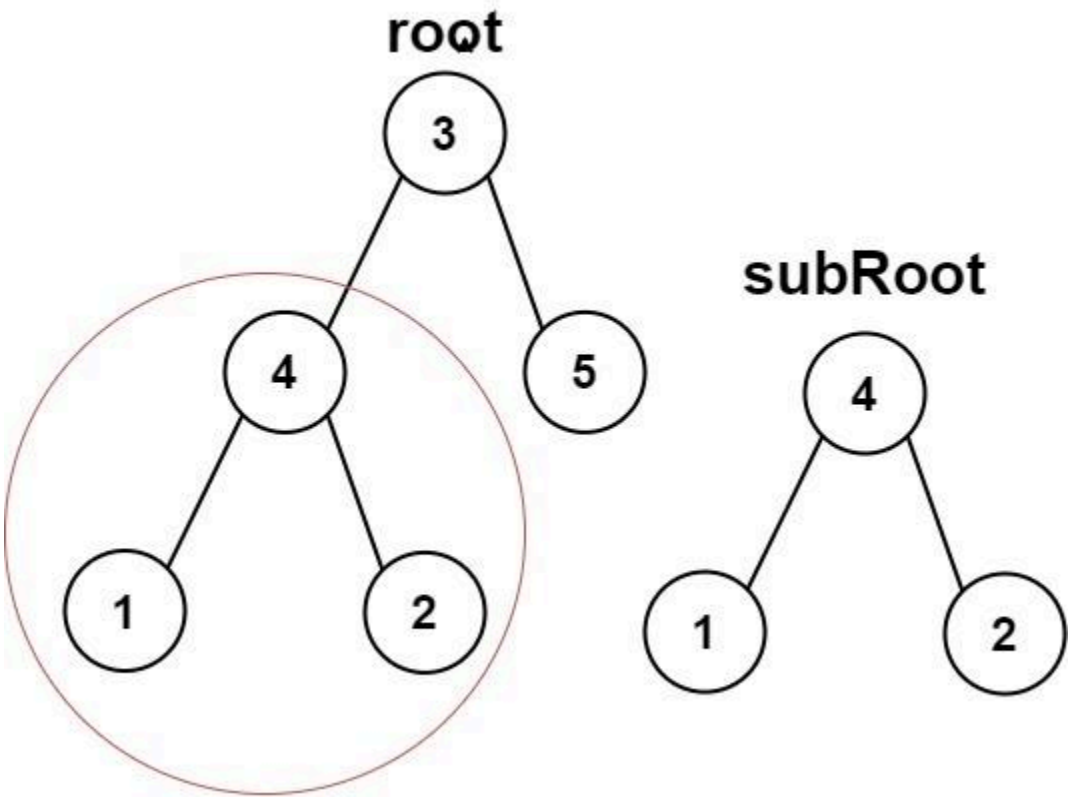
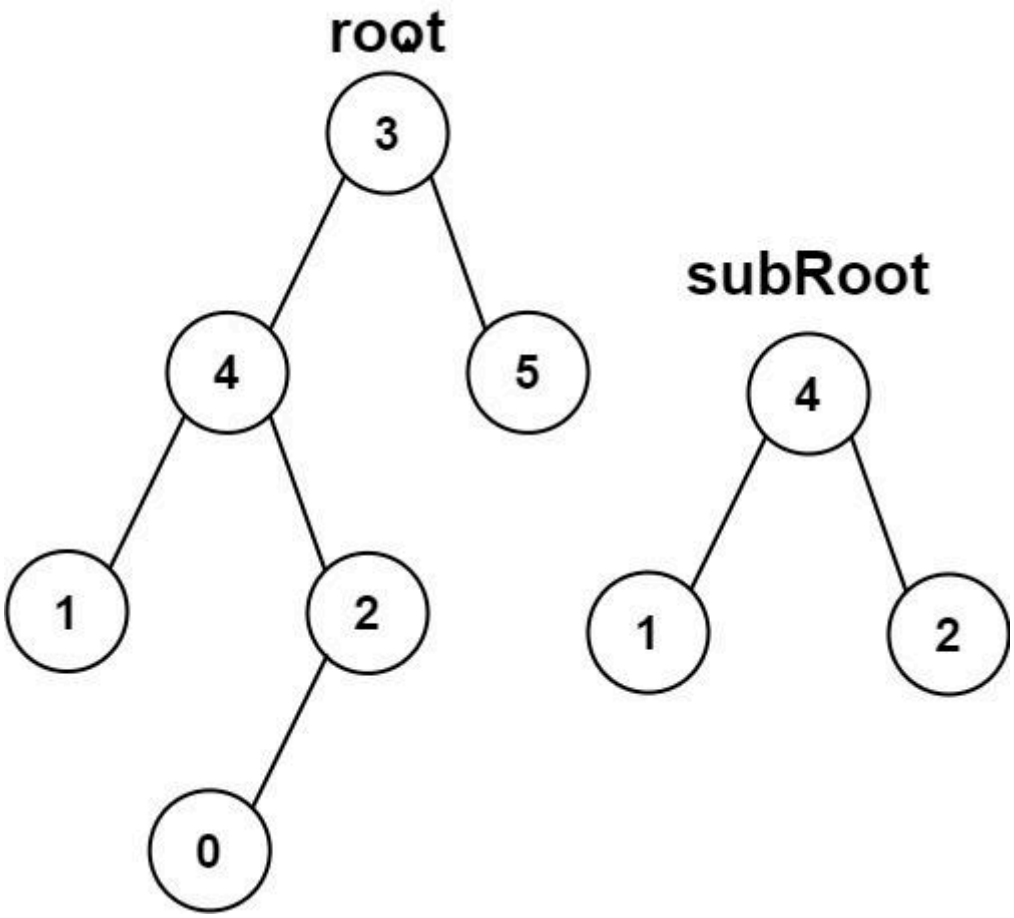
Given the roots of two binary trees root and subRoot, return true if there is a subtree of root with the same structure and node values of subRoot and false otherwise. A subtree of a binary tree tree is a tree that consists of a node in tree and all of this node's descendants. The tree tree could also be considered as a subtree of itself.

Example 1:



Input: root = [3,4,5,1,2], subRoot = [4,1,2]
Output: true

Example 2:



Input: root = [3,4,5,1,2,null,null,null,null,0], subRoot = [4,1,2]
Output: false

Constraints:

- **The number of nodes in the root tree is in the range [1, 2000].**
- **The number of nodes in the subRoot tree is in the range [1, 1000].**
- **-10^4 <= root.val <= 10^4**
- **-10^4 <= subRoot.val <= 10^4**

Solution:

```
class Solution {
    public boolean isIdentical(TreeNode root, TreeNode subRoot) {
        if(root==null&&subRoot==null){
            return true;
        }else if(root==null||subRoot==null||(root.val)!=(subRoot.val)){
            return false;
        }
        if(!isIdentical(root.left,subRoot.left)){
                return false;
        }
        if(!isIdentical(root.right,subRoot.right)){
                return false;
        }
        return true;
    }
    public boolean isSubtree(TreeNode root, TreeNode subRoot) {
        if(root==null){
            return false;
        }
        if((root.val)==(subRoot.val)){
            if(isIdentical(root,subRoot)){
                return true;
            }
        }
        return (isSubtree(root.left,subRoot)||isSubtree(root.right,subRoot));
    }
}
```