Problem 3: Construct A Binary Tree from Inorder and Postorder Traversal.

```
class TreeNode:
    def __init__(self, val=0, left=None, right=None):
        self.val = val
        self.left = left
        self.right = right


def build_tree(inorder, postorder):
    if not inorder or not postorder:
        return None
    root_val = postorder[-1]
    root = TreeNode(root_val)

    root_index = inorder.index(root_val)

    root.left = build_tree(inorder[:root_index], postorder[:root_index])
    root.right = build_tree(inorder[root_index + 1:], postorder[root_index:-1])

    return root


def inorder_traversal(root):
    if root:
        inorder_traversal(root.left)
        print(root.val, end=' ')
        inorder_traversal(root.right)


def postorder_traversal(root):
    if root:
```

```
        postorder_traversal(root.left)
        postorder_traversal(root.right)
        print(root.val, end=' ')


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

tree_root = build_tree(inorder, postorder)

print("Inorder traversal:")
inorder_traversal(tree_root)
```