# 题目

给定一个二叉树，返回其节点值自底向上的层次遍历。 （即按从叶子节点所在层到根节点所在的层，逐层从左向右遍历）

**例如：**

给定二叉树 [3,9,20,null,null,15,7],

3

/ \

9 20

/ \

15 7

返回其自底向上的层次遍历为：

[

[15,7],

[9,20],

[3]

]

# 分析

## 方法一：递归法

## 方法二：迭代法

思路：

**代码：**

class Solution

{

public:

vector<vector<int>> levelOrderBottom(TreeNode\* root)

{

vector<vector<int>> res;

if(root==NULL)return res;

queue<TreeNode\*> myQueue;

TreeNode\* p;

myQueue.push(root);

while(!myQueue.empty())

{

vector<int> temp;

int queueSize=myQueue.size();

for(int i=0;i<queueSize;++i)

{

p=myQueue.front();

myQueue.pop();

temp.push\_back(p->val);

if(p->left!=NULL)

myQueue.push(p->left);

if(p->right!=NULL)

myQueue.push(p->right);

}

res.push\_back(temp);

}

reverse(res.begin(),res.end());

return res;

}

};

或：

class Solution {

public:

vector<vector<int>> levelOrderBottom(TreeNode\* root) {

std::queue<TreeNode\*> Q;

Q.push(root);

std::vector<vector<int>> ret;

while(!Q.empty())

{

uint curLevelSize = Q.size();

vector<int> vec;

while(curLevelSize--)

{

TreeNode \*node = Q.front();

Q.pop();

if(!node)

continue;

vec.push\_back(node->val);

Q.push(node->left);

Q.push(node->right);

}

if(vec.size()!=0)

ret.push\_back(vec);

}

reverse(ret.begin(),ret.end());

return ret;

}

};