104 Maximum Depth of Binary Tree

2018年3月30日 18:58

Question:

Given a binary tree, find its maximum depth.
The maximum depth is the number of nodes along the longest path from the root node down to the farthest leaf node.
For example:
Given binary tree [3,9,20,null,null,15,7],

```
   3
  / \
 9  20
   / \
  15  7
```

return its depth = 3.

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

给定一个二叉树，找出其最大深度。
二叉树的深度为根节点到最远叶节点的最长路径上的节点数。

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
# Recursive Version:
class Solution1:
    def maxDepth(self, root):
        """
        :type root: TreeNode
        :rtype: int
        """
        if not root:
            return 0
        return 1 + max(self.maxDepth(root.left), self.maxDepth(root.right))

# Iterative Version:
class Solution2:
    def maxDepth(self, root):
        """
        :type root: TreeNode
        :rtype: int
        """
        if not root:
            return 0
        from collections import deque
        d = deque()
        d.append(root)
        res = 0
```

```
        while d:
            res += 1
            for i in range(len(d)):
                p = d.popleft()
                if p.left:
                    d.append(p.left)
                if p.right:
                    d.append(p.right)
        return res
```

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) {}
 * };
 */
// Recursive Version:
class Solution1 {
public:
    int maxDepth(TreeNode* root) {
        if (!root) {
            return 0;
        }
        return 1 + max(maxDepth(root->left), maxDepth(root->right));
    }
};

// Iterative Version:
class Solution2 {
public:
    int maxDepth(TreeNode* root) {
        if (!root) {
            return 0;
        }
        int res = 0;
        queue<TreeNode*> q;
        q.push(root);
        while (!q.empty()) {
            ++res;
            for (int i = 0, n = q.size(); i < n; i++) {
                TreeNode *p = q.front();
                q.pop();
                if (p->left != NULL) {
                    q.push(p->left);
                }
                if (p->right != NULL) {
                    q.push(p->right);
                }
            }
```

```
43          }
44          return res;
45      }
46  };
```