Heap

Heap sort basically recursively performs two main operations –

- Build a heap H, using the elements of array.
- Repeatedly delete the root element of the heap formed in 1st phase.

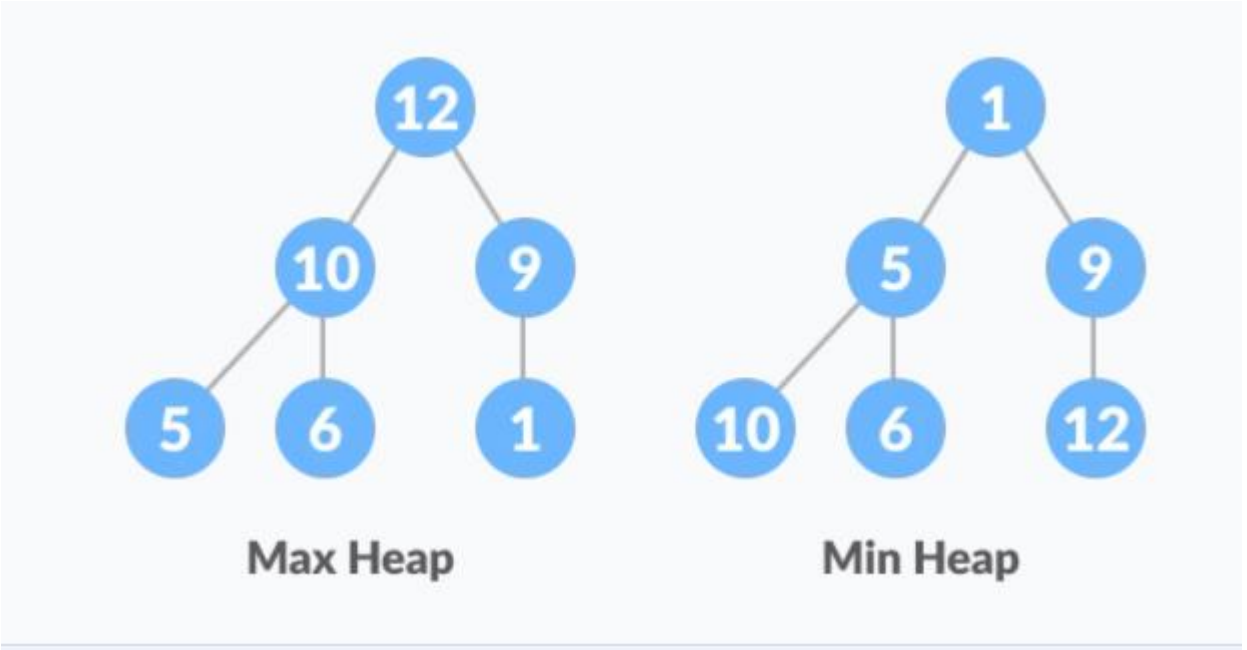
What is a heap?

- ✓ A heap is a complete binary tree, and the binary tree is a tree in which the node can have the utmost two children.

- ✓ A complete binary tree is a binary tree in which all the levels except the last level, i.e., leaf node, should be completely filled, and all the nodes should be left-justified.

- ✓ All nodes in the tree follow the property that they are greater than their children i.e. the *largest element is at the root* and both its children and smaller than the root and so on. Such a heap is called a max-heap.

- ✓ If instead, all nodes are smaller than their children, it is called a min-heap



How to "heapify" a tree?

Starting from a complete binary tree, we can modify it to become a Max-Heap by running a function called heapify on all the non-leaf elements of the heap.

Let's understand the max heap through an example.

Insertion in the Heap tree:
44, 33, 77, 11, 55, 88, 66

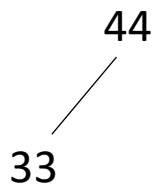
Suppose we want to create the max heap tree. (You can also create min heap)

To create the max heap tree, we need to consider the following two cases:

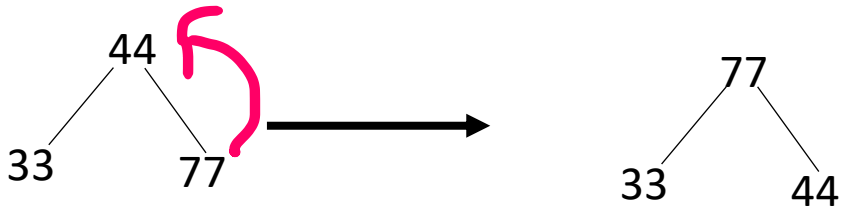
- First, we have to insert the element in such a way that the property of the complete binary tree must be maintained.
- Secondly, the value of the parent node should be greater than the either of its child.

