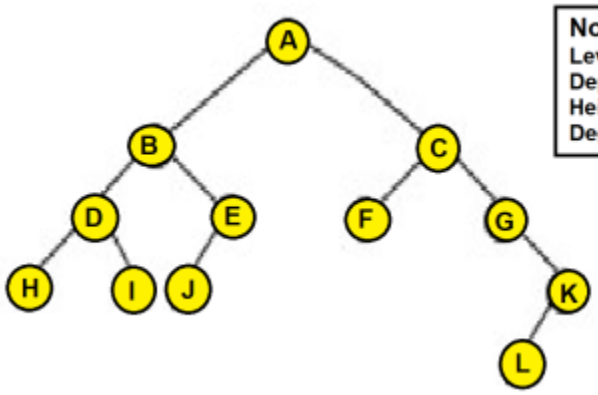
Practice Problems

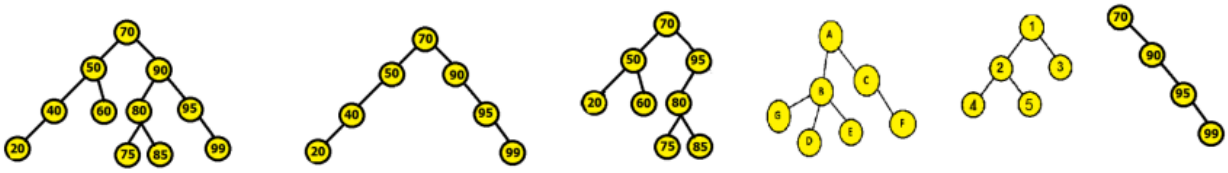
1. Find the level, depth, height and degree of the specified nodes of the following tree.



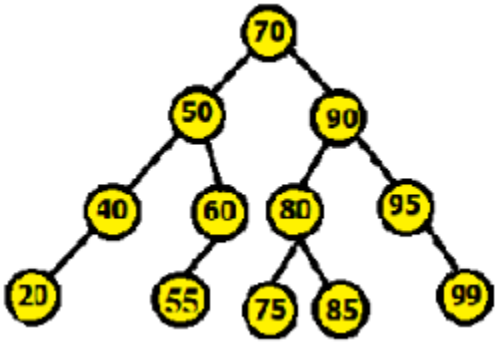
Node A	Node D	Node F	Node C
Level:	Level:	Level:	Level:
Depth:	Depth:	Depth:	Depth:
Height:	Height:	Height:	Height:
Degree:	Degree:	Degree:	Degree:

Node E	Node G	Node L
Level:	Level:	Level:
Depth:	Depth:	Depth:
Height:	Height:	Height:
Degree:	Degree:	Degree:

2. Identify which of the following trees are full, complete, perfect and balanced.



3. Traverse the following trees in pre-order, in-order and post-order and print the elements. Show both simulation and code.

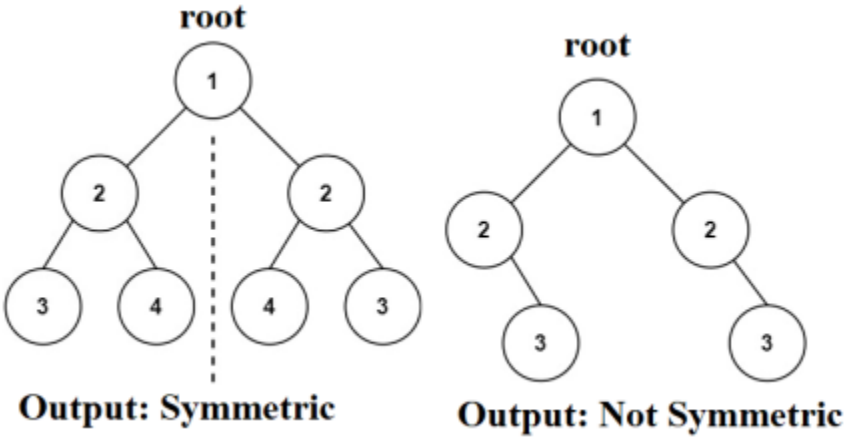


Pre-order:
In-order:
Post-order:

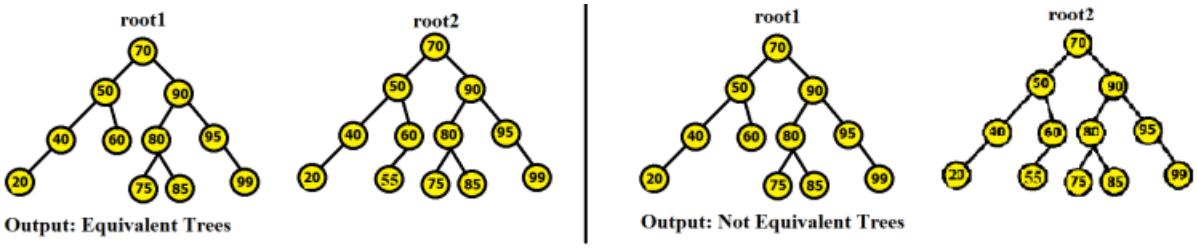
4. Consider the following array and convert it into a binary tree. Show simulation and code.

```
[None, 15, 25, 35, 10, 35, 15, 18, None, None, None, 33, None, 5, None, 19, None, None, None, 16]
```

