UNIT – I

BASIC HEAP OPERATIONS:

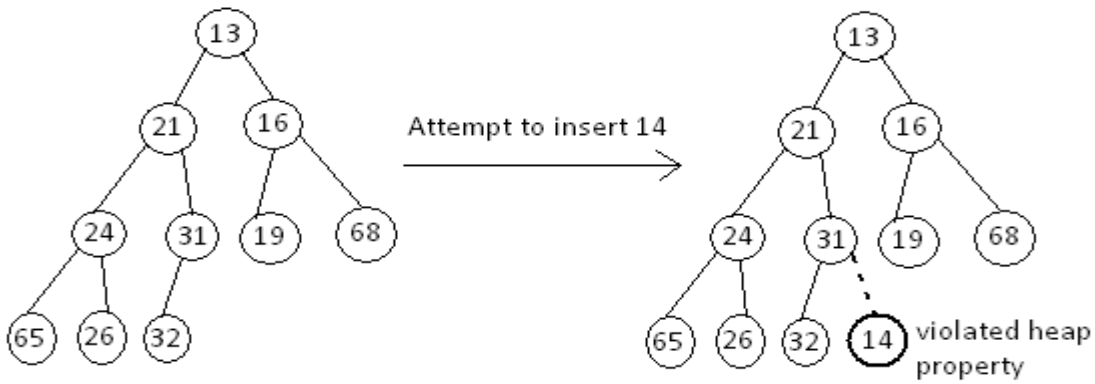
It is easy to perform the two required operations. All the work involves ensuring that the heap order property is maintained.

- Finding/searching an element
- Inserting a new element
- Deleting an element
- Merging of two heap trees

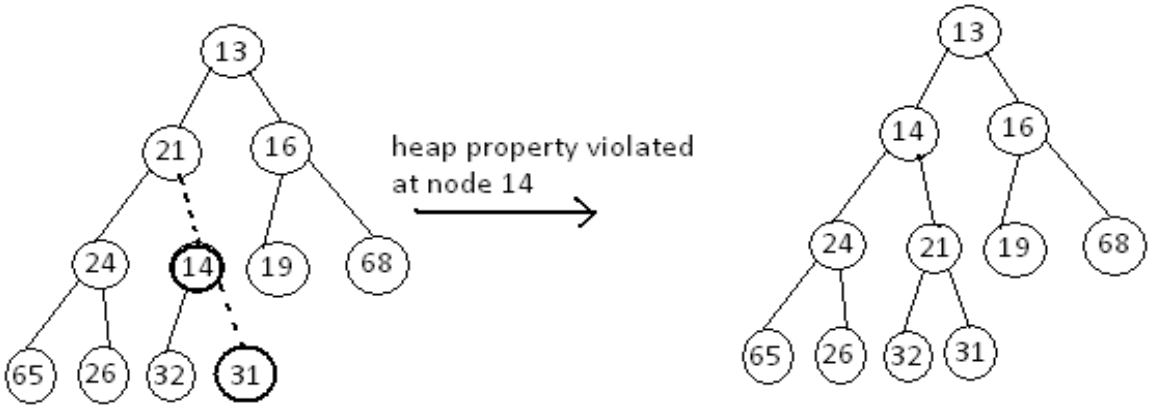
Insertion Operation On Binary Heap:

- To insert an element say x, into the heap with n elements, we first create a hole in position (n+1) and see if the heap property is violated by putting x into the hole.
- If the heap property is not violated, then we have found the correct position for x. Otherwise, we ``Re heap -up'' or ``percolate-up'' x until the heap property is restored.
- Percolate-up / Reheap-up: we slide the element that is in the hole's parent node into the hole, thus bubbling the hole up toward the root. We continue this process until x can be placed in the whole

Consider the heap



We create a hole in the next available heap location. Inserting 14 in the hole would violate the heap order property so 31 is slid down into the hole. This strategy is continued until the correct location for 14 is found.



This general strategy is known as a percolate up. i.e. the new element is percolated up the heap until the correct location is found.

NOTE: Worst case complexity of insert is O(h) where h is the height of the heap. Thus insertions are O(log n) where n is the no .of elements in the heap.

