652. Find Duplicate Subtrees

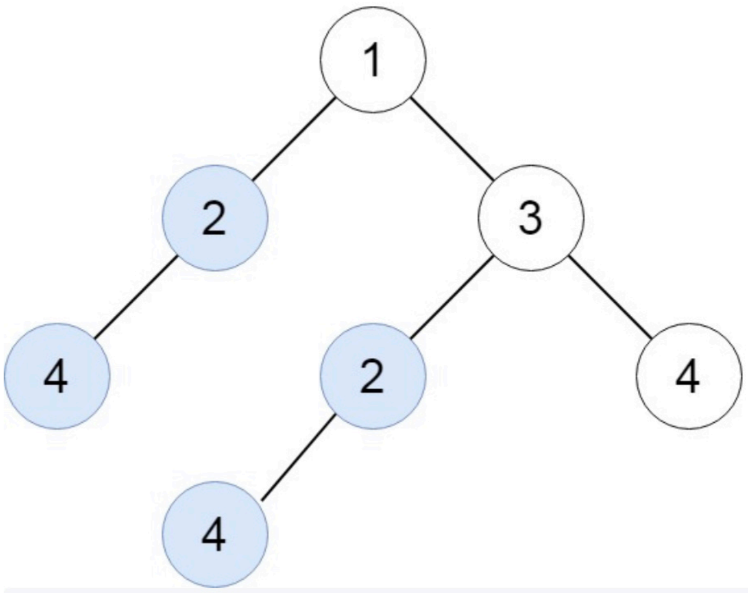
Medium 1788 233 Add to List Share

Given the `root` of a binary tree, return all **duplicate subtrees**.

For each kind of duplicate subtrees, you only need to return the root node of any **one** of them.

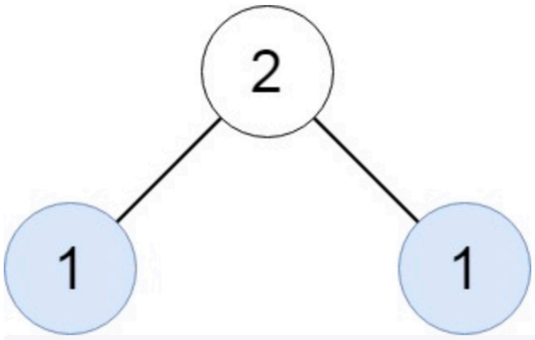
Two trees are **duplicate** if they have the **same structure** with the **same node values**.

Example 1:



```
Input: root = [1,2,3,4,null,2,4,null,null,4]
Output: [[2,4],[4]]
```

