CARNEGIE MELLON UNIVERSITY

10601a: Homework #2 - "hello, autolab"

TAs-in-charge:
Joe Runde (jrunde@andrew)
Yifan Xue (yifanxue@andrew)

Assigned: Friday, 15 January 2016.
Due: 11:59:59pm on Sunday, 24 January 2015.
Late Penalty: 20% per day.

Policy on Collaboration among Students

The purpose of student collaboration is to facilitate learning, not to circumvent it. Studying the material in groups is strongly encouraged. It is also allowed to seek help from other students in understanding the material needed to solve a particular homework problem, provided no written notes are shared, or are taken at that time, and provided learning is facilitated, not circumvented. **The actual solution must be done by each student alone**.
The purpose of programming assignments in this course is to make sure you truly understand the relevant techniques. **All code must be written by each student alone**. We will strictly enforce this policy, by carefully inspecting your code using sophisticated detection techniques. You have been warned!
The presence or absence of any form of help or collaboration, whether given or received, must be explicitly stated and disclosed in full by all involved. Specifically, **each assignment must contain a file named collaboration.txt where you will answer the following questions**:

- Did you receive any help whatsoever from anyone in solving this assignment? Yes / No. If you answered 'yes', give full details? (e.g."Jane explained to me what is asked in Question 3.4").
- Did you give any help whatsoever to anyone in solving this assignment? Yes / No. If you answered 'yes', give full details? (e.g. "I pointed Joe to section 2.3 to help him with Question 2").
- Did you find or come across code that implements any part of this assignment ? Yes / No. If you answered 'yes', give full details? (e.g. book & page, URL & location within the page, etc)

If you gave help after turning in your own assignment and/or after answering the questions above, you must update your answers before the assignment's deadline, if necessary by emailing the TA in charge of the assignment.

You are encouraged to read books and other instructional materials, both online and offline, to help you understand the concepts and algorithms taught in class. These materials may contain example code or pseudo code, which may help you better understand an algorithm or an implementation detail. However, when you implement your own solution to an assignment, you must put all materials aside, and write your code **completely on your own, starting "from scratch"**. Specifically, you may not use any code you found or came across. If you find or come across code that implements any part of your assignment, you must disclose this fact in your collaboration statement.
Collaboration without full disclosure will be handled severely, in compliance with CMU's Policy on Cheating and Plagiarism. All violations (even first one) of course policies will always be reported to the university authorities, will carry severe penalties, usually failure in the course, and can even lead to dismissal from the university. You have been warned!
Some of the homework assignments used in this class may have been used in prior versions of this class, or in classes at other institutions, or elsewhere. Avoiding the use of heavily tested assignments will detract from the main purpose of these assignments, which is to reinforce the material and stimulate thinking. Because some of these assignments may have been used before, solutions to them may be, or may have been, available online, or from other people or sources. It is explicitly forbidden to use any such sources, or to consult people who have solved these problems before. It is explicitly forbidden to search for these problems or their solutions on the internet. You must solve the homework assignments completely on your own. For programming assignments, this means you must write your programs completely by yourself, and not use any code from any source whatsoever. I will be actively monitoring your compliance, and any violation will be dealt with harshly. Collaboration with other students who are currently taking the class is allowed, but only under the conditions stated above.

General Instructions

The goal of this assignment is to introduce you to Java or Python as powerful tools for processing text and to introduce you to the autolab environment for homework submission. We've described functions that we want you to implement. Be sure to test your code before submitting.

In this homework you have to choose Python or Java as your programming language, and you must respect that choice for all questions in this homework. The autolab environment has **Python 2.6** installed—check your code runs on linux.andrew.cmu.edu.

0 INTRODUCTION

Bag of Words (BoW) is a widely used method in machine learning and natural language processing for representing text information. The idea is to represent a given document by the count of the frequency of each word in the document—a so-called "bag of words."

We illustrate with an example. Consider the following three documents:

*Document*1
come on you painter, you piper, you prisoner, and shine

Document2
The piper at the gates of dawn

Document3
Then the piper will lead us to reason.

There are 19 unique words that appear in the documents:

```
come, on, you, painter, piper, prisoner, and, shine, at, the, gates, of, dawn,
then, will, lead, us, to, reason
```

These 19 words are our vocabulary. Our BoW representation will be a 19-dimensional vector, one for each document. Every dimension of this vector represents the term frequency of a word in the vocabulary.

