**Binary Tree Level Order Traversal**

Given the root of a binary tree, return *the level order traversal of its nodes' values*. (i.e., from left to right, level by level).

**Example 1:**



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

**Output:** [[3],[9,20],[15,7]]

**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

/\*\*

\* Definition for a binary tree node.

\* public class TreeNode {

\* public int val;

\* public TreeNode left;

\* public TreeNode right;

\* public TreeNode(int val=0, TreeNode left=null, TreeNode right=null) {

\* this.val = val;

\* this.left = left;

\* this.right = right;

\* }

\* }

\*/

public class Solution {

public IList<IList<int>> LevelOrder(TreeNode root)

{

IList<IList<int>> retVal = new List<IList<int>>();

Traverse(root, retVal, 0);

return retVal;

}

void Traverse(TreeNode root, IList<IList<int>> retVal, int depth)

{

if(root == null)

{

return;

}

if(retVal.Count == depth)

{

retVal.Add(new List<int>());

}

retVal[depth].Add(root.val);

depth++;

Traverse(root.left, retVal, depth);

Traverse(root.right, retVal, depth);

}

}