5369 Anagrams by Stack

How can anagrams result from sequences of stack operations? 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. Your program should, given pairs of words produce sequences of stack operations which convert the first word to the second.

Input

The input will consist of several lines of input. The first line of each pair of input lines is to be considered as a source word (which does not include the end-of-line character). The second line (again, not including the end-of-line character) of each pair is a 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

```
[
]
```

and the sequences should be printed in "dictionary order". Within each sequence, each '`i`' and '`o`' is followed by a single space and each sequence is terminated by a new line.

Process

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.

Sample Input

```
madam
adamm
bahama
bahama
long
short
eric
rice
```

Sample Output

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