Delete min:

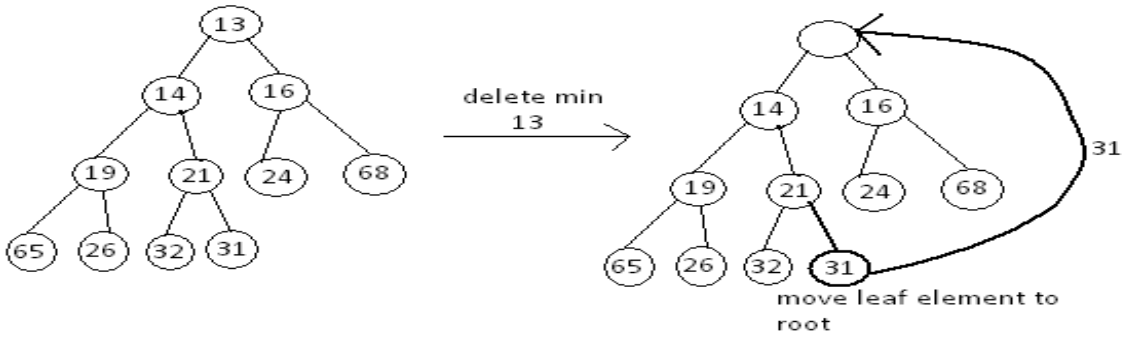
Where the minimum is deleted a hole is created at the root level. Since the heap now has one less element and the heap is a complete binary tree, the element in the least position is to be relocated.

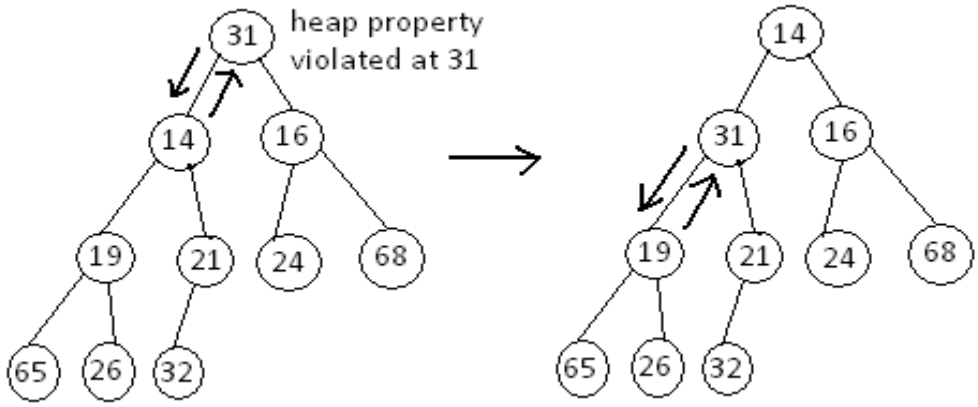
This we first do by placing the last element in the hole created at the root. This will leave the heap property possibly violated at the root level.

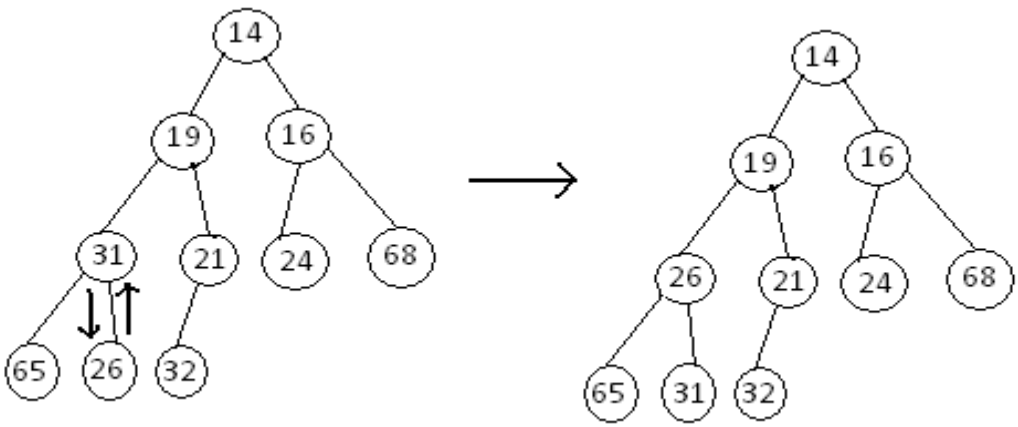
We now push – down or percolate – down the hole at the root until the violation of heap property is stopped. While pushing down the hole it is important to slide it down to the less of its two children (pushing up the latter). This is done so as not to create another violation of heap property.

