CSULA PROGFEST 2014
Problem 8
Text Formalization

One duty Jimmy has at the ACM is to formalize the language and grammar used in texts. Part of this job is expanding contractions and certain acronyms. A contraction in English is a word or phrase formed by omitting or combining some of the sounds of a longer phrase. For example, "don't" is a contraction for "do not" and "o'clock" comes from "of the clock."

An acronym is a series of letters (or word) formed from the initial letters of a name or from combining parts of a series of words. For example, "ACM" for "Association for Computing Machinery" or "radar" for "radio detecting and ranging."

Your job is to take a list of contractions and acronyms, and expand all contractions and some acronyms in a text.

Input

Input begins with two numbers, $C < 50$ and $A < 50$, indicating respectively the number of contractions and acronyms Jimmy must expand. The next C lines list a contraction and its formal expansion. Following will be a list of A acronyms and their expansions, each on individual lines. Both contractions and acronyms will be presented in the following format: "contraction or acronym" -> "expansion"

Output

Output each text exactly as input, except for necessary expansions. All contractions must be fully expanded. Each contraction may appear as listed, entirely uppercase, or capitalized (first letter uppercase, remaining letters as listed). The expansion should follow the same rule; if a contraction is uppercased, the expansion should be uppercased as well. If more than one case applies, choose the earliest matching case in the list: "as listed," "uppercased," and "capitalized."

Since acronyms are useful for understanding and identifying names, only modify the first instance of an acronym in each text. An instance of an acronym must match the case exactly ("acm" is not an instance of "ACM"). The modification consists of replacing the acronym with the expansion, followed by a space, followed by the acronym in brackets. This allows the reader to connect the acronym with the fully expanded term. Ignore recursive acronyms that are created after expanding some previous text.

The terminating line of '#' should be printed after each text. If more than one expansion or acronym match can be valid, use the one which starts earlier in the text. If several begin at the same letter, use the one appearing earliest in the input lists. Use the sample below to illustrate the process.

Sample Input

```
3 4
"doesn't" -> "does not"
"isn't" -> "is not"
"can't" -> "cannot"
"ACM" -> "Association for Computing Machinery"
"CSULA" -> "California State University, Los Angeles"
"IO" -> "input output"
"ASAP" -> "as soon as possible"
Creating problems for the CSULA Progfest programming contest
isn't an easy feat.
We have to solve all the problems before hand and we can't just use the sample IO files.
If you find any errors, please submit them on the website and we will correct them ASAP.
Signed,

ACM CSULA Chapter
#
The ACM CSULA Chapter,
welcomes students of all majors to join.
But let's be serious, non-programmers can't understand programming puns.
#
```

Sample Output

```
Creating problems for the California State University, Los Angeles (CSULA) Progfest programming contest
is not an easy feat.
We have to solve all the problems before hand and we cannot just use the sample input output (IO) files.
If you find any errors, please submit them on the website and we will correct them as soon as possible (ASAP).
Signed,

Association for Computing Machinery (ACM) CSULA Chapter
#
The Association for Computing Machinery (ACM) California State University, Los Angeles (CSULA) Chapter,
welcomes students of all majors to join.
But let's be serious, non-programmers cannot understand programming puns.
```