

GPT-3.5

问题出在第11行代码，在将node.right添加到tmp列表时，应该是tmp.append(node.right)而不是tmp.append(node.left)。修改后的代码如下：

```
def maxDepth(self, root: TreeNode) -> int:  
    if not root: return 0  
    queue, res = [root], 0  
    while queue:  
        tmp = []  
        for node in queue:  
            if node.left: tmp.append(node.left)  
            if node.right: tmp.append(node.right)  
        queue = tmp  
        res += 1  
    return res
```

The problem lies in line 11 of the code. When adding node.right to the tmp list, it should be tmp.append(node.right) instead of tmp.append(node.left). The modified code is as follows:

```
def maxDepth(self, root: TreeNode) -> int:  
    if not root: return 0  
    queue, res = [root], 0  
    while queue:  
        tmp = []  
        for node in queue:  
            if node.left: tmp.append(node.left)  
            if node.right: tmp.append(node.right)  
        queue = tmp  
        res += 1  
    return res
```

GPT-4

你的代码中有一个错误：

1095

你在这一行代码if node.right: tmp.append(node.left)中应该是想把右子节点加入到tmp列表中，但是你写的是node.left，这样就只会把左子节点加入到tmp列表中。正确的应该是node.right。

正确的代码如下：

```
def maxDepth(self, root: TreeNode) -> int:  
    if not root: return 0  
    queue, res = [root], 0  
    while queue:  
        tmp = []  
        for node in queue:  
            if node.left: tmp.append(node.left)  
            if node.right: tmp.append(node.right)  
        queue = tmp  
        res += 1  
    return res
```

这种解法的思路是：每次迭代时，都把当前层的所有节点遍历一遍，同时把下一层的所有节点都加到下一次迭代的队列中。所以每次迭代完，深度res就加1。直到遍历的队列为空，此时的深度就是二叉树的最大深度。