MSDS 7330 File Organization and Database Management Homework Anagram

Name:

This is a homework assignment for MSDS 7330, File Organization and Database Management. Your final submission shall be a single pdf document that includes this document plus your answers to each of the questions.

Collaboration is expected and encouraged; however, each student must hand in their own homework assignment. To the greatest extent possible, answers should not be copied but, instead, should be written in your own words. Copying answers from anywhere is plagiarism, this includes copying text directly from the textbook. Do not copy answers. Always use your own words. For each question list all persons with whom you collaborated and list all resources used in arriving at your answer. Resources include but are not limited to the textbook used for this course, papers read on the topic, and Google search results. Note that 'Google' is not a resource. Don't forget to place your name on the document.

This assignment was adapted from the requirements of Dr. Eric Larson.

Python

In this assignment you will be using Python to investigate the scrabble dictionary. All the words in the scrabble dictionary are available from http://www.puzzlers.org/pub/wordlists/ospd.txt

Using Python with Scrabble Word List

The text file contains a list of the words separated by newlines/carriage returns. Your first portion of the assignment is to find out how many unique anagrams are in the dictionary using Python. Note that this can be done efficiently in about 30 lines of code. However the method you use to store the words can wildly change the complexity of the problem. This can be completed procedurally or using object-oriented coding practices: either are fine implementations.

To find out what an anagram is, check out the Wikipedia page (http://en.wikipedia.org/wiki/Anagram). Essentially, anagrams are words with the same letters like 'ape' and 'pea'. This pair of words forms one anagram. As another example, 'ate', 'tea', and 'eat' form another anagram.

Turn in your Python code used to answer the following questions in addition to the answers to the questions.

Question 1 : How many unique anagrams are in the scrabble word list?
Collaborators:
Resources:

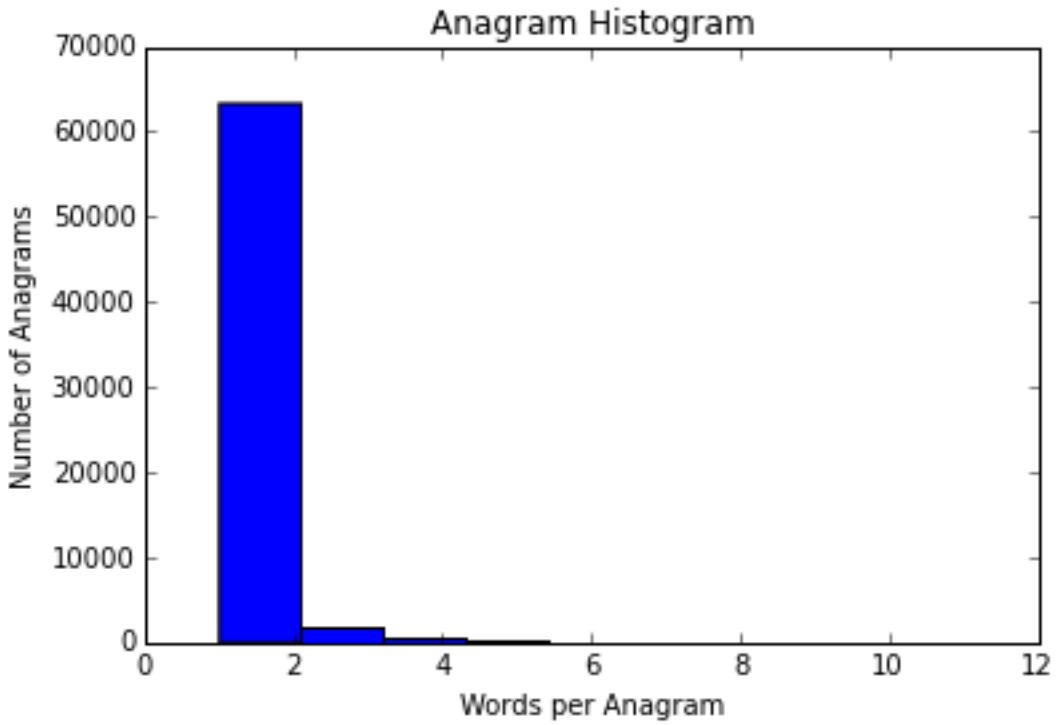
Python code and output screenshot appended to end of file. Output for this question is as follows:
QUESTION 1.) Number of unique anagrams in the Scrabble word list: 65783

Question 2 : What is the anagram with the largest number of words in the word list? How many words are in this anagram?
Collaborators:
Resources:

Python code and output screenshot appended to end of file. Output for this question is as follows:
QUESTION 2.) The anagram with the most instances is "aeprs" with 12 words.

