111 Minimum Depth of Binary Tree

2018年3月31日 19:48

Question:

Given a binary tree, find its minimum depth.
The minimum depth is the number of nodes along the shortest path from the root node down to the nearest leaf node.

来自 <https://leetcode.com/problems/minimum-depth-of-binary-tree/description/>

给定一个二叉树，找出其最小深度。
最小深度是从根节点到最近叶节点的最短路径的节点数量。

Solution for Python3:

```
# Definition for a binary tree node.
# class TreeNode:
#     def __init__(self, x):
#         self.val = x
#         self.left = None
#         self.right = None

class Solution1:
    def minDepth(self, root):
        """
        :type root: TreeNode
        :rtype: int
        """
        if not root:
            return 0
        if not root.left:
            return 1 + self.minDepth(root.right)
        if not root.right:
            return 1 + self.minDepth(root.left)
        return 1 + min(self.minDepth(root.left), self.minDepth(root.right))

class Solution2:
    def minDepth(self, root):
        """
        :type root: TreeNode
        :rtype: int
        """
        if not root: return 0
        d = map(self.minDepth, (root.left, root.right))
        return 1 + (min(d) or max(d))
        # 我们需要加上最小子树的深度，除了该子树深度为0即该子树为空。
        # 例如，左子树为空，右子树为1，当前节点深度为1+右子树的深度。

class Solution2:
    def minDepth(self, root):
        """
        :type root: TreeNode
        :rtype: int
        :rtype: int
        """
        if not root: return 0
        d, D = sorted(map(self.minDepth, (root.left, root.right)))
        return 1 + (d or D)
```

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) {}
 * };
 */
class Solution1 {
public:
    int minDepth(TreeNode* root) {
        if (root == NULL) {
            return 0;
        }
        if (root->left == NULL) {
            return 1 + minDepth(root->right);
        }
        if (root->right == NULL) {
            return 1 + minDepth(root->left);
        }
        return 1 + min(minDepth(root->left), minDepth(root->right));
};

class Solution2 {
public:
    int minDepth(TreeNode* root) {
        if (!root) return 0;
        int L = minDepth(root->left), R = minDepth(root->right);
        return 1 + (min(L, R) ? min(L, R) : max(L, R));
        // return 1 + (L && R ? min(L, R) : max(L, R));
        // return 1 + (!L - !R ? max(L, R) : min(L, R));
    }
};
```