Data Structure
Algorithms & Applications

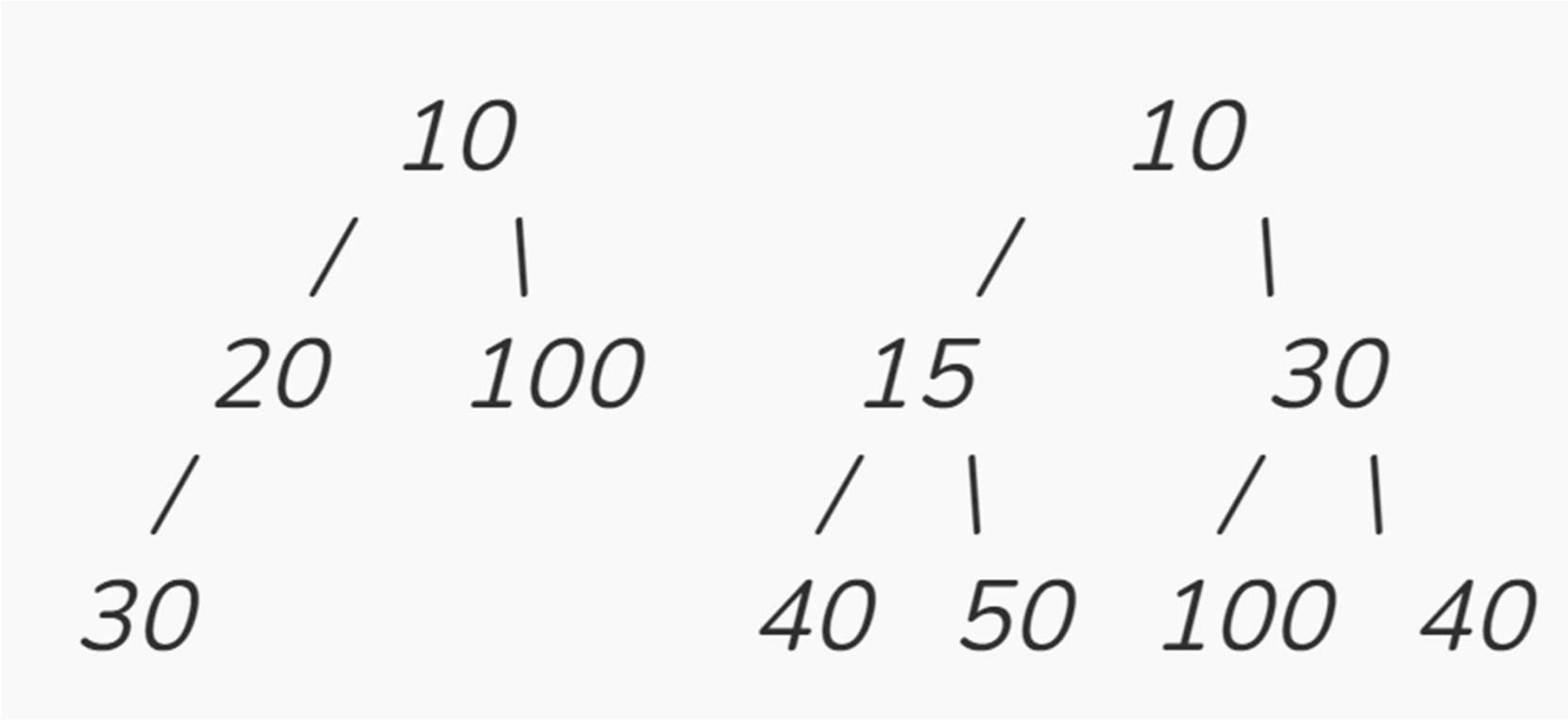
CT-159

Prepared by
Muhammad Kamran

Binary Heap

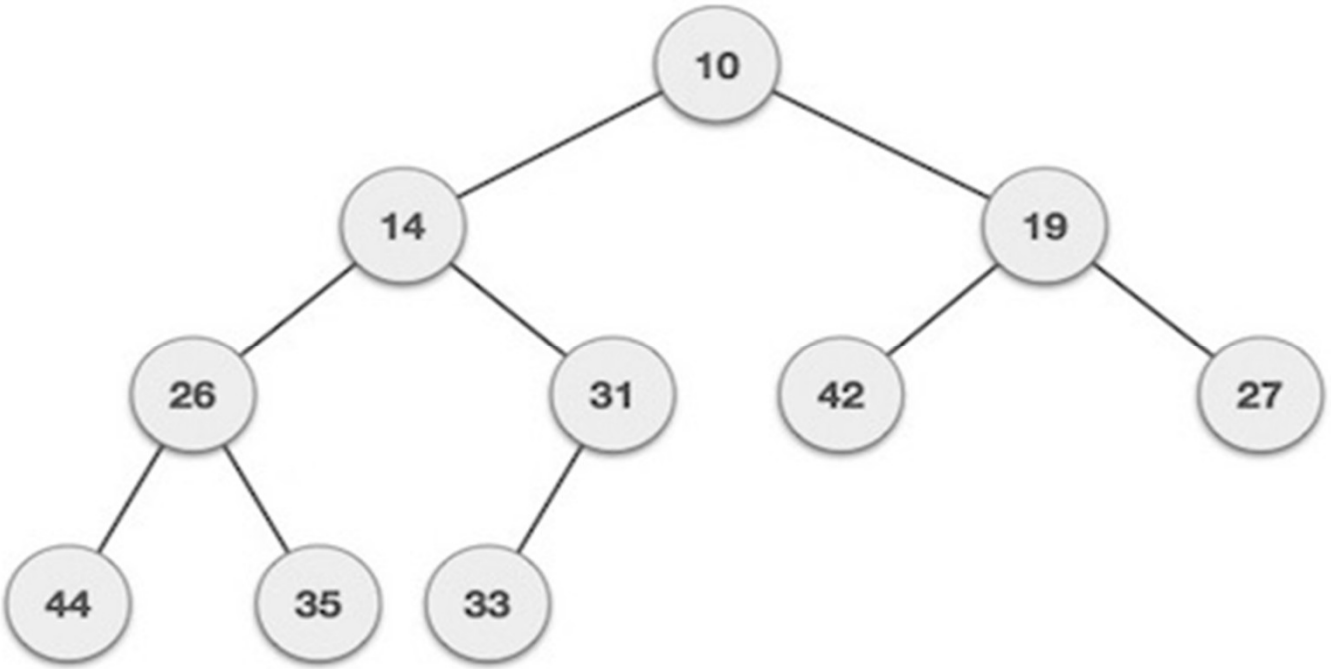
- A Binary Heap is a complete Binary Tree which is used to store data efficiently to get the max or min element based on its structure.
- A Binary Heap is either Min Heap or Max Heap.
- In a Min Binary Heap, the key at the root must be minimum among all keys present in Binary Heap. The same property must be recursively true for all nodes in Binary Tree.

