101 Symmetric Tree

101. Symmetric Tree

1. 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.

2. Implementation

(1) DFS recursion

```
class Solution {
    public boolean isSymmetric(TreeNode root) {
        return isSymmetric(root, root);
    }

    public boolean isSymmetric(TreeNode node1, TreeNode node2) {
        if (node1 == null && node2 == null) return true;
        if (node1 == null || node2 == null) return false;
        return node1.val == node2.val && isSymmetric(node1.left, node2.right)
    }
}
```

(2) BFS iteration

```
class Solution {
    public boolean isSymmetric(TreeNode root) {
        if (root == null) {
            return true;
        }

        Queue<TreeNode> queue = new LinkedList<>();
        queue.add(root);
        queue.add(root);

        while (!queue.isEmpty()) {
            TreeNode node1 = queue.remove();
            TreeNode node2 = queue.remove();

            if (node1 == null && node2 == null) {
                continue;
            }
            else if (node1 == null || node2 == null || node1.val != node2.val
                return false;
            }

            queue.add(node1.left);
            queue.add(node2.right);
            queue.add(node1.right);
            queue.add(node2.left);
        }
        return true;
    }
}
```

3. Time & Space Complexity

DFS recursion: 时间复杂度: O(n), 空间复杂度: O(n)

BFS iteration: 时间复杂度: O(n), 空间复杂度: O(n)

Previous
662 Maximum Width of Binary Tree

Next
102 Binary Tree Level Order Traversal