

**COMPSCI 121 / IN4MATX 141**  
**Assignment 1 - Text Processing Functions**  
**Due date: 4/18 at 11:59pm**

This assignment is to be done individually. You cannot use code written by your classmates. Use code found over the Internet at your own peril - it may not do exactly what the assignment requests. If you do end up using code you find on the Internet, you must disclose the origin of the code. **Concealing the origin of a piece of code is plagiarism.** Use the discussion section on Piazza (Assignment #1 on [https://piazza.com/uc\\_irvine/spring2018/compsci121in4matx141](https://piazza.com/uc_irvine/spring2018/compsci121in4matx141)) for general questions whose answers can benefit you and everyone.

### General Specifications

1. Programming language: You are free to write the program in any programming language of your choice. However, the next homework will work with a web-crawler written in Python 2.7, so we encourage you to use this homework to brush up your knowledge of Python.
2. Make sure to break down your program into classes/methods/functions corresponding to the parts in this specification. They will be tested separately.
3. The function signatures in this specification are informal; their purpose is to explain the inputs and outputs of the methods.
4. At points, the assignment may be underspecified. In those cases, make your own assumptions and be prepared to defend them.

### Part A: Word Frequencies (40 points)

Write a program that takes a text file as a command line argument and outputs the word frequency in decreasing order.

Briefly, implement the following functionality:

1. Read an input file passed as a *command line argument*. Please ensure you clearly understand what command line arguments are.
2. Tokenize the text of the file. A token is a sequence of alphanumeric characters, independent of capitalization (so “Apple” and “apple” are the same token)
3. Count the number of occurrences of each word in the tokens generated.
4. Print out the word frequency counts onto the screen. *The print out should be ordered by decreasing frequency. (so, highest frequency words first). Resolve ties alphabetically.*

Here is an example of an input file and corresponding output after computing the word frequency:

Input:

Here's a fun-fact! White tigers live mostly in "India".  
Wild lions live mostly in "Africa".

Output (Tokens and their counts):

in - 2  
live - 2  
mostly - 2  
a - 1  
africa - 1  
fact - 1  
fun - 1  
here - 1  
india - 1  
lions - 1  
s - 1  
tigers - 1  
white - 1  
wild - 1

NOTE: Tokens should only contain alphanumeric characters (alphabets and numbers). Any other characters such as punctuation marks, symbols, etc. should be eliminated.

The TA will use test text file(s) during the evaluation of your assignment. For this part, it is expected that your program will read this text file, tokenize its contents, count the tokens, and print out the word (token) frequencies in the order specified above.

**Part B: Intersection of two files (60 points)**

Write a program that reads the contents of two text files and outputs the *number of tokens they have in common*. Here is an example of input/output:

Input file 1:

We reviewed **the trip** and credited **the cancellation fee**. **The driver** **has** been notified.

Input file 2:

If a **trip** is cancelled more than 5 minutes after **the driver**-partner **has** confirmed **the** request, a **cancellation fee** will apply.

Output:

(optionally print out the common words, then...)

**6**

If required, you can reuse code you wrote for part A.

The TA will use test text file(s) during the evaluation of your assignment. Note that some of the text files may be VERY LARGE.

Once you have implemented your program, please perform an analysis of its runtime complexity: does it run in linear time relative to the size of the input? Polynomial time? Exponential time?

For this part, programs that perform better will be given more credit than those that perform poorly. It will help the TA if you specify the run-time complexity of your program in the form of comments in your code.

## Submitting Your Assignment

Your submission should be a single zip file containing two programs, one for part A, the other for part B. Submit it to Canvas.

## Grading Process

You **may be** required to meet with a TA for about 10 minutes during the two weeks after submission. During that meeting, be ready for the following:

1. Show your program running on your own computer, on an input file provided by the grader on a USB stick
2. Answer questions about your program (as submitted) and its behavior during the meeting

You should **not** continue to work on your program once you submit it to Canvas, as the TA will be looking at your submitted code.

## Evaluation Criteria

Your assignment will be graded on the following four criteria.

1. Correctness (40%)
  - a) How well does the behavior of the program match the specification?
  - b) How does your program handle bad input?
2. Understanding/structure/comments in the code (30%)
  - a) Do you demonstrate understanding of the code submitted?
3. Efficiency (30%)
  - a) How quickly does the program work on large inputs?