Binary Heap



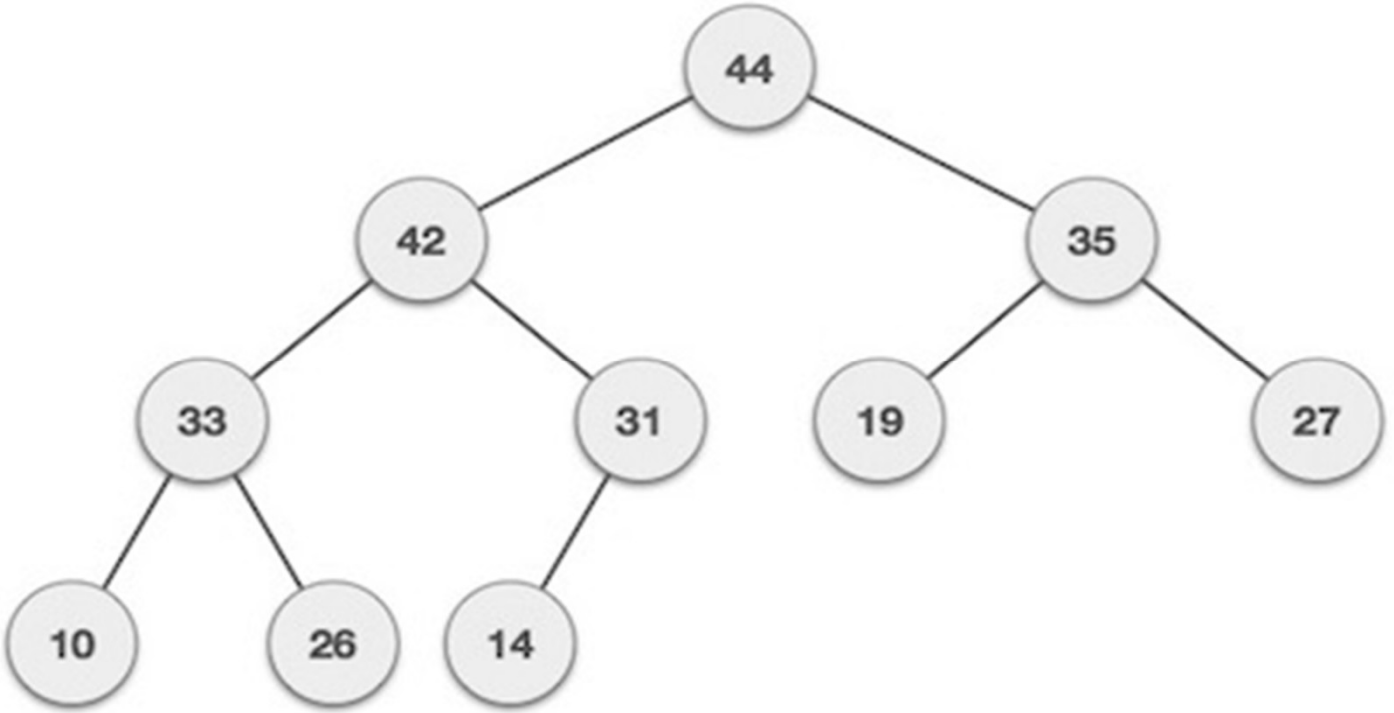
Min. Heap

- Where the value of the root node is less than or equal to either of its children.



Max. Heap

- Where the value of the root node is greater than or equal to either of its children.



Binary Heap Representation

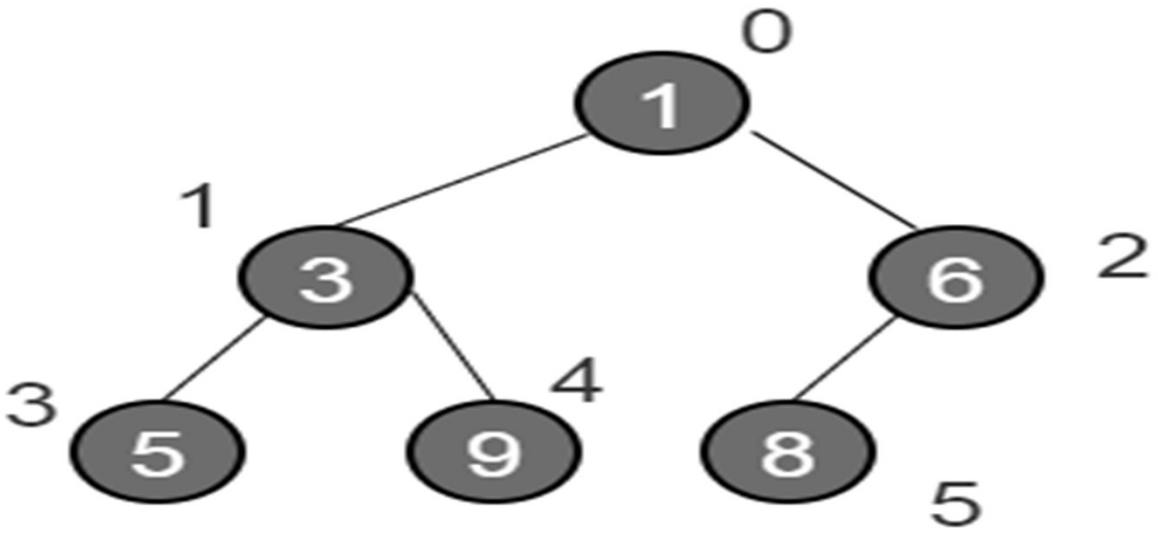
A binary heap is typically represented as an array.

The root element will be at Arr[0].