Step 1: First we add the 44 element in the tree as shown below:

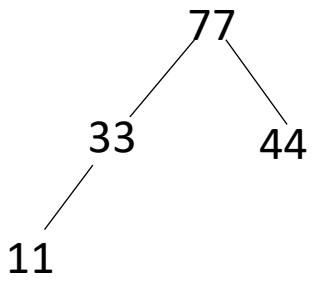
Step 2: The next element is 33. As we know that insertion in the binary tree always starts from the left side so 44 will be added at the left of 33 as shown below:



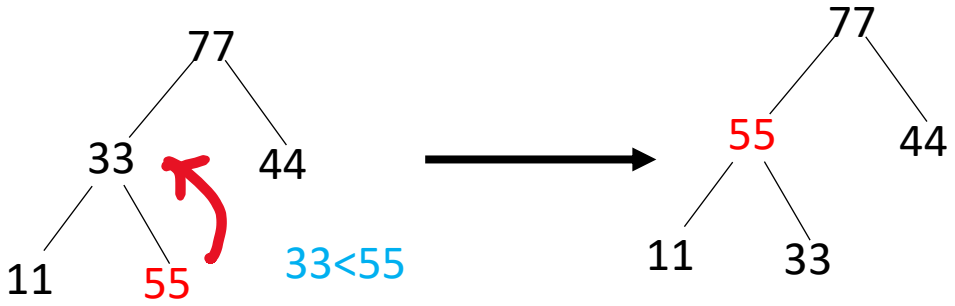
Step 3: The next element is 77 and it will be added to the right of the 44 as shown below:



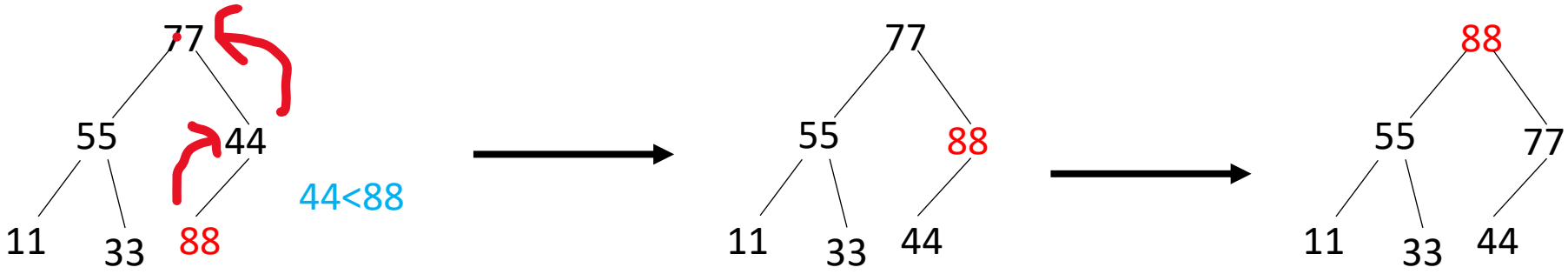
Step 4: The next element is 11. The node 11 is added to the left of 33 as shown below:



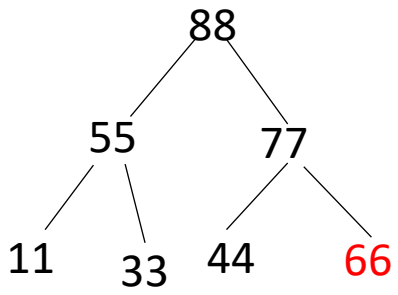
Step 5: The next element is 55. To make it a complete binary tree, we will add the node 55 to the right of 33 as shown below:



Step 6: The next element is 88. The left subtree is completed so we will add 88 to the left of 44 as shown below:



Step 7: The next element is 66. To make a complete binary tree, we will add the 66 element to the right side of 77 as shown below:



Try to create the min heap tree for the same input.

Problem-02:

