111. Minimum Depth of Binary Tree

link

判断最小深度 bfs最快, 但是得注意dfs的写法, 最小跟最大不一样, 最小返回min(left, right) + 1不行, 因为如果左边节点是0就会返回1, 而不会返回叶子结点.

```
public int minDepth(TreeNode root) {
    if (root == null) return 0;
    int L = minDepth(root.left), R = minDepth(root.right);
    int m = Math.min(L, R);
    return 1 + (m > 0 ? m : Math.max(L, R));
}
```

这个是我的写法, 不太好

```
int res = -1;
public int minDepth(TreeNode root) {
    if(root == null) return 0;
    helper(root, 1);
    return res;
}
public void helper(TreeNode root, int depth){
    if(root == null) return;
    if(root.left == null && root.right == null){
        if(res < 0)
            res = depth;
        res = Math.min(res, depth);
    }else{
        helper(root.left, depth + 1);
        helper(root.right, depth + 1);
    }
}
```

最快的bfs

```
public int helper(TreeNode root){
    if(root == null) return 0;
    Queue<TreeNode> queue = new LinkedList();
```

```
        queue.offer(root);
        int level = 1;
        while(!queue.isEmpty()){
            int size = queue.size();
            for(int i = 0; i < size; ++i){
                TreeNode node = queue.poll();
                if(node.left != null)
                    queue.offer(node.left);
                if(node.right != null){
                    queue.offer(node.right);
                }
                if(node.left == null && node.right == null)
                    return level;
            }
            level++;
        }
        return level;
    }
```