[CS 11] Prac 9i – Frequency Analysis

Problem Statement

Zipf's law is an empirical law that often holds, approximately, when a list of measured values is sorted in decreasing order. It states that the value of the nth entry is inversely proportional to n:

For example, if one tallies the words in a typical passage of text, one often finds Zipf's law to approximately hold:

$$\text{word frequency} \propto \frac{1}{\text{word rank}}.$$

You would like to test this!

As a first step, you would like to be able to do the following: Given a sequence of words representing text, give the number of times each word has appeared previously in the text.

Task Details

Your task is to implement a function called `with_freq_count`. This function has a single parameter, a `str`.

The function must return a `list` of pairs. The first element of each pair must be the next word in the text (a `str`), and the second element must be an `int` denoting how many times it has appeared previously in the sequence.

Restrictions

(See 9a for more restrictions)

For this problem in particular:

- Recursion is allowed.
- The source code limit is 2000.

Example Calls

Example 1 Function Call

```
with_freq_count("""
don't stop me now
I'm having such a good time
I'm having a ball
don't stop me now
if you wanna have a good time
just give me a call
""")
```

Copy

Example 1 Return Value

```
[
    ("don't", 0),
    ('stop', 0),
    ('me', 0),
    ('now', 0),
    ("I'm", 0),
    ('having', 0),
    ('such', 0),
    ('a', 0),
    ('good', 0),
    ('time', 0),
    ("I'm", 1),
    ('having', 1),
    ('a', 1),
    ('ball', 0),
    ("don't", 1),
    ('stop', 1),
    ('me', 1),
    ('now', 1),
    ('if', 0),
    ('you', 0),
    ('wanna', 0),
    ('have', 0),
    ('a', 2),
    ('good', 1),
    ('time', 1),
    ('just', 0),
    ('give', 0),
    ('me', 2),
    ('a', 3),
    ('call', 0),
]
```

Copy

Example 2 Function Call

```
with_freq_count('a A a A')
```

Copy

Example 2 Return Value

```
[('a', 0), ('A', 0), ('a', 1), ('A', 1)]
```

Copy

Constraints

- The function `with_freq_count` will be called at most $60{,}000$ times.
- The number of words per call will be at most $200{,}000$.
- The total number of words across all calls will be at most $200{,}000$.
- Each word is a nonempty string of up to 6 English letters or the `-` or `'` symbols
- Words will be separated by spaces or newlines.
- The text will have a length of at most 8 times the number of words.

Scoring

- You get 45 ❤ points if you solve all test cases where:
  - The number of words per call is at most has length at most 50.
  - The total number of words across all calls is at most 500.
- You get 40 ❤ points if you solve all test cases where:
  - The number of words per call is at most has length at most $4{,}000$.
  - The total number of words across all calls is at most $8{,}000$.
- You get 60 ❤ points if you solve all test cases.

Clarifications

Report an issue

No clarifications have been made at this time.

Submit solution [CS 11] Practice 9 ❤

My submissions

✓ **Points:** 145 (partial)

Time limit: 4.0s

Memory limit: 1G

Author:
kvatienza (Kevin Atienza)

Problem type

Allowed languages
~~NONE~~, py3