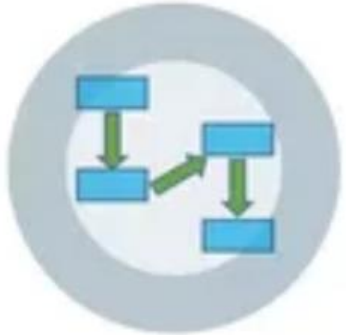
Heap

Min-Heap | Max-Heap



Data Structures and Algorithms

Arfan Shahzad

{ arfanskp@gmail.com }

Data Structures and Algorithms
Course Contents:
Abstract data types, complexity analysis, Big Oh notation, Stacks (linked lists and array implementations), Recursion and analyzing recursive algorithms, divide and conquer algorithms, Sorting algorithms (selection, insertion, merge, quick, bubble, heap, shell, radix, bucket), queue, dequeuer, priority queues (linked and array implementations of queues), linked list & its various types, sorted linked list, searching an unsorted array, binary search for sorted arrays, hashing and indexing, open addressing and chaining, trees and tree traversals, binary search trees, heaps, M-way tress, balanced trees, graphs, breadth-first and depth-first traversal, topological order, shortest path, adjacency matrix and adjacency list implementations, memory management and garbage collection

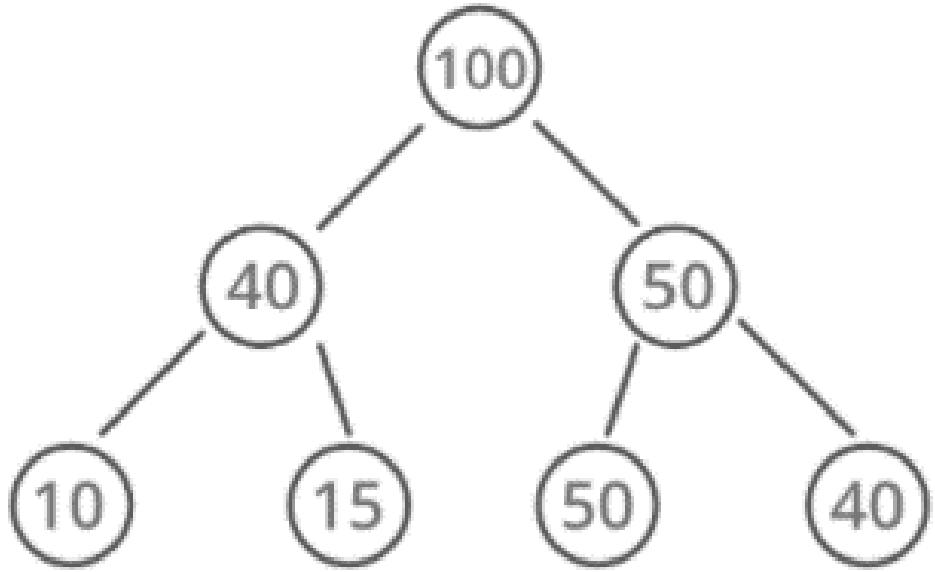
Tree (Heap)

- A Heap is a special Tree-based data structure in which the tree is a complete binary tree.
- Generally, Heaps can be of two types:

1. Max-Heap
2. Min-Heap

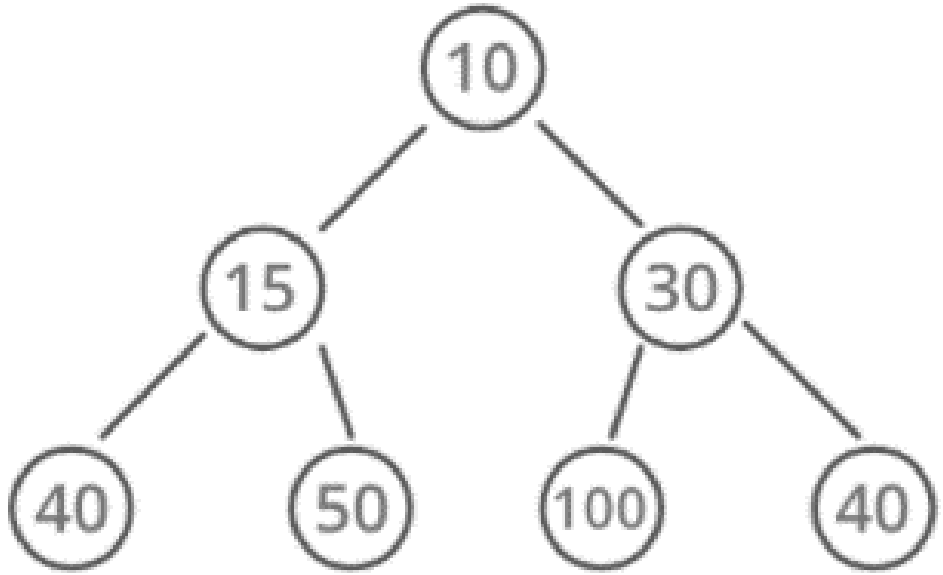
Tree (Heap) cont…

- **Max-Heap:** In a Max-Heap the key present at the root node must be greatest among the keys present at all of it's children.
- The same property must be recursively true for all sub-trees in that Binary Tree.

