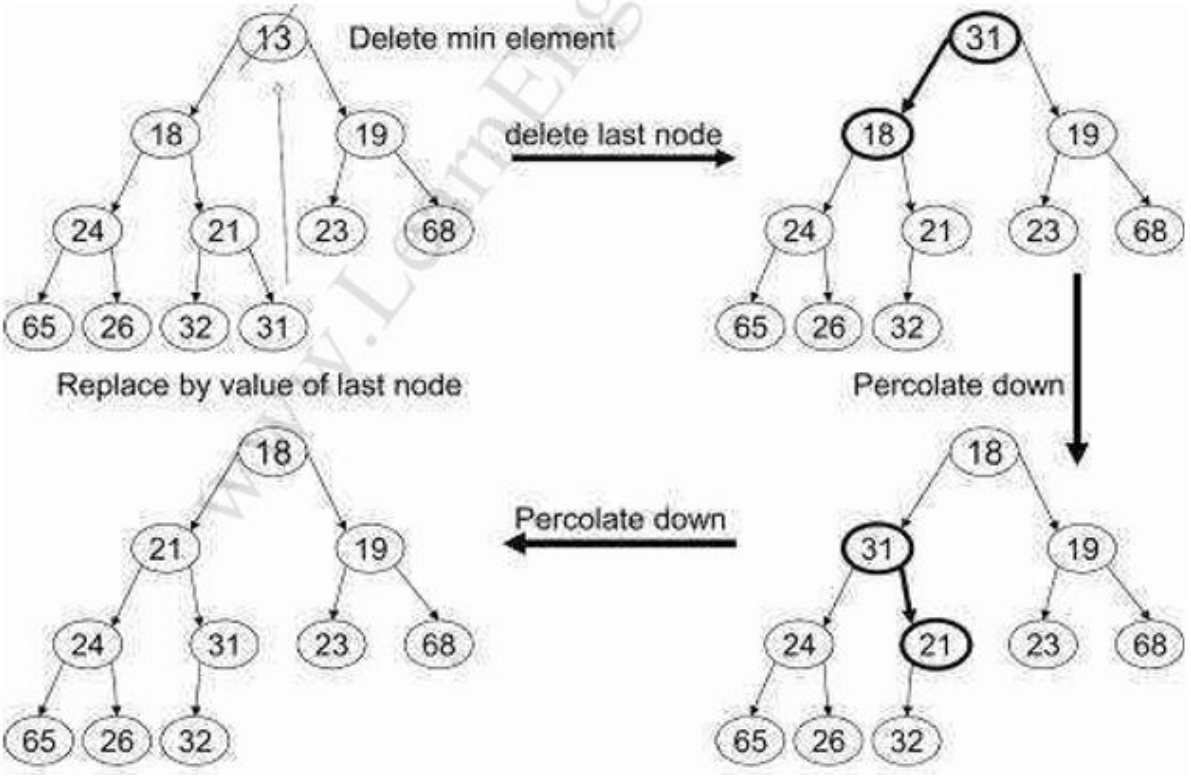


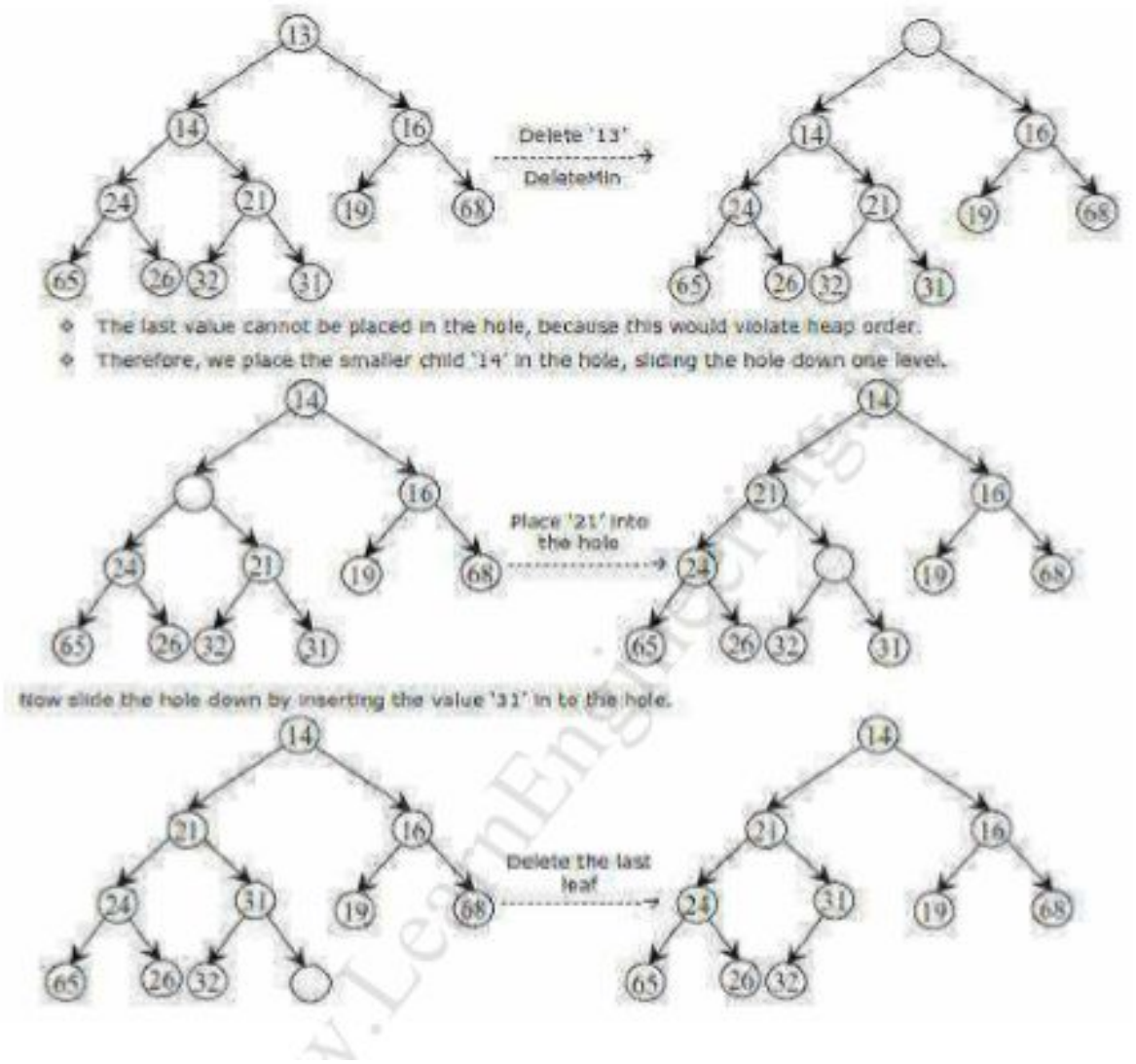
Next two steps in delete_min



Last two steps in delete_min

1.DELETE MIN





ALGORITHM

```
Heapify(array, size, i)
set i as largest
leftChild = 2i + 1
 rightChild = 2i + 2
 if leftChild &gt; array[largest]
 set leftChildIndex as largest
if rightChild &gt; array[largest]
set rightChildIndex as largest
 swap array[i] and array[largest]
```

To create a Max-Heap:

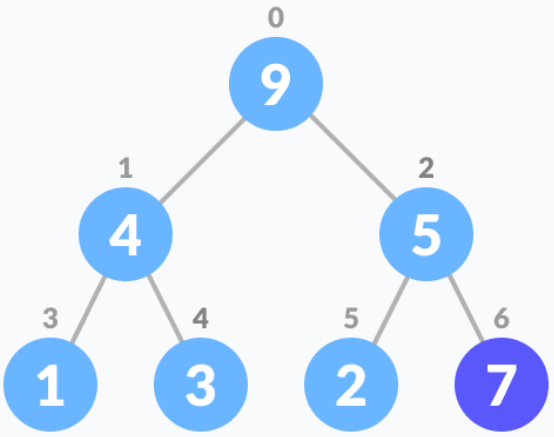
```
MaxHeap(array, size)
loop from the first index of non-leaf node down to zero
call heapify
```

For Min-Heap, both leftChild and rightChild must be smaller than the parent for all nodes.