The below table shows indices of other nodes for the ith node, i.e., Arr[i]:

Arr[(i-1)/2]	Returns the parent node
Arr[(2*i)+1]	Returns the left child node
Arr[(2*i)+2]	Returns the right child node

Binary Heap Representation



1	3	6	5	9	8
0	1	2	3	4	5

Applications

- Heap Sort uses Binary Heap to sort an array in O(nLogn) time.
- Priority queues can be efficiently implemented using Binary Heap because of its operation.
- Minimum Spanning Tree.

Max. Heap Creation Algorithm

- **Step 1** − Create a new node at the end of heap.
- **Step 2** − Assign new value to the node.
- **Step 3** − Compare the value of this child node with its parent.
- **Step 4** − If value of parent is less than child, then swap them.
- **Step 5** − Repeat step 3 & 4 until Heap property holds.

Max. Heap Creation

Input 35 33 42 10 14 19 27 44 26 31

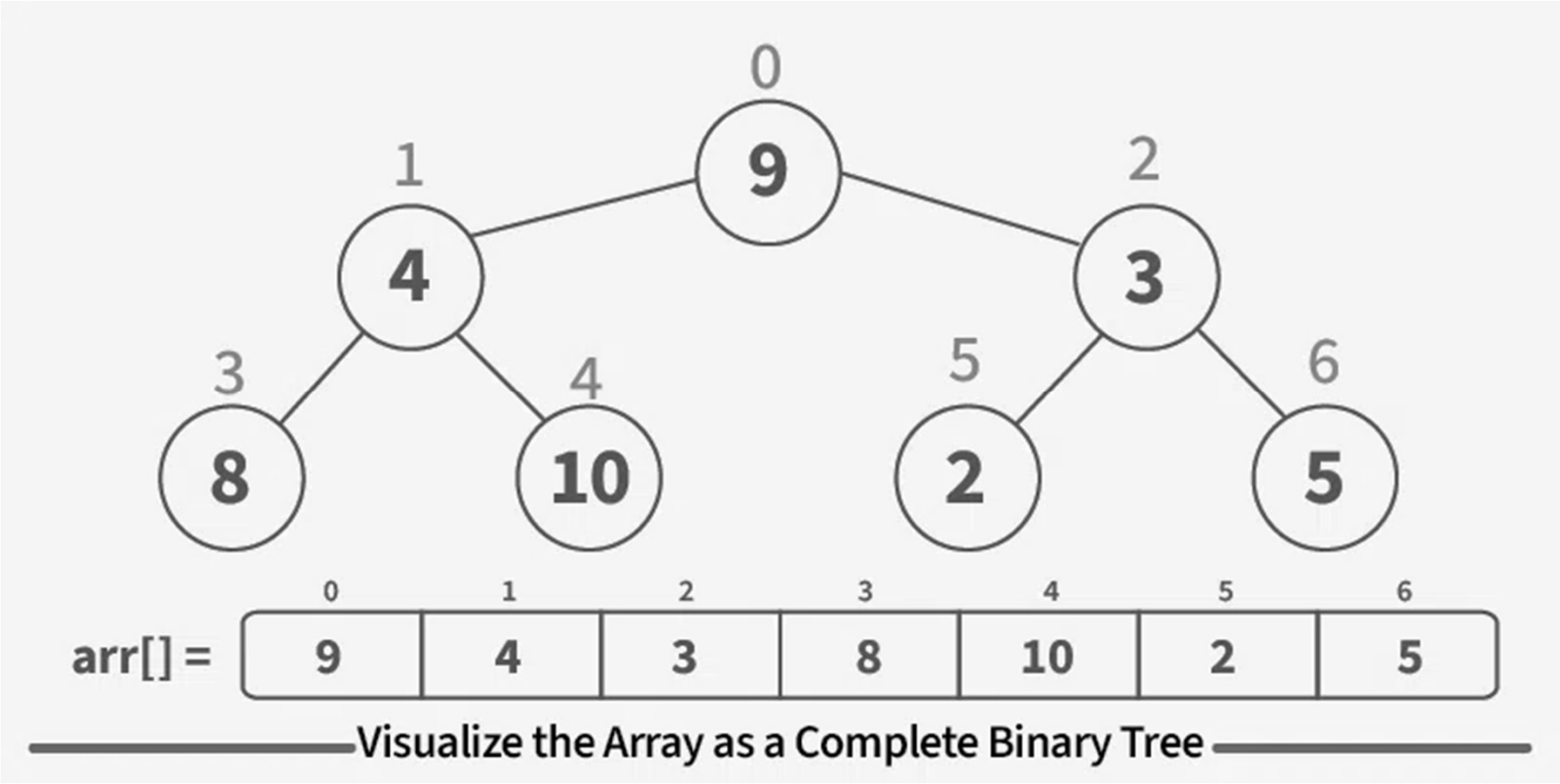
Heap Sort Algorithm

First convert the array into a max heap using heapify,

The array elements are re-arranged to follow heap properties. Then one by one delete the root node of the Max-heap and replace it with the last node and heapify. Repeat this process while size of heap is greater than 1.