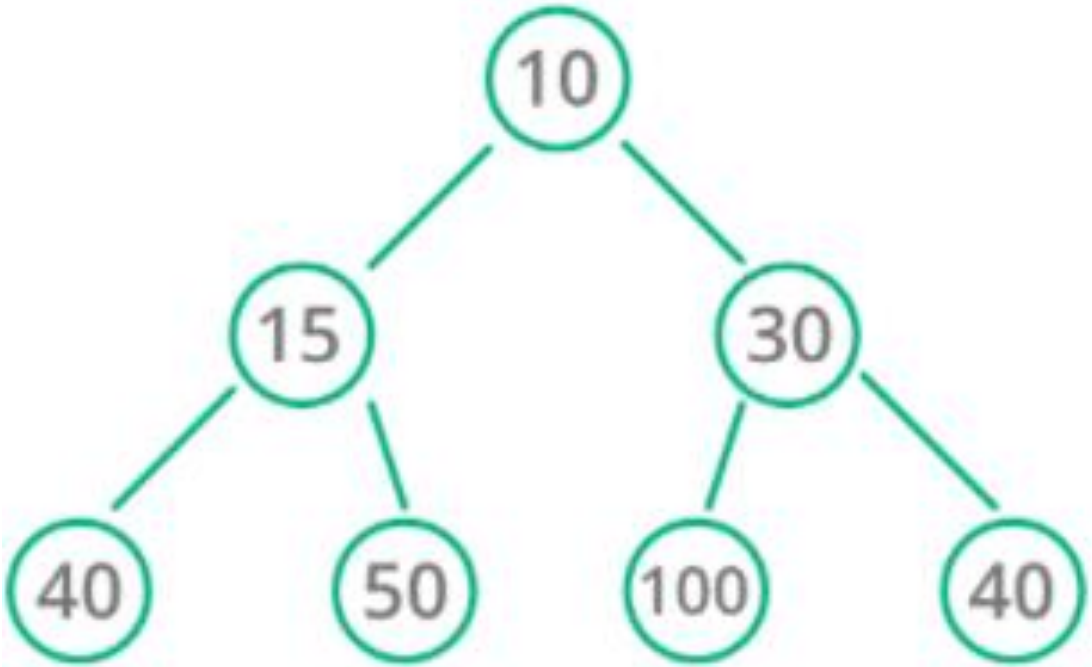


Tree (Heap) cont…

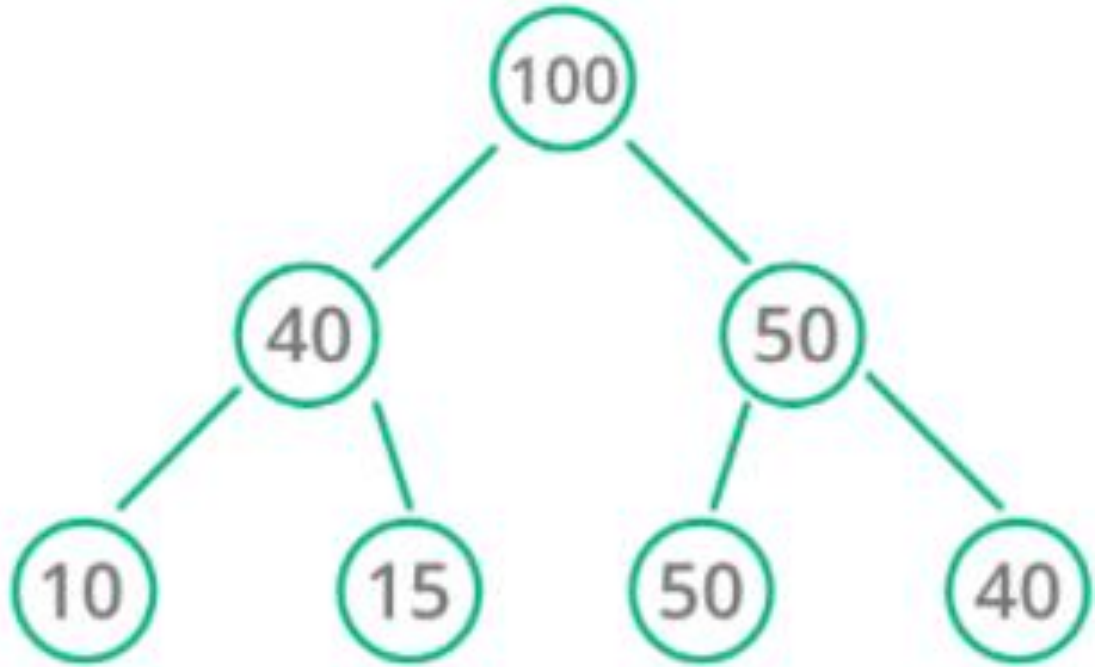
- **Min-Heap:** In a Min-Heap the key present at the root node must be minimum among the keys present at all of it's children.
- The same property must be recursively true for all sub-trees in that Binary Tree.



Tree (Heap) cont...



Min Heap



Max Heap

Tree (Heap) cont…
Insertion

- Suppose we want to create **max heap** tree for **44, 33, 77, 11, 55, 88, 66**.
- To create the max heap tree, we need to consider the following two cases:

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

Tree (Heap) cont…
Insertion

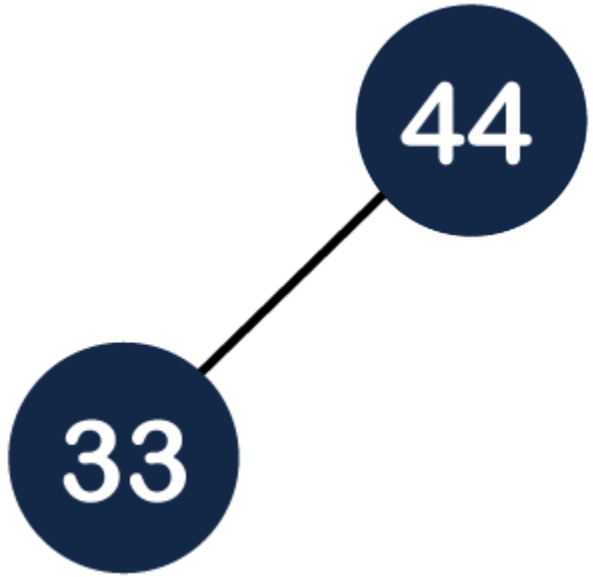
44, 33, 77, 11, 55, 88, 66

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

Tree (Heap) cont...
Insertion

44, 33, 77, 11, 55, 88, 66

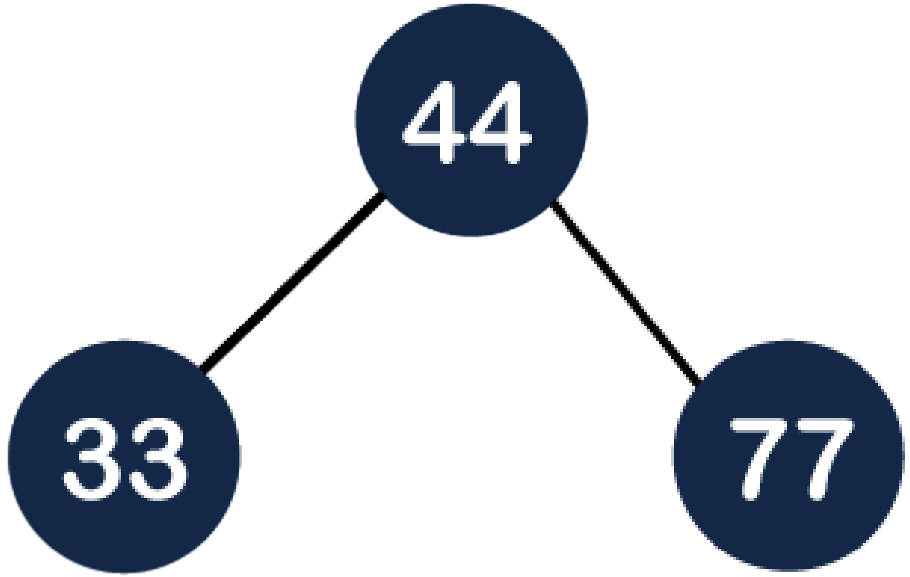
- **Step 2:** The next element is 33.
- As we know that in max heap root node will be maximum. So 33 will be child.
- Furthermore insertion in the binary tree always starts from the left side so 33 will be added at the left of 44 as shown below:



