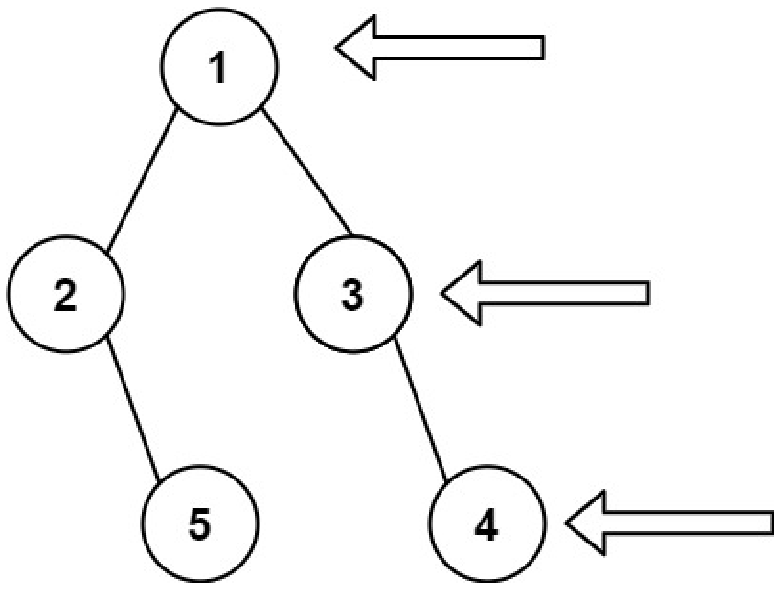
# 199. Binary Tree Right Side View

| | |
|---|---|
| Created | @November 26, 2022 3:59 PM |
| Difficulty | Medium |
| LC Url | https://leetcode.com/problems/binary-tree-right-side-view/ |
| Importance | |
| Tag | NEET Tree |
| Video | https://leetcode.cn/problems/binary-tree-right-side-view/solution/python3-gui-na-liao-san-dao-ti-yi-ci-gao-jx9u/ |

Given the `root` of a binary tree, imagine yourself standing on the **right side** of it, return *the values of the nodes you can see ordered from top to bottom*.

**Example 1:**



```
Input: root = [1,2,3,null,5,null,4]
Output: [1,3,4]
```

### Example 2:

```
Input: root = [1,null,3]
Output: [1,3]
```

### Example 3:

```
Input: root = []
Output: []
```

### Constraints:

- The number of nodes in the tree is in the range `[0, 100]`.
- `100 <= Node.val <= 100`

# Solution

## BFS

相当于记录每一层的最后一个节点，可用队列实现层序遍历，在内层循环时判断是否为当前层最后一个节点，若是则记录该节点值到res，遍历完成后返回res即可。

```
# 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 rightSideView(self, root: Optional[TreeNode]) -> List[int]:
        if not root:
            return []

        res = []
        queue = collections.deque([root])
        while queue:
            sz = len(queue)
            for i in range(sz):
                cur = queue.popleft()
                if i == sz - 1:
                    res.append(cur.val)
                if cur.left:
                    queue.append(cur.left)
                if cur.right:
                    queue.append(cur.right)
        return res
```

右视图即每层最右边的值，因此只需要做一个非常简单的小变化，即返回的列表只需要对每一层的列表的最后一个元素入结果列表就可以了。

```
# 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 rightSideView(self, root: Optional[TreeNode]) -> List[int]:
        if root is None:
            return []
        res = []
        queue = []
        queue.append(root)

        while queue:
            sz = len(queue)
            res.append(queue[-1].val)
            level = []
            for i in range(sz):
                cur = queue[i]
                if cur.left:
                    level.append(cur.left)
                if cur.right:
                    level.append(cur.right)
            queue = level
        return res
```

## DFS

若使用深度优先遍历，先遍历右子树，则每次遍历到的第一个节点就是最右边的节点。使用深度depth来判断当前节点是否是该深度首次访问到的节点（depth==len(res)），每次递归更新depth，若是首次则加到结果res中，最后返回res。

```
# 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 rightSideView(self, root: Optional[TreeNode]) -> List[int]:
        res = []

        def dfs(root, depth):
            if root is None:
                return

            if depth == len(res):
                res.append(root.val)
            depth += 1
            dfs(root.right, depth)
            dfs(root.left, depth)

        dfs(root, 0)
        return res
```

链接：https://leetcode.cn/problems/binary-tree-right-side-view/solution/er-cha-shu-de-you-shi-tu-bfs-dfs-by-huan-b6nh/