**Homework #1: Basic Text Pre-processing**

**1. Overview**

Your task in this assignment is to write a program that accepts as input a plain text document and extract some properties of the content.  Specifically, your program must output:

1. Number of paragraphs. (Splitted by \n)
2. Number of sentences.
3. Number of words (i.e., "tokens").
4. Number of distinct words (i.e., "word types").
5. List of word frequency counts.  Words are ordered by frequency (in the descending order), and words which have the same frequency count are ordered by lexicographical order (in the ascending order).
6. Remove the stopwords (i.e., words that are frequent but do not contribute much to the meaning of a sentence.) A list of stopwords is provided for English.
7. How would what you did be different if you did it for another language (e.g. Greek or French). List as many changes in your approach as you can. Do you think there can be a universal methodology for identifying and counting words / sentences / paragraphs, or is it language specific?

The purpose of this assignment is that you **write the code from scratch**, without using any library function (except for those that are fundamentally necessary such as those for file I/O or collection classes).

**Exception:** You can use library in **sentence segmentation**.  In general, a document can be formatted in a variety of ways, especially regarding line breaks.  To save time, you are allowed to use a library function (though only for sentence segmentation).

* In Python, you may use **sent\_tokenize()**in [Natural Language Toolkit (NLTK)](http://www.nltk.org/) toolkit.  But do not use other NLTK functions (such as word\_tokenize()) or external libraries/packages.

Try to implement all parts by yourselves and experiment if you have time on how to make your code efficient: e.g., in memory, or in processing time. For example, read the text just one time or write fast code by simplifying some operations.

Other specifics:

1. Do **not** convert the text into lower/upper case or apply stemming (if you know what it is).  Use the text as is.
2. Words must be tokenized such that:
   * **Punctuations** are separated.  Although this is not truly correct, in this assignment you may assume punctuations are the same as all *non-alphanumeric characters* (i.e., neither alphabet/letter or digit).  Also:
     + You only separate **leading and trailing** punctuations.  If a punctuation occurs within a word, you do not separate the word.  For example, "$3.19" should be separated into "$" and "3.19".
     + Multiple punctuations should be separated individually.  For example, "(3A):" should be separated into four tokens ("(", "3A", ")" and ":").
     + Also if a word ends with a period (.), you don't have to check if it is a known acronym (e.g. "Mr.", "m.p.g.", "e.g.", "etc.").  For this assignment, you can just look at the symbols in both ends of a word string.
   * [**Contractions**](http://dictionary.cambridge.org/us/grammar/british-grammar/writing/contractions) are separated into individual tokens (without expanding to true words).  Although some contractions are ambiguous (e.g. "they'd" could be "they would" or "they had"), in this assignment you can make these simple assumptions:
     + words ending with n't -- separate the stem and change "n't" to "not"; e.g. "don't" -> "do" and "not" -- but except for "won't" and "can't" (see **(\*)** below).
     + words ending with 'll -- assume "will" for all instances; e.g. "they'll" -> "they" and "will"
     + words ending with 've -- assume "have" for all instances; e.g. "they've" -> "they" and "have"
     + words ending with 'd -- assume "would" for all instances; e.g. "they'd" -> "they" and "would"
     + words ending with 're -- assume "are" for all instances; e.g. "they're" -> "they" and "are"
     + words ending with 's :
       - -- assume contraction of "is" if the preceding word is a **singular pronoun**; e.g. "it's" -> "it" and "is".  Pronouns that apply here (for this assignment) are "he", "she", "it", "that", "here" and "there" (and the versions with a capitalized first letter).
       - -- assume possessive (i.e., an *apostrophe-s*) for all other instances; e.g. "phone's" -> "phone" and "'s"
     + special ones:
       - I'm (or i'm) -- "I" (or "i") and "am".
       - **(\*)** won't (or Won't) -- "will" (or "Will") and "not"
       - **(\*)** can't (or Can't) -- "can" (or "Can") and "not"
     + Also if a word contains multiple contractions (e.g. "shouldn't've"), you must **separate ALL of them** (e.g. "should", "not", "have").
3. Output must be written to a file.

**2. Data files**

To help you develop and debug your code, a sample input text (sample-text) is provided, as well as a list of stopwords

Also note that the sample text does not cover all cases of punctuations and contractions.  You should create your own test input file to ensure the correctness of your implementation.

**4. Deliverables**

Submit the following:

1. Your code.  All files involved (which you wrote).
2. A write up document explaining what you did and how you addressed the different questions in the assignment (in docx or pdf; 2 page minimum).
3. (optional) An extra credit word cloud. For 10 extra credits use a library and create a word cloud with the 50 most frequent non-stopwords in the sample-text.

Requirements:

* IMPORTANT: Your code file must have your name at the top of the file (in a comment section).
* IMPORTANT: The write-up document must **your name**, the **course name** and the **assignment number** (HW#1) at the top of the files you submit (in the header of the code file also as comments).
* Write-up must also include:
  + I**nstructions on how to run your code**, including the programming language, development environment and library/package installation (besides Python 3 and NLTK), the location of the input file (and where the output file is generated).
  + A **short description** of what you did and all the various assumptions you made in the code.
  + **Final/overall comments**.  For instance, how difficult you felt this assignment was, any difficulties you encountered, how you would improve your code.

**5. Submission**

Submit the necessary files on blackboard by Monday April 26 EOD.