Consider the previous example:

First remove or delete min is 13.







This general strategy is known as a percolate down. We use same technique as in the insert routine to avoid the use of swaps in this routine.

NOTE: The worst case running time of delete min is O(log n) where n is the no. of elements in the heap.

Creating Heap:

The build heap operation takes as input n elements. The problem here is to create a heap of these elements i.e. places them into an empty heap.

Obvious approach is to insert the n element one at a time into an initially empty heap. Since each insert will take O(1) average and O(log n) worst case time, the total running time of this algorithm would be O(n) average but O(n log n) worst case

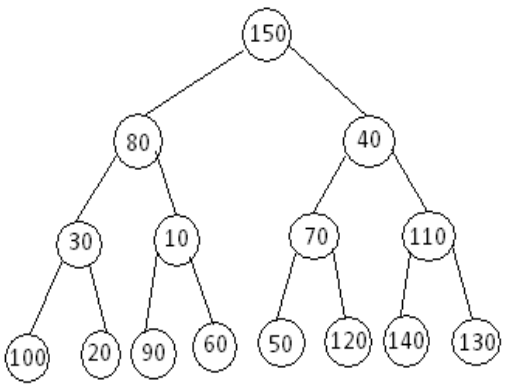
Another approach proposed by Floyd in 1964 is to use a procedure called push – down or percolate – down repeatedly. Starting with the array consisting of the given n elements in the input – order.

If percolate – down (i) percolate down from node i, perform the algorithm to create a heap – ordered tree.

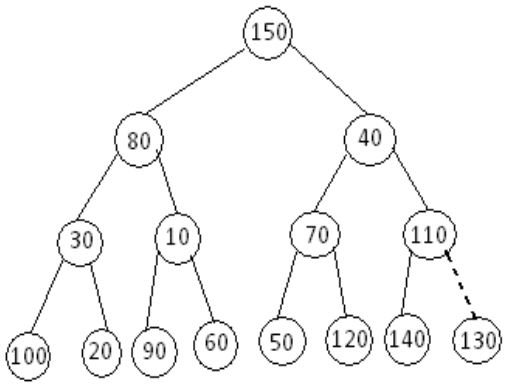
percolate - down (i)

Consider the tree is the unordered tree

Left: initial tree

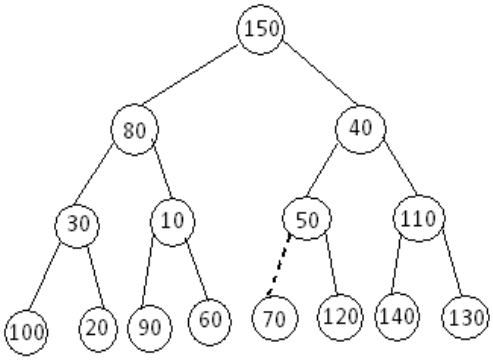


Right: after percolate – down (7)

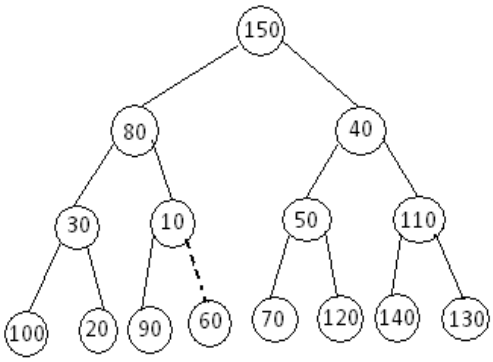


Left: after percolate – down (6)

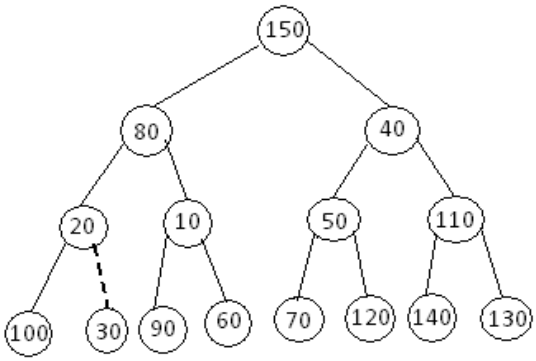
Right: after percolate – down (5)



