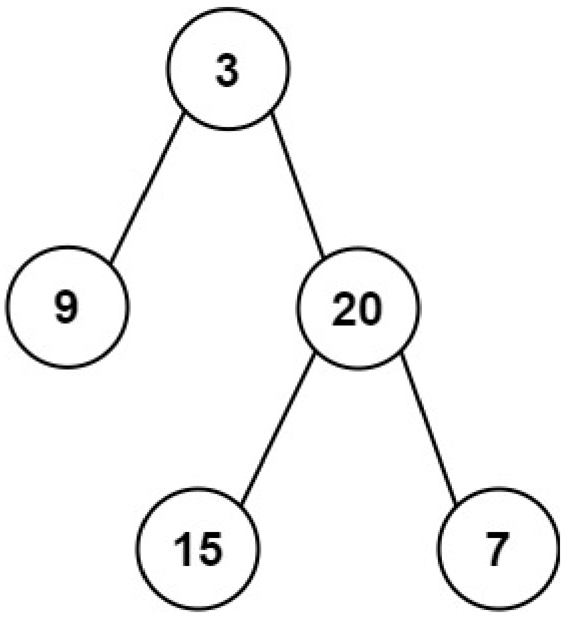
105. Construct Binary Tree from Preorder and Inorder Traversal

Given two integer arrays `preorder` and `inorder` where `preorder` is the preorder traversal of a binary tree and `inorder` is the inorder traversal of the same tree, construct and return *the binary tree*.

Example 1:



```
Input: preorder = [3,9,20,15,7], inorder = [9,3,15,20,7]
Output: [3,9,20,null,null,15,7]
```

Example 2:

```
Input: preorder = [-1], inorder = [-1]
Output: [-1]
```

Constraints:

- `1 <= preorder.length <= 3000`
- `inorder.length == preorder.length`
- `-3000 <= preorder[i], inorder[i] <= 3000`
- `preorder` and `inorder` consist of **unique** values.
- Each value of `inorder` also appears in `preorder`.
- `preorder` is **guaranteed** to be the preorder traversal of the tree.
- `inorder` is **guaranteed** to be the inorder traversal of the tree.

```
def buildTree(self, preorder: List[int], inorder: List[int]) ->
Optional[TreeNode]:
        return self.buildTreeHelper(preorder,inorder)

    def buildTreeHelper(self,preorder,inorder):
        if len(inorder)==0:
            return None

        node = TreeNode(preorder[0])
        data = preorder[0]
        idx = inorder.index(data)
        count = idx
        node.left = self.buildTreeHelper(preorder[1:count+1],inorder[:idx])
        node.right = self.buildTreeHelper(preorder[idx+1:],inorder[idx+1:])
        return node
```