A spelling suggestion is a part of spelling correction program that generates a set of plausible replacements for words that are likely to be misspelled. One way to measure the plausibility of these replacements is to compute their edit distance against a given misspelled word. The edit distance between two words is the total number of edit operations that have to be done in order to transform one word into the other. Normally these edit operations are insertion, deletion and substitution of a single character including transposition of 2 consecutive characters.

For example, for a word "`wonder`", if the deletion is applied at the character '`o`', this word will transform into "`wnder`". And if the substitution with '`a`' is applied at '`o`', it will become "`wander`". And if the transposition is applied at "`er`", it will become "`wondre`".

In this edit distance strategy, the degree of similarity between two words is up to their minimum edit distance. If the minimum edit distance between $word_1$ and $word_2$ is lower than the distance between $word_1$ and $word_3$, then $word_1$ is more similar to $word_2$ than to $word_3$. So the $word_2$ is a better spelling suggestion for $word_1$, comparing with $word_3$.

Suppose that you are an employee of a software company which needs to build up a prototype of spelling suggestion program. This prototype tends to be a part of word processing software. The requirement is that it has to use the edit distance strategy for their spelling suggestion. But the substitution operation has to be redefined to match the behavior of mistyping. The cost of substitution of a character with another character depends on the position of them on the keyboard layout. If they are close to each other, the cost is only half of the normal one. For this purpose, the substitution is categorized into near-substitution and far-substitution. Their costs are defined as 1 and 2 respectively. And the costs for insertion, deletion and transposition are 2. In addition, this program must run fast enough to pass the time limit that is set by your manager. By the way for this prototype, an English QWERTY keyboard layout is chosen to be used.

Goal

To generate optimum spelling suggestions for each input word, where each optimum spelling suggestion is a word in dictionary that has the least minimum edit distance from the given input word. The time limit for 5,000 misspelled words is less than or equal to 5 minutes.

Input

Input is a standard input which contains 3 parts of data. Each of these three parts ends with a blank line.

- The first part is a set of near-substitution rules, where each rule is kept in one line. Each line has two fields. Each filed is separated by a space. The total no. of rules is less than or equal to 150
  - The first field is a character where it can be near-substituted with other characters.
  - The second field is a sequence of characters which can be near-substituted for the character in the previous field. There is no space in this field.
  - The characters that may be contained in this part are characters that can be typed in using a generic English keyboard layout, which are alpha-numeric characters and some punctuations without space or tab. They are listed as the following.

    ```
    abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
    0123456789‘~!@#$%^&*()-_=+\|[{]};:’",<.>/?
    ```

- The second part is a sequence of words in dictionary, where each word is kept in one line. The total no. of words is less than or equal to 150,000.
  - The characters in dictionary are alphabetical characters with an apostrophe punctuation.
    - `abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ’`
- The third part is a sequence of words that need to be checked for their spellings. Each of these words is also kept in one line. The total no. of words is less than or equal to 5,000.
  - The characters that may be contained in this part are the same as characters in the first part, which are alpha-numeric characters and some punctuation. (See the first part.)
- Since each of these three parts end with a blank line, the third blank line is the termination of the input.

Output

For each word in the third part, write a line which contains 3 parts of information, separated with a colon.

- The first part is the given input word.
- The second part is the minimum edit distance between the input word and suggestion word(s).
- The third part is an ascending sorted sequence of suggestion word(s), separated with a space. There is no space left after the last word.

More Explanations:

In this sample input, there are 5 near-substitution rules (line no. 1-5), 10 words in dictionary (line no. 7-16) and 12 words looking for their suggestions (line no. 18-29).

In the sample output, there are 12 lines for each corresponding 12 words from the input.

For the 1st word, the minimum edit distance between `x` and its suggestions (`A B Z a b`) is 2. All of them are the (far) substitution costs.

For the 2-nd word, the minimum edit distance between s and its suggestion (`a`) is 1, which is a near-substitution cost, guided by the 1st substitution rule.

For the 3rd word, the minimum edit distance between `z` and its suggestion (`A Z a`) are 1, which is a near-substitution cost, guided by the 1st or 5th substitution rule.

For the 4th word, the minimum edit distance is 4, which are summed from one far-substitution cost and one deletion cost.

For the 5th word, the minimum edit distance is 6. The costs between `xxx` and (`A B Z a b`) are from two deletion and one far-substitution costs. The cost between `xxx` and `ABC` re from three far-substitution costs.

For the 6th word, the cost between `angre` and `anger` is from one transposition costs. The costs between `angre` and (`angle angry`) are from one far-substitution costs.

For the 7th word and 8th word, seem to be similar. But the cost of 7th word is from one far-substitution costs. But the cost between `angrt` and `anger` is from 2 near-substitution costs (`r` substitutes with `e` and `t` substitutes with `r`) according to the near-substitution rule no. 4.

For the 9th word, the word is exactly matched within dictionary. So the cost is 0.

For the 10th and 11th words, each cost is from one transposition costs.

For the 12th words, the cost between `CAB` and (`A B`) is from two deletion costs. The cost between `CAB` and (`ABC`) is from one deletion and one insertion costs. Please be notify that it is not from 2 transpositions of `CAB` to `ACB` and then `ACB` to `ABC`.

Sample Input

```
a AqQsSzZ
b BgGvVnN
p P0);:oO[{
r R4$fFeEtT
z ZaAxX

a
A
b
B
Z
angel
angle
anger
angry
ABC

x
s
z
xx
xxx
angre
angri
angrt
angel
ACB
BAC
CAB
```

Sample Output

```
x:2:A B Z a b
s:1:a
z:1:A Z a
xx:4:A B Z a b
xxx:6:A ABC B Z a b
angre:2:anger angle angry
angri:2:angry
angrt:2:anger angry
angel:0:angel
ACB:2:ABC
BAC:2:ABC
CAB:4:A ABC B
```