Tree (Heap) cont…
Insertion

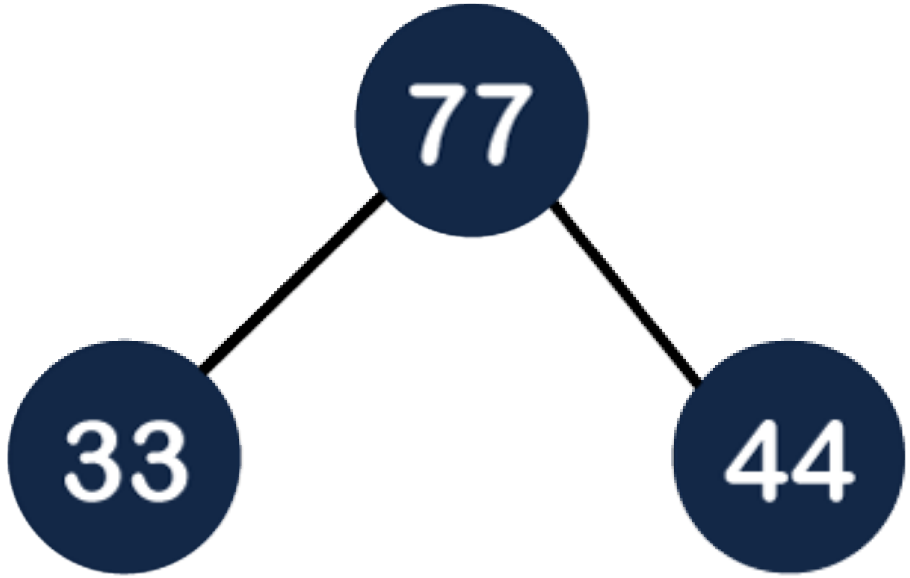
44, 33, 77, 11, 55, 88, 66

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



Tree (Heap) cont…
Insertion

44, 33, 77, 11, 55, 88, 66

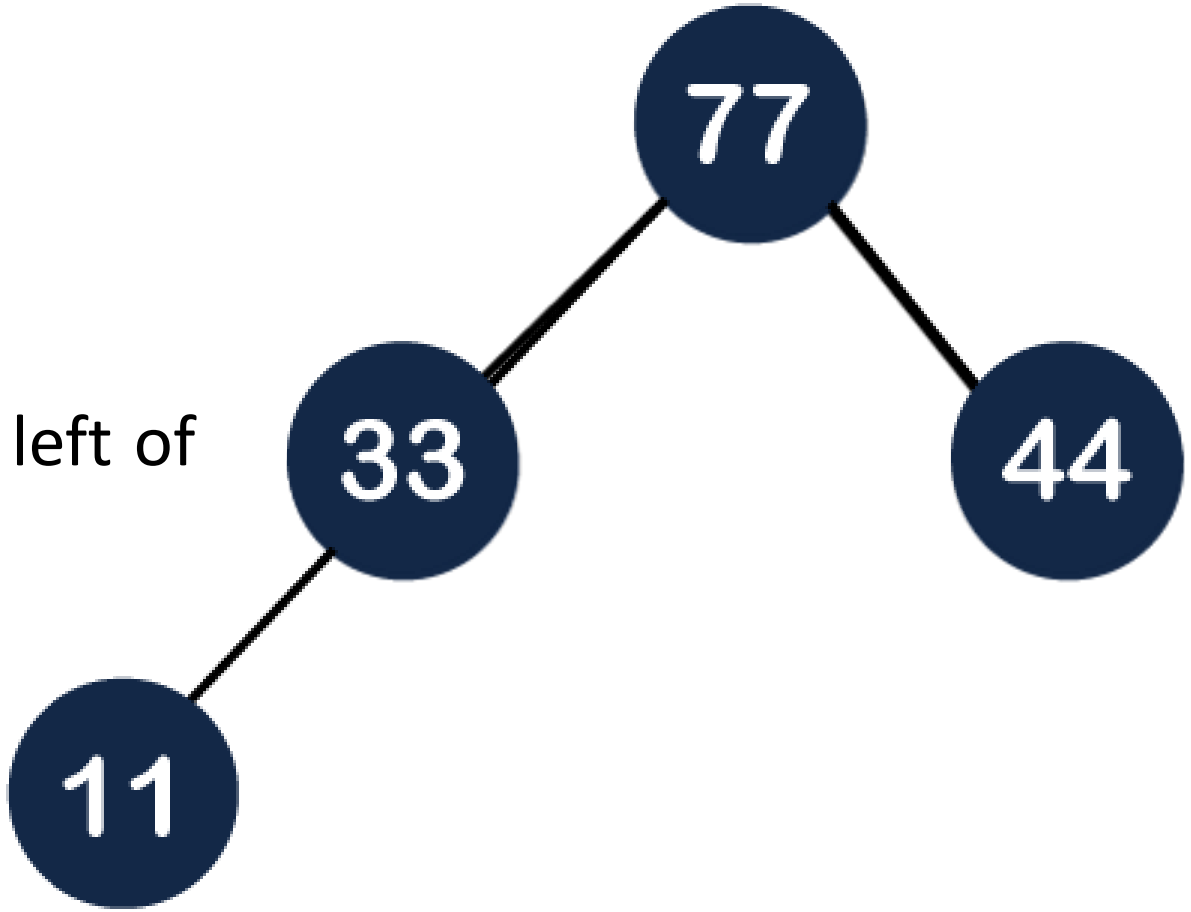


- **Step 3:**
- As we can observe in the tree that it ***does not satisfy the max heap property***, i.e., parent node 44 is less than the child 77.
- So, we will swap these two values as shown here:

Tree (Heap) cont…
Insertion

44, 33, 77, 11, 55, 88, 66

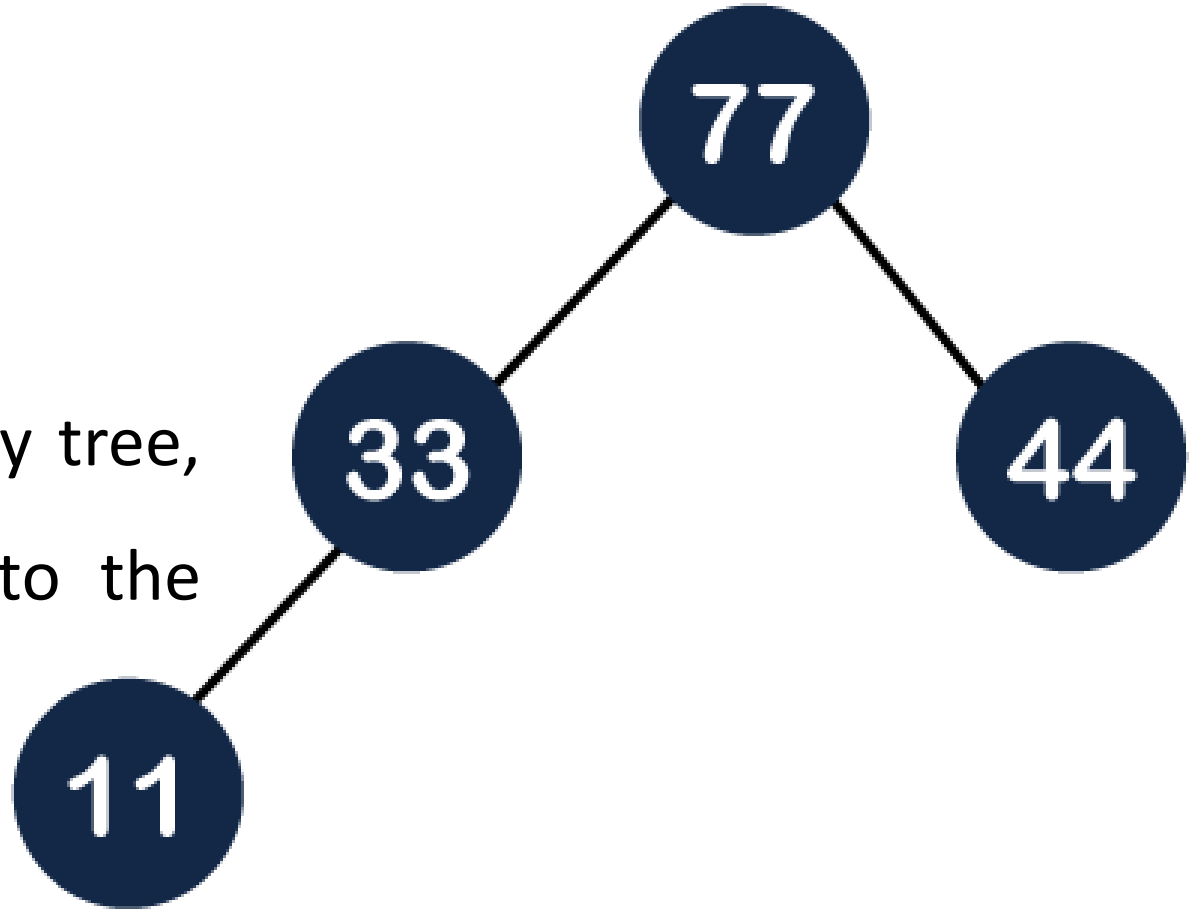
- **Step 4:**
- The next element is 11.
- The node 11 is added to the left of 33 as shown below:



Tree (Heap) cont…
Insertion

44, 33, 77, 11, 55, 88, 66

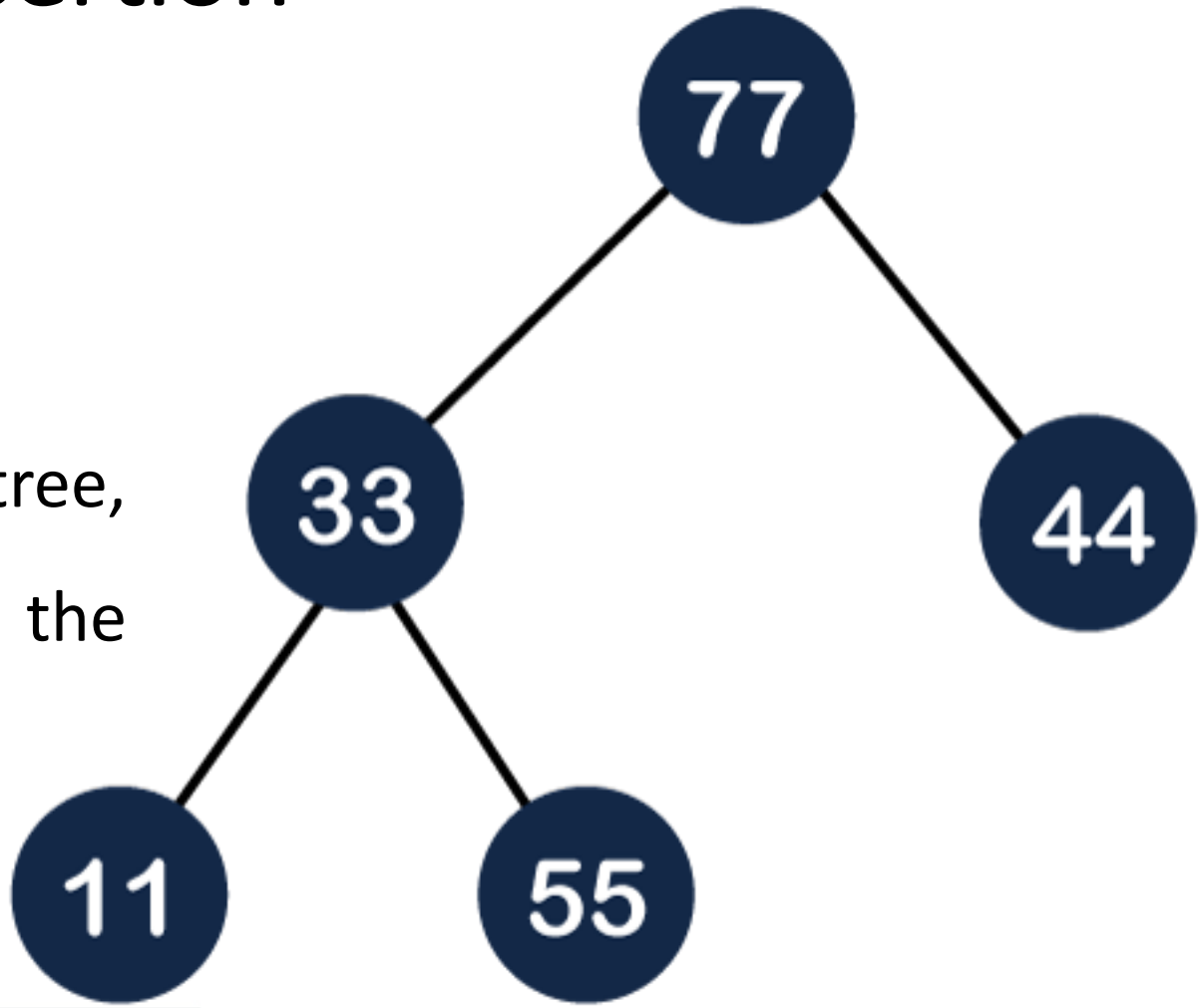
- **Step 5:**
- The next element is 55.
- To make it a complete binary tree, we will add the node 55 to the right of 33 as shown below:



