101 Symmetric Tree

2018年3月30日 16:28

Question:

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
```

Note:
Bonus points if you could solve it both recursively and iteratively.

来自 <https://leetcode.com/problems/symmetric-tree/description/>
给定一个二叉树，检查它是否是它自己的镜像（即，围绕它的中心对称）。

Solution for Python3:

```python
# Definition for a binary tree node.
# class TreeNode:
#     def __init__(self, x):
#         self.val = x
#         self.left = None
#         self.right = None
#Iterative Version:
class Solution1:
    def isSymmetric(self, root):
        """
        :type root: TreeNode
        :rtype: bool
        """
        if not root:
            return True
        d = deque()
        d.append(root.left)
        d.append(root.right)
        while d:
            L = d.popleft()
            R = d.popleft()
            if not L and not R:
                continue
            if L and R:
                if L.val != R.val:
                    return False
                d.append(L.left)
                d.append(R.right)
                d.append(L.right)
                d.append(R.left)
            else:
                return False
        return True

#Recursive Version:
class Solution2:
    def isSymmetric(self, root):
        """
        :type root: TreeNode
        :rtype: bool
        """
        if not root:
            return True
        return Solution.isMirror(self, root.left, root.right)

```

```
    def isMirror(self, L, R):
        if not L and not R:
            return True
        if not L or not R:
            return False
        return L.val == R.val and Solution.isMirror(L.left, R.right) and Solution.isMirror(L.right, R.left)
```

Solution for C++:

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
// Iterative Version:
class Solution1 {
public:
    bool isSymmetric(TreeNode* root) {
        if (!root) {
           return true;
        }
        stack<TreeNode*> s;
        TreeNode *left, *right;
        s.push(root->left);
        s.push(root->right);
        while (!s.empty()) {
           L = s.top();
           s.pop();
           R = s.top();
           s.pop();
           if (!L && R) || (L && !R) {
               return false;
           }
           if (L && R) {
               if (L->val != R->val) {
                   return false;
               }
               s.push(L.left);
               s.push(R.right);
               s.push(L.right);
               s.push(R.left);
           }
        }
        return true;
    }
};

// Recursive Version:
class Solution2 {
public:
    bool isSymmetric(TreeNode* root) {
       if (!root) {
           return true;
       }
       return isMirror(root->left, root->right);
    }
    bool isMirror(TreeNode* L, TreeNode* R) {
       if (!L and !R) {
           return true;
       }
       if (!L or !R) {
           return false;
       }
       return (L->val == R->val) && isMirror(L->left, R->right) && isMirror(L->right, R->left);
    }
};
```

Appendix:

Python 类内函数互相调用:

1) 调用函数前加类名：ClassName.function(self, args),该方法函数定义和调用参数都有self
2) 调用函数前加self: self.function(args)
3) deque双端队列可模拟队列和栈