102 二叉树的层序遍历

Label：二叉树
给你一个二叉树，请你返回其按 层序遍历 得到的节点值。 （即逐层地，从左到右访问所有节点）。

二叉树：`[3,9,20,null,null,15,7]`，

```
    3
   / \
  9  20
    /  \
   15   7
```

返回其层序遍历结果：

```
[
  [3],
  [9,20],
  [15,7]
]
```

- 广度优先（最后一个用例超出时间限制）

```java
class Solution {
    public List<List<Integer>> levelOrder(TreeNode root) {
        Queue<TreeNode> queue = new LinkedList<>();
        List<List<Integer>> re = new ArrayList<>();
        if (root == null) return re;
        List<Integer> init = new ArrayList<>();
        init.add(root.val);
        re.add(init);
        queue.add(root.left);
        queue.add(root.right);
        int deep = 0;

        while (true) {
            init = new ArrayList<>();
            for (int i = 0; i < 2<<deep; i++) {
                TreeNode curr = queue.poll();
                if (curr != null) {
                    init.add(curr.val);
                    queue.offer(curr.left); // 是null 也加上
                    queue.offer(curr.right);
                } else {
                    queue.offer(null); // 填充上
                    queue.offer(null);
                }
            }

            if (init.size() > 0){
                re.add(init);
                deep++;
            } else break;
        }
        return re;
    }
}
```

- count计数

```java
class Solution {
    public List<List<Integer>> levelOrder(TreeNode root) {

        if(root == null) return new ArrayList<>();
        List<List<Integer>> res = new ArrayList<>();
        Queue<TreeNode> queue = new LinkedList<TreeNode>();
        queue.add(root);

        while(!queue.isEmpty()){
            int count = queue.size(); // 用count 计数上一层的节点数
            List<Integer> list = new ArrayList<Integer>();
            while(count > 0) {
                TreeNode node = queue.poll();
                list.add(node.val);
                if(node.left != null)
                    queue.add(node.left);
                if(node.right != null)
                    queue.add(node.right);
                count--;
            }
            res.add(list);
        }
        return res;
    }
}
```