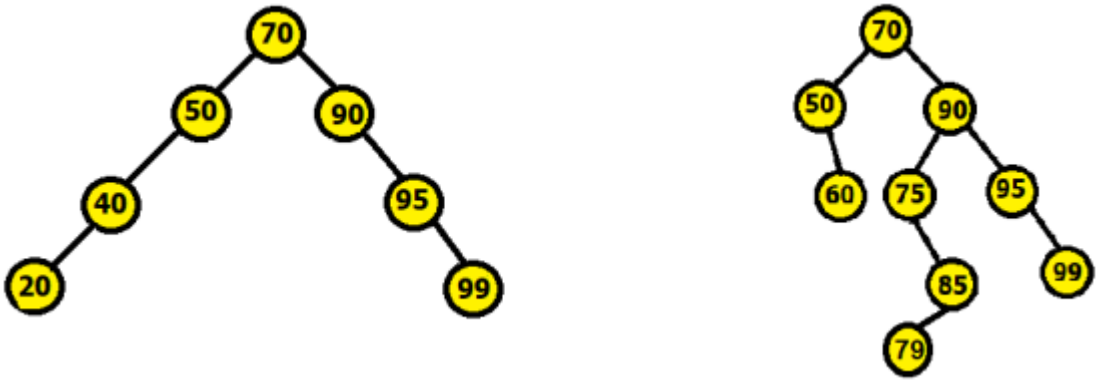
5. Write a Python function **isSymmetric(root)** that takes the root of a binary tree, check whether it is a mirror of itself (i.e., symmetric around its center).



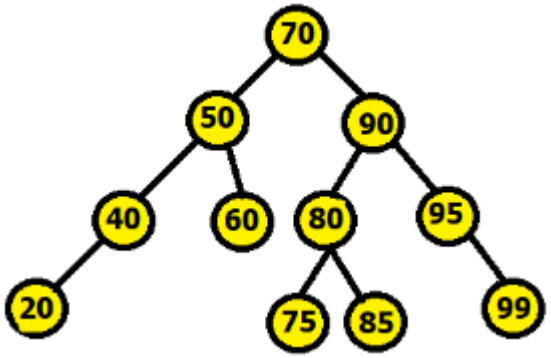
6. Write a Python function **isIdentical(root1, root2)** that takes the roots of two binary trees, check whether they are identical or not).



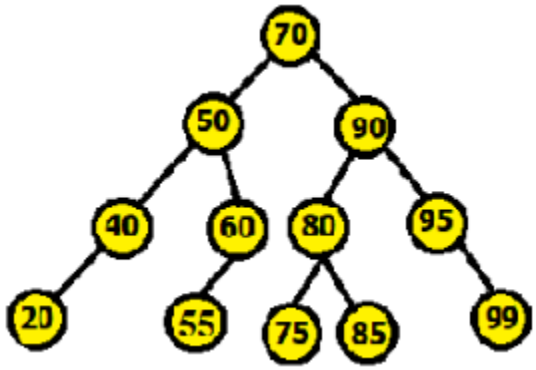
7. Convert the following unbalanced BSTs into balanced BSTs. Show simulation.



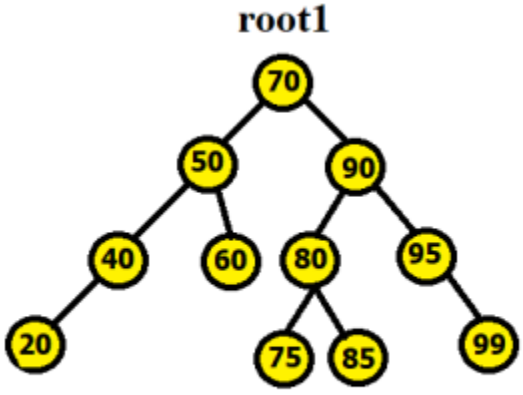
8. Insert keys 65, 105, 69 into the following BST and show the steps. Show simulation and code.



9. Delete keys 20, 95, 50, 70, 75 into the following BST and show the steps. Show simulation and code.



10. Write a python program that takes the root of a tree and finds its inorder successor and predecessor.

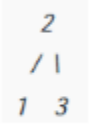


Output: In-order Successor: 75
In-order Predecessor: 60

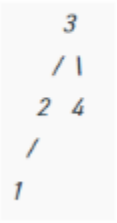
11. Given a sorted array, write a function that creates a Balanced Binary Search Tree using array elements. Follow the steps mentioned below to implement the approach:
    a. Set The middle element of the array as root.
    b. Recursively do the same for the left half and right half.
        i. Get the middle of the left half and make it the left child of the root created in step 1.
        ii. Get the middle of the right half and make it the right child of the root created in step 1.
    c. Print the preorder of the tree.

Given Array#1: [1, 2, 3]
Output: Pre-order of created BST: 2 1 3

BST#1



BST#2



Given Array#2: [1, 2, 3,4]
Output: Pre-order of created BST: 3 2 1 4

12. How can you print the contents of a tree in descending order with and without using stack? Solve using code.