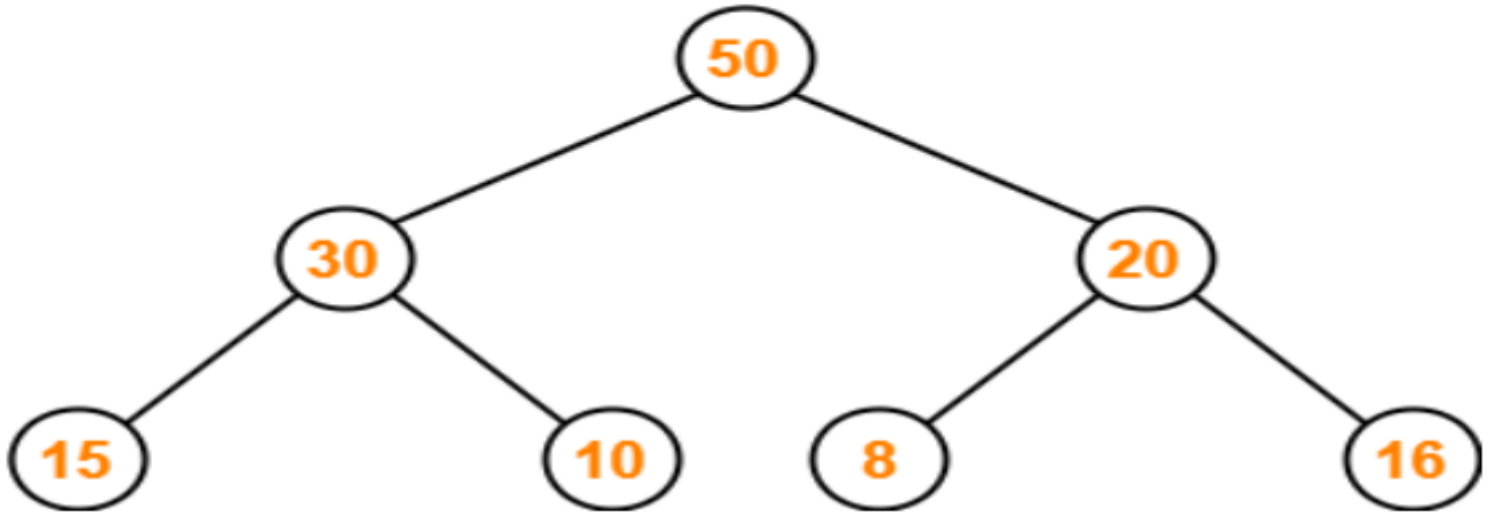
Consider the following max heap-

50, 30, 20, 15, 10, 8, 16

Insert a new node with value 60.

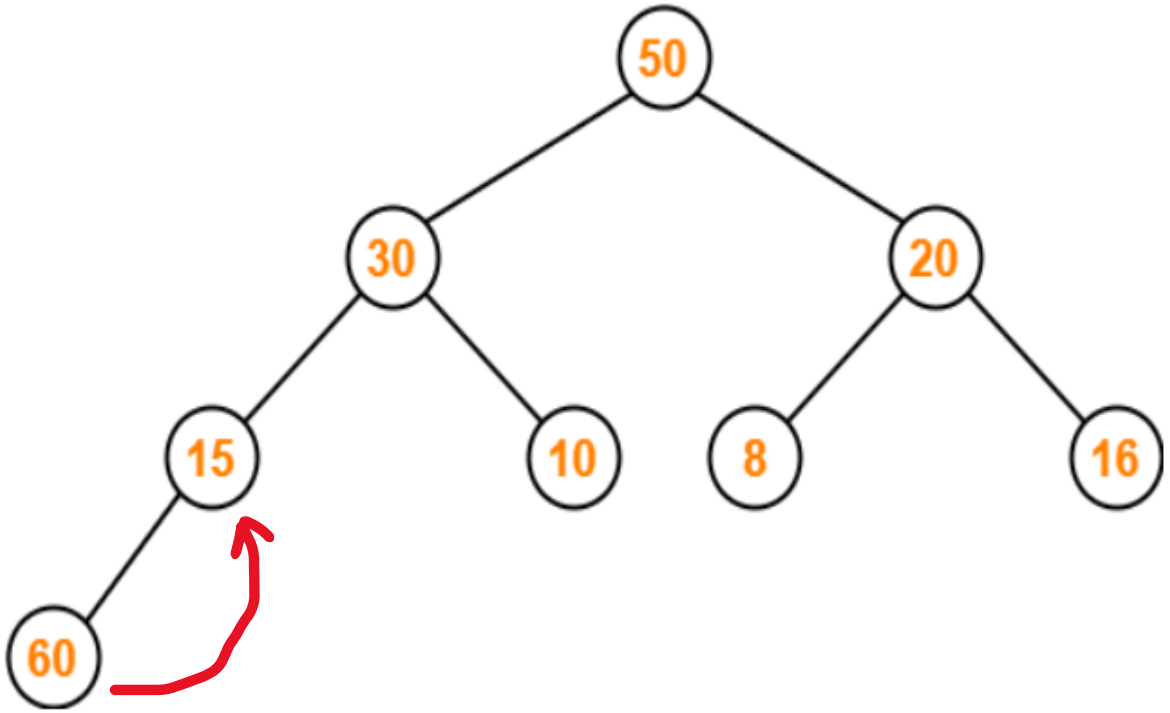
Step-01:

We convert the given array of elements into a heap tree-



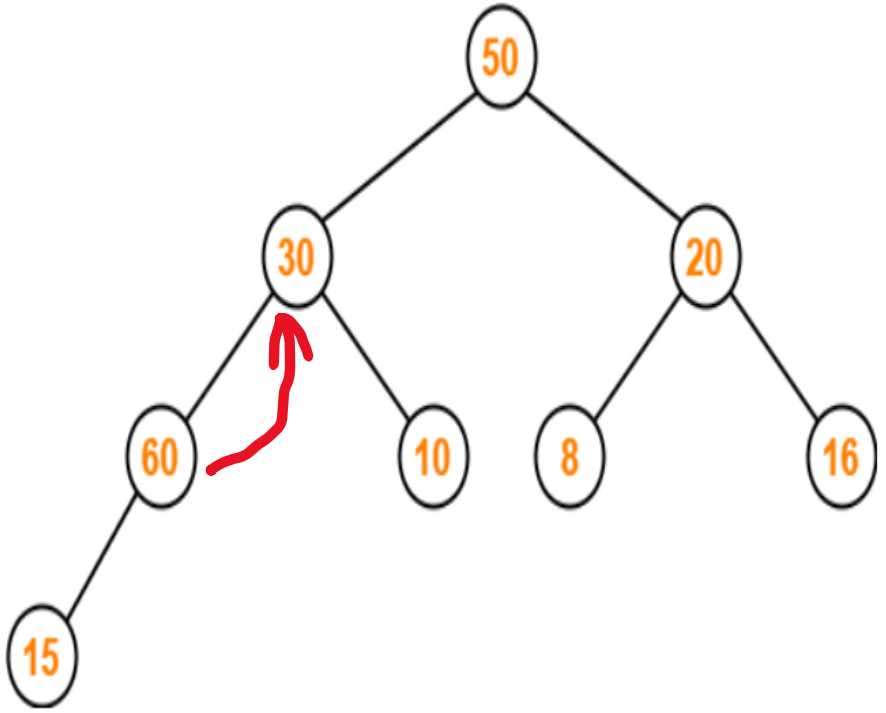
Step-02:

