Problem 4 - Relevance Index

Given several paragraphs (lines) of text and a search word, the **relevance index of a paragraph** is the **number of occurrences** of that **search word** (as a separate word) in **the paragraph**. Here, **comparison is case insensitive**, i.e. the only thing that is important is the words have the same letters, ignoring casing.

Hence, ordering text by relevance index means creating a **new text** and

- Finding the paragraph with the highest relevance index (in the old text) and placing it first
- Then finding the paragraph with the second-highest relevance index (in the old text) and placing it second
- and so on…

You are tasked with **ordering paragraphs by relevance index, given a search word**. Furthermore, **all letters** of **all occurrences of the search word** must be **made uppercase** in the **resulting ordered text** and **any punctuation should be removed and words should be separated by a single space**.

The **paragraphs can contain English letters** (upper- and lower-case), **spaces and punctuation** to **separate words**. Punctuation characters are: **`","`, `"."`, `"("`, `")"`, `";"`, `"-"`, `"!"`, `"?"`** (comma, dot, parentheses, semicolon, dash, exclamation mark, question mark).

Input

The input data should be read from the console.

On the first line of the input, there will be a **single string – the search word**.

On the second line there will be a single **integer number L – the number of paragraphs in the text**.

On **each of the next L lines** there will be a **single string – the text in the respective paragraph**.

The input data will always be valid and in the format described. There is no need to check it explicitly.

Output

The output data should be printed on the console. Print **exactly L lines** – the **paragraphs** with **removed punctuation**, each two **words separated by a single space** and the **words matching the search word converted to uppercase**.

Constraints

- **L** will be no more than **100**
- Each **line will be no longer than 1000 characters**
- Each symbol in the original text will have an ASCII code from **0** to **255**
- Time limit: **0.10 seconds**
- Memory limit: **16 MB**

Examples

Input	Output
a 3	c A A A b A A

a b c b a a c a a a	A b c

Input
text 5 if you have someone to text but the text is more than text to text and you need a better text to text try to text what the text would want to text another text with text cause this text is too much about text, and a text will never teach you how to text
Output
try to TEXT what the TEXT would want to TEXT another TEXT with TEXT cause this TEXT is too much about TEXT and a TEXT will never teach you how to TEXT but the TEXT is more than TEXT to TEXT and you need a better TEXT to TEXT if you have someone to TEXT