## 题目描述

## 请实现一个函数按照之字形打印二叉树，即第一行按照从左到右的顺序打印，第二层按照从右至左的顺序打印，第三行按照从左到右的顺序打印，其他行以此类推。

## 解题思路

/\*\*

 \* 大家的实现很多都是将每层的数据存进ArrayList中，偶数层时进行reverse操作，

 \* 在海量数据时，这样效率太低了。

 \* （我有一次面试，算法考的就是之字形打印二叉树，用了reverse，

 \* 直接被鄙视了，面试官说海量数据时效率根本就不行。）

 \*

**优化**下面的实现：不必将每层的数据存进ArrayList中，偶数层时进行reverse操作，直接按打印顺序存入

 \* 思路：利用Java中的LinkedList的底层实现是双向链表的特点。

 \*     1)可用做队列,实现树的层次遍历

 \*     2)可双向遍历,奇数层时从前向后遍历，偶数层时从后向前遍历

 \*/

算法描述：

/\*

struct TreeNode {

int val;

struct TreeNode \*left;

struct TreeNode \*right;

TreeNode(int x) :

val(x), left(NULL), right(NULL) {

}

};

\*/

class Solution {

public:

vector<vector<int> > Print(TreeNode\* pRoot) {

vector<vector<int> > res;

if(pRoot == NULL)

return res;

queue<TreeNode\*> que;

que.push(pRoot);

bool even = false;

while(!que.empty())

{

vector<int> vec;

const int size = que.size();

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

{

TreeNode\* p = que.front();

que.pop();

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

if(p->left != NULL)

que.push(p->left);

if(p->right != NULL)

que.push(p->right);

}

if(even)

std::reverse(vec.begin(), vec.end());

res.push\_back(vec);

even = !even;

}

return res;

}

};