Problem 142: Codebreaker

Difficulty: Medium

Author: Brett Reynolds, Annapolis Junction, Maryland, United States

Originally Published: Code Quest 2021

Problem Background

A substitution cipher is a cipher that replaces letters (or sometimes groups of letters) with other letters (or groups), effectively scrambling a message and making it illegible. For example, a Caesar cipher, likely the first cipher ever created, works by "shifting" the alphabet. Each letter in the original "plaintext" message is replaced with another letter a certain number of positions further down the alphabet to create the encrypted "ciphertext." For example, in English with a shift of 3, A becomes D, B becomes E, and so on (X, Y, and Z wrap around to be replaced with A, B, and C respectively).

Plaintext: `ABCDEFGHIJKLMNOPQRSTUVWXYZ`

Caesar Shift 3: `DEFGHIJKLMNOPQRSTUVWXYZABC`

Substitution ciphers have a major vulnerability, however; some letters in the alphabet occur more frequently than others. Vowels in particular (in English, A, E, I, O, and U) are very common letters in any language, because there are so few of them and yet they are required to make words pronounceable. Certain consonants also occur more frequently than others (R, S, and T are much more common in English than Q, X, and Z). By counting how often certain letters occur within a message encrypted with a substitution cipher, a cryptanalyst (codebreaker) might be able to crack the cipher. By creating a chart showing the relative frequency of each letter in the ciphertext and comparing it against the average frequency of each letter in the expected language, a cryptanalyst can guess the substitutions used and begin to piece together the original message.

Problem Description

You're working on a new cybersecurity team within Lockheed Martin to help the National Security Agency develop a new frequency analysis database. You must write a program that is able to read in a large amount of text and count the number of occurrences of each letter of the English alphabet. It must then print the relative frequency of each number in a list, for use in future codebreaking endeavors.

The relative frequency of a letter is determined by the following formula, and is expressed as a percentage value:

$$Relative\ frequency = \left(\frac{Occurrences\ of\ letter}{Total\ number\ of\ letters}\right) \times 100\%$$

For example, in the sentence "The cow is brown," there are 13 letters. The letter 'o' appears twice, giving it a relative frequency of $(2 \div 13) \times 100\% = 15.38\%$. Punctuation and spaces should not be taken into account when calculating relative frequencies. Uppercase and lowercase versions of a letter should be counted as the same letter.

Sample Input

The first line of your program's input, received from the standard input channel, will contain a positive integer representing the number of test cases. Each test case will include:

- A line containing a positive integer, X, indicating the number of lines of text to be provided.
- X lines, containing text to be analyzed. Lines may contain up to 2000 characters each, and can contain any US ASCII character.

```
1
3
The quick red fox jumps over the lazy brown dog.
The above sentence contains every letter in the English language.
Don't forget to ignore punctuation and numbers; they're not relevant!
```

Sample Output

For each test case, your program must print a line for each letter in the English language, in order, using the following format:

- The letter, in uppercase
- A colon (:)
- A space
- The relative frequency of that letter within the text provided in the test case, rounded to two decimal places and including any trailing zeroes
- A percent sign (%)

A: 5.37%
B: 2.01%
C: 2.68%
D: 2.68%
E: 15.44%
F: 1.34%
G: 4.03%
H: 4.03%
I: 4.03%
J: 0.67%
K: 0.67%
L: 3.36%
M: 1.34%
N: 10.74%
O: 8.05%
P: 1.34%
Q: 0.67%
R: 6.71%
S: 3.36%
T: 10.74%
U: 4.03%
V: 2.68%
W: 0.67%
X: 0.67%
Y: 2.01%
Z: 0.67%