## HasSubtree

https://www.acwing.com/problem/content/35/

# 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, p1, p2):

"""

:type pRoot1: TreeNode

:type pRoot2: TreeNode

:rtype: bool

"""

self.res = False

def check(p1,p2):

#if p2:

# if p1:

# if p1.val==p2.val:

# return True and check(p1.left,p2.left) and check(p1.right,p2.right)

if p1 and p2 and p1.val==p2.val:

return True and check(p1.left,p2.left) and check(p1.right,p2.right)

if not p2:

return True

return False

def rec(p1,p2):

if p1 and p2:

if p1.val==p2.val:

if check(p1,p2):

self.res = True

return

rec(p1.left,p2)

rec(p1.right,p2)

rec(p1,p2)

return self.res