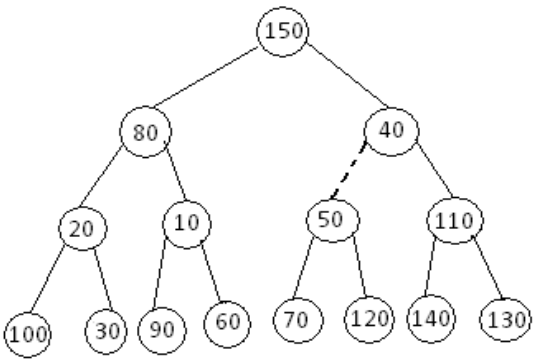
Left: after percolate – down (4)



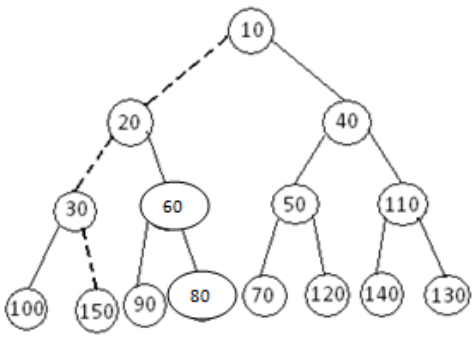
Right: after percolate – down (3)



Left: after percolate – down (2)



Right: after percolate – down (1)



Each dashed line corresponds to two comparisons: one to find the smallest child and one to compare the smaller child with the node.

Notice that there are only 10 dashed lines in the entire algorithm (there could been an 11th where?) corresponding to 20 comparisons.

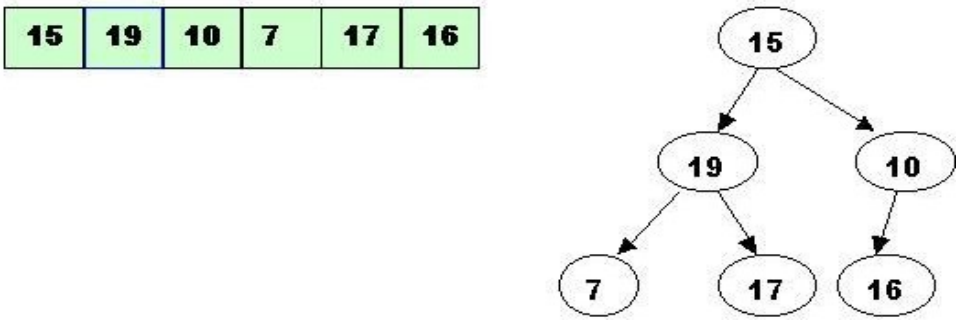
To bounding the running time of build heap, we must bound the no. of dashed lines. This can be done by computing the sum of the heights of all the nodes in the heap, which is the maximum no. of dashed lines. What we would like to show is that this is O(n).

Example:

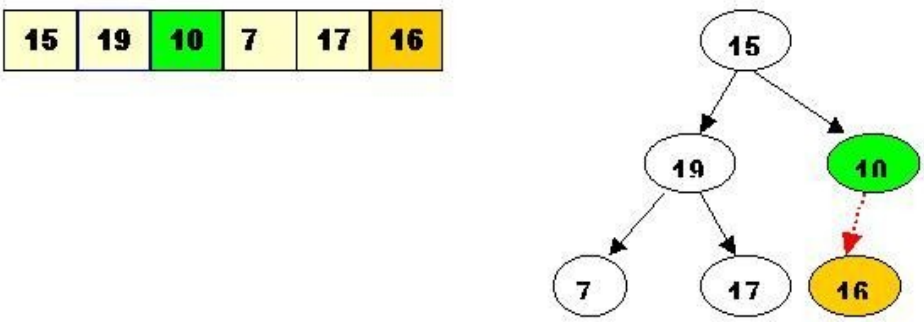
Given an array of 6 elements: 15, 19, 10, 7, 17, 16, sort it in ascending order using heap sort.

