CS 353 – Functional Programming
Program 1

The first program of the semester is fairly straightforward text processing. This is to give you a chance to get familiar with Racket and the functional paradigm without having to struggle to understand the problem.

You are given several text files, with various types of text – short stories, news articles, etc. Your task is to break each article into words and begin a classification by topic, by identifying the 5 most-frequently-occurring meaningful words in the text (more on this below).

Your program will ask the user for the name of the file, which can be stored in the same folder as your program. Your program will then need to carry out the following operations. How you arrange them or carry them out is up to you:

- Remove punctuation: Most punctuation marks can simply be dropped or replaced with empty strings. The exception is the single quote mark, which might be part of a contraction (such as *can't* or *won't*) or a possessive (*Fred's book*); those should not be removed. Single quotes used for emphasis or to set a term apart (*the word 'focus' originally referred to the hearth)* should be removed. A lone single quote at the end of a word might be a possessive for a name ending in S (*Iris' books*). *Hint*: look at where the single quote mark is in the string.
- Break everything into single words, and convert those words into a single case, either all lower-case or all UPPER-CASE. It doesn't matter which case, but you want everything to be consistent.
- Filter out the common 'utility' words such as *for, in, the, on,* etc. You are given a file of words to remove, "stop_words.txt". Anything from this file should be removed from your working text.
- For the remaining words, count the number of times each word appears in the list, and select the 10 most-frequently-appearing words. If there is a tie for 10th place, include all tied values.
- List the frequently appearing words to the screen and ask the user whether to process another file. Continue until the user says to quit.

Programming Notes

- Your program is to be written in Racket, using a functional style. Note that the Racket libraries have a wide variety of data structures already defined—hash tables, trees, etc.—though simple lists may be enough for this assignment.
  - If you do use the more advanced data structures, be sure to use the functional versions. Most of the data structures are implemented two ways—as mutable data structures (pass an item to add or delete, and that's it) or immutable (functional) structures (pass an item to add or delete, get back a reference to a new structure with the desired changes).
  - Each use of the set! function will result in a penalty of one letter grade.
- Use of Generative AI such as ChatGPT, Copilot, etc: You may use these tools as you wish, but save your prompts. You will probably find them most useful for problem decomposition and for producing draft code once you've broken the problem down. (Given a sufficiently specific description of what a function should take as input and output, most of these tools can do fairly well at providing draft code.) They can also be useful for queries such as "I need a function to do thus-and-such, is there a library function to do that?" (But note that sometimes it makes things up or mis-describes things; *anything* from a large language model needs to be checked.)

Deliverables:

- Your source code .rkt file(s).
- A document (Word, pdf, or text) listing the sources you used in developing your project, including any AI use (what tools you used, what you used them for). Also, any discussion or comments about your code, the project, etc.