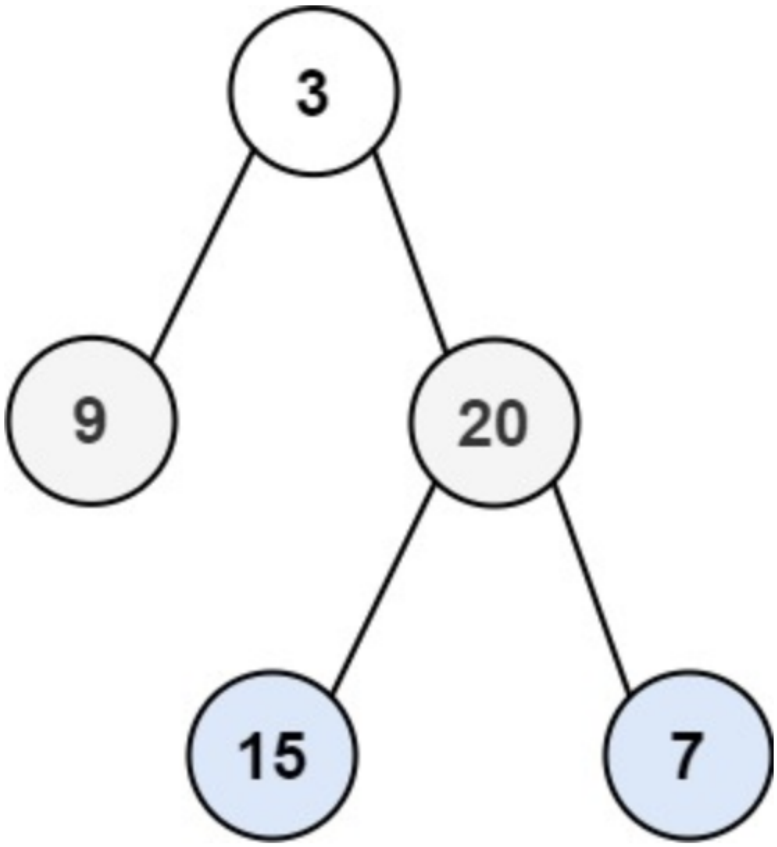
Binary Tree Level Order Traversal

Difficulty	Medium
Category	Tree
Question	https://leetcode.com/problems/binary-tree-level-order-traversal/
Solution	https://www.youtube.com/watch?v=6ZnyEApgFYg
Status	Done

Question

> 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:** []

Solution

```
# 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 levelOrder(self, root: Optional[TreeNode]) -> List[List[int]]:
    # Check if the tree is empty
        if not root:
            return []

        # Initialize an empty list to store the result
        result = []

        # Initialize a queue with the root node
        queue = [root]

        while queue:
            # Initialize a list to store nodes at the current level
            current_level = []

            # Get the number of nodes at the current level
            level_size = len(queue)

            for i in range(level_size):
                # Dequeue the first node from the queue
                node = queue.pop(0)
                current_level.append(node.val)

                # Enqueue the left and right children, if they exist
                if node.left:
                    queue.append(node.left)
                if node.right:
                    queue.append(node.right)

            # Add the current level to the result
            result.append(current_level)

        return result
```

> Basically we implement BFS for Level order Traversal