Tree (Heap) cont…
Insertion

44, 33, 77, 11, 55, 88, 66

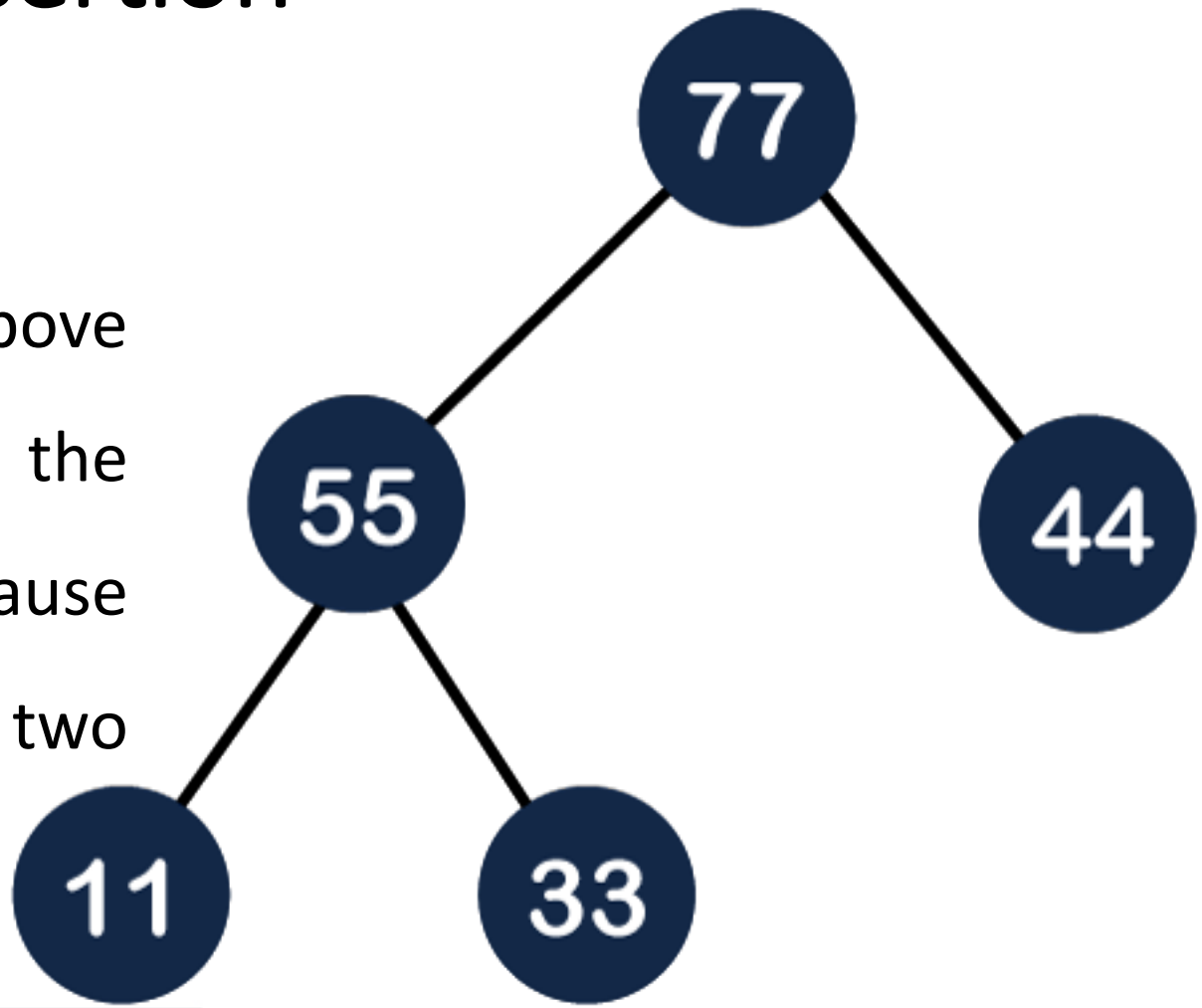
- **Step 5:**
- The next element is 55.
- To make it a complete binary tree, we will add the node 55 to the right of 33 as shown below:



Tree (Heap) cont…
Insertion

44, 33, 77, 11, 55, 88, 66

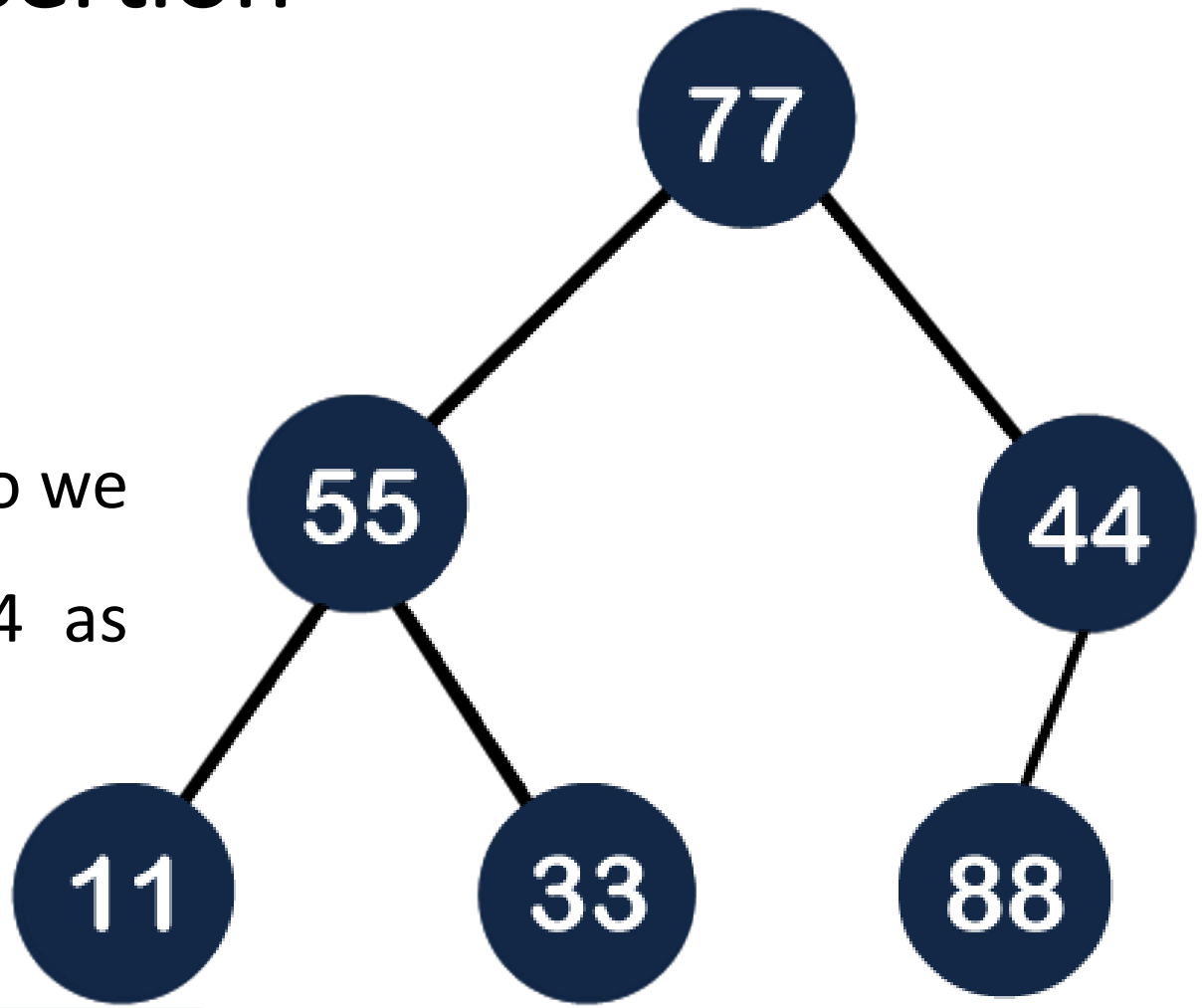
- **Step 5:**
- As we can observe in the above figure that it does not satisfy the property of the **max heap** because 33<55, so we will swap these two values as shown below:



