Q3. Construct a binary tree using inorder and post order traversal given below. Inorder Traversal: 9, 3, 15, 20, 7 Post Order Traversal: 9, 15, 7, 20, 3 (10 marks)

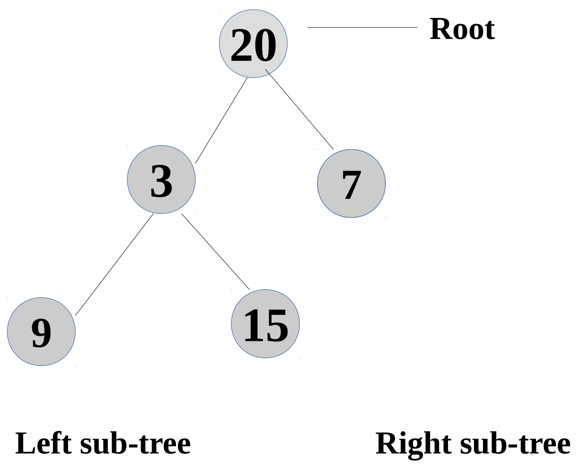
Note: You would need to explain all the steps.

In-order Traversal:-

In this traversal method, the left subtree is visited first, then the root and later the right sub-tree. We should always remember that every node may represent a subtree itself.

If a binary tree is traversed **in-order**, the output will produce sorted key values in an ascending order.

```
class Node:
        def __init__(self,val):
        self.data = val
        self.left = None
        self.right = None
def inorder(root):
        if root is None:
        return
        inorder(root.left)
        print(root.data)
        inorder(root.right)
```



Output:

9-->3-->15-->20-->7

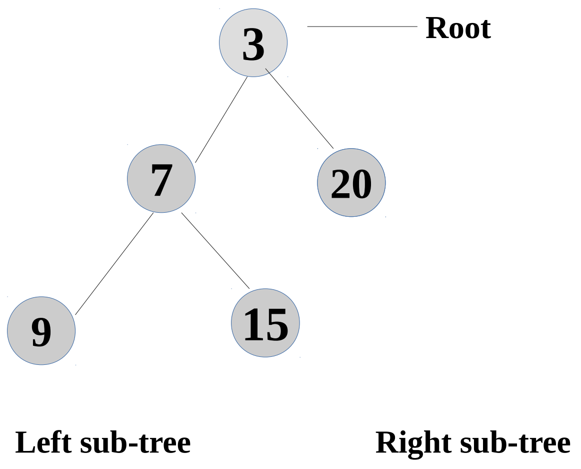
Algorithm

```
Until all nodes are traversed −
Step 1 − Recursively traverse left subtree.
Step 2 − Visit root node.
Step 3 − Recursively traverse right subtree.
```

Post-order Traversal:-

In this traversal method, the root node is visited last, hence the name. First we traverse the left subtree, then the right subtree and finally the root node.

```
class Node:
        def __init__(self,val):
        self.data = val
        self.left = None
        self.right = None
def postorder(root):
        if root is None:
        return
        inorder(root.left)
        inorder(root.right)
        print(root.data)
```



Output:

9-->15-->7-->20-->3

Algorithm

```
Until all nodes are traversed −
Step 1 − Recursively traverse left subtree.
Step 2 − Recursively traverse right subtree.
Step 3 − Visit root node.
```