Book Cipher

Time limit: *2000 ms*
Memory limit: *256 MB*

IEEE Xplore

Warm greetings to all IEEEXtreme Participants from the Xplore API Team!

In this challenge, which is described below, you will be tasked with a programming challenge that uses documents in the form retrieved from the IEEE Xplore API.

For a full dynamic database search IEEE Xplore API is available for your IEEE research needs. Xplore API provides metadata on 4.9mm academic works and is now delivering full-text content on 50k 'Open Access' articles. Xplore API will make your research needs fast and easy. The Xplore API Portal supports PHP, Python and Java as well as providing output in Json and XML formats. Many API use cases are listed within the API Portal.

Xplore API registration is free. To learn more about IEEE Xplore API please visit developer.ieee.org/ and register for an API key TODAY!

Challenge

A book cipher is a cipher in which the key is some aspect of a book. Book ciphers work by replacing words in the plaintext of a message with the location of letters from the book being used.

Your challenge is to create a row/column cipher based on the letters in the XML representation of an IEEE Xplore article and create the encrypted code for the phrase. Rows of the cipher are numbered 1 to R from top to bottom, and columns of the cipher are numbered 1 to C from left to right. Each character in the cipher is identified by its row and cell (r, c). The ciphertext contains the r, c positions of each letter in the cipher.

Note that you should remove all XML tags before creating the cipher from the article. Any string starting with the '<' symbol and ending with the '>' symbol is an XML tag.

Standard input

The first line of the input contains an integer p, giving the number of phrases to be encrypted.

The second line of input contains an integer n, indicating the number of lines in the XML representation of Xplore Article.

The third line of input has two integers R, C which give the number of rows and columns, respectively, in the cipher grid. You will fill this grid with the content of the Xplore document. The grid may not be large enough to contain the entire document.

The fourth line of input contains a character that indicates the position code in the cipher for each character. Since a character can appear in the cipher grid more than once you must select the appropriate instance. S denotes finding the first (R_i, C_i) instance of the character in the grid and L denotes finding the last (R_i, C_i) instance of the character. For ordering, consider the order that characters are added to the grid.

The next p lines of the testcase give the phrases to be encrypted.

The final n lines of the testcase are the XML representation of the Xplore document.

Standard output

For each of the p phrases to be encrypted, output either:

- The encoded message in the form $R_1, C_1, R_2, C_2, \ldots, R_z, C_z$

- 0, if the ciphertext cannot be calculated using the cipher.

Constraints and notes

- $1 \leq p \leq 100$
- $1 \leq R, C \leq 2000$
- $1 \leq n \leq 1000$

Each phrase to be encrypted will be between 1 and 1300 characters.

Each input file will be less than 250KB in size.

Input

```
2
8
10,5
S
Hi
Dad
<?xml version="1.0" encoding="UTF-8"?>
<response><body>
<p>abcd efg hijk</p>
<p>ABCD EFG HIJK</p>
<p>abcd efg hijk</p>
<p>ABCD EFG HIJK</p>
</body>
</response>
```

Output

```
5,3,3,1
4,2,1,1,1,4
```

Explanation

The sample input has two phrases to be encrypted.

The grid is calculated by removing all XML tags and printing each character (which includes spaces, numbers, and special characters) in a matrix 10 rows by 5 columns. Notice some characters at the end of the XML document do not appear since the matrix is not sufficient in size. We are using the underscore character for a space in the grid below.

```
abcd_
efg_h
ijkAB
CD_EF
G_HIJ
Kabcd
_efg_
hijkA
BCD_E
FG_HI
```

We are using the smallest coordinate for our encryption.

The first phrase to be encrypted is "Hi".

- Capital H appears at the smallest coordinate 5,3 (r,c).
- Lower case i appears at the smallest coordinate 3,1 (r,c).

Therefore, the first line of output is 5,3,3,1.

The second phrase to be encrypted is "Dad".

- Capital D appears at the smallest coordinate 4,2 (r,c).
- Lower case a appears at the smallest coordinate 1,1 (r,c).
- Lower case d appears at the smallest coordinate 1,4 (r,c).

Therefore, the second line of output is 4,2,1,1,1,4.