A state of Concordance

T. Reinhardt

1 What this is about

A "concordance," in the sense it's being used here, is an analysis of written text wherein we count the number of occurrences of each word. A common, well-known concordance is Strong's Concordance of the King James Bible. We, however. will not undertake such an enormous task for this assignment; instead, we will use JAVA's `HashMap` class, along with some other libraries, to perform a simplified concordance over a small body of text. Specifically, in this Lab you will

- Use File and I/O methods to open, read, and close a text file.
- Use a *Regular Expression* library to interpret the tokens contained within this file.
- Create associations `<key,value>` pairs that reside in a hash table-like structure, and
- Implement some additional methods that perform the concordance and interpret what's in the table.

1.1 Looking at the pieces ...

By the time you read this document we should have discussed, in some detail, the mechanics of Files and I/O in the JAVA language. You will use the `Scanner` class in conjunction with the `File` class to access a text file, which is included in your project. Note: after you've passed the various tests, etc., you could substitute another source for that file and see how far your code extends.

The more *interesting* piece is how you need to handle *tokens*, i.e., `String` objects that are separated by spaces... think of these as words. Several things should come to mind here:

- How will I handle case? Is "Thing" the same word as "thing?"
- How will I handle punctuation? Is "thing" the same as "thing."?
- How will I handle numbers?

It so happens that many modern languages provide libraries that support "regular expressions." You will certainly study regular expressions if you take a course in Compilers or any course in Theoretical Computer Science. Essentially, regular expressions are strings that can be constructed by the application of certain "rules," such as concatenation. Without getting into the gory details, think about the kinds of tokens, complete with "wildcards" that you type into Find and Replace programs. Intuitively, the rules that allow those programs to correctly interpret your request comprise the rules of a regular grammar.

For this assignment, you will either have to write the code that distinguishes tokens by removing punctuation, handling case, etc., or, preferably, learn to use JAVA's `regex` package. Without giving away too much, you'll just need to figure out what is the correct "pattern" (the code that describes what constitutes "'words" to your application), compile it as a pattern, and then let `regex` do the work for you. (Hint: the quickest way might be to have the "pattern" object `split` the stream from the file into an array of `String`s ....)

Assuming these tasks are completed, you then need only design the associations, i.e., the `Map` entries and manage the table, which is likely a `HashMap`.