We insert the new element 60 as a next leaf node from left to right.
The resulting tree is-



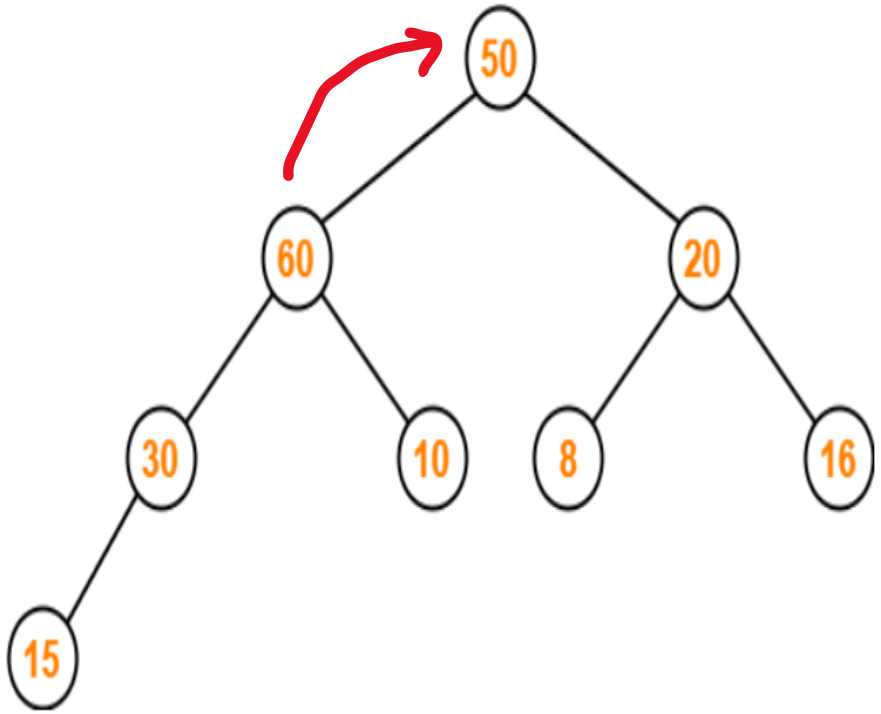
Step-03:

- We ensure that the tree is a max heap.
- Node 15 contains greater element in its left child node.
- So, we swap node 15 and node 60.



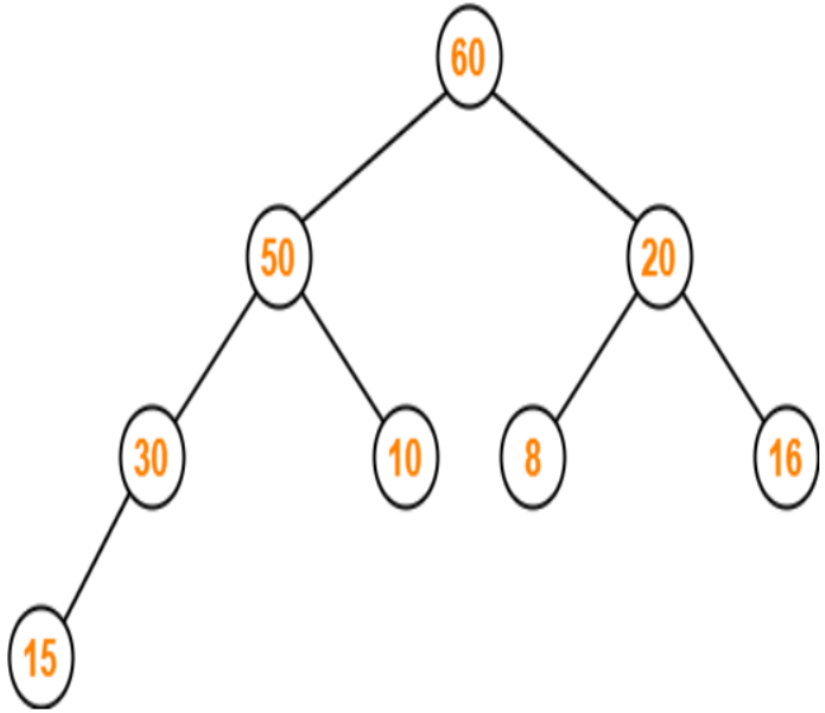
Step-04:

•Node 30 contains greater element in its left child node.
•So, we swap node 30 and node 60.



Step-05:

•Node 50 contains greater element in its left child node.
•So, we swap node 50 and node 60.



