Given a binary tree, check whether it is a mirror of itself (ie, symmetric around its center).

For example, this binary tree [1,2,2,3,4,4,3] is symmetric:

1

/ \

2 2

/ \ / \

3 4 4 3

But the following [1,2,2,null,3,null,3] is not:

1

/ \

2 2

\ \

3 3

就不是对称二叉树

找对称二叉树关键的就在于recursion

前面三个都是最后才用的，只有你确认到树的底端，两个点左右都是空，两个点值相等，会return true，不然就在确认这两个点值相等的情况下一直往下，最终这个·true会层层上传

class Solution {

public boolean isSymmetric(TreeNode root) {

return isSame(root.left,root.right);

}

public static boolean isSame(TreeNode left, TreeNode right){

if(left==null&&right==null) return true;

if(left==null&&right!=null) return false;

if(left!=null&&right==null) return false;

if(left.val!=right.val) return false;

return isSame(left.left,right.right)&&isSame(left.right,right.left);

}

}