Question 3 : Use matplotlib (or Excel) to visualize a histogram of the size of the anagrams. That is, make a bar chart where the X-axi from from X = 1 to X =max anagram size. Each bar will represent the number of anagrams of size X.
Collaborators:
Resources:

Python code and output screenshot appended to end of file. Output using matplotlib for this question is as follows:



Using Python with Databases and Scrabble Word List

Create a database using Python with one table. The table will have three columns: Unique Anagram identifier (i.e. the sorted letters in the anagram), number of words in the anagram, and the actual words as a comma separated value string.

Use Python to create and populate the database. You can use MySQL to create a bare bones database, but Python should create the table and setup the variable types. Alternatively, you can use sqlite3 that ships with Python.

Now create queries to the database that answer the following questions (which are the same as Questions 1 and 2). Turn in your Python code used to answer these questions with the database.

Question 4 : How many unique anagrams are in the scrabble word list?

Collaborators:

Resources:

Python code and output screenshot appended to end of file. Output for this question is as follows:

QUESTION 4.) There are 65783 unique anagrams in the Scrabble dictionary according to the database.

Question 5 : What is the anagram with the largest number of words in the word list? How many words are in this anagram?

Collaborators:

Resources:

Python code and output screenshot appended to end of file. Output for this question is as follows:

QUESTION 5.) The anagram with the most instances is "aeprs" with 12 words.

Using Python with Databases and Merriam-Webster English Language Dictionary

Use Python to create a new table in the database and again find anagrams in the word list using three columns. However, instead of using the small dataset of scrabble words, use the Merriam-Webster English language dictionary (approximately 300,000 words).

Now create queries to the database that answer the following questions (which are the same as Questions 1 and 2). Turn in your Python code used to answer these questions with the database.

Question 6 : How many unique anagrams are in the Merriam-Webster word list?

Collaborators:

Resources:

Python code and output screenshot appended to end of file. Output for this question is as follows:

QUESTION 6.) There are 315379 unique anagrams in the Merriam-Webster dictionary according to the database.

Question 7 : What is the anagram with the largest number of words in the word list? How many words are in this anagram?

Collaborators:

Resources:

Python code and output screenshot appended to end of file. Output for this question is as follows:

QUESTION 7.) The anagram with the most instances is "aelrst" with 15 words.

