**Trees Traeversal**

<https://youtu.be/PJX9T2oVKmk?si=ZjV_oGMkYV9R0HpA>

* Pre Order(root,left,right)
* In order(left,root,right)
* Post Order(left,right,root)

**Build Tree using inorder and postorder**

class TreeNode:

def \_\_init\_\_(self, val):

self.val = val

self.left = None

self.right = None

def buildTree(inorder, postorder):

if not inorder or not postorder:

return None

# The last element in postorder is the root of the tree

root\_val = postorder.pop()

root = TreeNode(root\_val)

# Find the root in the inorder traversal to split it into left and right subtrees

root\_index = inorder.index(root\_val)

# Recursively construct the right subtree first

root.right = buildTree(inorder[root\_index + 1:], postorder)

# Recursively construct the left subtree

root.left = buildTree(inorder[:root\_index], postorder)

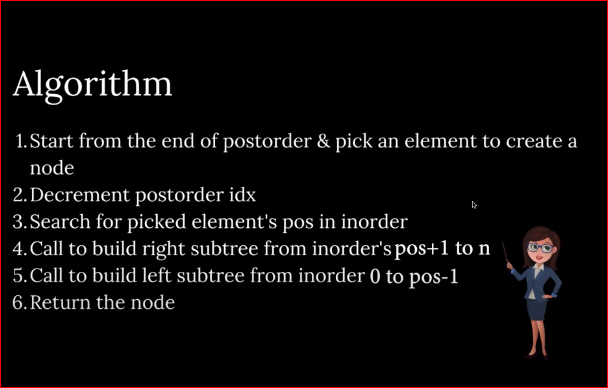
return root

# Example usage:

inorder = [9, 3, 15, 20, 7]

postorder = [9, 15, 7, 20, 3]

root = buildTree(inorder, postorder)



**Basically we take root node from preorder or post order then go to inorder and find its right and left nodes. If root has no element on left or right side it means they have no left or right node.**

**Build Tree using preorder and postorder**