Lab 07: Dictionaries and Sets

Goals for this lab:

1. Use `Dictionaries` to quickly store and access data
2. Use `Sets` to store and access data
3. Determine the term frequency of words in a file
4. Prevent stopwords from being included in the term frequency calculation
5. Use pickling to store objects in a file for later use
6. Use pickling to restore objects from a file
7. Write and execute simple Python programs using the given instructions

General Guidelines:

- Use meaningful variable names
- Use appropriate indentation
- Use comments, especially for the header, variables, and blocks of code.

Example Header:

```
# Author: Your name
# Date: Today's date
# Description: A small description in your own words that describes what
the program does. Any additional information required for the program to
execute should also be provided.
```

Example Function Docstring:

```
"""
General function description
:param p1: DESCRIPTION
:type p1: TYPE
:param p2: DESCRIPTION
:type p2: TYPE
:return: DESCRIPTION.
"""
```

1. Preparation

You will need two input files for this lab, so be sure to download **lab07Files.zip** from the Canvas Lab07 link before you begin the lab.

2. Term Frequency

The area of natural language processing is typically an intersection between linguistics, computer science, and artificial intelligence. Algorithms and software are developed in order to allow the computer to infer meaning and make decisions based on human languages. This could be as complex as teaching a computer to carry on a reasonable conversation with a human, or as "simple" as trying to determine if people liked a movie based on their social media posts. In both cases, sentences must be broken down into individual words or phrases so that the computer can process them. One of the first steps is often term frequency analyses, where the frequency of words are counted, so that the most important or significant words can be determined.

For this exercise you will be writing a program to count the number of words in some text. The input file will contain multiple lines, and punctuation that is not part of a word has already been removed, however not all of the words are completely lowercase. You will need to break up each line into individual words, and maintain a count of how many times each word appears in the overall text using a `Dictionary`. Finally, you will need to output the list of words and their frequencies to the user.

To complete this program:

- In a file named **freqCountV1.py**, perform the text analysis by creating the following functionality:
    - Define a function named `readTextFile`, which will take in a `Dictionary` as a parameter, will return nothing, and should:
        - Prompt the user for the name of the file containing the text
        - Check if the file exists using the appropriate functions and if it does not, repeatedly prompt the user for a new filename until the provide the name of a file that does exist
        - Read in the file one line at a time and for each line
            - Remove any extra whitespace or newlines from the line
            - Set all of the characters to lowercase
            - Split the line up into individual words and for each word
                - If the word exists in the `Dictionary` then increase its count by 1
                - If the word does not exist in the `Dictionary` set its initial count to 1
        - Close the file
    - Define a function named `outputFreq`, which will take in a `Dictionary` as a parameter, will return nothing, and should:
        - Output the number of unique words
        - Output each word and its associated count from the `Dictionary`
    - Define a `main` function, which should:
        - Create an empty `Dictionary` that will index word counts using Strings
        - Call `readTextFile()` and pass it your `Dictionary`
        - Call `outputFreq()` and pass it your `Dictionary`

Make sure you save **freqCountV1.py** as you will be submitting this file for Lab 07.

3. Stopwords

One of the initial challenges facing proper term frequency analysis is the occurrence of stopwords, or words that provide little to no information regarding the text. These are typically common words such as “the”, “a”, “and”, “have”, or single letters. Thus, to focus on the more important words, we want to perform stopword removal (or at least prevent their inclusion) in our analysis.

For this exercise you will be improving upon your previous code from part 1 of this lab. You will perform all of the same steps from last time, but before you read in the text file, you will need to read in a list of stopwords and store them in a `Set`. Then, as you are reading in your text file, use the `Set` to prevent the adding of stopwords to the `Dictionary` containing your term frequencies.

To complete this program:

- In a file named **freqCountV2.py**, perform the text analysis by adding the following functionality to your previous version of the program:
    - Define a function named `readStopWordsFile`, which will take in a `Set` as a parameter, will return nothing, and should:
        - Prompt the user for the name of the file containing the stopwords
        - Check if the file exists using the appropriate functions and if it does not, repeatedly prompt the user for a new filename until the provide the name of a file that does exist
        - Read in the file one line at a time and for each line
            - Remove any extra whitespace or newlines from the line
            - Add the line to your `Set`
        - Close the file
    - Update your `readTextFile` function such that:
        - It also takes in your stopwords `Set` as a parameter
        - Before a word is added to your `Dictionary` your program should check if the word is in your `Set`
            - If the word is not in the `Set`, add it as normal to your `Dictionary`
            - If the word is in the `Set`, do not add it to your `Dictionary`
    - Update `main` function such that:
        - It creates an empty `Set` to store stopwords
        - It calls `readStopWordsFile` and you pass it your `Set`
        - It calls `readTextFile` and you pass it your `Dictionary` and `Set`

Make sure you save **freqCountV2.py** as you will be submitting this file for Lab 07.

