Pepperstone

Pepperstone Data Team Code Challenge

Thank you for completing this code challenge so we can get to know how you like to write your programs.

Deliverable

You can choose the language you want to use however it would be better if we can see your code in a language that is specific to the data engineer role. E.g. Python, Scala or Go.

Place the source code in a git repository. It should contain everything, including a README, we need to:

1. Run the tests;
2. Build the program;
3. Run the command.

The command should be:

```
./scrmabled-strings --dictionary [PATH TO DICTIONARY FILE] --input [PATH TO INPUT FILE]
```

Problem

This code challenge is based on this problem from Google Code Competitions. You can check the Analysis section for suggestions on how to solve this problem efficiently.

Count how many of the words from a dictionary appear as substrings in a long string of characters either in their original form or in their scrambled form. The scrambled form of the dictionary word must adhere to the following rule: the first and last letter must be maintained while the middle characters can be reorganised.

The scrambled or original form of the dictionary word may appear multiple times but we only count it once since we only need to know whether it shows up at least once.

For example, if we had the word **this** in the dictionary, the possible valid words which would be counted are **this** (original version) and **tihs** (scrambled version). **tsih**, **siht** and other variations are **not valid** since they do not start with **t** and end with **s**. Also, **tis**, **tiss**, and **thiss** are **not** scrambled forms, because they are not reorderings of the original set of letters.

Pepperstone

Input

Your input will consist of:

1. a dictionary file, where each line comprises one dictionary word from which you can create your dictionary. E.g. “and”, “bath”, etc, but note the dictionary words do not need to be real words.
2. an input file that contains a list of long strings, each on a newline, that you will need to use to search for your dictionary words. E.g. “btahand”

Output

Treating each line of the input file as one search string, your program should output a line *Case #x: y* per input file string, where *x* is the line number (starting from 1) and *y* is the number of words from the dictionary that appear (in their original or scrambled form) as substrings of the given string.

E.g.

Case #1: 2

Limits

- No two words in the dictionary are the same.
- Each word in the dictionary is between 2 and 105 letters long, inclusive.
- The sum of lengths of all words in the dictionary does not exceed 105.

Example

Note this sample only has one search line in the input, but multiple should be handled.

Dictionary File

```
axpaj
apxaj
dnrbt
pjxdn
abd
```

Input File (1 search line example)

```
aapxjdnrbtvldptfzbbdbbzxtndrvjblnzjfpvhdhhpxjdnrbt
```

Program Output

```
Case #1: 4
```

In the example above,

- `axpaj` occurs in its scrambled form as `aapxjdnrbtvldptfzbbdbbzxtndrvjblnzjfpvhdhhpxjdnrbt`.
- `apxaj` occurs in its scrambled form as `aapxjdnrbtvldptfzbbdbbzxtndrvjblnzjfpvhdhhpxjdnrbt`. Note that even though `apxaj` is the scrambled form of another dictionary word `axpaj`, both should be counted.
- `dnrbt` occurs twice in its original form as `aapxjdnrbtvldptfzbbdbbzxtndrvjblnzjfpvhdhhpxjdnrbt`, though it should be counted only once.
- `pjxdn` occurs in its scrambled form as `aapxjdnrbtvldptfzbbdbbzxtndrvjblnzjfpvhdhhpxjdnrbt`. Note this occurrence overlaps with occurrence of another dictionary word, but still they're counted independently.
- `abd` doesn't occur at all.

Stretch Goals

- Dockerfile that we can build and run the code
- Documentation for public interfaces
- Logging