1、以二叉链表作存储结构，建立一棵二叉树， 输出该二叉树的中序遍历序列，层序遍历序列，并求二叉树的深度。

考察：中序遍历(栈)、层次遍历、二叉树深度

二叉链表的类型描述：

typedef char ElemType;

typedef struct BiNode

{ ElemType data;

struct BiNode \*lchild,\*rchild;

}BiNode,\*BiTree;

输入格式:

输入一个二叉树的**先序**序列，孩子为空的位置以#替代。

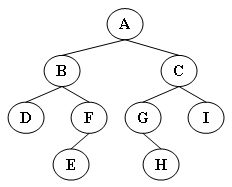
输出格式:

第一行输出该二叉树的**中序**遍历序列。输出的遍历序列中字符间均无间隔。第二行输出二叉树的**层序**遍历序列，第三行输出二叉树的深度。第四行输出二叉树删除F的中序遍历序列。第五行输出第一个叶子节点的父亲节点，第六行输出第一个叶子节点的兄弟节点

**注：**

1. **删除指删除该点及其所有子树**
2. **若节点无父节点或兄弟节点则输出提示信息**
3. 叶子节点的顺序是从左往右  
   具体格式参看输出样例。

对于下图中给出的二叉树：



输入样例:

ABD##FE###CG#H##I##

输出样例:

DBEFAGHCI

ABCDFGIEH

4

DBAGHCI

B

F

1. 创建一个二叉树，结点存放的是整型数据，遵循的规则是：第一个数值作为二叉树的树根，小于父节点的值放在左子节点，大于父节点的值放在右子节点。 在创建好二叉树的基础之上，进行结点统计和删除。

考察点：平衡二叉树建树、遍历、平衡二叉树节点的删除  
统计的内容包括：结点的个数；树的深度；度为0的点的个数；度为1的点的个数；度为2的点的个数。  
删除的策略：如果删除结点是叶子结点，直接删除该点；如果删除结点只有一棵子树，子树的根结点直接替代该结点；如果删除结点同时有左子树和右子树，将右子树中的最小值点移动到删除点的位置上来替代删除结点。

**输入格式:**

输入数值为整型，多行输入。 第一行：二叉树的结点个数N（1≤N≤30）。 第二行：结点的值，以逗号间隔。 第三行：要删除的结点值。

**输出格式:**

多行输出。第1行输出结点个数。 第2行输出树的深度。 第3行度为0的结点个数。 第4行输出度为1的结点的个数。 第5行输出度为2的结点的个数。第6行为删除前的中序遍历结果。第7行为删除前的广度优先遍历结果 。第8行为删除后的中序遍历结果。第9行为删除后的广度优先遍历结果。具体的格式于要求参见输入输出样例。

**输入样例:**

在这里给出一组输入。例如：

5

7,2,5,9,12

7

**输出样例:**

在这里给出相应的输出。例如：

NodeNumbers:5

TreeDepth:3

numbers\_0:2

numbers\_1:2

numbers\_2:1

InOrder:2,5,7,9,12

LayerOrder:7,2,9,5,12

InOrder after:2,5,9,12

LayerOrder after:9,2,12,5

**输入样例2:**

在这里给出一组输入。例如：

1

3

3

**输出样例2:**

在这里给出相应的输出。例如：

NodeNumbers:1

TreeDepth:1

numbers\_0:1

numbers\_1:0

numbers\_2:0

InOrder:3

LayerOrder:3

InOrder after:null

LayerOrder after:null

3、本题要求根据给定的一棵二叉树的先序遍历和中序遍历结果，输出该树的后序遍历结果。

**输入格式:**

第一行给出正整数*N*(≤30)，是树中结点的个数。随后两行，每行给出*N*个字母，分别对应先序遍历和中序遍历结果，字母间以空格分隔。题目保证输入正确对应一棵二叉树。

**输出格式:**

在一行中输出该树的后序遍历结果。字母间有1个空格。

**输入样例:**

10

a b c d e f g h i j

c b e d a h g i j f

**输出样例:**

c e d b h j i g f a