

Problem J. Anagrams by Stack

Source file name: J.c, J.cpp, J.java
 Input: Standard
 Output: Standard
 Author(s):

A school teacher wants to simplify the concept of a stack and its operations to his classroom. He wants a program to show them how anagrams can result from sequences of stack operations: 'push' and 'pop'. He'll be illustrating on 4- letter words. There are two sequences of stack operators which can convert TROT to TORT:

```
[
i i i i o o o o
i o i i o o i o
]
```

where i stands for Push and o stands for Pop. Given pairs of words, your program should produce sequences of stack operations which convert the first word to the second.

A stack is a data storage and retrieval structure permitting two operations:

Push - to insert an item and

Pop - to retrieve the most recently pushed item

We will use the symbol i (in) for push and o (out) for pop operations for an initially empty stack of characters. Given an input word, some sequences of push and pop operations are valid in that every character of the word is both pushed and popped, and furthermore, no attempt is ever made to pop the empty stack. For example, if the word FOO is input, then the sequence:

i i o i o o is valid, but

i i o

Is not (it's too short), neither is

i i o o o i

(there's an illegal pop of an empty stack)Valid sequences yield rearrangements of the letters in an input word. For example, the input word FOO and the sequence i i o i o o produce the anagram OOF. So also would the sequence i i i o o o. You are to write a program to input pairs of words and output all the valid sequences of i and o which will produce the second member of each pair from the first.

Input

The input file will consist of several pairs of input lines. The first line of the file will include a single integer, indicating the number of pairs of input lines that will follow. The first line of each pair of input lines is to be considered as a 4-letter source word .The second line is the 4-letter target word.

Output

For each input pair, your program should produce a sorted list of valid sequences of i and o which produce the target word from the source word. Each list should be delimited by

```
[
]
```

on a separate line. The sequences should be printed in "dictionary order". Within each sequence, each i and o should be followed by a single space. Each sequence should be on a separate line.

Example

Input	Output
3 mada adam long nice eric rice	Output for mada adam [i i i i o o o o i i o i o i o o] Output for long nice [] Output for eric rice [i i o i o i o o]