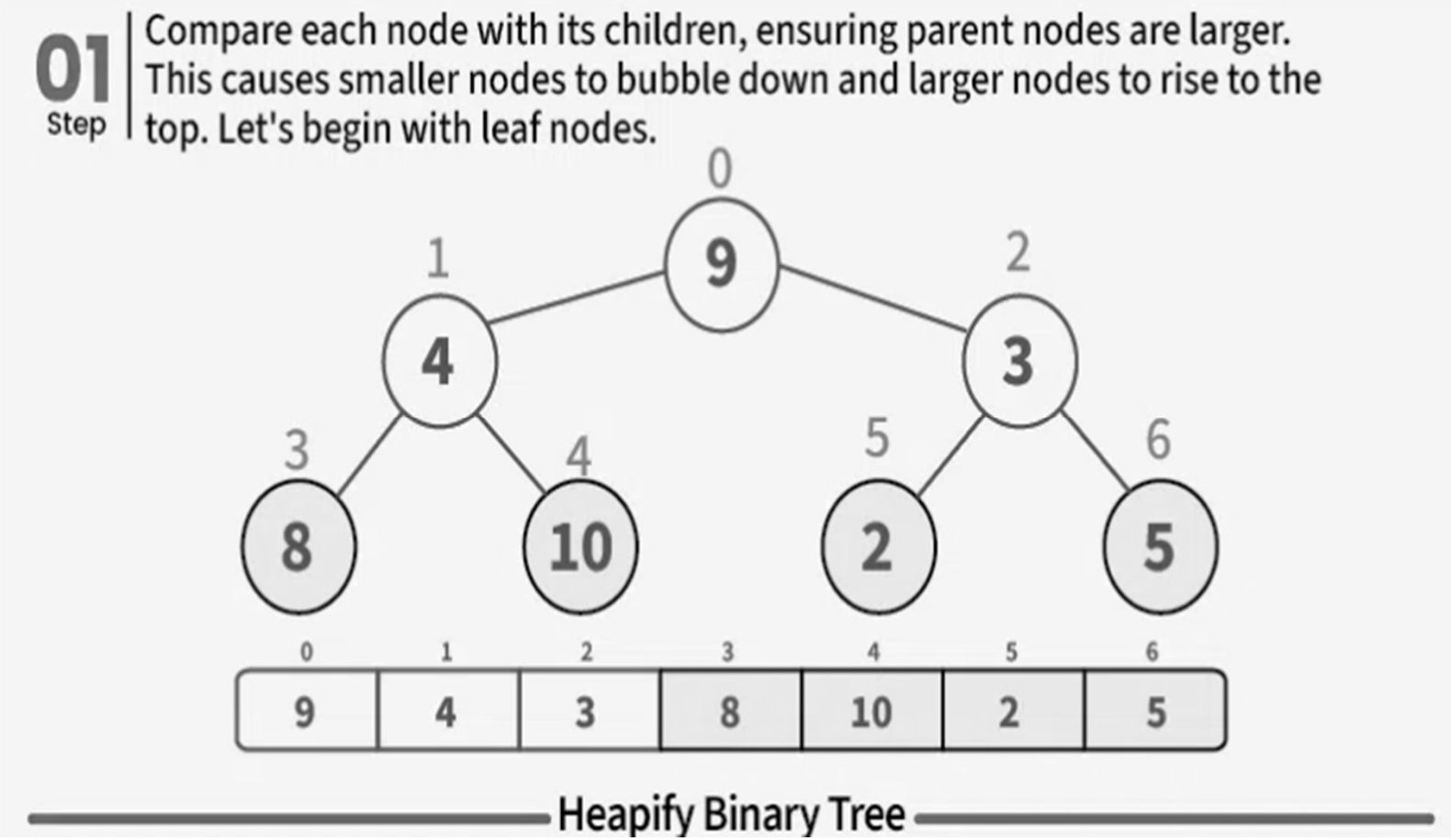
- Rearrange array elements so that they form a Max Heap.
- Repeat the following steps until the heap contains only one element:
    - Swap the root element of the heap (which is the largest element in current heap) with the last element of the heap.
    - Remove the last element of the heap (which is now in the correct position). We mainly reduce heap size and do not remove element from the actual array.
    - Heapify the remaining elements of the heap.
- Finally we get sorted array.

Step 1: Treat the Array as a Complete Binary Tree

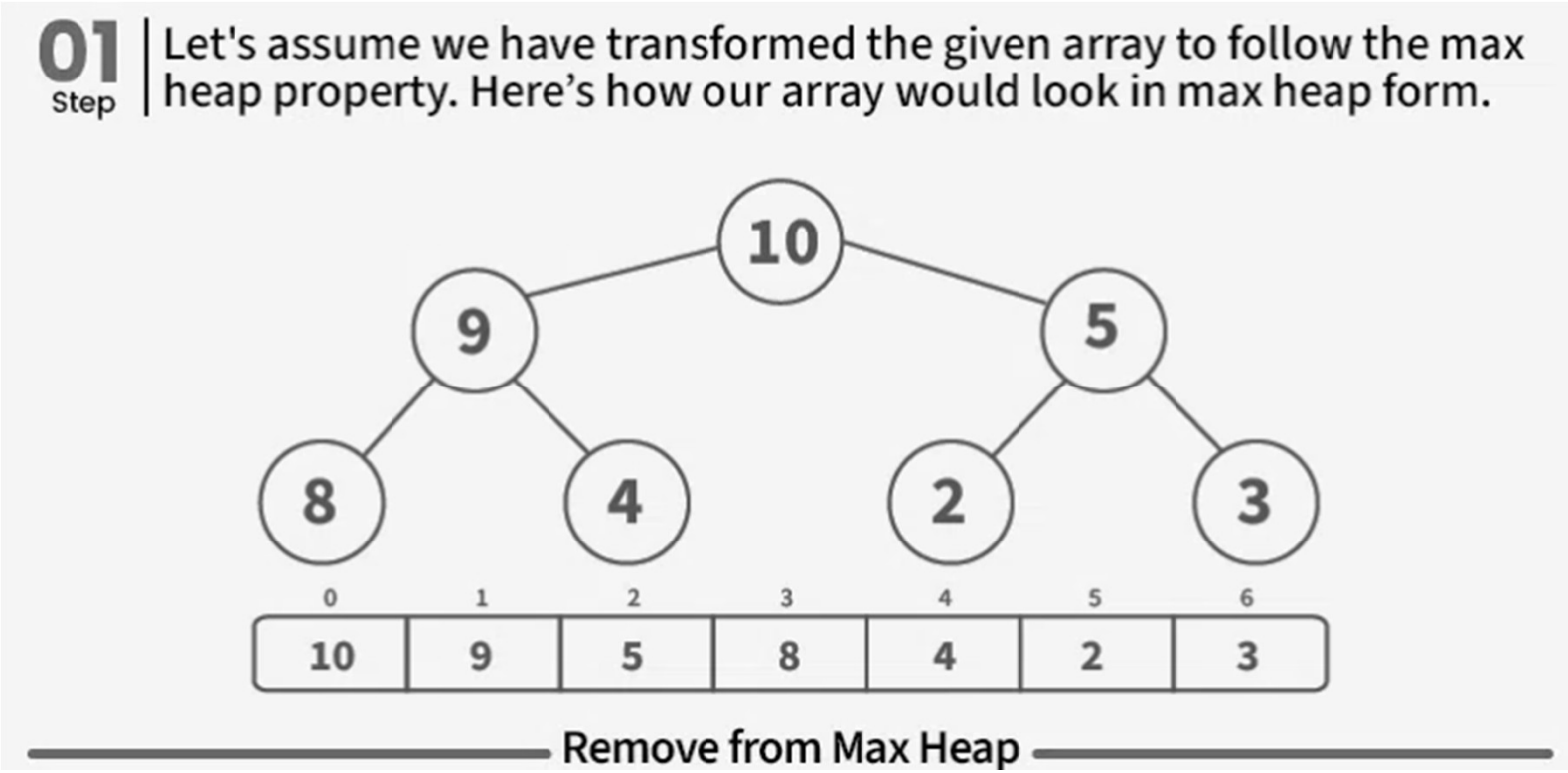


For an array of size **n**, the root is at index **0**, the left child of an element at index **i** is at **2i + 1**, and the right child is at **2i + 2**.

