

226 Invert Binary Tree

2018年4月5日 16:00

Question:

Invert a binary tree.

```
    4
   / \
  2   7
 / \  / \
1  3 6  9
```

to

```
    4
   / \
  7   2
 / \  / \
9  6 3  1
```

来自 <https://leetcode.com/problems/invert-binary-tree/description/>

Solution for Python3:

```
1  # Definition for a binary tree node.
2  # class TreeNode:
3  #     def __init__(self, x):
4  #         self.val = x
5  #         self.left = None
6  #         self.right = None
7
8  class Solution1:
9      def invertTree(self, root):
10         """
11         :type root: TreeNode
12         :rtype: TreeNode
13         """
14         if not root:
15             return root
```

```

16         T = self.invertTree(root.right)
17         root.right = self.invertTree(root.left)
18         root.left = T
19         return root
20
21     class Solution2:
22         def invertTree(self, root):
23             """
24             :type root: TreeNode
25             :rtype: TreeNode
26             """
27             if not root:
28                 return None
29             from collections import deque
30             deq = deque([root])
31             while deq:
32                 node = deq.popleft()
33                 t = node.left
34                 node.left = node.right
35                 node.right = t
36                 if node.left:
37                     deq.append(node.left)
38                 if node.right:
39                     deq.append(node.right)
40             return root

```

Solution for C++:

```

1     class Solution1 {
2     public:
3         TreeNode* invertTree(TreeNode* root) {
4             if (!root) {
5                 return root;
6             }
7             TreeNode* T = invertTree(root->right);
8             root->right = invertTree(root->left);

```

```

9         root->left = T;
10        return root;
11    }
12 };
13
14 class Solution2 {
15 public:
16     TreeNode* invertTree(TreeNode* root) {
17         if (!root) {
18             return root;
19         }
20         queue<TreeNode*> que;
21         que.push(root);
22         while (!que.empty()) {
23             TreeNode* node = que.front();
24             que.pop();
25             TreeNode* t = node->left;
26             node->left = node->right;
27             node->right = t;
28             if (node->left) {
29                 que.push(node->left);
30             }
31             if (node->right) {
32                 que.push(node->right);
33             }
34         }
35         return root;
36     }
37 };

```