**题目: 请实现一个函数，用来判断一颗二叉树是不是对称的。注意，如果一个二叉树同此二叉树的镜像是同样的，定义其为对称的。**

1. **代码(示例):**

class Solution {

public:

bool isSymmetrical(TreeNode \*root) {

if (!root) return true;

return compRoot(root->left, root->right);

}

private:

bool compRoot(TreeNode\* lroot, TreeNode\* rroot){

**①**if (lroot==NULL&&rroot==NULL) return true;

**②**if (NULL == rroot&&lroot != NULL || NULL != rroot&&lroot == NULL) return false;

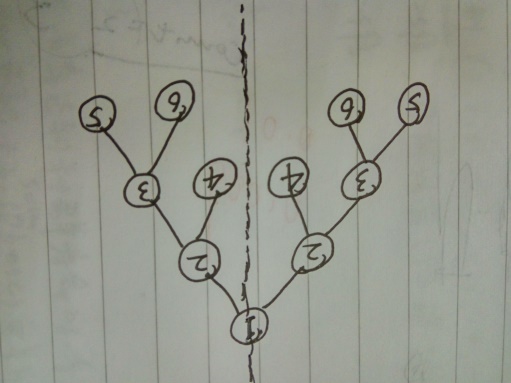
**③**if (lroot->val != rroot->val) return false;

**④return (compRoot(lroot->left, rroot->right) && compRoot(lroot->right, rroot->left));**

}

};

**分析：**

****

1. **Compact()函数比较两个等立结点的对称情况(共有4种情况)，算法精妙之处就是对比较结点的选取(如果两个根结点的是相等的，就将左根的左结点和右根的右结点以及左根的右结点和右根的左结点分别比较).**
2. **利用&&来联立表示对称轴两边的所有等立结点对都相等才返回true.**

**B(我的).**

class Solution {

public:

void LDR(TreeNode\* head){

if (head != NULL){

LDR(head->left);

data.push\_back(head->val);

LDR(head->right);

}

else data.push\_back(-1);

}

bool isSymmetrical(TreeNode\* pRoot){

bool flag = true;

LDR(pRoot);//中序遍历

int size = data.size();

for (int i = 0; i < size / 2; i++){

if (data[i] != data[size - i - 1]){

flag = false;

break;

}

}

return flag;

}

vector<int>data;

};

**错误分析:**程序的本意是想通过中序遍历一颗二叉树来判断是否对称，若对称则中序遍历序列关于根结点对称，且遍历数组data是基数个。但是有一种bug，对称二叉树的遍历序列一定关于根结点对称，但非对称二叉树遍历数组data关于根结点对称，情况如下：

