**Given a binary tree, return the *bottom-up level order* traversal of its nodes' values. (ie, from left to right, level by level from leaf to root).**

For example:  
Given binary tree {3,9,20,#,#,15,7},

3

/ \

9 20

/ \

15 7

return its bottom-up level order traversal as:

[

[15,7],

[9,20],

[3]

]

**My solution: （8ms，Your runtime beats 21.35% of cppsubmissions.）**

class Solution {

public:

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

if (root != NULL){

level++;

data[total\_level-level].push\_back(root->val);

add(root->left,level);

add(root->right,level);

}

}

int caculate\_level(TreeNode \*root,int level) { **//统计树的层数**

if(root!=NULL) {

level++;

if (data.size() < level){ //如果没有那么多层则需要新建

vector<int>tmp;

data.push\_back(tmp);

}

if(level>count)count=level;

caculate\_level(root->left,level);

caculate\_level(root->right,level);

}

return count;

}

vector<vector<int> > levelOrderBottom(TreeNode\* Root) {**//将每一层加入到对应的容器中**

if (Root == NULL)return data;

count=0;

total\_level=caculate\_level(Root,0);

int level = 0;

add(Root, level);

return data;

}

vector<vector<int> >data;

int count;

int total\_level;

};