Problem-03:

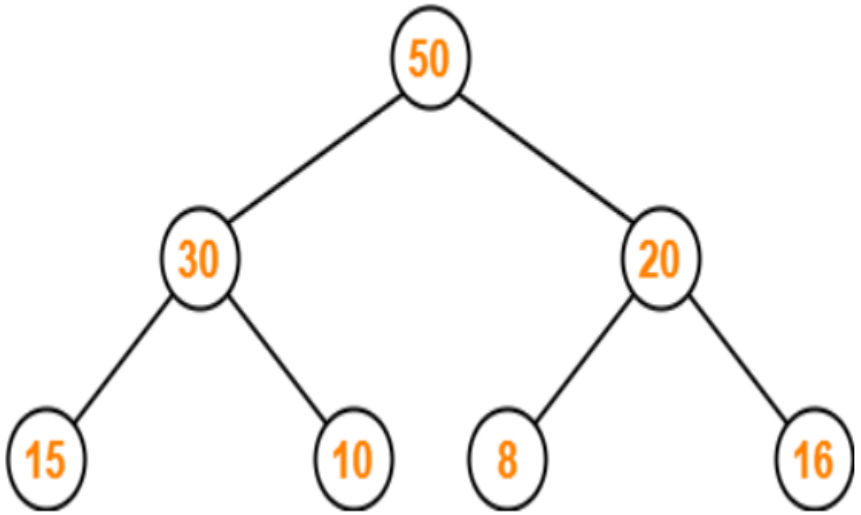
Consider the following max heap-

50, 30, 20, 15, 10, 8, 16

Delete a node with value 50.

Step-01:

We convert the given array of elements into a heap tree

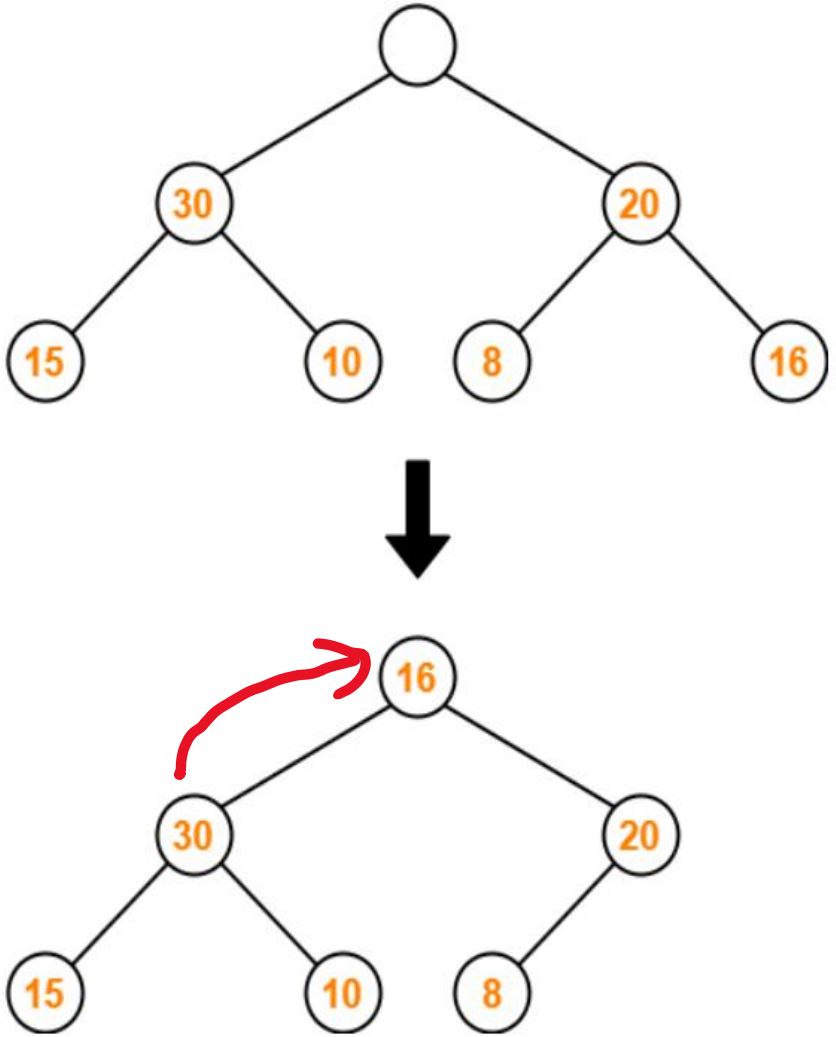


Step-02:

We delete the element 50 which is present at root node.