Building the heap tree

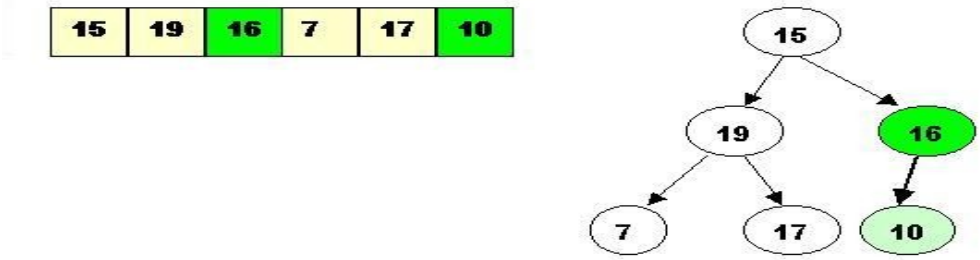
The array represented as a tree, complete but not ordered:



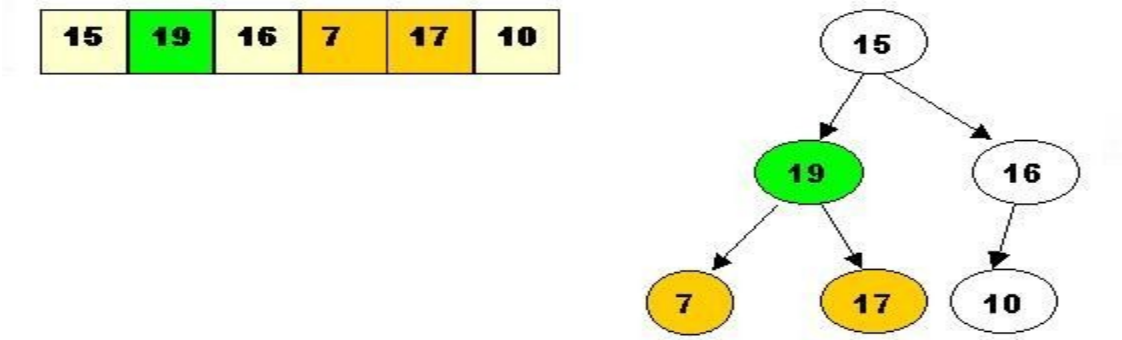
Start with the rightmost node at height 1 - the node at position 3 = Size/2. It has one greater child and has to be percolated down:



After processing array[3] the situation is:

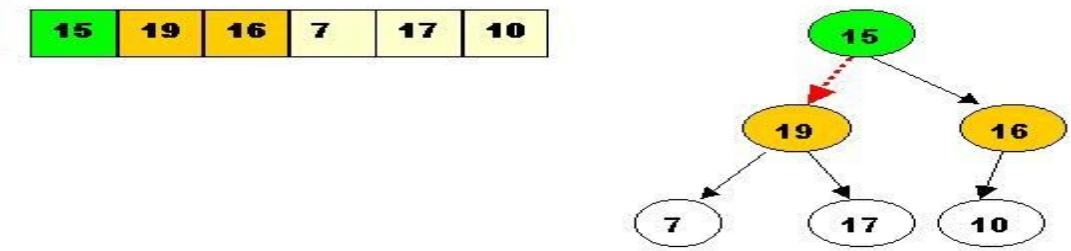


Next comes array[2]. Its children are smaller, so no percolation is needed.

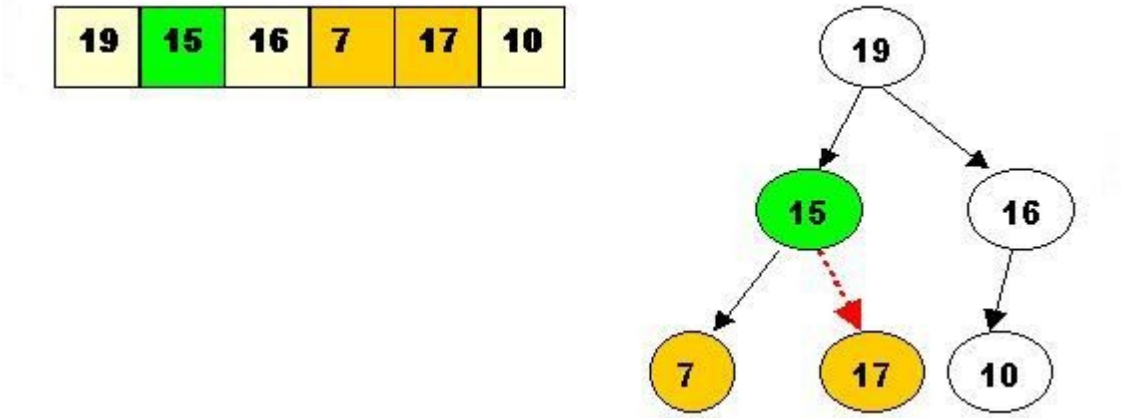


The last node to be processed is array[1]. Its left child is the greater of the children.