Step 2: Build a Max Heap



Step 3: Sort the array by placing largest element at end of unsorted array.

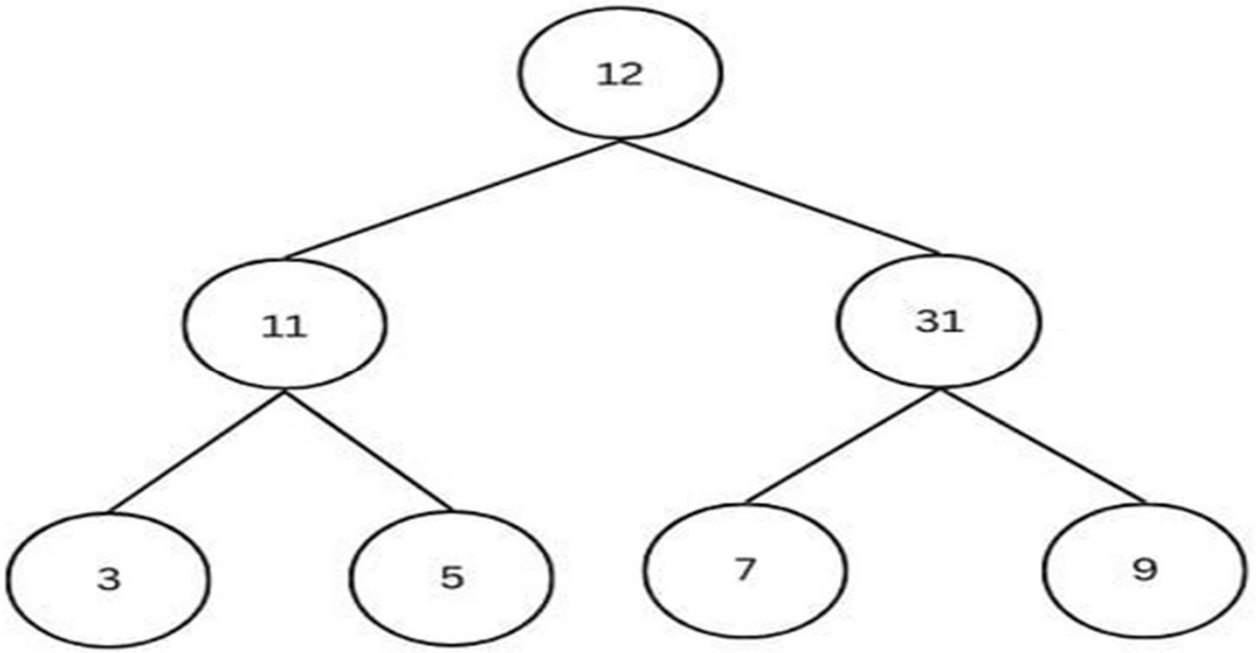


Heap Sort Algorithm Revisited

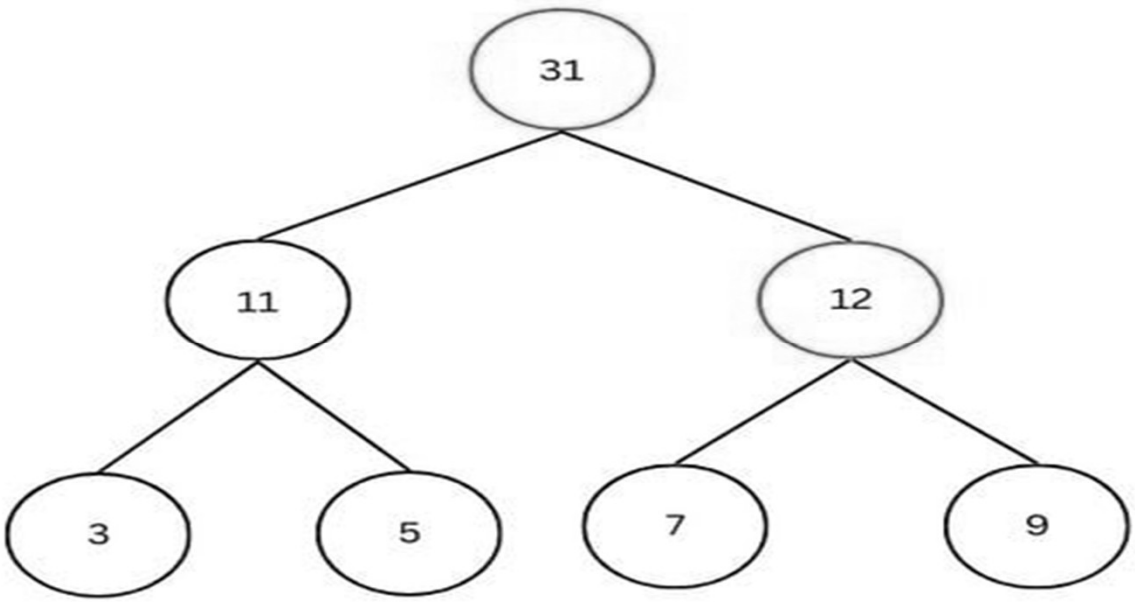
1. Transform the array into a binary tree by inserting each element as a node in a breadth-first manner.
2. Convert the binary tree into a max heap, ensuring that all parent nodes are greater than or equal to their child nodes.
3. Swap the root node — the largest element — with the last element in the heap.
4. Call the heapify() function to restore the max heap property.
5. Repeat steps 3 and 4 until the heap is sorted, and exclude the last element from the heap on each iteration.
6. After each swap and heapify() call, ensure that the max heap property is satisfied.