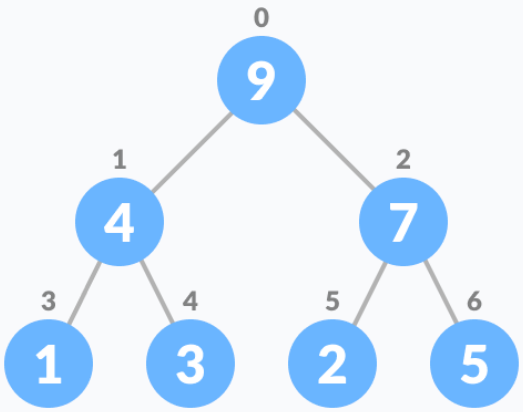
Insert Element into Heap

```
If there is no node,
 create a newNode.
else (a node is already present)
insert the newNode at the end (last node from left to right.)
heapify the array
```

1. Insert the new element at the end of the tree.



2. Heapify the tree.



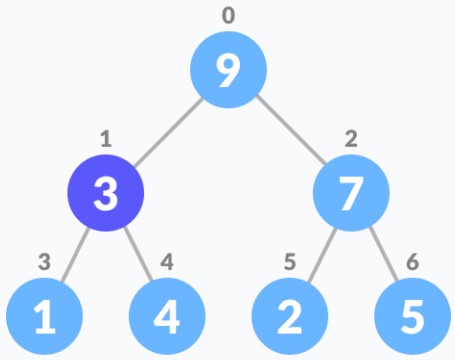
For Min Heap, the above algorithm is modified so that parentNode is always smaller than newNode. Delete Element from Heap

Algorithm for deletion in Max Heap

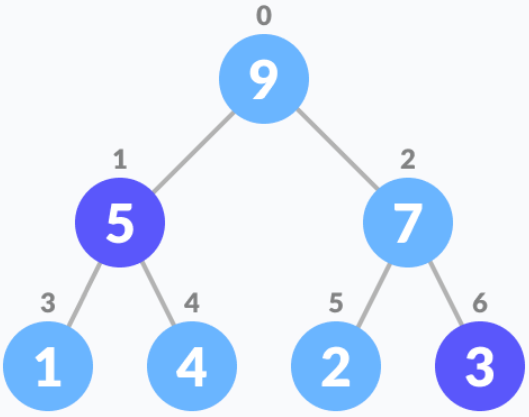
```
If nodeToBeDeleted is the leafNode
 remove the node
Else swap nodeToBeDeleted with the lastLeafNode
  remove noteToBeDeleted

heapify the array
```

1.

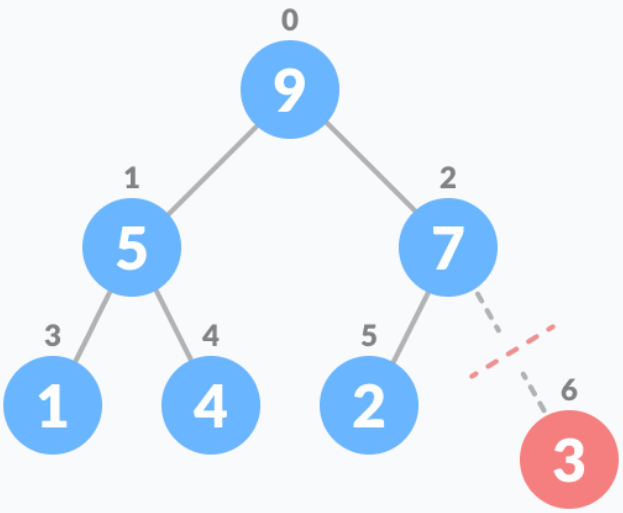
1. Select the element to be deleted.



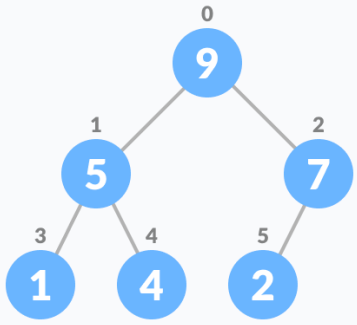
2. Swap it with the last element.