4. Repository

As you may have guessed, our current **stopwords.txt** file is not a complete listing of all possible stopwords in the English language. We could expand upon it by manually combining it with other stopwords lists, but this could introduce duplicates into our list which would waste space. To solve this, we are going to create a program that acts as a repository for stopwords. Then, we can add many different lists to the repository and have it store the words and automatically remove duplicates. However, our repository should keep continual track of all of the stopwords it knows, but rather than manually writing each of these words out to a file we can use pickling to dump and load our set of stopwords automatically each time the program starts and finishes.

To complete this program create a file named **stopwordRepo.py**, which should contain**:**

- A global constant to store the name of the file we will use for pickling, **stopwordset.data**
- A function named `readStopWordsFile` that takes in a `Set` of stopwords, return nothing, and should:
    - Prompt the user for the name of the file to be read in
    - Check if the file exists, and if not, repeatedly prompt the user for the name of the file
    - Read in the file one line at a time and for each line
        - Remove any extra whitespace or newlines from the line
        - If the word is already in the `Set`, report it to the user
        - Otherwise add the word to the `Set`
    - Close the file
- A function named `writeStopWordsFile` that takes in a `Set` of stopwords, return nothing, and should:
    - Prompt the user for the name of the file to be written to
    - Writes each word in the `Set` to the file on a separate line
    - Close the file
- A function named `displayStopWords` that takes in a `Set` of stopwords, return nothing, and should:
    - State the number of stopwords currently in the `Set`
    - Output each word to the screen on a separate line
- A function named `storeStopWords` that takes in a `Set` of stopwords, return nothing, and should:
    - Using the global constant, open a file for binary writing
    - Dump the `Set` to the file
    - Close the file
- A function named `restoreStopWords` that return a `Set` of stop words, and should:
    - Using the global constant, open a file for binary reading
    - Load the `Set` in the file to a variable
    - Close the file
    - Return the variable containing the `Set` of stopwords

- A `main` function which should:
    - Using the global constant, check if stopwords data file exists
        - If it does, call the `restoreStopWords` function and store the results in a variable
        - Otherwise, create an empty `Set` and store it in a variable
    - Welcome the user and provide them with a list of options: Add a new list of stopwords; write the current set of stopwords to a file; display the current set of stopwords; quit.
    - While the user wishes to do anything other than quit, call the appropriate functions
    - Once the user has decided to quit, call the `storeStopWords` function and provide it with the `Set` of stopwords

Make sure you save **stopwordRepo.py** as you will be submitting this file for Lab 07.

Sample Output

freqCountV1.py:

```
Please enter the name of the file to analyze:backgammon
backgammon does not exist! Please enter the name of the file to
analyze:backgammon.txt
The file contained 110 unique words.
backgammon:5
is:5
one:2
of:12
the:16
oldest:2
known:1
board:3
games:3
its:1
history:1
can:1
be:2
traced:1
back:1
nearly:1
5000:1
years:1
to:6
archaeological:1
discoveries:1
in:1
mesopotamia:1
it:1
a:6
two-player:1
game:4
where:1
each:2
player:2
has:3
```

fifteen:2
pieces:1
that:2
move:2
between:1
twenty-four:1
triangles:1
according:1
roll:2
two:1
dice:3
objective:1
first:1
bear:1
off:2
or:1
all:1
checkers:2
member:1
tables:1
family:1
classes:1
involves:1
combination:1
strategy:1
and:2
luck:1
while:1
may:1
determine:1
outcome:1
single:1
better:2
will:1
accumulate:1
record:1
over:1
series:1
many:1
with:2
players:3
must:1
choose:1
from:1
numerous:1
options:1
for:1
moving:1
their:1
anticipate:1
possible:1
counter-moves:1

```
by:2
opponent:1
optional:1
use:1
doubling:1
cube:1
allows:1
raise:1
stakes:1
during:1
like:1
chess:1
been:2
studied:1
great:1
interest:1
computer:1
scientists:1
owing:1
this:1
research:1
software:1
developed:1
capable:1
beating:1
world-class:1
human:1
```

freqCountV2.py:

```
Please enter the name of the stopwords file:stopwords
stopwords does not exist! Please enter the name of the stopwords
file:stopwords.txt
Please enter the name of the file to analyze:backgammon.txt
The file contained 82 unique words.
backgammon:5
one:2
oldest:2
known:1
board:3
games:3
history:1
can:1
traced:1
back:1
nearly:1
5000:1
years:1
archaeological:1
discoveries:1
mesopotamia:1
two-player:1
```