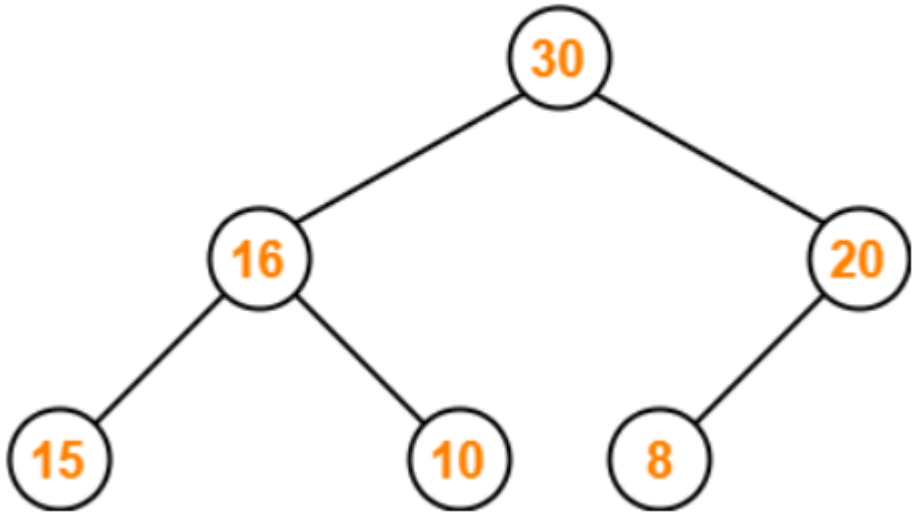
•We pluck the last node 16 and put in place of the deleted node.

•Heap: always root node is deleted



Step-03:

- We ensure that the tree is a max heap.
- Node 16 contains greater element in its left child node.
- So, we swap node 16 and node 30.



Heap sort

- Basically, there are two phases involved in the sorting of elements using heap sort algorithm they are as follows:

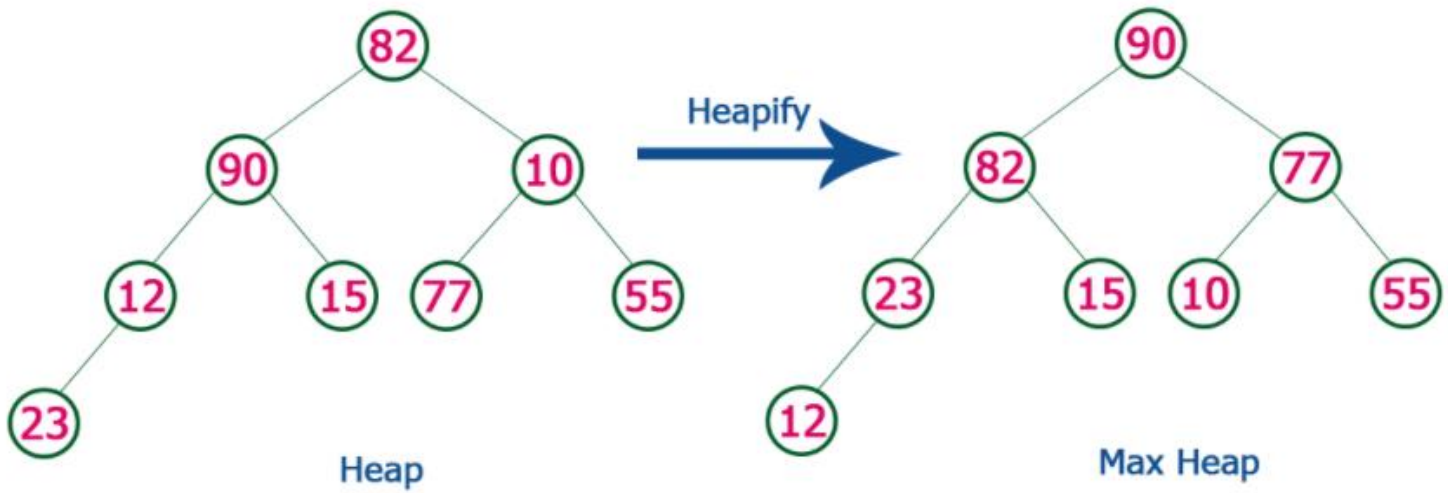
  - Consider an array which is to be sorted using Heap Sort.

  - Initially build a max heap of elements in .
  - The root element, that is , will contain maximum element of . After that, swap this element with the last element of and heapify the max heap excluding the last element which is already in its correct position and then decrease the length of heap by one.
  - Repeat the step 2, until all the elements are in their correct position.

Consider the following list of unsorted numbers which are to be sort using Heap Sort

