Little Valentine liked playing with binary trees very much. Her favorite game was constructing randomly looking binary trees with capital letters in the nodes.   
This is an example of one of her creations:

D

/ \

/ \

B E

/ \ \

/ \ \

A C G

/

/

F

To record her trees for future generations, she wrote down two strings for each tree: a preorder traversal (root, left subtree, right subtree) and an inorder traversal (left subtree, root, right subtree). For the tree drawn above the preorder traversal is DBACEGF and the inorder traversal is ABCDEFG.   
She thought that such a pair of strings would give enough information to reconstruct the tree later (but she never tried it).   
  
Now, years later, looking again at the strings, she realized that reconstructing the trees was indeed possible, but only because she never had used the same letter twice in the same tree.   
However, doing the reconstruction by hand, soon turned out to be tedious.   
So now she asks you to write a program that does the job for her!

**Input**

The input will contain one or more test cases.   
Each test case consists of one line containing two strings preord and inord, representing the preorder traversal and inorder traversal of a binary tree. Both strings consist of unique capital letters. (Thus they are not longer than 26 characters.)   
Input is terminated by end of file.

**Output**

For each test case, recover Valentine's binary tree and print one line containing the tree's postorder traversal (left subtree, right subtree, root).

**Sample Input**

DBACEGF ABCDEFG

BCAD CBAD

**Sample Output**

ACBFGED

CDAB

就是个简单的二叉树已知前序和中序，写出后序

#include <iostream>

#include<stdio.h>

#include<cstring>

using namespace std;

void build(int n,char\* s1,char\* s2)

{

if(n<=0)return;

int p=strchr(s2,s1[0])-s2;//s1是先序，s2是中序//strchr返回的是地址，s2也是地址

//找根在s2中的位置

//p的大小就是左子树的长度

build(p,s1+1,s2);//左子树//因为s2开头就是左子树,这个传递只是传递了个地址（并不是想的那样传递了整个s2）

build(n-p-1,s1+p+1,s2+p+1);//右子树

//n-p-1是右子树的长度，p是左子树长度，1是根节点

//不管s1，s2最后的一定都是右子数树

cout<<s1[0];

}

int main()

{

//freopen("input.txt","r",stdin);

char s1[30],s2[30];

while(cin>>s1>>s2)

{

build(strlen(s1),s1,s2);

cout<<"\n";

}

return 0;

}

注意这里对指针的用法