Thus, the BoW representation of the three documents will be:

	come	on	you	painter	piper	prisoner	and	shine	at	the	gates	of	dawn	then	will	lead	us	to	reason
document1	1	1	3	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
document2	0	0	0	0	1	0	0	0	1	2	1	1	1	0	0	0	0	0	0
document3	0	0	0	0	1	0	0	0	0	1	0	0	0	1	1	1	1	1	1

Now, instead of raw text, each document is represented by a feature vector. In this assignment you will build a simplified version of a BoW.

1 Splitting Text

Write a program that takes the location path of a text file as its argument and generates an output with a list of **unique, lower case words, sorted in reverse alphabetical order**. You should only split the input text on **spaces (' ')**. Don't worry about tabs, line breaks or any kind of punctuation marks. The input will be a very simple sentence.

For example, if the input text file contains the following sentence

```
The quick brown fox jumps over the lazy dog
```

your output should be

```
the,quick,over,lazy,jumps,fox,dog,brown
```

Your output is **not** supposed to contain a line break ('\n') at the end. Please observe that there are **no spaces** between words on the output.

You have to output your answer to the standard output (stdout).

We will run your code as below:

For Python:

```
python question1.py "path/to/input"
```

For Java:

```
javac question1.java
java question1 "path/to/input"
```

Make sure your program is called question1.py or question1.java.

2 COUNTING WORDS

Write a program that takes the location path of a text file as argument, splits the text as in question 1, and counts the number of occurrences of each word. The words must also be displayed in reverse alphabetical order.

For example, if the input text file contains the following text:

```
The quick brown fox jumps over the lazy dog
```

your output should be

```
the:2,quick:1,over:1,lazy:1,jumps:1,fox:1,dog:1,brown:1
```

Your output is **not** supposed to contain a line break ('\n') at the end. Please observe that there are no spaces between words or numbers on the output.

You have to output your answer to the standard output (stdout).

We will run your code as below:

For Python:

```
python question2.py "path/to/input"
```

For Java:

```
javac question2.java
java question2 "path/to/input"
```

Make sure your program is called question2.py or question2.java.

3 REMOVING STOP WORDS

Alter your program from question 2 so that it now removes stop words from a list provided. Stop words are words such as “the” or “at” that carry little meaning—they are usually removed during text processing for learning tasks.

For example, if the input text file contains the following text

```
The quick brown fox jumps over the lazy dog
```

and the stop words list is

```
as
the
in
over
on
```

your output should be

```
quick:1,lazy:1,jumps:1,fox:1,dog:1,brown:1
```

The stopwords file will be in the format that each line contains one and only one stop word.

Your output file is **not** supposed to contain a line break (‘\n’) at the end. Please observe that there are no spaces between words or numbers on the output.

You have to output your answer to the standard output (stdout).

We will run your code as below:

For Python:

```
python question3.py "path/to/input/file" "path/to/stopwords/file"
```

For Java:

```
javac question3.java
java question3 "path/to/input/file" "path/to/stopwords/file"
```

Make sure your program is called question3.py or question3.java.

4 AUTOLAB SUBMISSION

For Java:

You must submit a .tgz file named **hw2.tgz** containing all your files (the three .java files + collaboration.txt).
autolab is case sensitive, so observe that all your files should be named in **lowercase**.
You can create that file by running " *tar -cvf hw2.tgz *.java collaboration.txt"*.
DO NOT put the above files in a folder and then tar gzip the folder.
You must submit this file to the **"homework2java"** link on Autolab:
https://autolab.andrew.cmu.edu/courses/10601a-s16/assessments

For Python:

You must submit a .tgz file named **hw2.tgz** containing all your files (the three .py files + collaboration.txt).
autolab is case sensitive, so observe that all your files should be named in **lowercase**.
You can create that file by running " *tar -cvf hw2.tgz *.py collaboration.txt"*.
DO NOT put the above files in a folder and then tar gzip the folder.
You must submit this file to the **"homework2python"** link on Autolab:
https://autolab.andrew.cmu.edu/courses/10601a-s16/assessments