Practical Regex: Dictionary Search

Eckel, TJHSST AI2, Spring 2022

Background & Explanation

One frequent use of regular expressions is to search a dictionary – in our case, a txt file containing a lot of words, one on each line – for words that match a particular pattern. It's easy to imagine applications here like a script to help you in solving crosswords or make Scrabble / Words With Friends moves.

For this assignment, you'll get the filename for a dictionary as a command line argument and then you'll need to output all the words matching certain conditions from that dictionary. So, your code will need to be written so it works for any given dictionary, even if that dictionary contains strings that aren't actual English words. For example, one question asks about words with the largest number of one consecutive character repeated over and over. English clearly won't have a word like "rrrrrrrrrrrrrrt" but the dict that your code gets might; don't make any assumptions about what could plausibly be in the words you're searching!

There are basically two different approaches here. You can:

- Read in the entire file as a single string with a bunch of \n characters in it and then regex search the entire string using line anchors (^ and $) to make sure you're matching full words. You have all the tools you need for this approach.
- Alternately, read the file into a list of individual strings of separate words stripped of \n characters and then run your regex on each word individually, checking to see if there is a match each time. This approach is just as good as the former, and definitely faster (fewer initial characters for the regex engine to begin a match check), but you might want to look up the `.match()` or `.fullmatch()` methods on compiled regular expression objects; searching for "python re documentation" should get you the Python 3 documentation page for the re library, then scroll down to "Regular Expression Objects".

NOTE: This assignment is adapted from the RE5 assignment on Dr. Gabor's grader. If you just want the challenge, you can submit to that assignment as well, though this assignment is different enough from his that your regexes won't necessarily be the same (especially if you take the second approach above). This is totally optional, though – you don't have to look at his RE5 assignment or submit anything to that assignment on his grader. I'll grade this one through the usual Dropbox type of submission.

Importantly, this means that **there is no length restriction on the regexes you use for this assignment**.

The next page lists all of the regexes you need to write; after that, there are two sample runs for you to use to check your work.

Assignment

You'll get a single command line argument specifying a file name – the file will contain a word list, with one word on each line. For each question below, use regex to find all the words that match the specified criteria, and then output your results as follows:

- A line with the question number and your regex
- A line with the total number of matches in the given dictionary
- The first (up to) five results, in alphabetical order, one per line (ONLY OUTPUT UP TO 5 RESULTS)
- A blank row

Like so:

```
#1 re.compile(YOUR REGEX GOES HERE)
16 total matches
word
word
word
word
word
(leave a blank space before the next question)
```

Some additional details:

- You should always ignore case. (And/or just .lowercase() the dictionary as it's being read.)
- Unlike Ghost, if the dictionary has words that contain non-alphanumeric characters like digits, that's ok. Leave them in. (You can see one in one of the sample runs below.)
- Vowels are [aeiou] unless otherwise specified.
- You may assume the dictionary you're reading is in alphabetical order to begin with.

Your challenges are:

1) Match all words of **minimum** length that contain each vowel at least once. (That is, from all of the words that contain each vowel at least once, find the shortest length and output the number of matching words of that length, ignoring the rest. Then print up to the first five of them, in alphabetical order. You don't need to use a regex to filter by length; you do need to use a regex to find the words in the first place.)
2) Match all words of **maximum** length that contain precisely 5 vowels. (Same additional instructions as #1.)
3) Match all words of **maximum** length where the first letter reappears as the last letter but does not appear anywhere else in the word. (Same additional instructions as #1.)
4) Match all words where the first three letters, reversed, are the last three letters. Overlapping is ok; for instance, "ada" and "abba" should both count.
5) Match all words where there is exactly one "b", exactly one "t", and they are adjacent to each other.
6) Match all words with the longest contiguous block of a single letter. (That is, figure out what the longest contiguous block of a single letter is, and print out all words with a block of that length. This may require more than one regex to find, or a loop modifying and re-searching with a series of regexes.)
7) Match all words with the greatest number of a repeated letter (which may occur anywhere in the word). Same additional instructions as #6.
8) Match all words with the greatest number of adjacent pairs of identical letters. Order matters; that is, "ab" is not the same pair as "ba". Same additional instructions as #6.
9) Match all words with the greatest number of consonants. Same additional instructions as #6.
10) Match all words of **maximum** length where no letter is repeated more than once. (That is, where no letter appears more than twice!)

Sample Runs

IMPORTANT CAVEAT: This assignment is brand new this year. I could definitely have made a mistake somewhere. If you think I did, let me know!

Using the words_all.txt file that you should have from Word Ladders Any Length or Ghost, this is the output you should get:

```
#1: re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
3 total matches
eulogia
miaoued
sequoia

#2: re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
787 total matches
abstractionisms
abstractionists
acceptingnesses
accommodatingly
accountantships

#3 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
83 total matches
centrosymmetric
chorioallantoic
chromatographic
cinematographic
contortionistic

#4 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
146 total matches
aba
abba
aga
aha
ala

#5 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
424 total matches
antbear
antbears
carpetbag
carpetbagged
carpetbagger

#6 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
2 total matches
brrr
zzz

#7 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
4 total matches
classlessnesses
possessednesses
senselessnesses
stresslessness
```

```
#8 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
1 total matches
restlessnesses

#9 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
8 total matches
bremsstrahlungs
cryptorchidisms
crystallography
plethysmographs
plethysmography

#10: re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
884 total matches
accommodatingly
accomplishments
accountableness
accountantships
achondroplastic
```

And, using the wordlist.txt file from the Crosswords assignment, here's your output. (Many of these are quite silly, as you'd expect from crossword puzzle answers! Take this as an example of some of the wacky things that may appear in dictionaries when you submit.)

```
#1: re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
1 total matches
aeiou

#2: re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
1 total matches
showsphotographsshows

#3: re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
3 total matches
needsadultsupervision
spaghettiandmeatballs
stillwetbehindtheears

#4 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
390 total matches
212
aaa
aaaa
aba
ababa

#5 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
576 total matches
abt
abtoner
accountbooks
actbig
adebt
```

```
#6 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
2 total matches
grrrrrrrrrrrrr
shhhhhhhhhhhhh

#7 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
2 total matches
grrrrrrrrrrrrr
shhhhhhhhhhhhh

#8 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
1 total matches
falalalalalalala

#9 re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
1 total matches
dotdotdotdashdashdashdotdotdot

#10: re.compile(YOUR REGEX GOES HERE, FLAGS HERE)
4 total matches
awolfinsheepsclothing
breakingcupishardtodo
dontgobreakingmyheart
noonehasratedthisfilm



Process finished with exit code 0
```

Specification

Submit a single Python script to the link on the course website.

This assignment is **complete** if:

- The “Name” field on the Dropbox submission form contains your **class period**, then your **last name**, then your **first name**, in that order.
- Your code operates exactly as specified above, with no additional print statements.