```cpp
// 之字形打印二叉树
/*
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) {

        // 存储结果
        vector<vector<int> > results;
        vector<int> rlt_temp;

        // 边界条件
        if(pRoot == nullptr)
            return results;

        // 辅助容器
        stack<TreeNode*> stk[2]; // stk[0]是奇数层，stk[1]是偶数层
        int now = 0;
        int next = 1;
        TreeNode* temp=pRoot;

        // 根节点入栈
        stk[now].push(temp);

        // 遍历两个栈，当两个栈均为空时，跳出循环
        while(!stk[now].empty() || !stk[next].empty()){
            // 存储遍历结果
            temp = stk[now].top();
            rlt_temp.push_back(temp->val);
            stk[now].pop();

            // 当前层是奇数或偶数
            if(now==0)
            {
                // 当前层是奇数时，左子树先入栈，右子树后入栈
                if(temp->left!=nullptr)
                    stk[next].push(temp->left);
                if(temp->right!=nullptr)
                    stk[next].push(temp->right);
            }
            else
            {
                // 当前层是偶数时，右子树先入栈，左子树后入栈
                if(temp->right!=nullptr)
                    stk[next].push(temp->right);
                if(temp->left!=nullptr)
                    stk[next].push(temp->left);
            }

            // 当前层为空时，打印下一层
            if(stk[now].empty())
            {
                results.push_back(rlt_temp);
                rlt_temp.clear();
                now=1-now;
                next = 1-next;
            }
        }
        return results;
    }
};

```