Second Homework Assignment (10% of grade)

Due on Monday, April 27 at 5 pm.

For each assignment, deliver the following in folder /home/public/assignment (you should have write permissions; after you copy every file, execute: chmod 400 file_name)

1. Your java source code file: name the file lastname1_exercisex.java where x is either 1,2,3, and 4.
2. One output file from a reducer: name the file lastname1_exercisex.txt where x is either 1, 2,3, and 4.

There should be two files per student per exercise.

1. Google ngrams: Google counted occurrences of words in the entire collection of their Google Books. There are two files: one file that reports 1 grams (single words) and the other file that reports 2 grams (pairs of words; one following each other in text). Note that the two files are only 2 sample files and not the entire data set which is too big to be stored by each one of you.
   The 1gram file format: word \t year \t number of occurrences \t number of volumes \t potential other irrelevant fields
   The 2 gram file format: word \t word \t year \t number of occurrences \t number of volumes \ potential other irrelevant fields
   The data represents the number of occurrences of a particular word (or pairs of words) in a given year in all books available by Google Books. The number of different volumes/books including the word (or pairs of words) is also reported.

   Write a mapreduce java routine that will for each year report the average number of volumes including words containing substrings: 'nu', 'die', 'kla'

   The output should be: year substring average; Example: 2000 nu 345; 1998 die 31; …
   Do this in the most efficient way with a single mapreduce job. Beyond the file formats described above, you are not allowed to make any structural assumptions on the data; e.g., that there are no extra fields – some records can have extra fields. The 'year' column may include erroneous values which can be a string. If the year field is a string, the record should be discarded.

   The data set is available in /home/public/course/google
2. Google ngram average (not easy – only if you want to be challenged): The data set is the same as in the previous exercise. You need to compute in mapreduce the standard deviation of all volume values (across all records and both files). The output should be a single value.
   You must not use more than 2 mapreduce jobs.
3. Million songs: These is a subset from the million song database. From the data set you have to extract the records for years 2000-2010 (field 165). For each extracted record your output file should only have fields: artist name (field 2), duration (field 3), song title (field 1)
   You should do this as efficiently as possible in mapreduce.
   The file is in /home/public/course/music

4. Million songs: The data set is the same as in the previous exercise. For each artist in the data set, compute the maximum duration of a song, i.e., the output should be: artist, max duration of all his/her songs.
   In addition, the management of your firm wants the artists to be sorted across all files based on the first character. Your firm is very conservative and does not have Microsoft licenses and they oppose open software tool such as Open Office. The bottom line, you cannot take the output files and then sort them in a spreadsheet software. You are only allowed to concatenate them at the end.
   In order to save computing resources in your conservative firm, you have to use 5 reducers.