

1. One use of a spell-checking program is to allow one to write in abbreviations, which a program replaces when there is a unique expansion. For this problem, we want a program that takes as input a sequence of words constituting a *lexicon*, followed by an abbreviated message. Each word, W , in the message is replaced by a word, W' , from the lexicon if W' is the unique shortest word in the lexicon that can be formed by inserting 0 or more letters into W . If no W' satisfies this criterion, then W is passed through to the output unchanged.

For example, if the lexicon contains

```
arrived truly tally shipment has yours your our congratulations
```

and the message is

```
yr pmt hs arrd.  our congr. yrs try, jack
```

the output would be

```
your shipment has arrived.  our congratulations. yours truly, jack
```

Had the input contained the word `tly` instead of `try`, on the other hand, it would have remained unchanged, since both `truly` and `tally` match it.

The input to your program will consist of a sequence of words in free format (a ‘word’ here is simply a contiguous sequence of letters and digits), followed by a word consisting of a single slash—preceded by whitespace and followed by a newline—after which comes an arbitrary sequence of lines of abbreviated text. The words up to the slash are the lexicon. The abbreviated words are delimited by any characters that are not letters or digits.

The output should consist of the lines after the slash, with all abbreviated words replaced as indicated above (and others unchanged). Preserve all characters that are not letters or digits (in particular, punctuation and whitespace).

Example:

Input	Output
arrived truly shipment tally	your shipment has arrived.
has yours your	our congratulations.
our	yours truly, jack
congratulations /	
yr pmt hs arrd.	
our congr.	
yrs try, jack	

2. [From the 23rd Annual ACM Collegiate Programming Contest World Finals] To enable housebuyers to estimate the cost of flood insurance, a real-estate firm provides clients with the elevation of each 10-meter by 10-meter square of land in regions where houses may be purchased. Water from rain, melting snow, and burst water mains will collect first in those squares with the lowest elevations, since water from squares of higher elevation will run downhill. For simplicity, we also assume that storm sewers enable water from high-elevation squares in valleys (completely enclosed by still higher elevation squares) to drain to lower elevation squares, and that water will not be absorbed by the land.

From weather data archives, we know the typical volume of water that collects in a region. As prospective housebuyers, we wish to know the elevation of the water after it has collected in low-lying squares, and also the percentage of the region's area that is completely submerged (that is, the percentage of 10-meter squares whose elevation is strictly less than the water level). You are to write the program that provides these results.

The input consists of a sequence of region descriptions. Each begins with a pair of integers, P and Q, each less than 30, giving the dimensions of the rectangular region in 10-meter units. Immediately following are P lines of Q integers giving the elevations of the squares in row-major order. Elevations are given in meters, with positive and negative numbers representing elevations above and below sea level, respectively. The final value in each region description is an integer that indicates the number of cubic meters of water that will collect in the region. A pair of zeroes follows the description of the last region. For each region, display the region number (1, 2, ...), the water level (in meters above or below sea level) and the percentage of the region's area under water, each on a separate line. The water level and percentage of the region's area under water are to be displayed accurate to two fractional digits. Follow the output for each region with a blank line.

Example:

Input	Output
<pre>3 3 25 37 45 51 12 349 4 83 27 13000 0 0</pre>	<pre>Region 1 Water level is 46.67 meters. 66.67 percent of the region is under water.</pre>

3. [S. Skiena and M. A. Revilla] A sequence $x_1x_2\dots x_n$ is a *subsequence* of $y_1y_2\dots y_m$, where $n \leq m$, if $x_1 = y_{k_1}, \dots, x_n = y_{k_n}$, where $0 < k_1 < k_2 < \dots < k_n \leq m$. For input strings X and Y , you are to count the number of ways in which X appears as a subsequence of Y (that is, the number of different index sequences k_1, \dots, k_n that satisfy the definition of subsequence above). For example, if X is “bag” and Y is “ababgbag”, then X appears in Y in five different ways: characters (2, 3, 5), (2, 3, 8), (2, 7, 8), (4, 7, 8), and (6, 7, 8).

The input to your program will be a sequence of strings in free format. Each string will consist of no more than 10,000 lower-case letters. Each test case consists of two of these strings, the first being Y and the second being X .

For each (Y, X) pair, output the number of ways X appears as a subsequence of Y in the format shown in the example below. Assume that each answer is expressible as a 63-bit unsigned integer (Java type `long` or C/C++ type `long long int`).

Input	Output
babgbag bag	"bag" occurs 5 times in "babgbag" "rabbit" occurs 3 times in "rabbbit"
rabbit rabbit tarmack cam	"cam" does not occur in "tarmack"