1. Transform the Array Into a Binary Tree

[12, 11, 31, 3, 5, 7, 9]

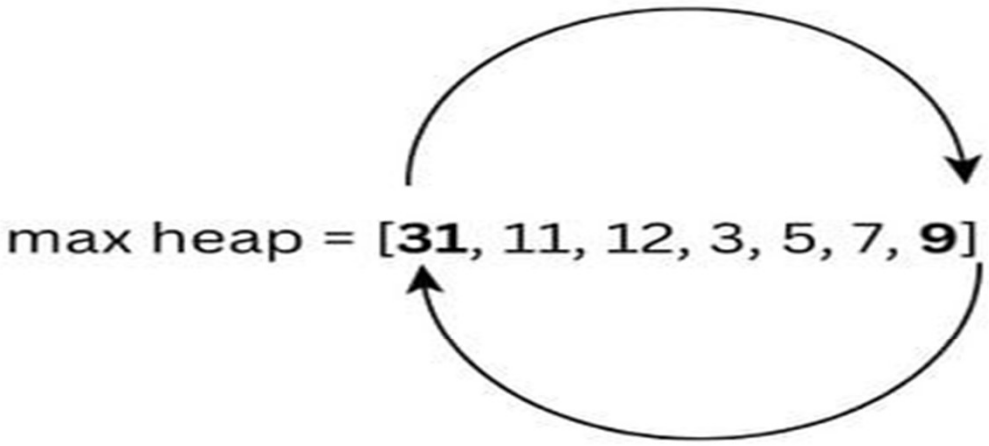


2. Convert the Binary Tree Into a Max Heap



max heap = [31, 11, 12, 3, 5, 7, 9]

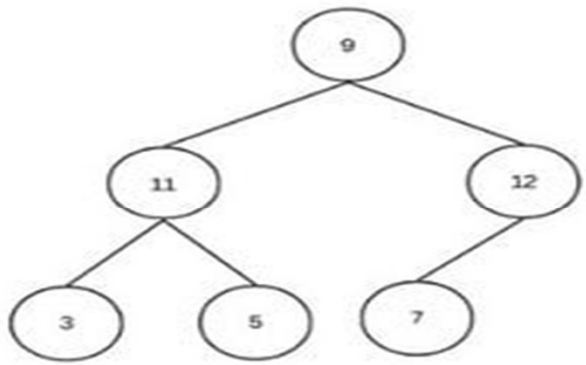
3. Swap the Root Node With the Last Element in the Heap



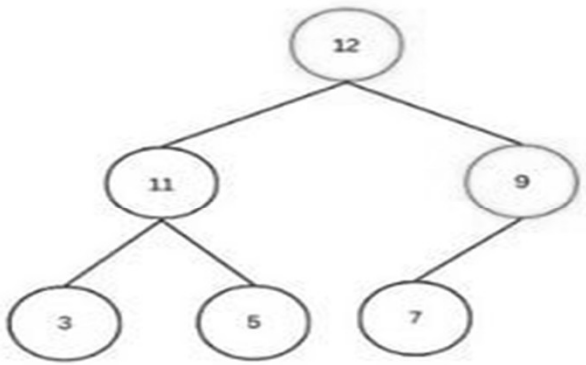
[9, 11, 12, 3, 5, 7, 31]

4. Call the heapify() Function

[9, 11, 12, 3, 4, 7]



