遍历序列还原树:

1. 还原二叉树

由二叉树的先序序列和中序序列，或者中序序列和后序序列，可以唯一地还原一棵二叉树。

注意：由二叉树的先序序列和后序序列不能唯一地还原一棵二叉树。

先序序列和中序序列还原二叉树。

算法步骤:

1）先序序列的第一个字符为根。

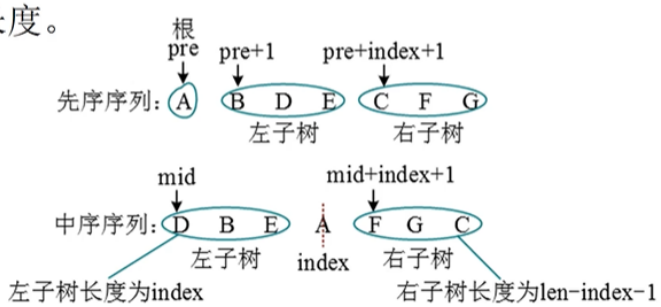
2）在中序序列中，以根为中心划分左右子树。

3）还原左右子树。

函数设置三个参数： pre，mid，len。

pre、mid为指针类型，分别指向先序、中序序列的首地址；

len为序列的长度。



```
Btree pre_mid_createbtree(char *pre,char *mid,int len){ //先序、中序还原二叉树
    if(len==0)
        return NULL;
    char ch=pre[0]; //先序中的第1个节点为根
    int index=0;
    while(mid[index]!=ch){//在中序中找到根的位置，根的左边为该节点的左子树，右边为右子树
        index++;
    }
    Btree T=new Bnode;//创建根节点
    T->data=ch;
    T->lchild=pre_mid_createbtree(pre+1,mid,index);//创建左子树
    T->rchild=pre_mid_createbtree(pre+index+1,mid+index+1,len-index-1);//创建右子树
    return T;
}
```

先序遍历和中序遍历还原二叉树**秘籍**：先序找根，中序分左右。

后序遍历和中序遍历还原二叉树**秘籍**：后序找根，中序分左右。

练习：已知一棵二叉树的后序序列DEBGFCA和中序序列DBEAFGC，画出这棵二叉树。

2. 还原树

树的先根遍历和后根遍历与其对应二叉树的先序遍历和中序遍历相同，先根据对应关系还原为二叉树，再把二叉树转换为树。

当一颗树给出了先根遍历和后根遍历时，我们应当知道它和二叉树之间的对应关系：
先根遍历：对应先序遍历
后根遍历：对应中序遍历

已知一棵树的先根遍历序列ABEFCDGIH和后根遍历序列EFBCIGHDA，画出这棵树。

3. 还原森林

森林的先序遍历和中序遍历与其对应二叉树的先序遍历和中序遍历相同，先根据对应关系还原为二叉树，再把二叉树转换为森林。

已知森林的先序遍历序列ABCDEFGHJI和中序遍历序列BCDAFEJHIG，画出该森林。

二叉树转森林->
我们说把森林转为二叉树非常简单，就是把所有树的树根都看作是兄弟就可以了
那么把二叉树转为森林就是它的反操作，即它的右斜线上都是兄弟