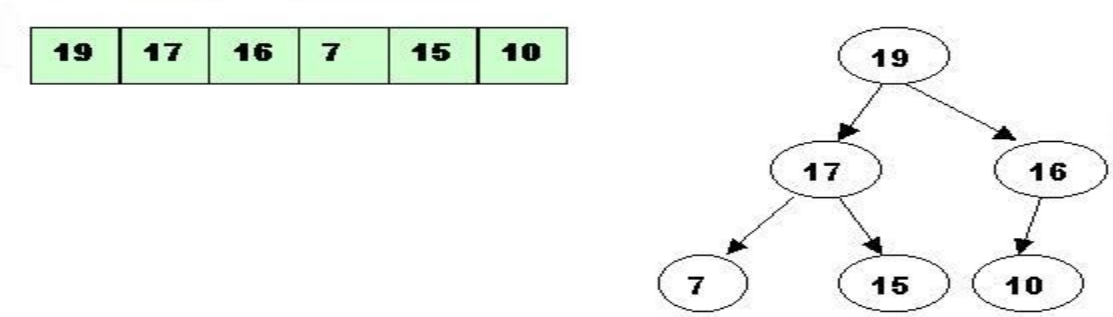
The item at array[1] has to be percolated down to the left, swapped with array[2].



As a result the situation is:



The children of array[2] are greater, and item 15 has to be moved down further, swapped with array[5].

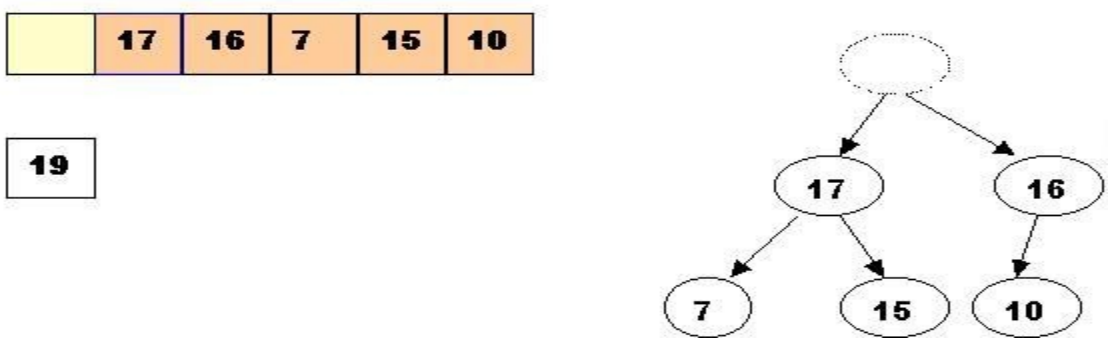


Now the tree is ordered, and the binary heap is built.

Sorting - performing deleteMax operations:

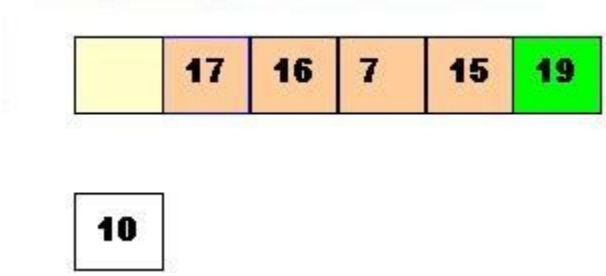
Delete the top element 19.

