Problem 107: Binary Tree Level Order Traversal II

Problem Information

Difficulty: Medium
Acceptance Rate: 67.10%
Paid Only: No
Tags: Tree, Breadth-First Search, Binary Tree

Problem Description

Given the `root` of a binary tree, return _the bottom-up level order traversal of its nodes ' values_. (i.e., from left to right, level by level from leaf to root).

Example 1:

Input: root = [3,9,20,null,null,15,7] **Output:** [[15,7],[9,20],[3]]

Example 2:

Input: root = [1] **Output:** [[1]]

Example 3:

Input: root = [] **Output:** []

Constraints:

* The number of nodes in the tree is in the range `[0, 2000]`. * `-1000 <= Node.val <= 1000`

Code Snippets

C++:

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 * int val;
 * TreeNode *left;
 * TreeNode *right;
 * TreeNode() : val(0), left(nullptr), right(nullptr) {}
 * TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
 * TreeNode(int x, TreeNode *left, TreeNode *right) : val(x), left(left),
right(right) {}
 * };
 */
class Solution {
public:
vector<vector<int>> levelOrderBottom(TreeNode* root) {


}
};
```

Java:

```
/**
 * Definition for a binary tree node.
 * public class TreeNode {
 * int val;
 * TreeNode left;
 * TreeNode right;
 * TreeNode() {}
 * TreeNode(int val) { this.val = val; }
 * TreeNode(int val, TreeNode left, TreeNode right) {
 * this.val = val;
 * this.left = left;
 * this.right = right;
 * }
 * }
 */
class Solution {
public List<List<Integer>> levelOrderBottom(TreeNode root) {


}
}
```

Python3:

```python
# Definition for a binary tree node.
# class TreeNode:
# def __init__(self, val=0, left=None, right=None):
# self.val = val
# self.left = left
# self.right = right
class Solution:
def levelOrderBottom(self, root: Optional[TreeNode]) -> List[List[int]]:
```