# 题目

给定一个二叉树的 根节点root，请找出该二叉树的 最底层 最左边 节点的值。

假设二叉树中至少有一个节点。

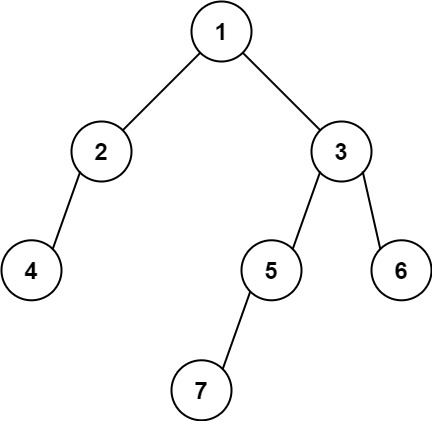
示例 1:



输入: root = [2,1,3]

输出: 1

示例 2:



输入: [1,2,3,4,null,5,6,null,null,7]

输出: 7

提示:

二叉树的节点个数的范围是 [1,104]

-231 <= Node.val <= 231 - 1

# 分析

## 方法一：递归法

class Solution {

public:

int res = 0;

int maxlevel = 0;

int findBottomLeftValue(TreeNode\* root) {

helper(root, 1);

return res;

}

void helper(TreeNode\* root, int level){

if(root == NULL) return;

helper(root->left, level + 1);

if(level > maxlevel){

maxlevel = level;

res = root->val;

}

helper(root->right, level + 1);

}

};

## 方法二：迭代法/宽度优先遍历

class Solution {

public:

int findBottomLeftValue(TreeNode\* root) {

if(root->left == NULL && root->right == NULL)

return root->val;

//if(root == NULL) return -1;

queue<TreeNode\*> qu;

qu.push(root);

while(!qu.empty()){

root = qu.front();

qu.pop();

if(root->right != NULL){

//先把右子节点入队，最后的就是左叶子节点

qu.push(root->right);

}

if(root->left != NULL){

qu.push(root->left);

}

}

return root->val;

}

};