Tree (Heap) cont…
Insertion

44, 33, 77, 11, 55, 88, 66

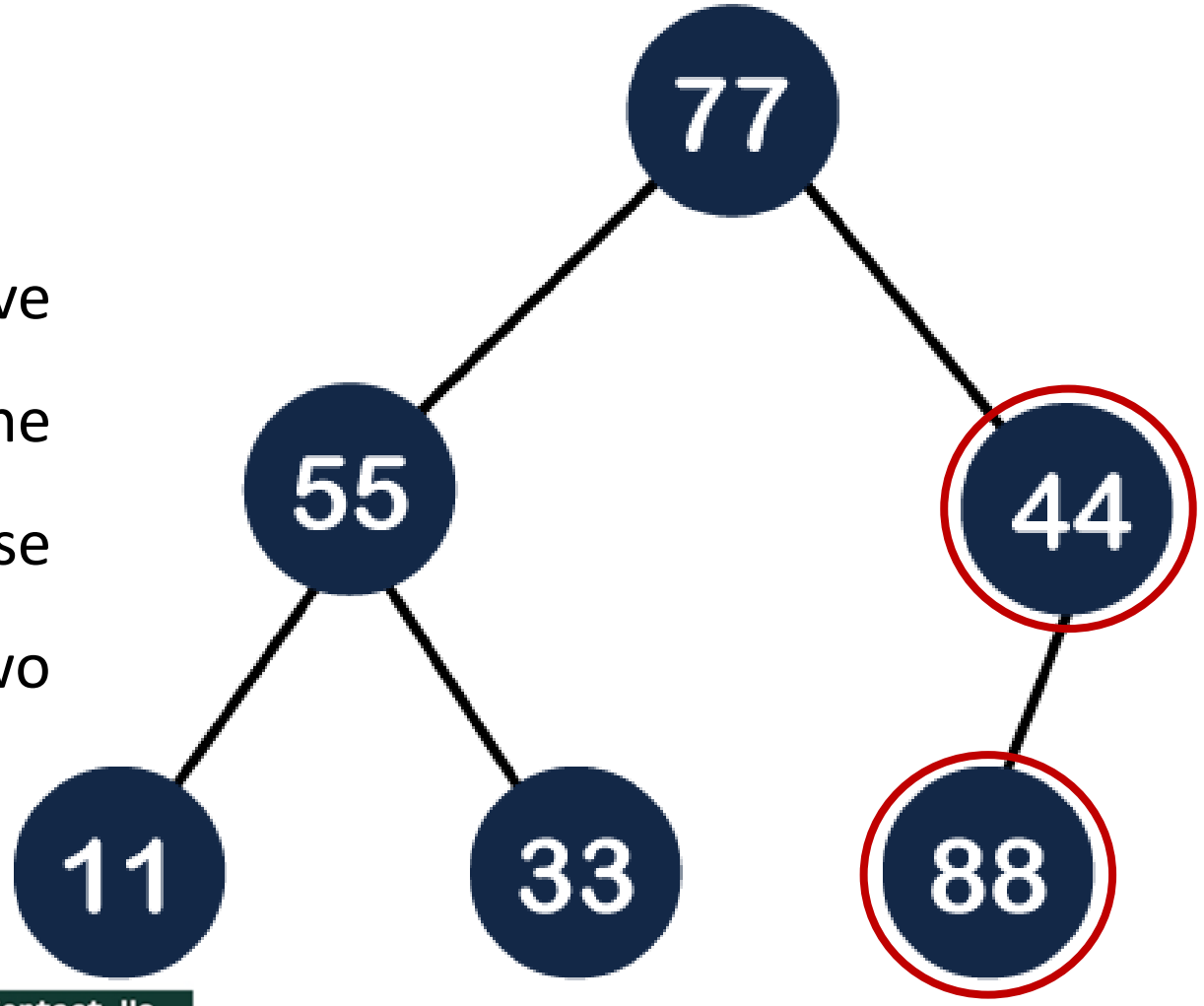
- **Step 6:**
- The next element is 88.
- The left subtree is completed so we will add 88 to the left of 44 as shown below:



Tree (Heap) cont…
Insertion

44, 33, 77, 11, 55, 88, 66

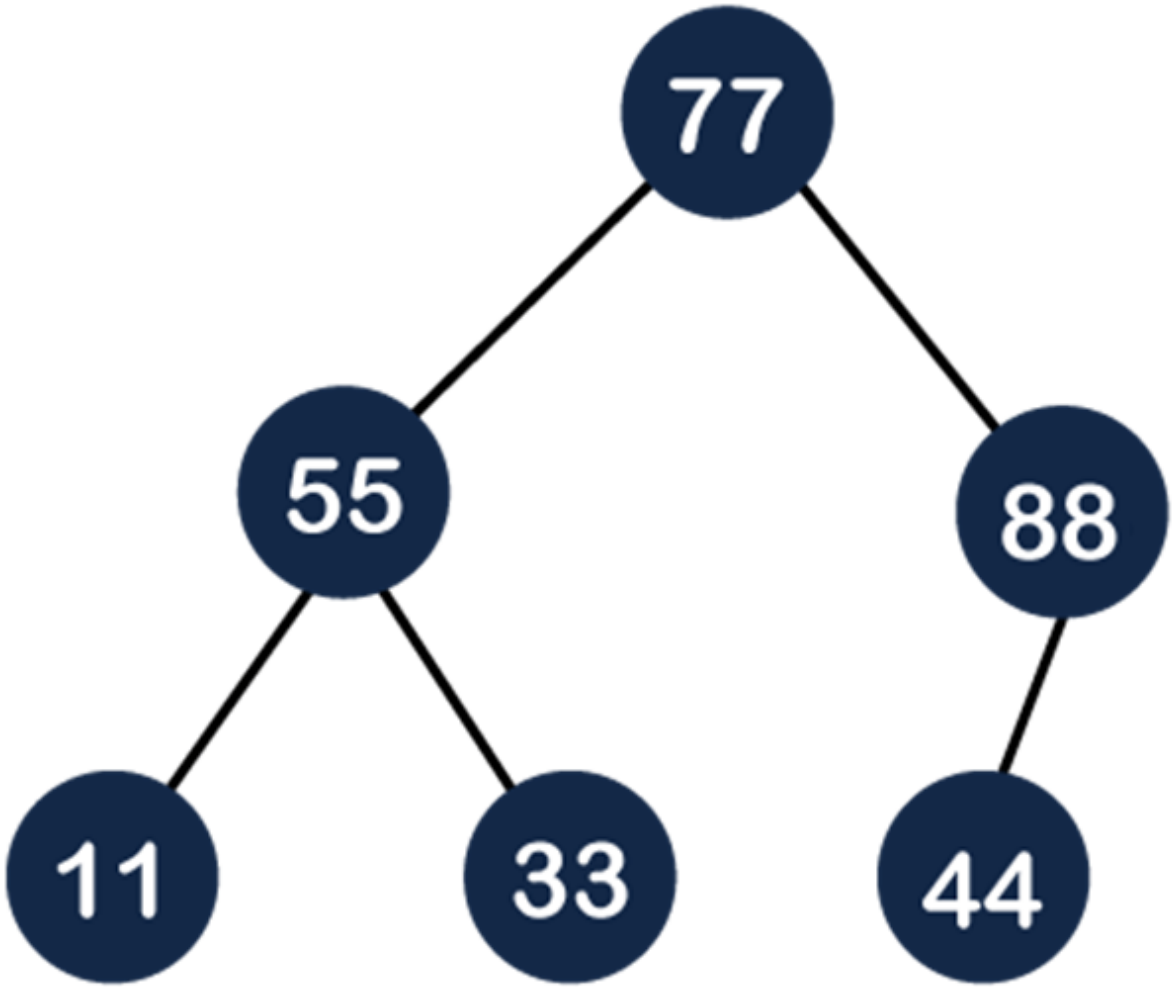
- **Step 6:**
- As we can observe in the above figure that it does not satisfy the property of the max heap because 44<88, so we will swap these two values as shown below:



Tree (Heap) cont…
Insertion

44, 33, 77, 11, 55, 88, 66

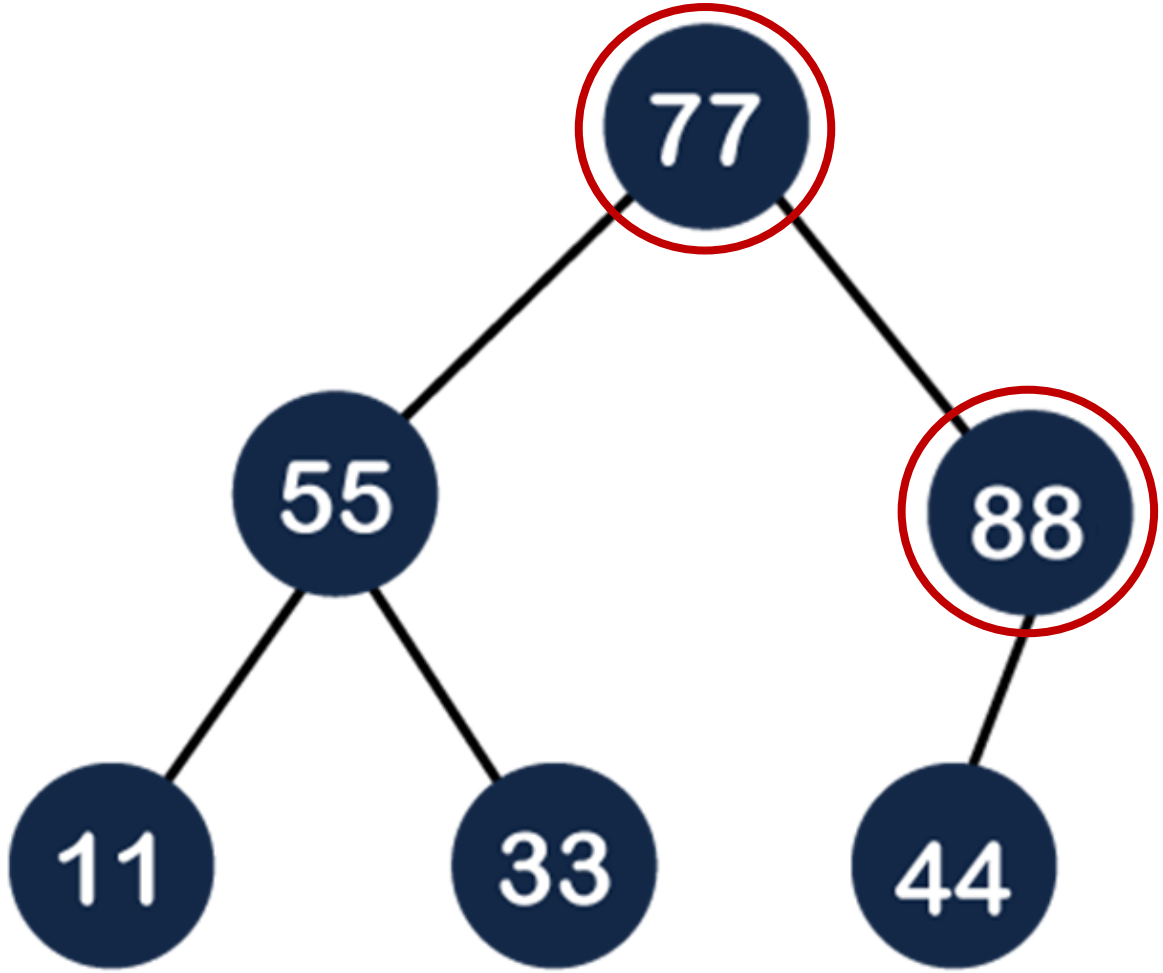
- **Step 6:**
- As we can observe in the above figure that it does not satisfy the property of the max heap because 44<88, so we will swap these two values as shown below:



