**题目描述：输入一棵二叉树，判断该二叉树是否为平衡二叉树.**

代码：

class Solution {

public:

    int caculate\_len(TreeNode \*head){  //求子树的深度

        if (head == NULL)return 0;

        int left = caculate\_len(head->left)+1;

        int right = caculate\_len(head->right)+1;

        return left > right ? left : right;

    }

    bool IsBalanced\_Solution(TreeNode\* pRoot) {

        if (pRoot == NULL)return true;

        int left\_len = caculate\_len(pRoot->left);

        int right\_len = caculate\_len(pRoot->right);

        if (abs(right\_len - left\_len) > 1){

            return false;

        }

        else{

            return IsBalanced\_Solution(pRoot->left) && IsBalanced\_Solution(pRoot->right);

        }

    }

};

题目分析：此题关键主要分为两个部分

A. 求任意一二叉树的深度;

B.  **return IsBalanced\_Solution(pRoot->left)** **&& IsBalanced\_Solution(pRoot->right),** 分别求左右子树是否满足平衡二叉树条件，中间的**&&**表示左右子树都满足条件才返回**true，当左右递归子树中的一个不满足条件时，则整个树都不满足条件。**

C. 三个return先后顺序不能颠倒！

