LEET CODE:106

By Priyansh

Problem Statement: To Construct a binary tree with An Array containg the postorder traversal and Inorder traversal of A binary tree...

Ex: inorder = [9, 3, 15, 20, 7] Postorder = [9, 15, 7, 20, 3]

one simple logic: The last element or the element at the last index of the postorder array would be the root of the binary tree...

Another point is the root element of the inarray is the pivot And left of inorder Array is left Subtree and right side is the right Subtree...

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

Now we $\overset{\text{know}}{\wedge}$ last indexed element of postorder Array is the root as postorder : Left Right Root

Now Reverse the postorder Array...

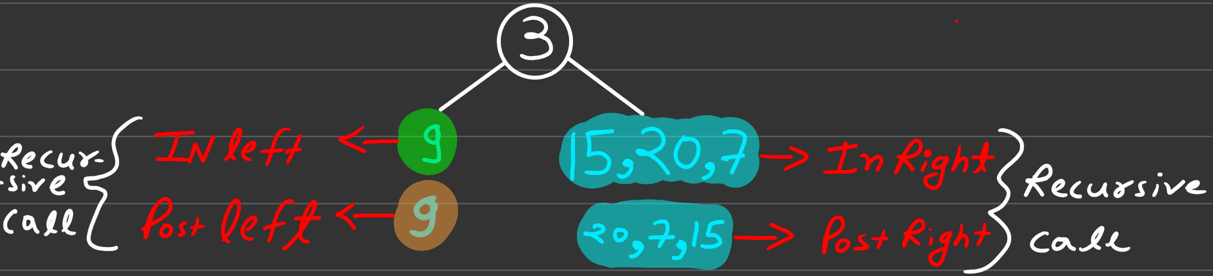
inorder = [9, 3, 15, 20, 7]   Postorder = [9, 15, 7, 20, 3]

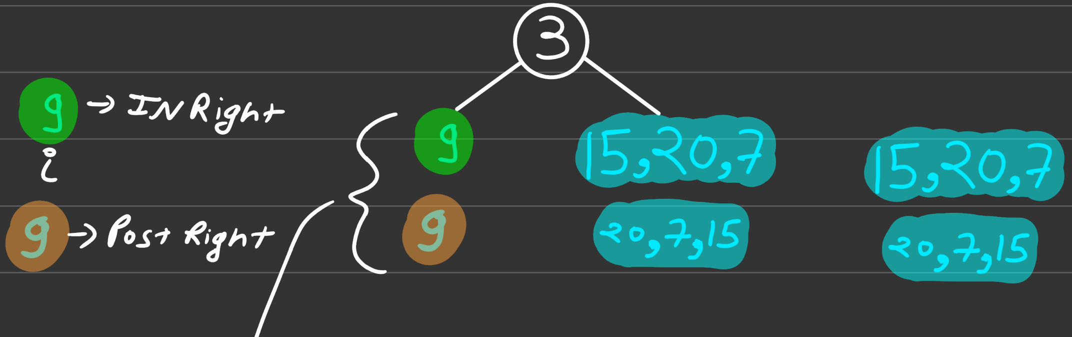
i   Postrev = [3, 20, 7, 15, 9]

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

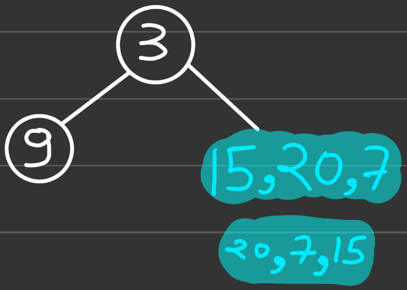
left subtree   i   Right subtree

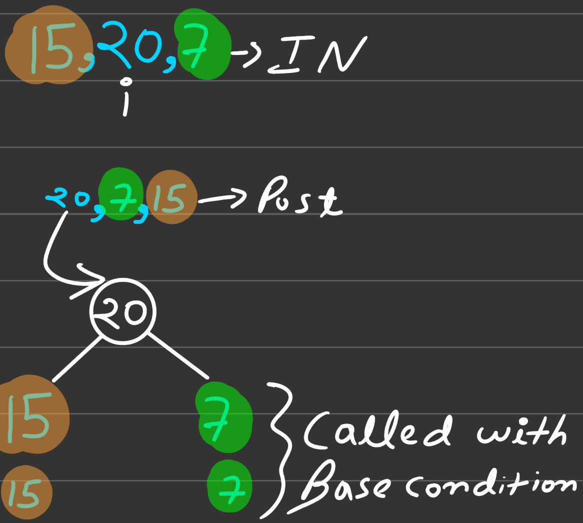
Now find the index of root in inorder traversal...