3. Remove the last element.



4 .Heapify the tree.



For Min Heap, above algorithm is modified so that both childNodes are greater smaller than currentNode.

Peek (Find max/min)

Peek operation returns the maximum element from Max Heap or minimum element from Min Heap without deleting the node. For both Max heap and Min Heap. return rootNode

Extract-Max/Min

Extract-Max returns the node with maximum value after removing it from a Max Heap whereas Extract-Min returns the node with minimum after removing it from Min Heap.

Other Heap Operations

1. Decrease Key.
2. Increase Key.
3. Delete.
4. Build Heap.

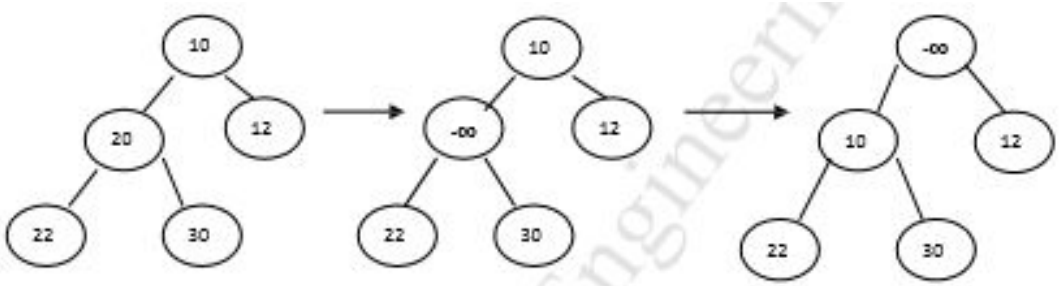
DELETE OPERATION

The delete (P,H) operation removes the node at the position P, from the heap H. This can be done by,

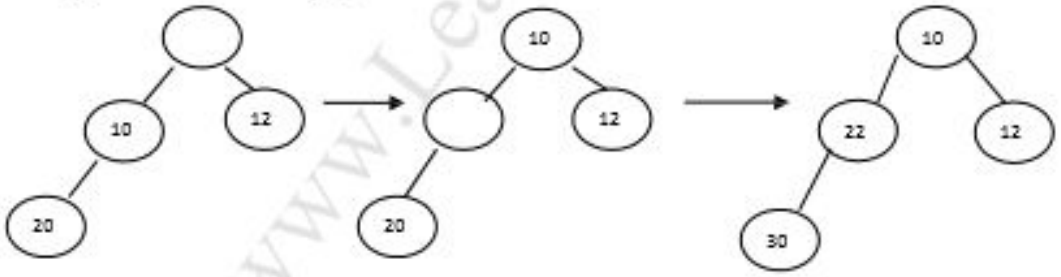
Step 1: Perform the decrease key operation, decrease key(P,∞,H).

Step 2: Perform deletemin(H) operation.

Step 3: Decreasekey(2, ∞,H)



Step 2 : Deletemin(H)



HEAP APPLICATIONS

A. Heap sort
B. Selection algorithms
C. Graph algorithms

Heap sort **:** One of the best sorting methods being in-place and with no quadratic worst-case scenarios.

Selection algorithms: Finding the min, max, both the min and max, median, or even the k-th largest element can be done in linear time using heaps.

Graph algorithms: By using heaps as internal traversal data structures, run time will be reduced by an order of polynomial. Examples of such problems are Prim's minimal spanning tree algorithm and Dijkstra's shortest path problem.

ADVANTAGE The biggest advantage of heaps over trees in some applications is that construction of heaps can be done in linear time.

DISADVANTAGE Heap is expensive in terms of

- Safety
- Maintenance
- Performance

Performance : Allocating heap memory usually involves a long negotiation with the

Maintenance: Dynamic allocation may fail; extra code to handle such exception is required.

Safety : Object may be deleted more than once or not deleted at all .

Heap Data Structure Applications

- Heap is used while implementing a priority queue.
- Dijkstra’s Algorithm
- Heap Sort