Store 19 in a temporary place. A hole is created at the top

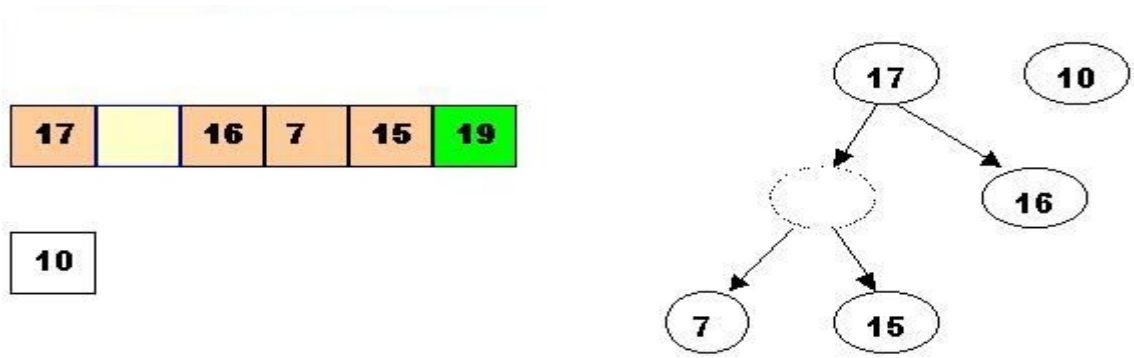


Swap 19 with the last element of the heap.

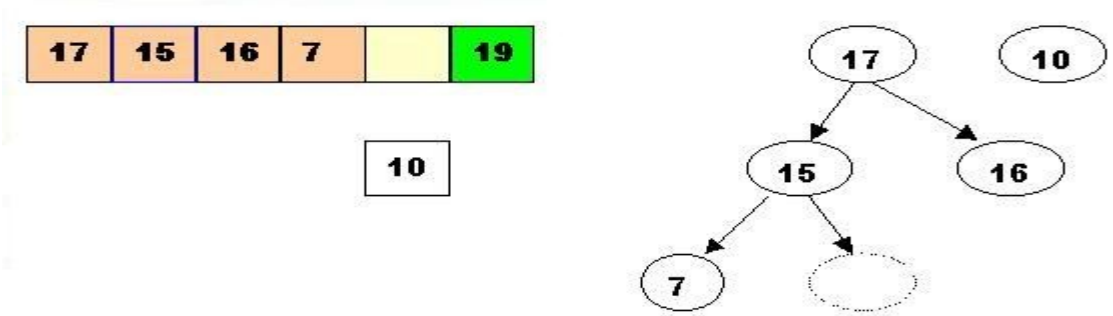
As 10 will be adjusted in the heap, its cell will no longer be a part of the heap. Instead it becomes a cell from the sorted array



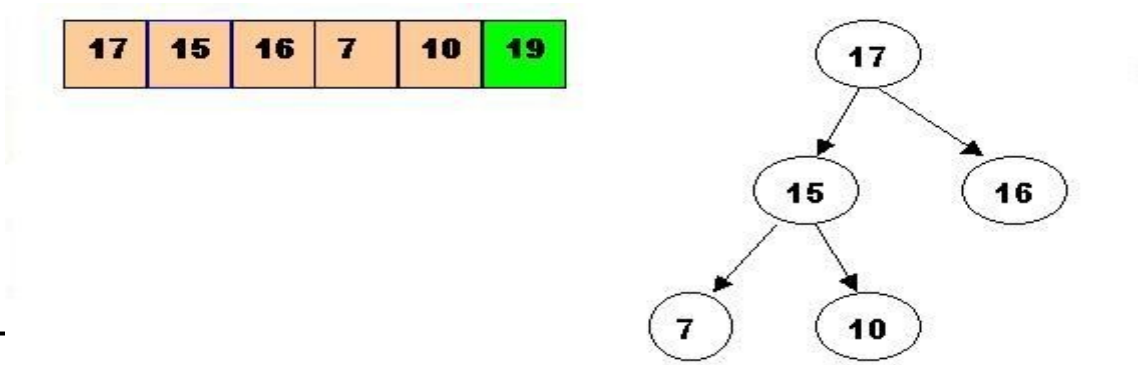
Percolate down the hole



Percolate once more (10 is less that 15, so it cannot be inserted in the previous hole)



Now 10 can be inserted in the hole



Reference Links

1. https://www.tutorialspoint.com/data_structures_algorithms/heap_data_structure.htm
2. https://www.programiz.com/dsa/heap-data-structure

Video Links

1. https://www.youtube.com/watch?v=iAPjvPA7O_M
2. https://www.youtube.com/watch?v=ioPtKrAx65g

Questions

1. The elements 32, 15, 20, 30, 12, 25 and 16 are inserted one by one in the given order into a max-heap. What is the resultant Max-heap?
2. Write an algorithm to insert an element in to a heap. Explain with a suitable example.
3. What is binary heap? Explain the procedure to insert an element into binary heap.
4. Construct a heap using the following list of numbers: 12,9,8,3,7,5,10,18