Documentation: Construct Binary Tree from Inorder and Postorder Traversal

Problem Description

- Given two integer arrays inorder and postorder where:
- inorder is the inorder traversal of a binary tree.
- postorder is the postorder traversal of the same tree.

the task is to construct and return the binary tree.

Example 1

Input:

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

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

Output:

[3, 9, 20, null, null, 15, 7]

Example 2

Input:

inorder = [-1]

postorder = [-1]

Output:

[-1]

Constraints

- (1 leq text{inorder.length} leq 3000)
- (text{postorder.length} == text{inorder.length})
- (-3000 leq text{inorder}[i], text{postorder}[i] leq 3000)
- inorder and postorder consist of unique values.
- Each value of postorder also appears in inorder.
- inorder is guaranteed to be the inorder traversal of the tree.
- postorder is guaranteed to be the postorder traversal of the tree.

Solution

To solve this problem, we can use a recursive approach. Here are the steps:

1. **Base Case:** If either inorder or postorder is empty, return None.
2. **Root Extraction:** The last element in the postorder list is the root of the binary tree.
3. **Find Root in Inorder:** Locate the index of this root in the inorder list to divide the tree into left and right subtrees.
4. **Recursive Construction:**
    - Recursively construct the right subtree first (since we are popping from the end of postorder).
    - Recursively construct the left subtree.
5. **Return Root:** Return the constructed tree's root node.

Explanation

- **TreeNode Class:** Defines the structure of a binary tree node with val, left, and right attributes.

- **buildTree Method:**
  - *Base Case:* Checks if either inorder or postorder is empty and returns None.
  - *Root Extraction:* Pops the last element from postorder to use as the root value.
  - *Inorder Index:* Finds the index of this root value in inorder.
  - *Recursive Calls:* Constructs the right subtree first, followed by the left subtree using slices of the inorder list.
  - *Return:* Returns the constructed tree's root node.

This solution effectively reconstructs the binary tree by leveraging the properties of inorder and postorder traversals.