Contents of Python Script

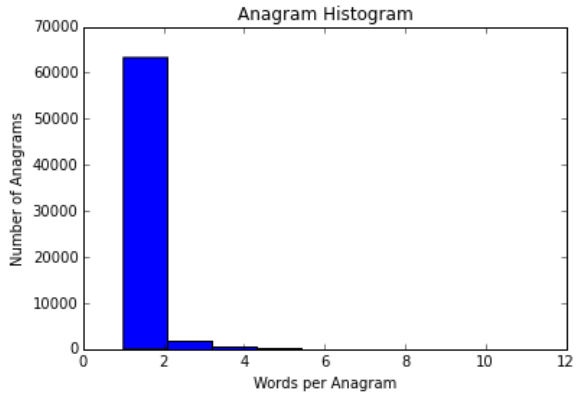
```
# Zach Brown
# MSDS 7330 403
# anagram.py
# 10 Nov 2015

# Import libraries
import matplotlib.pyplot as plt
import sqlite3

# Initialize anagram dictionary
anagrams = dict()

# Read in Scrabble dictionary txt file
with open("C:\\Users\\ubrowza\\Google Drive\\SMU\MSDS 7330 - File Organization and
Database Management\\Week 11\\ospd.txt") as file:
    scrabble = file.readlines()

# Populate anagram dictionary
for line in scrabble:
    # Sort word in alphabetical order and strip off newline character
    word = line[:-1]
    sorted_word = "".join(sorted(word))

    # If sorted_word is not already a key in the anagrams dictionary,
    # set the value to a set containing the word
    if sorted_word not in anagrams:
        anagrams[sorted_word] = {word}

    # Else add the word to the set
    else:
        anagrams[sorted_word].add(word)

# Print the number of anagrams
print "QUESTION 1.) Number of unique anagrams in the Scrabble word list: " +
str(len(anagrams))

# Determine anagram with the most instances
most_words = max(anagrams, key=lambda k: len(anagrams[k]))

# Determine number of instances for the most used anagram
most_words_count = len(anagrams[most_words])

# Print the anagram with the most instances
print "QUESTION 2.) The anagram with the most instances is \"%s\" with %s words." %
(most_words, most_words_count)

# Create doctionary of word counts for all anagrams
counts = dict()
for key in anagrams:
    counts[key] = (len(anagrams[key]))

# Plot a histogram of distribution of anagram instances
print "QUESTION 3.) "
plt.hist(counts.values())
plt.title("Anagram Histogram")
plt.xlabel("Words per Anagram")
plt.ylabel("Number of Anagrams")
plt.show()
```

```
# Initialize database connection and cursor
conn = sqlite3.connect("C:\\Users\\ubrowza\\Google Drive\\SMU\\MSDS 7330 - File
Organization and Database Management\\Week 11\\anagram.db")
c = conn.cursor()

# Create database table
c.execute("DROP TABLE scrabble")
c.execute("""CREATE TABLE scrabble
            (anagram TEXT PRIMARY KEY, word_count INTEGER, words TEXT)""")

# Create list of tuples to insert into scrabble table
anagram_list = []
for key in anagrams:
    anagram_list.append((key, str(len(anagrams[key])), ','.join(anagrams[key])))

# Populate database table
c.executemany('insert into scrabble values (?,?,?)', anagram_list)

# Commit changes to database
conn.commit()

# Query the number of unique anagrams from the scrabble database table
c.execute("SELECT COUNT(*) FROM scrabble")
anagram_count = c.fetchone()[0]

# Print the number of unique anagrams
print "QUESTION 4.) There are %s unique anagrams in the Scrabble dictionary according
to the database." % anagram_count

# Run a query to find the anagram with the largest number of words in the database
c.execute("""SELECT anagram, word_count
            FROM scrabble
            WHERE word_count = (SELECT max(word_count) FROM scrabble)""")
row = c.fetchone()
max_words = row[0]
max_word_count = row[1]

# Print the anagram with the most words and the number of words it can make
print "QUESTION 5.) The anagram with the most instances is \"%s\" with %s words." %
(max_words, max_word_count)

# Close the database connection
conn.close()

# Initialize anagram dictionary for the Merriam-Webster word list
mw_anagrams = dict()

# Read in Merriam-Webster txt file
with open("C:\\Users\\ubrowza\\Google Drive\\SMU\\MSDS 7330 - File Organization and
Database Management\\Week 11\\words.txt") as file:
    mw = file.readlines()

# Populate anagram dictionary
for line in mw:
    # Sort word in alphabetical order and strip off newline character
    word = line[:-1]
    sorted_word = "".join(sorted(word))

    # If sorted_word is not already a key in the mw_anagrams dictionary,
```

```
    # set the value to a set containing the word
    if sorted_word not in mw_anagrams:
        mw_anagrams[sorted_word] = {word}

    # Else add the word to the set
    else:
        mw_anagrams[sorted_word].add(word)

# Create doctionary of word counts for all Merriam-Webster anagrams
mw_counts = dict()
for key in mw_anagrams:
    mw_counts[key] = (len(mw_anagrams[key]))

# Initialize database connection and cursor
conn = sqlite3.connect("C:\\Users\\ubrowza\\Google Drive\\SMU\\MSDS 7330 - File
Organization and Database Management\\Week 11\\anagram.db")
c = conn.cursor()

# Create database table
c.execute("DROP TABLE mw")
c.execute("""CREATE TABLE mw
            (anagram TEXT PRIMARY KEY, word_count INTEGER, words TEXT)""")

# Create list of tuples to insert into mw table
mw_anagram_list = []
for key in mw_anagrams:
    mw_anagram_list.append((key, str(len(mw_anagrams[key])),
','.join(mw_anagrams[key])))

# Populate database table
c.executemany('insert into mw values (?,?,?)', mw_anagram_list)

# Commit changes to database
conn.commit()

# Query the number of unique anagrams from the Merriam-Webster database table
c.execute("SELECT COUNT(*) FROM mw")
mw_anagram_count = c.fetchone()[0]

# Print the number of unique anagrams
print "QUESTION 6.) There are %s unique anagrams in the Merriam-Webster dictionary
according to the database." % mw_anagram_count

# Run a query to find the anagram with the largest number of words in the database
c.execute("""SELECT anagram, word_count
            FROM mw
            WHERE word_count = (SELECT max(word_count) FROM mw)""")
row = c.fetchone()
mw_max_words = row[0]
mw_max_word_count = row[1]

# Print the anagram with the most words and the number of words it can make
print "QUESTION 7.) The anagram with the most instances is \"%s\" with %s words." %
(mw_max_words, mw_max_word_count)

# Close the database connection
conn.close()
```

Code Output

```
IPython console
Console 1/A
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.
%guiref   -> A brief reference about the graphical user interface.

In [1]: runfile('C:/Users/ubrowza/Google Drive/SMU/MSDS 7330 - File Organization and Database Management/Week 11/anagram.py',
wdir='C:/Users/ubrowza/Google Drive/SMU/MSDS 7330 - File Organization and Database Management/Week 11')
QUESTION 1.) Number of unique anagrams in the Scrabble word list: 65783
QUESTION 2.) The anagram with the most instances is "aeprs" with 12 words.
QUESTION 3.)
```




```
QUESTION 4.) There are 65783 unique anagrams in the Scrabble dictionary according to the database.
QUESTION 5.) The anagram with the most instances is "aeprs" with 12 words.
QUESTION 6.) There are 315379 unique anagrams in the Merriam-Webster dictionary according to the database.
QUESTION 7.) The anagram with the most instances is "aelrst" with 15 words.

In [2]:
Console | History log | IPython console
Permissions: RW | End-of-lines: CRLF | Encoding: ASCII | Line: 50 | Column: 22 | Memory: 86 %
```