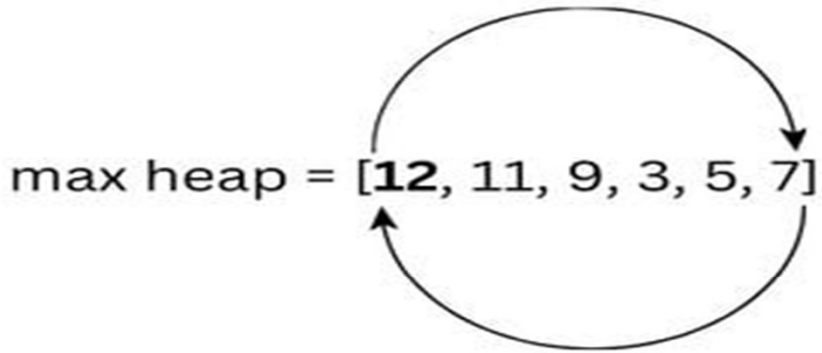
Binary Tree



Max-Heap

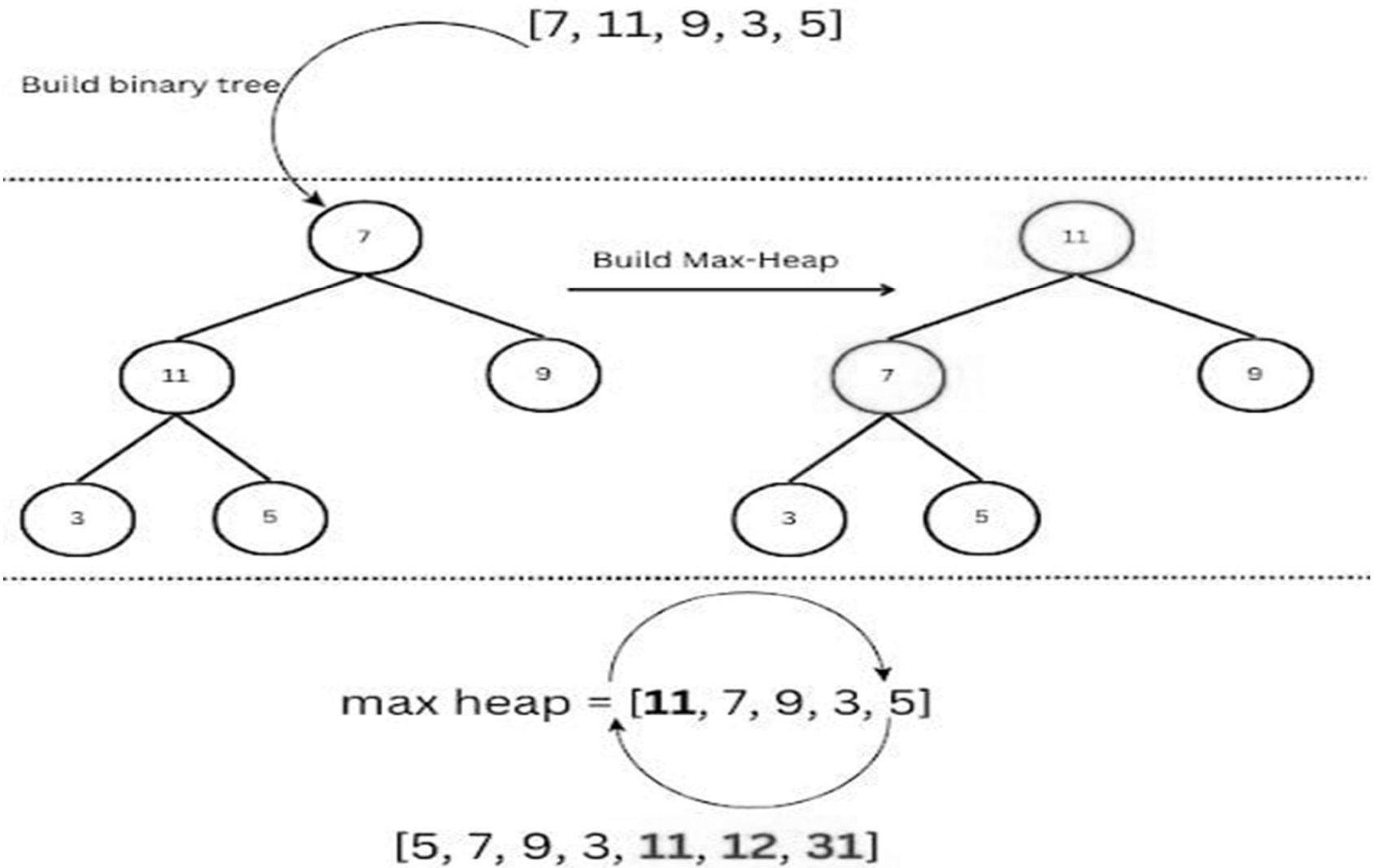
4. Call the heapify() Function

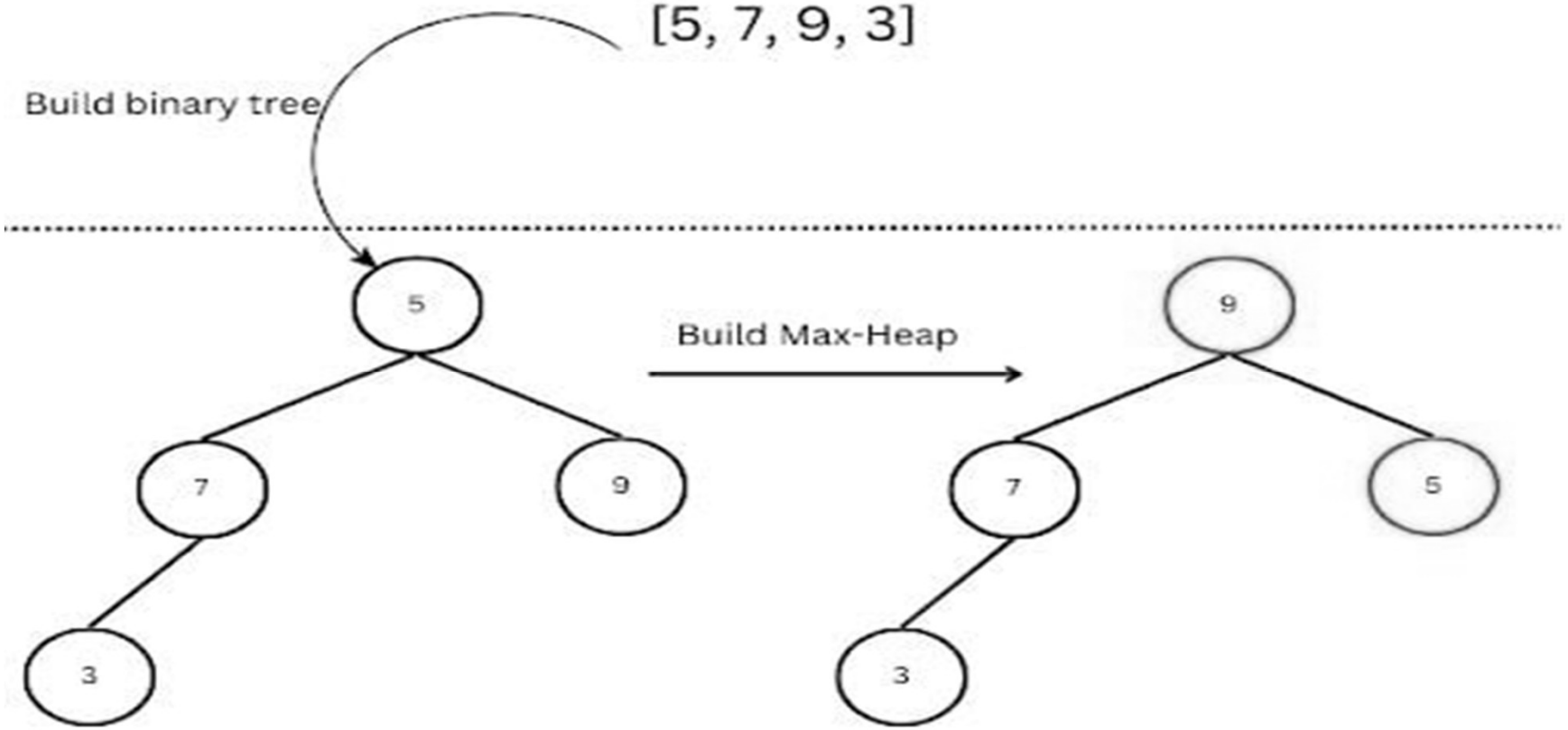
[12, 11, 9, 3, 5, 7]



[7, 11, 9, 3, 5, **12**, **31**]