Tree (Heap) cont…
Insertion

44, 33, 77, 11, 55, 88, 66

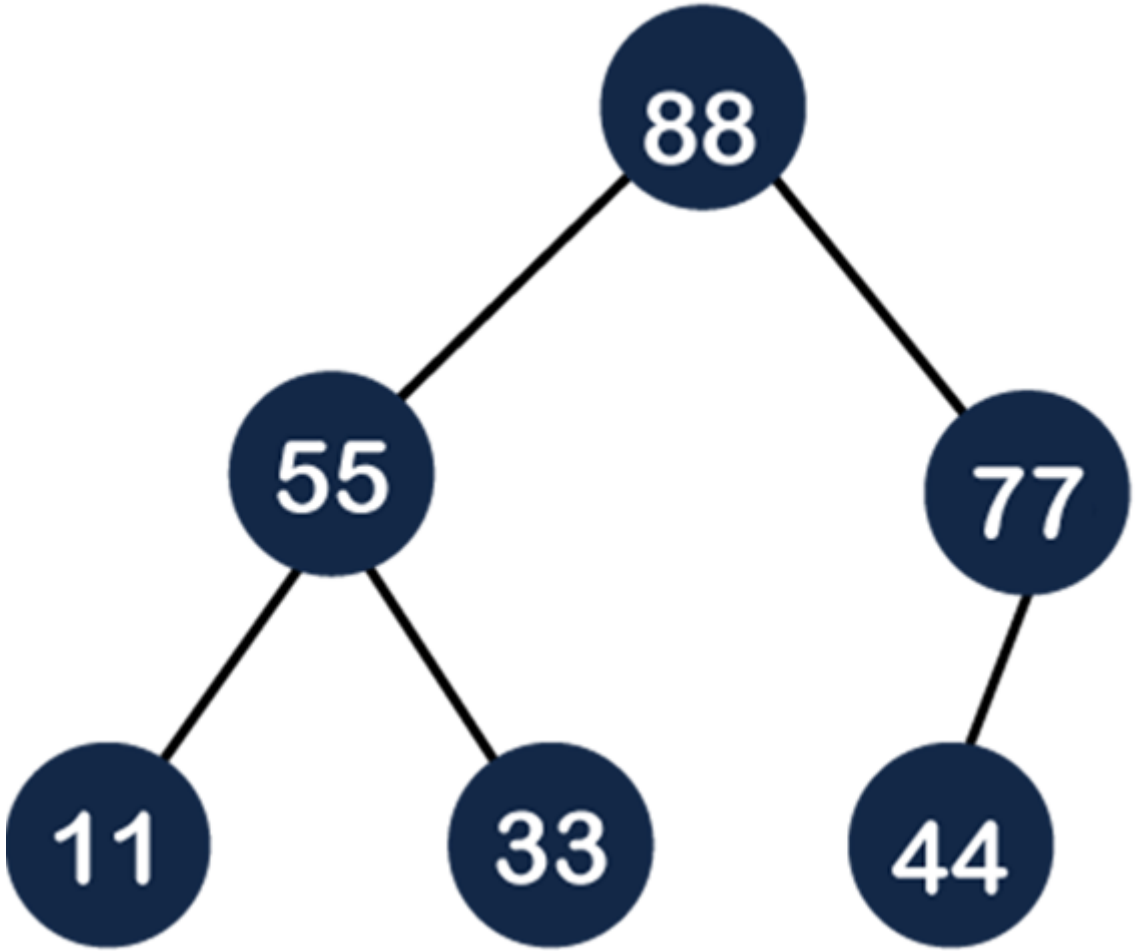
- **Step 6:**
- Again, it is violating the max heap property because 88>77 so we will swap these two values as shown below:



Tree (Heap) cont…
Insertion

44, 33, 77, 11, 55, 88, 66

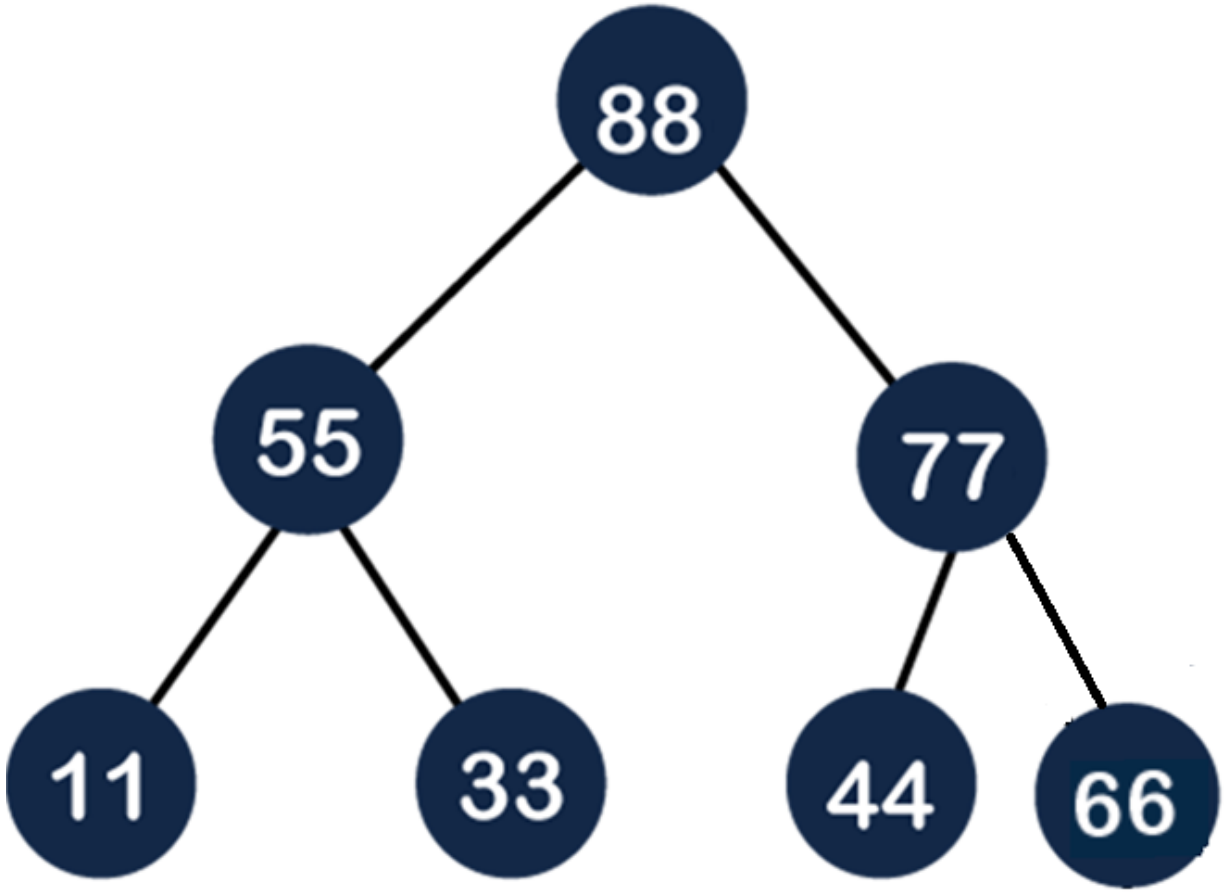
- **Step 6:**
- Again, it is violating the max heap property because 88>77 so we will swap these two values as shown below:



Tree (Heap) cont…
Insertion

44, 33, 77, 11, 55, 88, 66

- **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, then right of 88, as shown here:



Tree (Heap) cont…
Insertion

44, 33, 77, 11, 55, 88, 66

- **Step 7:**
- In this above figure, we can observe that the tree satisfies the property of max heap; therefore, it is a max-heap tree.