ACM (ACronym Maker)

Input: `acm.in`
Output: `standard output`

The sadists who design problems for ACM programming contests often like to include the abbreviation "ACM" somewhere in their problem descriptions. Thus, in years past, the World Finals has had problems involving "Apartment Construction Management," the "Atheneum of Culture and Movies", the "Association of Cover Manufacturers", "ACM Airlines", the "Association for Computa- tional Marinelife", and even an insect named "Amelia Cheese Mite". However, the number of word combinations beginning with A, C, and M that make sense is finite and the problem writers are starting to run out of ideas (it's hard to think of problems about "Antidisestablishmentarianistic Chthonian Metalinguistics"). Fortunately, modern culture allows more flexibility in designing abbreviations – consider, for example:

GDB – Gnu DeBugger
LINUX – either "LINus's UniX" or "LINUs's miniX" or "Linux Is Not UniX"
SNOBOL – StriNg Oriented symBOlic Language
SPITBOL – SPeedy ImplemenTation of snoBOL

The rules used in these examples seem to be:

- Insignificant words (such as "of", "a", "the", etc.) are ignored.
- The letters of the abbreviation must appear, in the correct order, as an ordered sublist of the letters in the significant words of the phrase to be abbreviated.
- At least one letter of the abbreviation must come from every significant word (multiple occurrences of a letter are, of course, treated as distinct).

Of course these rules are often broken in real life. For instance, RADAR is an abbreviation for "RAdio Detecting And Ranging". Under our rules (assuming that "and" is an insignificant word), this would not be a valid abbreviation (however, RADR or RADRAN or DODGING would be valid). You have been asked to take a list of insignificant words and a list of abbreviations and phrases and to determine in how many ways each abbreviation can be formed from the corresponding phrase according to the rules above.

Input

The input file consists of multiple scenarios. Each scenario begins with an integer $100 \geq n \geq 1$ followed by n insignificant words, all in lower case, one per line with no extra white space. (A line containing 0 indicates end of input.) Following this are one or more test cases for this scenario, one per line, followed by a line containing the phrase "LAST CASE". Each line containing a test case begins with an abbreviation (uppercase letters only) followed by a phrase (lowercase letters and spaces only). The abbreviation has length at least 1 and the phrase contains at least one significant word. No input line (including abbreviation, phrase, and spaces) will contain more than 150 characters. Within these limits, however, abbreviations and phrase words may be any length.

Output

For each test case, output the abbreviation followed by either
`is not a valid abbreviation`
or
`can be formed in` i `ways`
where i is the number of different ways in which the letters of the abbreviation may be assigned to the letters in the phrase according to the rules above. The value of i will not exceed the range of a 32-bit signed integer.

Sample Input

```
2
and
of
ACM academy of computer makers
RADAR radio detection and ranging
LAST CASE
2
a
an
APPLY an apple a day
LAST CASE
0
```

Sample Output

```
ACM can be formed in 2 ways
RADAR is not a valid abbreviation
APPLY can be formed in 1 ways
```