# Definition for a binary tree node.

# class TreeNode(object):

# def \_\_init\_\_(self, x):

# self.val = x

# self.left = None

# self.right = None

class Solution(object):

def hasSubtree(self, pRoot1, pRoot2):

"""

:type pRoot1: TreeNode

:type pRoot2: TreeNode

:rtype: bool

"""

# 以A树的每一个节点作为根节点，对树A进行遍历

if pRoot2 is None:

return False

if pRoot1 is None:

return False

if self.isEqual(pRoot1,pRoot2):

return True

else:

left = self.hasSubtree(pRoot1.left,pRoot2)

right = self.hasSubtree(pRoot1.right,pRoot2)

return (left or right)

def isEqual(self,node1,node2):

# node1表示A的节点

# node2表示B的节点

if node2 is None:

return True

if node1 is None:

return False

if not self.isEqualNum(node1.val,node2.val):

return False

else:

left = self.isEqual(node1.left,node2.left)

right = self.isEqual(node1.right,node2.right)

return (left and right)

def isEqualNum(self,val1,val2):

# if val1 - val2<0.000000001 or val2-val1<0.000000001:

if val1 == val2:

return True

else:

return False