题目: 输入两颗二叉树A，B，判断B是不是A的子结构。

代码:

class Solution {

public:

bool IsMatch(TreeNode \* p1, TreeNode \* p2)

递归出口

顺序不能变

{

if (p2 == NULL)return true; //右子树的对应结点没有，则返回匹配

if (p1 == NULL)return false; // 左子树的对应结点没有，则返回不匹配

if (p1->val == p2->val) //若相等则以此为根结点递归比较左右结点

return IsMatch(p1->left, p2->left) && IsMatch(p1->right, p2->right);

else return false;

}

bool HasSubtree(TreeNode\* pRoot1, TreeNode\* pRoot2)

{

bool flag = false;

//if(pRoot1 == NULL || pRoot2 == NULL) return false;

if (pRoot1 != NULL && pRoot2 != NULL)

{

if (pRoot1->val == pRoot2->val)

flag = IsMatch(pRoot1, pRoot2);

if (!flag)

flag = HasSubtree(pRoot1->left, pRoot2); //是否为pRoot1为根的子树，一旦根不相等就不可能匹配

if (!flag)

flag = HasSubtree(pRoot1->right, pRoot2);

}

return flag;

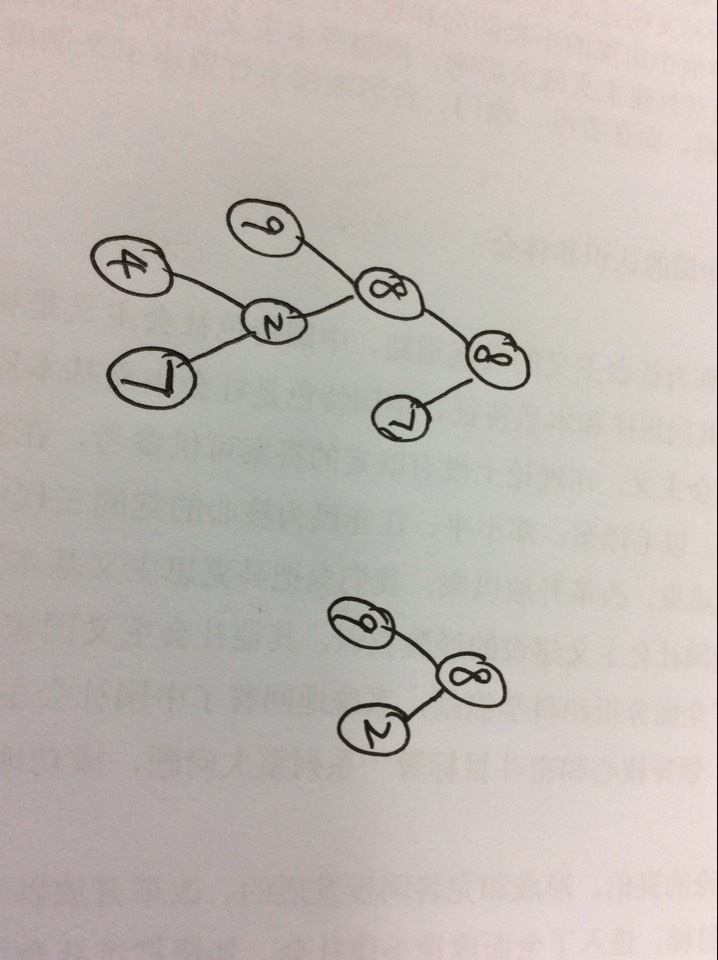
}

};

算法思想:

（1）先找出母树中和子树相匹配的根结点；

（2）顺着根结点依次向下比较每个结点的左右孩子，在对应的位置中，如图：



此题为双递归，具有很好的代表性。