82, 90, 10, 12, 15, 77, 55, 23

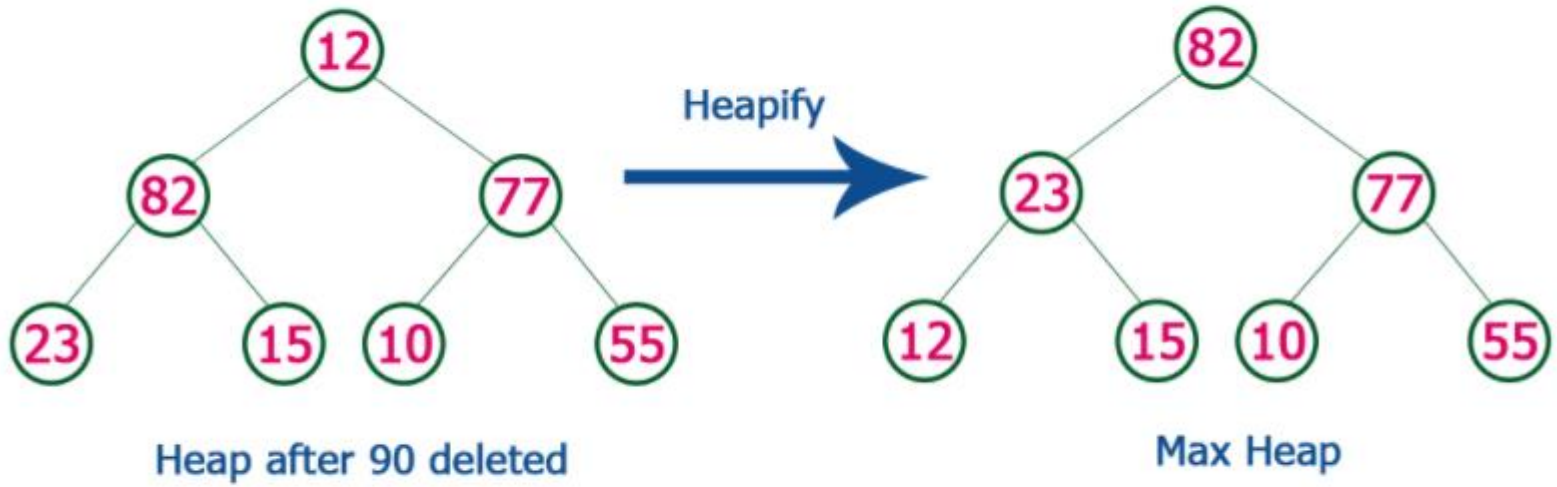
Step 1 - Construct a Heap with given list of unsorted numbers and convert to Max Heap



list of numbers after heap converted to Max Heap

90, 82, 77, 23, 15, 10, 55, 12

Step 2 - Delete root (**90**) from the Max Heap. To delete root node it needs to be swapped with last node (**12**). After delete tree needs to be heapify to make it Max Heap.



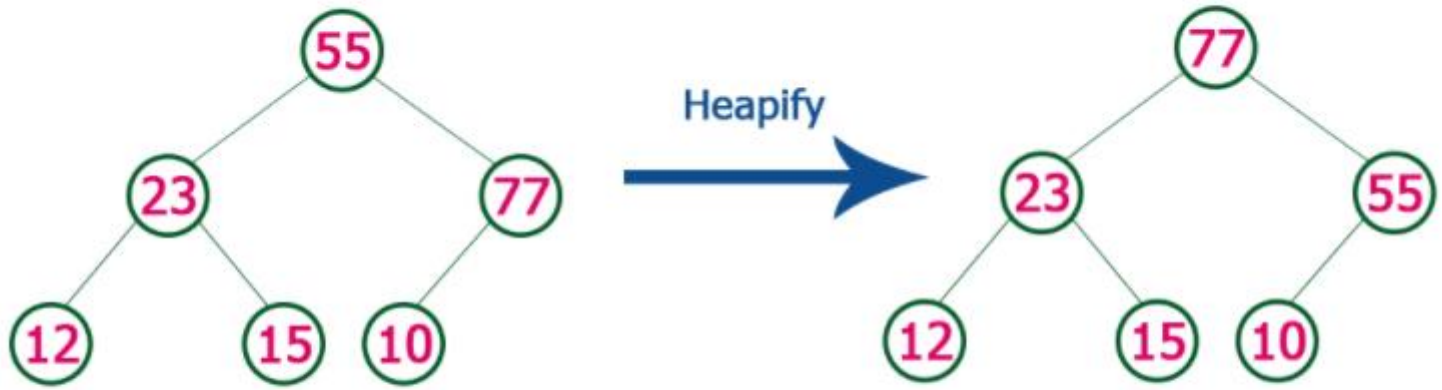
Heap after 90 deleted

Max Heap

list of numbers after swapping 90 with 12.

12, 82, 77, 23, 15, 10, 55, **90**

Step 3 - Delete root (**82**) from the Max Heap. To delete root node it needs to be swapped with last node (**55**). After delete tree needs to be heapify to make it Max Heap.



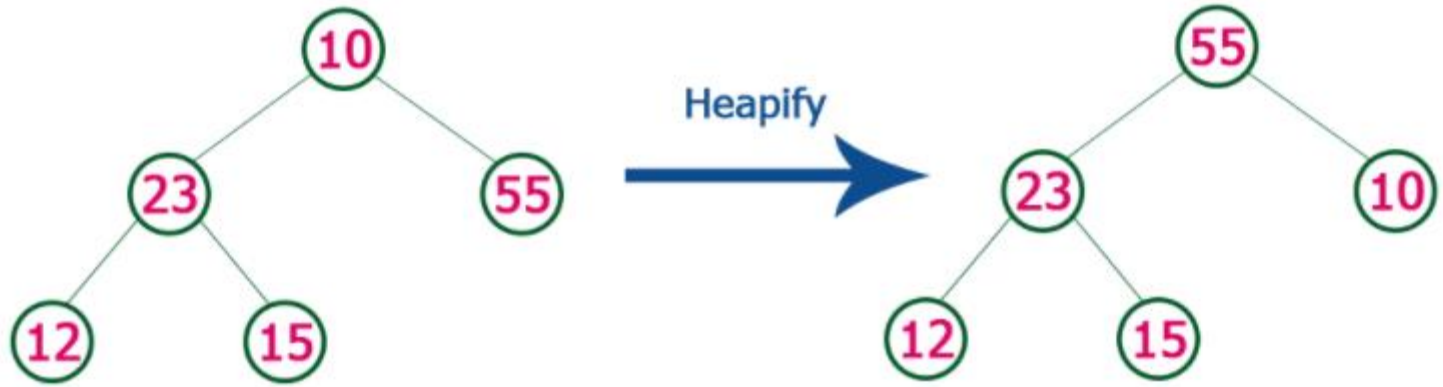
Heap after 82 deleted

Max Heap

list of numbers after swapping 82 with 55.

