给定一个二叉树，检查它是否是镜像对称的。

例如，二叉树 `[1,2,2,3,4,4,3]` 是对称的。

```
    1
   / \
  2   2
 / \ / \
3  4 4  3
```

但是下面这个 `[1,2,2,null,3,null,3]` 则不是镜像对称的:

```
    1
   / \
  2   2
   \   \
   3    3
```

说明:

如果你可以运用递归和迭代两种方法解决这个问题，会很加分。

分析，这题我没想通递归如何做，后面看别人代码才知道自己还是狭隘了，递归的树不一定一次只递归一个分支，可以多个分支同时进行递归，查找左右分支是否对称，不要被惯性思维固定了大脑。迭代做的话，应该就是入栈出栈操作。每遍历一层，进行回文比较，对称则进入下一层。

代码

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
class Solution {
public:
    bool isSymmetric(TreeNode* root) {
        if(root==NULL){
            return true;
        }
        return isSymmetric(root->left,root->right);
    }
    bool isSymmetric(TreeNode* left,TreeNode* right){
        if(left==NULL&&right==NULL){
            return true;
        }else if(left==NULL||right==NULL){
            return false;
```

```
        }
        if(left->val==right->val){
            return isSymmetric(left->left,right-
>right)&&isSymmetric(left->right,right->left);
        }
        return false;
    }
};
```