一、编程题

ACM：二叉树的广度优先遍历

题目描述：

有一棵二叉树，每个节点由一个大写字母标识(最多26个节点）。现有两组字母，分别表示后序遍历（左孩子->右孩子->父节点）和中序遍历（左孩子->父节点->右孩子）的结果，请输出层次遍历的结果。

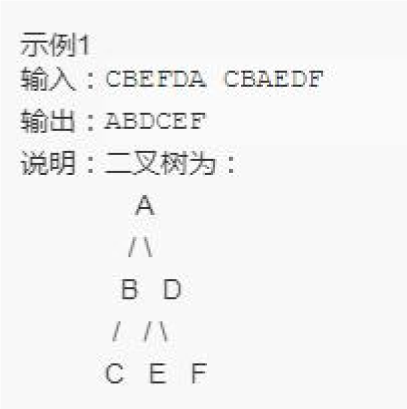
输入描述：

输入为两个字符串，分别是二叉树的后续遍历和中序遍历结果。

输出描述：

输出二叉树的层次遍历结果。

补充说明：



示例1

输入：CBEFDA CBAEDF

输出：ABDCEF

说明：二叉树为：

```
     A
    / \
   B   D
  /   / \
 C   E   F
```

代码：

```
class Node:
    def __init__(self, id, left= None, right = None):
        self.id = id
        self.left = left
        self.right = right



tree1, tree2 = input().split()

def getSup(t1, t2):
    if len(t1)==1:
        return Node(t1[0])
    if len(t1) == 0:
        return None
    sup = t1[-1]
    ind = t2.find(sup)
    leftt2 = t2[:ind]
    rightt2 = t2[ind+1:]
    leftt1 = [i for i in t1 if i in leftt2]
    rightt1 = [i for i in t1 if i in rightt2]
    left = getSup(leftt1, leftt2)
    right = getSup(rightt1, rightt2)
    supnode = Node(sup, left, right)
    return supnode


whole = getSup(tree1, tree2)
ans = ''
temp = [whole]
while temp != []:
    next_level = []
    for n in temp:
        ans += n.id
        if n.left!= None:
            next_level.append(n.left)
        if n.right != None:
            next_level.append(n.right)
    temp = next_level

print(ans)
```