Word Co-occurrence

Evaluation:

We are looking for people who can write great code, solve problems correctly, and do so rapidly. The following is a genuine problem that you might need to work on, and we'll be assessing you on both correctness, and speed.

If your program gives the correct answer for our test input, then we'll look at how quickly you finished your solution. You should aim to submit your solution within 1 hour of receiving this task.

See the example input/output on the second page of this spec. You must match this exactly.

Background:

Word co-occurrence is a way of determining how commonly two words appear together in a language. For example, the words "for" and "example" appear more commonly together than "programming" and "evaluations" in common English.

One way of defining co-occurrence is: for each occurrence of word A, how often does word B appear within K words?

For example, in the sentence "$I\ like\ programming, and\ I\ like\ programming\ evaluations.$", if $A =$ "$programming$", $B =$ "$evaluations$", and $k = 3$, then both occurrences of "$programming$" have the word "$evaluatios$" within 3 words range. So the

$$co\_occurrence(A = \text{“}programming\text{”}, B = \text{“}like\text{”}, k = 3) = 2$$

(Note that the first "$programming$" had the word "$like$" within ±3 words' range twice, but we only count this once.)

The co-occurrence probability is then defined as:

$$P(a, b, k) = \frac{co\_occurrence(a, b, k)}{\#a}$$

This can be thought of as: "out of every time the word A appears, how many of those times does B appear within k words".

$$P(programming, like, 3) = \frac{2}{2} = 1$$

$$P(programming, evaluations, 3) = \frac{1}{2} = 0.5$$

$$P(evaluations, programming, 3) = \frac{1}{1} = 1$$

Task:

Given an input document, write a program to calculate all co-occurrence probabilities for a document and given range k. Your program will then be given multiple input lines from stdin with one word pair A and B per line; for each input line, your program should print the co-occurrence probability of A, B.

- Your program must load and process the input document on startup; You should do any expensive processing at this point.
- Your program will then receive word pairs on standard input. Output must be fast! K will be an integer with $1 <= k <= 5$.
- Output should be printed rounded to 2 decimal places.
- Case of words should be ignored, and all whitespace and punctuation stripped.
- You may assume all characters are ASCII, and that the filename provided is valid.
- If word A does not appear in the text, you should output 0.00.

Match the output of the examples below exactly. Do not print anything else, or your program may fail automated evaluation.

Performance Requirements:

- Initial processing of the input document should be O(NK) worst case time complexity, where N is the number of words in the input document.
- Results for each pair of words needs to be produced in O(1) worst case time complexity.
- Your solution needs to be both correct and meet the above performance requirements in order to pass!

Example 1:

$ java Cooccurrence i-like-programming.txt 3

```
programming like
programming evaluations
evaluations programming
```

Expected output

```
1.00
0.50
1.00
```

Example 2:

$ java Cooccurrence cat-in-the-hat.txt 3

```
hat cat
cat hat
sit sit
programming evaluations
```

Expected output

```
0.71
0.38
0.80
0.00
```