1. Symmetric Tree

Easy

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

**Solution**

判断左右子树是否镜像

递归

/\*\*  
 \* Definition for a binary tree node.  
 \* struct TreeNode {  
 \* int val;  
 \* TreeNode \*left;  
 \* TreeNode \*right;  
 \* TreeNode(int x) : val(x), left(NULL), right(NULL) {}  
 \* };  
 \*/  
typedef TreeNode\* pnode;  
class Solution {  
public:  
 bool isSymmetric(TreeNode\* root) {  
 return isMirror(root, root);  
 }  
 bool isMirror(pnode t1, pnode t2){  
 if(t1 == NULL && t2 != NULL)return false;  
 if(t1 != NULL && t2 == NULL)return false;  
 if(t1 == NULL && t2 == NULL)return true;  
 return t1->val == t2->val && isMirror(t1->left, t2->right) && isMirror(t1->right, t2->left);  
 }  
};

非递归

/\*\*  
 \* Definition for a binary tree node.  
 \* struct TreeNode {  
 \* int val;  
 \* TreeNode \*left;  
 \* TreeNode \*right;  
 \* TreeNode(int x) : val(x), left(NULL), right(NULL) {}  
 \* };  
 \*/  
typedef TreeNode\* pnode;  
class Solution {  
public:  
 bool isSymmetric(TreeNode\* root) {  
 stack<pnode>q;  
 q.push(root);  
 q.push(root);  
 while(!q.empty()){  
 pnode t1 = q.top();  
 q.pop();  
 pnode t2 = q.top();  
 q.pop();  
 if(t1 == NULL && t2 == NULL)continue;  
 if(t1 == NULL || t2 == NULL)return false;  
 if(t1->val != t2->val)return false;  
 q.push(t1->right);  
 q.push(t2->left);  
 q.push(t1->left);  
 q.push(t2->right);  
 }  
 return true;  
 }  
};