Joined up writing

Say that two words *join up* if a proper suffix of one is a proper prefix of the next. For instance the words "suffix" and "fixture" join up. For a pair of joined up words, the suffix of one which is a prefix of the other will be called the *common part*. We specify two slight variations:

singly joined the common part is at least half as long as one of the two words, and

doubly joined the common part is at least half as long as both words.

The basic problem will be to find, for a given "dictionary" a shortest sequence of joined up words that link a beginning word to an end word. For instance:

```
bard ardent entire
```

is a sequence linking "bard" to "entire" in which each pair is doubly joined. On the other hand,

```
suffix fixture read
```

is a singly joined sequence.

Task

The I/O requirements for this task are **strict**.

- Your program should take two command line parameters (words to join up).
- A dictionary of available words will be input from `stdin`. This dictionary could contain up to 100,000 "words" (which need not be English words). However, the words will consist only of the characters `a` through `z` (i.e., lower case Latin letters).

The output (to `stdout`) from any single run should be two lines:

- The first line should consist of a non-negative integer which is the length of the shortest singly joined sequence between the two words (0 if no chain exists), followed by such a sequence (if it exists).
- The second line should be similar, but report on doubly joined chains.

For instance, using the example above, one might see (depending on the words in the dictionary of course):

```
$ java JoinUp suffix read < dict.txt
3 suffix fixture read
6 suffix fixage agent entire ire read
```