# 题目

给定一个非空二叉树, 返回一个由每层节点平均值组成的数组。

**示例 1：**

输入：

3

/ \

9 20

/ \

15 7

输出：[3, 14.5, 11]

解释：

第 0 层的平均值是 3 , 第1层是 14.5 , 第2层是 11 。因此返回 [3, 14.5, 11] 。

**提示：**

节点值的范围在32位有符号整数范围内。

# 分析

**思路：**

这里是对层次的操作，因此使用queue进行广度优先遍历。

**代码：**

/\*\*

\* Definition for a binary tree node.

\* struct TreeNode {

\* int val;

\* TreeNode \*left;

\* TreeNode \*right;

\* TreeNode(int x) : val(x), left(NULL), right(NULL) {}

\* };

\*/

class Solution {

public:

vector<double> averageOfLevels(TreeNode\* root) {

vector<double> ret;

queue<TreeNode\*> q;

q.push(root);//广度优先遍历，先将根节点添加到队列

while(!q.empty())//只要队列不为空就继续遍历

{

double sum=0;//用sum存每一层节点值的和

int n=q.size();

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

{

TreeNode\* tmp=q.front();//取出队列的第一个元素

q.pop();

sum+=tmp->val;//将每一层节点的值加起来

//若该节点的左右孩子不为空，则将左右孩子入队

if(tmp->left) q.push(tmp->left);

if(tmp->right) q.push(tmp->right);

}

ret.push\_back(sum/n); //当前层数值平均值

}

return ret;

}

};