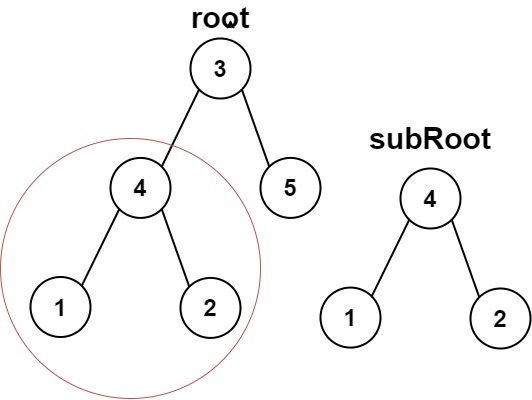
# 题目

给你两棵二叉树root和subRoot。检验root中是否包含和subRoot具有相同结构和节点值的子树。如果存在，返回true；否则，返回false。

二叉树tree的一棵子树包括tree的某个节点和这个节点的所有后代节点。tree也可以看做它自身的一棵子树。

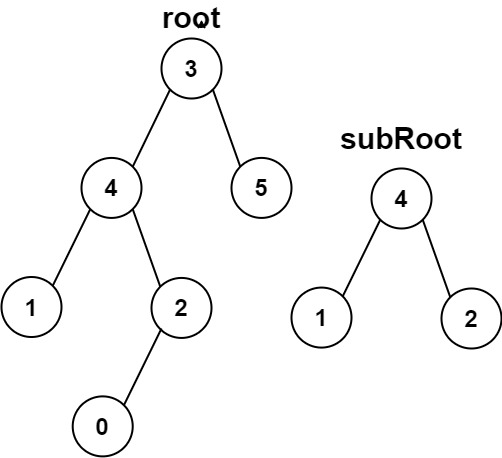
示例 1：



输入：root = [3,4,5,1,2], subRoot = [4,1,2]

输出：true

示例 2：



输入：root = [3,4,5,1,2,null,null,null,null,0], subRoot = [4,1,2]

输出：false

提示：

root 树上的节点数量范围是 [1, 2000]

subRoot 树上的节点数量范围是 [1, 1000]

-104 <= root.val <= 104

-104 <= subRoot.val <= 104

# 分析

/\*\*

 \* 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:

    bool isSubtree(TreeNode\* root, TreeNode\* subRoot) {

        if(!root)   return false;

        return isSubtree(root->left,subRoot) || isSame(root,subRoot) || isSubtree(root->right,subRoot);

    }

    bool isSame(TreeNode \*p,TreeNode \*q)

    {

        if(nullptr==p && nullptr==q)    return true;

        if(nullptr==p || nullptr==q)    return false;

        return p->val==q->val && isSame(p->left,q->left) && isSame(p->right,q->right);

    }

};