Comp160
Project 2
Spring 2018

The Project

For your second project you'll be completing and adding to the batch program for English dictionary analysis as described in Section 12.1. You'll be working with the English dictionary found on Linux systems and carrying out some analysis in order to determine which letters occur most frequently as the first letter of a word. Just like with the first project, your work is broken down by tiers.

Project Tier 1

For the first tier you'll need to complete all the exercises in Section 12.1. These problems walk you through the design of functions for analyzing which letter occurs most frequently at the start of a word. Before getting too far into the exercises you need to design a function named *string-downcase-all*[1] which will apply the string-downcase function for strings to every word in a dictionary to produce a dictionary that only contains lower case words. Using this function allows us to focus on letters without getting hung-up on upper vs. lower case.

[1] look up string-downcase in the documentation

Project Tier 2

For the second and final tier you need to add the following functions:

- Design a function *sort-counts* that sorts a list of letter courts in most to least frequent order. This functions is really just an adaptation of the sort discussed in Section 11.3.
- Design a function *top-n* that takes a list of letter counts and a positive number n and returns a list containing the n most frequent letter counts.
- Design a function *top-cumulative* that takes a list of letter counts and a positve number n and returns the relative frequency of the n most frequent letters as a percentage. As an example, say we only had five letters, a, b, c, d, and e, and that those letters had frequencies of 10, 15, 7, 6, and 8 respectively. This is a total of 46 letters giving the letter a a frequency of roughly 21.7%, letter b a frequency of 32.6%, letter c a frequency of 15.2%, letter d a frequency of 13%, and letter d a frequency of 17.4%[2]. If the user

[2] all values are rounded off

calls top-cumulative with an n of 2 then they should expect to get a result of roughly 54.4% because the two most frequent letters, a and b, account for 54.4% of the letters counted.

Homework 6

To ensure you get started in a timely fashion the following items will be submitted as homework assignment 6.

1. A complete set of data definitions for the project that includes concrete examples of all defined data types.
2. Function templates for all the defined data types.
3. Finish exercise 195
4. Finish implementation of *string-downcase-all*.

Notice that by successfully completing this homework assignment you ensure at least a passing grade on the project as you will have completed two function design tasks from the first tier of the project.

Grading

Your grade will be determined by two factors: your progress through the tiers determines the letter grade range that is open to you and the quality of your work will determine where you land within that range.

Tiers and Grade Ranges

Programs that complete at least one of the function design tasks in the first tier will receive something between a D and a B depending on how many tasks they complete and the overall quality of the code. Programs that produce syntax errors when run can expect to receive a failing grade. Programs that run but crash often or lack any testing can expect receive something in the D range at best. High quality programs that complete tasks in the second tier can expect a B+ or better.

Tier	Grade Range
1	D to B
2	B+ to A

Quality

A high quality program exhibits all the earmarks of intensional, systematic design and good programming style. Program data is

appropriately defined. Functions are well documented. Incomplete functions are stubbed as opposed to commented out. Complete functions typically exhibit template structure when applicable. Complete and incomplete functions have a full set of tests. Signatures, purpose statements, tests and function definitions are all appropriately placed. Function and variable names are helpful and meaningful. Purpose statements are specific and concrete. Indentation of code follows expected Racket standards[3]. Lines are terminated to avoid print wrapping. Comments are used to aid the human reader. Whitespace is used to break up logical blocks of definitions. The degree to which you meet these standards, along with progress within the tier, determines where you fall within the grade range associated with your tier. It is entirely possible the poorly designed and styled code the completes three extensions can get a lower grade than a well designed and styled program that completes one or two extensions. Do not short yourself points by writing sloppy code.

[3] It's styled like the code in the book.

Important Dates

Date	What's Happening
Monday 4/23	Open Lab Time to For Project Work
Tuesday 4/24	Open Lab Time to For Project Work
Friday 4/27	Homework 6 Due No Later than Today
Monday 4/30	Open Lab Time to For Project Work
Tuesday 5/1	Open Lab Time to For Project Work
Wednesday 5/2	Code due by the end of the day.