game:4
player:2
fifteen:2
pieces:1
move:2
twenty-four:1
triangles:1
according:1
roll:2
two:1
dice:3
objective:1
first:1
bear:1
checkers:2
member:1
tables:1
family:1
classes:1
involves:1
combination:1
strategy:1
luck:1
may:1
determine:1
outcome:1
single:1
better:2
will:1
accumulate:1
record:1
series:1
many:1
players:3
must:1
choose:1
numerous:1
options:1
moving:1
anticipate:1
possible:1
counter-moves:1
opponent:1
optional:1
use:1
doubling:1
cube:1
allows:1
raise:1
stakes:1
like:1
chess:1

```
studied:1
great:1
interest:1
computer:1
scientists:1
owing:1
research:1
software:1
developed:1
capable:1
beating:1
world-class:1
human:1
```

stopwordRepo.py

```
Welcome to the stopword repository!
    Please choose from the following options:
    1. Add a new list of stopwords
    2. Write current set of stopwords to a file
    3. Display current set of stopwords
    4. Quit

Please enter your choice: 1
Please enter the name of the new stopwords file: lettersNumbers.txt
Please choose from the following options:
        1. Add a new list of stopwords
        2. Write current set of stopwords to a file
        3. Display current set of stopwords
        4. Quit
Please enter your choice: 3
Currently we have 36 stopwords:
0
m
s
j
c
d
t
l
7
2
9
8
p
b
h
r
e
g
n
x
```

```
i
3
6
f
y
u
v
o
w
a
k
z
1
q
4
5
Please choose from the following options:
        1. Add a new list of stopwords
        2. Write current set of stopwords to a file
        3. Display current set of stopwords
        4. Quit
Please enter your choice: 4
Thank you for using the stopword repository!

Welcome to the stopword repository!
    Please choose from the following options:
    1. Add a new list of stopwords
    2. Write current set of stopwords to a file
    3. Display current set of stopwords
    4. Quit

Please enter your choice: 1
Please enter the name of the new stopwords file: shortList.txt
We already have the word: i
We already have the word: a
We already have the word: s
We already have the word: t
Please choose from the following options:
        1. Add a new list of stopwords
        2. Write current set of stopwords to a file
        3. Display current set of stopwords
        4. Quit
Please enter your choice: 2
Please enter the name of the stopwords file to write to: newstopwords.txt
Please choose from the following options:
        1. Add a new list of stopwords
        2. Write current set of stopwords to a file
        3. Display current set of stopwords
        4. Quit
Please enter your choice: 3
Currently we have 159 stopwords:
theirs
```

am
will
j
a
both
s
he
from
while
it
its
f
further
our
can
been
she
themselves
these
for
such
v
yours
9
them
out
d
b
ours
her
why
no
more
into
herself
than
this
some
h
we
8
an
now
were
him
do
6
had
very
c
off
your

against
0
about
have
has
u
once
so
who
his
q
that
myself
i
until
o
n
only
because
yourself
are
here
4
yourselves
of
me
should
between
under
the
is
y
down
too
ourselves
what
just
my
not
nor
l
m
other
there
5
any
most
2
after
was
1
above

their
whom
when
which
itself
or
few
before
same
w
k
g
having
those
himself
in
t
over
z
doing
they
again
each
7
during
up
and
by
p
through
how
does
did
all
be
don
with
3
below
you
then
where
hers
as
being
but
x
e
if
to
on
r

```
at
own
Please choose from the following options:
        1. Add a new list of stopwords
        2. Write current set of stopwords to a file
        3. Display current set of stopwords
        4. Quit
Please enter your choice: 4
Thank you for using the stopword repository!

Process finished with exit code 0
```

Submission

Once you have completed this lab it is time to turn in your work. Please submit the following files to the Lab 07 assignment in Canvas:

- **freqCountV1.py**
- **freqCountV2.py**
- **stopwordRepo.py**

Rubric

For each program in this lab, you are expected to make a good faith effort on all requested functionality. Each submitted program will receive a score of either 100, 75, 50, or 0 out of 100 based upon how much of the program you attempted, regardless of whether the final output is correct (See table below). The scores for all of your submitted programs for this lab will be averaged together to calculate your grade for this lab. Keep in mind that labs are practice both for the exams in this course and for coding professionally, so make sure you take the assignments seriously.

Score	Attempted Functionality
100	100% of the functionality was attempted
75	<100% and >=75% of the functionality was attempted
50	<75% and >=50% of the functionality was attempted
0	<50% of the functionality was attempted