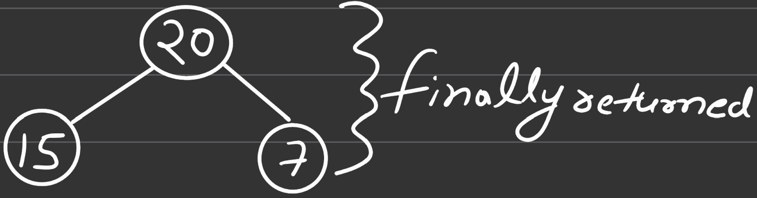


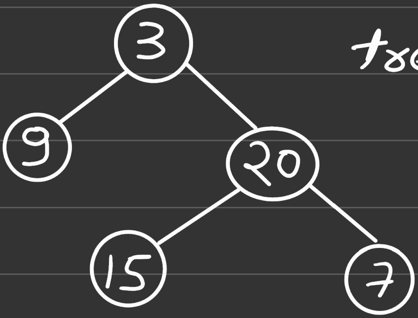


Base case in this call
inlo == prelo so just a single root would
be formed and returned!!!



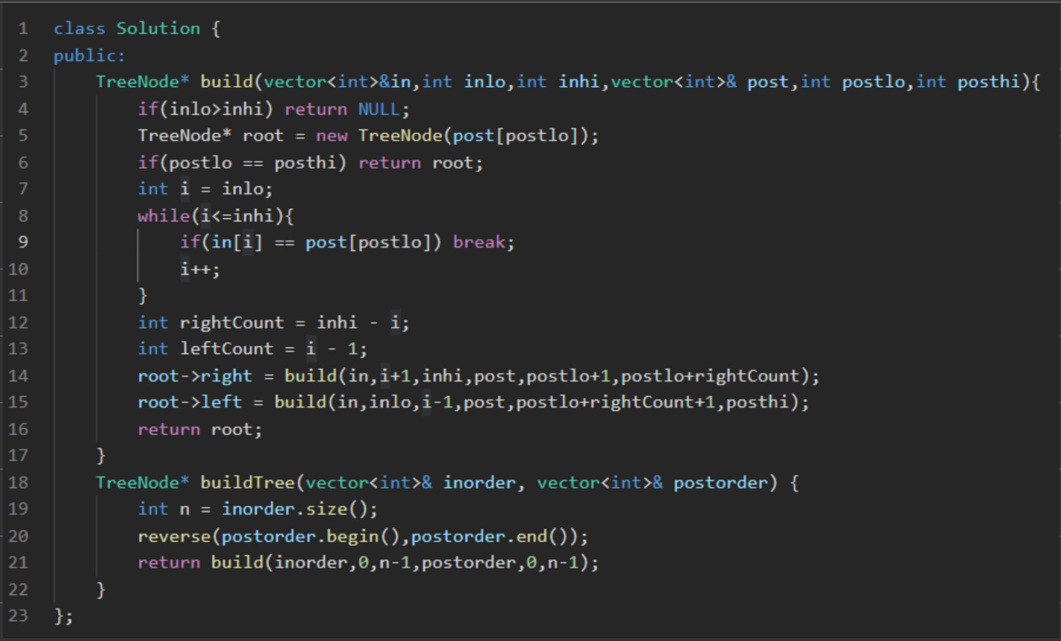






tree constructed!!!

Code Implementation!!!



```cpp
class Solution {
public:
    TreeNode* build(vector<int>&in,int inlo,int inhi,vector<int>& post,int postlo,int posthi){
        if(inlo>inhi) return NULL;
        TreeNode* root = new TreeNode(post[postlo]);
        if(postlo == posthi) return root;
        int i = inlo;
        while(i<=inhi){
            if(in[i] == post[postlo]) break;
            i++;
        }
        int rightCount = inhi - i;
        int leftCount = i - 1;
        root->right = build(in,i+1,inhi,post,postlo+1,postlo+rightCount);
        root->left = build(in,inlo,i-1,post,postlo+rightCount+1,posthi);
        return root;
    }
    TreeNode* buildTree(vector<int>& inorder, vector<int>& postorder) {
        int n = inorder.size();
        reverse(postorder.begin(),postorder.end());
        return build(inorder,0,n-1,postorder,0,n-1);
    }
};
```