**IR Phase 2 Requirements**

Phase two involves your corpus and ask you to do the following requirements:

REQ1) implement Porter's Algorithm (stemming) on the words in your inverted index and store the results in some structure.

REQ2) for both the inverted index and for Porter' algorithm you should ensure Persistence. Persistence means that you do not want to have to compute your inverted index and now Porter's algorithm more than once. The problem is that any time you exit your program you lose the data. So, you need persistence so that the data will exist even if program exited and then can be read in when program starts again.

There are three common ways to persist:

#1) You can write out to a text file, which you then read in at beginning of each program;

#2) You can communicate with an actual database since the database can persist the data for you, IF you set things up correctly.

#3) You can Use Java Serialization (see attached document). This allows for a Java Object to "wrap" around all your data and be outputted (and then inputted) as one.

*For this semester*, you can only use #1 or #3 for REQ2.

REQ3) The program should allow the user to decide whether s/he wants stemming *as part of the search process* by setting a switch/flag on command line. This means that if the user requests stemming, you need to search for all documents that share the same Porter Stem for a given query term (instead of just matching the specific word).

REQ4) Test your corpus by searching for one of the original queries. You search through the ENTIRE CORPUS and return the set of all documents that have ALL terms from the original Query. You will be doing this two times, the first time as the original query and the second time where STEMMING is turned on. For EACH document that you want to return, calculate a "snippet" where the first time the word is found in the document. Your inverted index should have the locations (ie positions) in each such document of each such word. A snippet means is a quote from the document starting from a fixed number of words before where the word appears and upto the same fixed number of words after the position where the word appears. The actual fixed number should be provided in command line by a flag with a value the user chooses. (These snippets will likewise be displayed in Phase 3 in the results of your search engine.)

You may use any code found on web specifically for Porter's Algorithm BUT You will need to COMMENT before the code where exactly the code is from (even the URL).  ALL OTHER CODE MUST BE WRITTEN BY YOU ALONE.

Submission protocol.

1) The Subject Line for submission should be as follows:

Subject:CS370 Phase(#) Project <Name of Website Assigned> <Due-Date-of-Phase>

2) The submission is done in one email from your Qmail to Class Projects email.

3) Attach to this one email all of the project .java files, (not the .class or .project files a Readme.txt how to install and start your program from command line (elaborated next in #4), a "User Manual" (MS Word document) explaining how to use all the features of your program both from command line and from GUIs and attach any input and output files (described in #6 below).

4) The Readme.txt (mentioned in #3 above) should discuss how to compile (javac) and run (java) your project including what each flag/switch provided at command line refers to/how to use.

5) An input text file of 5-10 different queries that will be read in from command line and processed by your system. You will be testing each query with and then without Porter's stemming algorithm.

6) Two combined Output files that provides for each of these queries that state the query, followed by the document names that are returned with snippets of each document. The reason that there are two output files is that one is for Porter's algorithm used and the other is for the same queries with Porter's algorithm not used.