12, 55, 77, 23, 15, 10, **82**, **90**

Step 4 - Delete root (**77**) from the Max Heap. To delete root node it needs to be swapped with last node (**10**). After delete tree needs to be heapify to make it Max Heap.



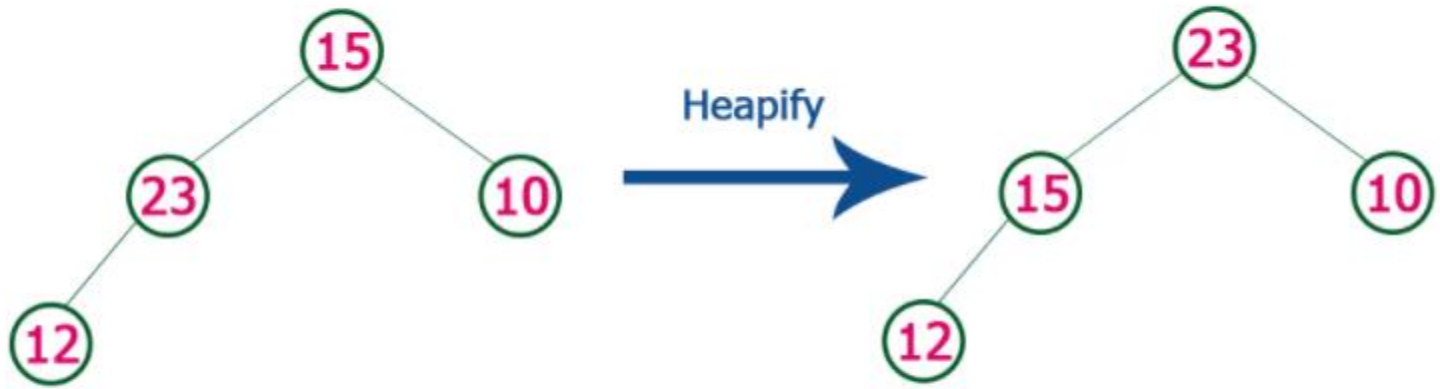
Heap after 77 deleted

Max Heap

list of numbers after swapping 77 with 10.

12, 55, 10, 23, 15, **77, 82, 90**

Step 5 - Delete root (**55**) from the Max Heap. To delete root node it needs to be swapped with last node (**15**). After delete tree needs to be heapify to make it Max Heap.



Heap after 55 deleted

Max Heap

list of numbers after swapping 55 with 15.

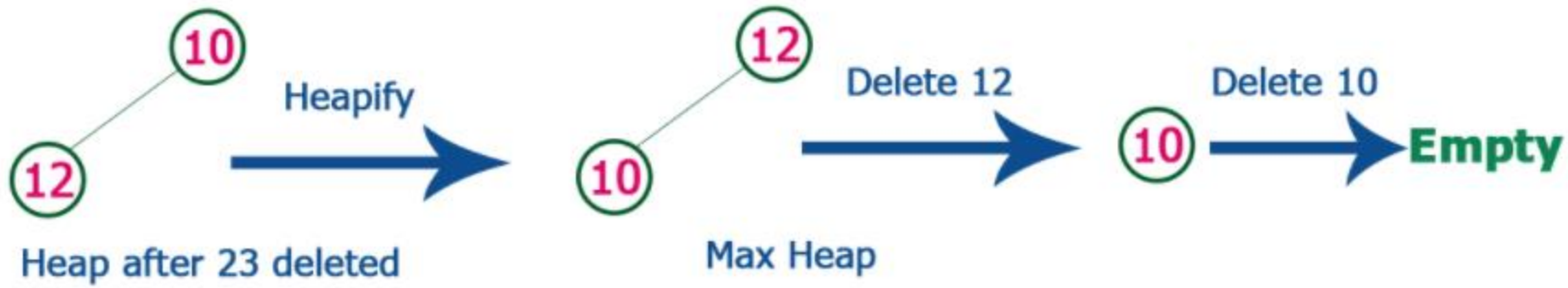
12, 15, 10, 23, **55**, **77**, **82**, **90**

Step 6 - Delete root (**23**) from the Max Heap. To delete root node it needs to be swapped with last node (**12**). After delete tree needs to be heapify to make it Max Heap.

list of numbers after swapping 23 with 12.

12, 15, 10, **23, 55, 77, 82, 90**

Step 7 - Delete root (**15**) from the Max Heap. To delete root node it needs to be swapped with last node (**10**). After delete tree needs to be heapify to make it Max Heap.



list of numbers after Deleting 15, 12 & 10 from the Max Heap.

10, 12, 15, 23, 55, 77, 82, 90

Whenever Max Heap becomes Empty, the list get sorted in Ascending order