0107. 二叉树的层序遍历 II

ITCharge 大约 1 分钟

- 标签：树、广度优先搜索、二叉树
- 难度：中等

题目链接

- 0107. 二叉树的层序遍历 II - 力扣

题目大意

描述：给定一个二叉树的根节点 $root$。

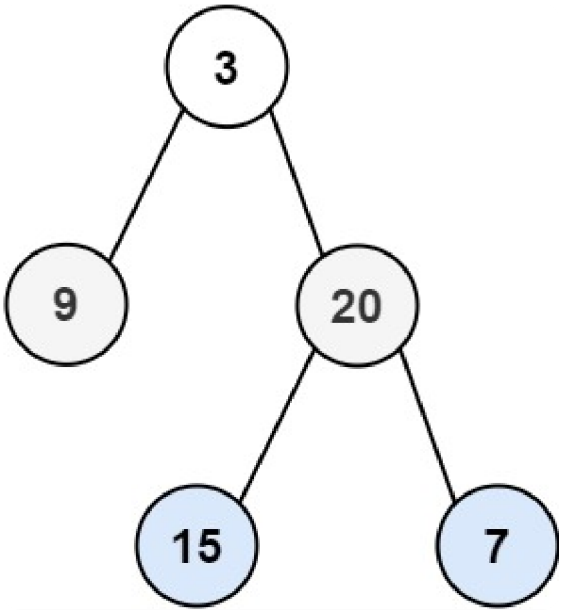
要求：返回其节点值按照「自底向上」的「层序遍历」（即按从叶子节点所在层到根节点所在的层，逐层从左向右遍历）。

说明：

- 树中节点数目在范围 $[0, 2000]$ 内。
- $-1000 \leq Node.val \leq 1000$。

示例：

- 示例 1：



```py
输入：root = [3,9,20,null,null,15,7]
输出：[[15,7],[9,20],[3]]
```

- 示例 2：

```py
输入：root = [1]
输出：[[1]]
```

解题思路

思路 1：二叉树的层次遍历

先得到层次遍历的节点顺序，再将其进行反转返回即可。

其中层次遍历用到了广度优先搜索，不过需要增加一些变化。普通广度优先搜索只取一个元素，变化后的广度优先搜索每次取出第 i 层上所有元素。

具体步骤如下：

1. 根节点入队。
2. 当队列不为空时，求出当前队列长度 s_i。
3. 依次从队列中取出这 s_i 个元素，将其左右子节点入队，然后继续迭代。
4. 当队列为空时，结束。

思路 1：代码

```py
class Solution:
    def levelOrderBottom(self, root: TreeNode) -> List[List[int]]:
        if not root:
            return []
        queue = [root]
        order = []
        while queue:
            level = []
            size = len(queue)
            for _ in range(size):
```

```
                curr = queue.pop(0)
                level.append(curr.val)
                if curr.left:
                    queue.append(curr.left)
                if curr.right:
                    queue.append(curr.right)
            if level:
                order.append(level)
        return order[::-1]
```

思路 1：复杂度分析

- **时间复杂度**：$O(n)$，其中 n 为树中节点个数。
- **空间复杂度**：$O(n)$。