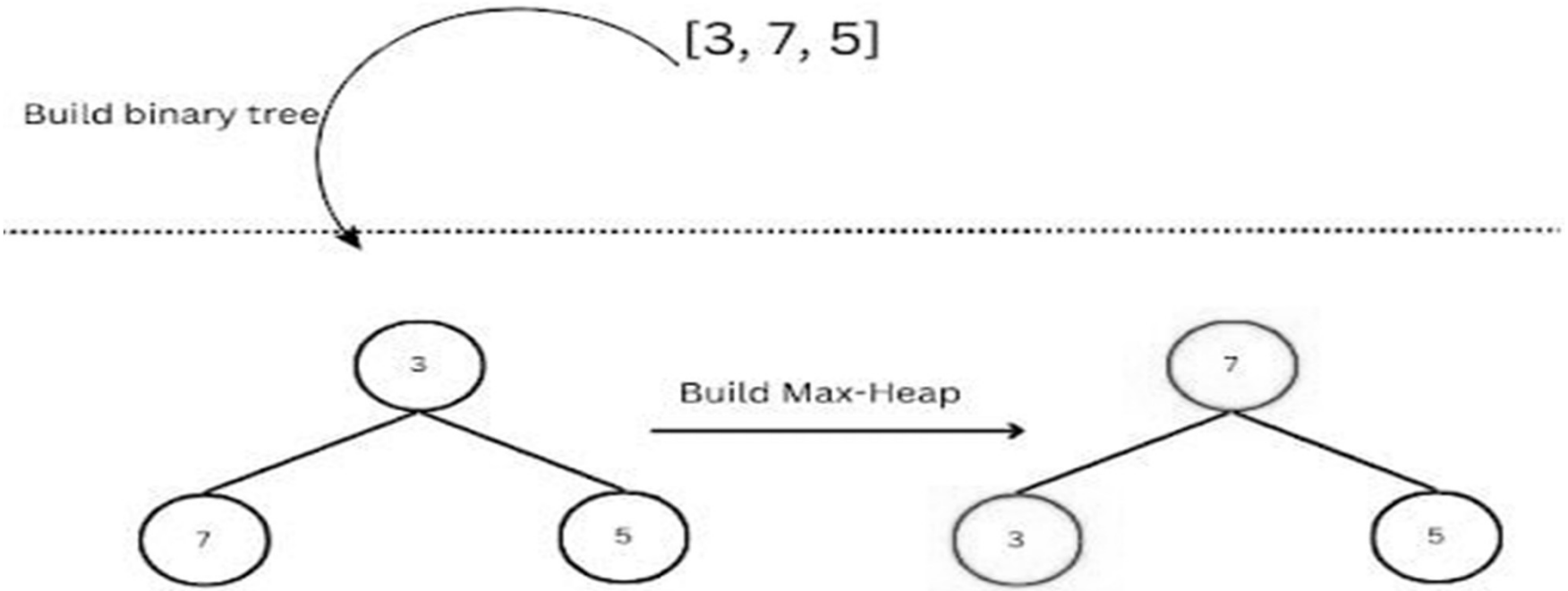
5. Repeat Steps 3 and 4 Until the Heap Is Sorted





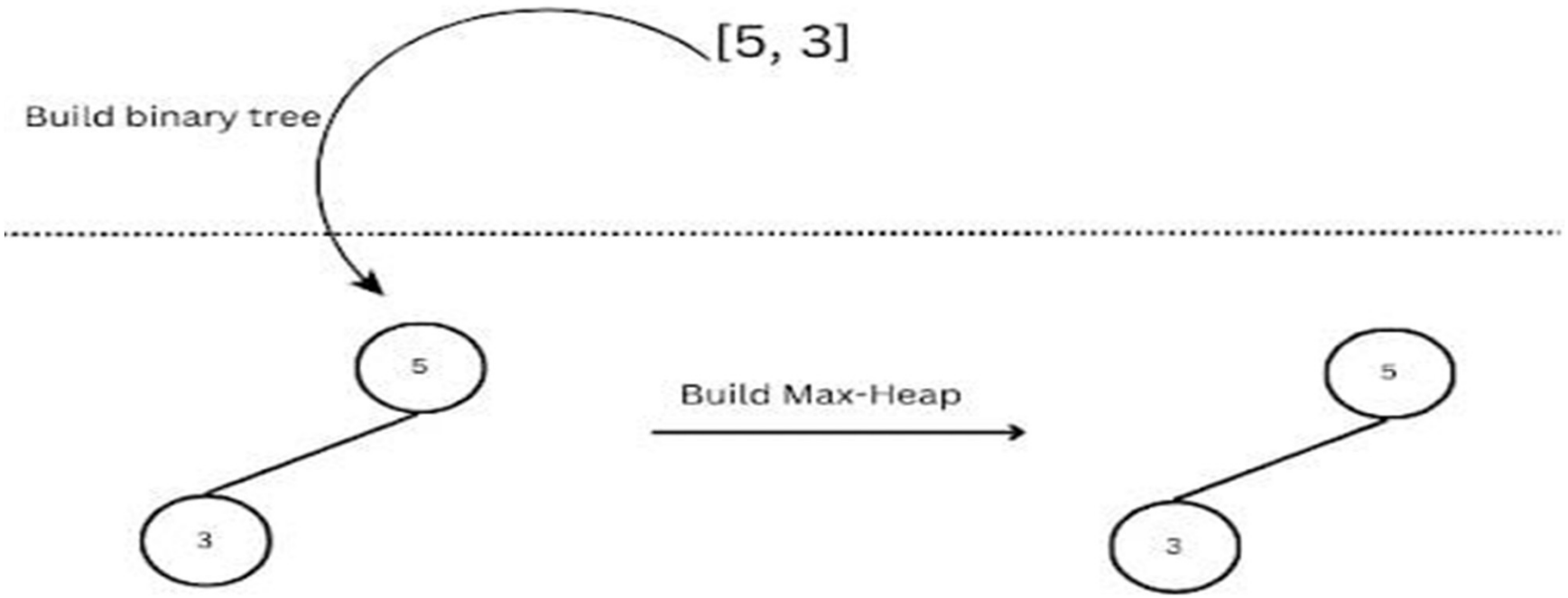
max heap = [**9**, 7, 5, **3**]

[3, 7, 5, **9**, **11**, **12**, **31**]



max heap = [**7**, 3, **5**]

[5, 3, **7**, **9**, **11**, **12**, **31**]



max heap = [**5**,**3**]

[3, **5**, **7**, **9**, **11**, **12**, **31**]

Thank You