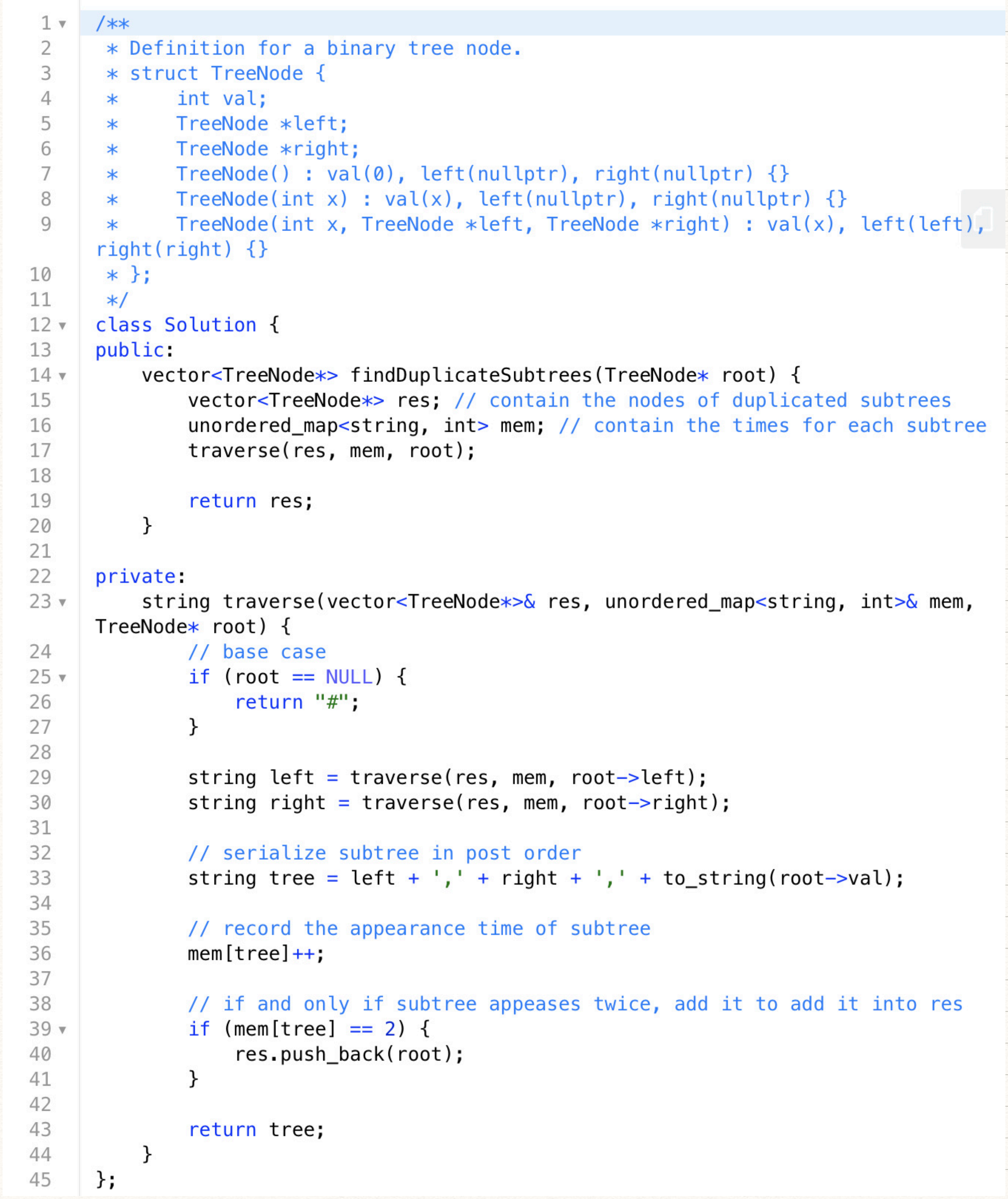
Example 2:



```
Input: root = [2,1,1]
Output: [[1]]
```

1. Serialize tree to describe it.

2. Use map to store times for each subtree.



```cpp
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode() : val(0), left(nullptr), right(nullptr) {}
 *     TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
 *     TreeNode(int x, TreeNode *left, TreeNode *right) : val(x), left(left), right(right) {}
 * };
 */
class Solution {
public:
    vector<TreeNode*> findDuplicateSubtrees(TreeNode* root) {
        vector<TreeNode*> res; // contain the nodes of duplicated subtrees
        unordered_map<string, int> mem; // contain the times for each subtree
        traverse(res, mem, root);

        return res;
    }

private:
    string traverse(vector<TreeNode*>& res, unordered_map<string, int>& mem, TreeNode* root) {
        // base case
        if (root == NULL) {
            return "#";
        }

        string left = traverse(res, mem, root->left);
        string right = traverse(res, mem, root->right);

        // serialize subtree in post order
        string tree = left + ',' + right + ',' + to_string(root->val);

        // record the appearance time of subtree
        mem[tree]++;

        // if and only if subtree appeases twice, add it to add it into res
        if (mem[tree] == 2) {
            res.push_back(root);
        }

        return tree;
    }
};
```