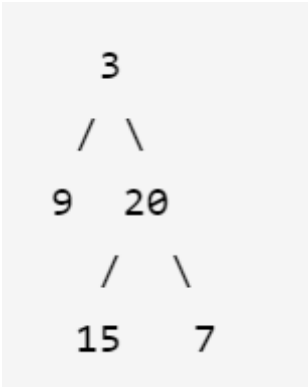
104 二叉树的最大深度

Label：二叉树

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子节点的最长路径上的节点数。
说明：叶子节点是指没有子节点的节点。
    [3,9,20,null,null,15,7]



- 递归 深度优先（从上到下依次加1）

```
class Solution {
    public int maxDepth(TreeNode root) {
        if (root == null) return 0;
        int leftMaxDepth = depth(root.left, 1);
        int rightMaxDepth = depth(root.right, 1);
        return Math.max(leftMaxDepth, rightMaxDepth);
    }

    private int depth (TreeNode root, int depth) {
        if (root == null) return depth;
        int leftMaxDepth = depth(root.left, depth+1);
        int rightMaxDepth = depth(root.right, depth+1);
        return Math.max(leftMaxDepth, rightMaxDepth);
    }
}
```

- 递归合并 有点类似于回溯（从下到上依次加1）

```
class Solution {
    public int maxDepth(TreeNode root) {
        if (root == null) {
            return 0;
        } else {
            int leftHeight = maxDepth(root.left);
            int rightHeight = maxDepth(root.right);
            return Math.max(leftHeight, rightHeight) + 1;
        }
    }
}
```

- 广度优先

```
class Solution {
    public int maxDepth(TreeNode root) {
        if (root == null) return 0;

        Queue<TreeNode> queue = new LinkedList<TreeNode>();
        queue.offer(root);
        int deep = 0;
        while (!queue.isEmpty()) {
            int size = queue.size();
            while (size > 0) {
                TreeNode node = queue.poll();
                if (node.left != null) {
                    queue.offer(node.left);
                }
                if (node.right != null) {
                    queue.offer(node.right);
                }
                size--;
            }
            deep++;
        }
        return deep;
    }
}
```