SENG265, FALL 2017
ASSIGNMENT 3
UNIVERSITY OF VICTORIA

Due: Nov 17, 2017 by 10:00 pm, by "git push".
(Late submissions **not** accepted)

1 Assignment Overview

This assignment has two distinct parts. The first part of the assignment involves writing a program to solve the same problem as Assignments 1 and 2 except using Python 3. As a reminder, part 1 will read lines of text from a given file, compute the frequency of words of certain length from the file and print these frequencies to standard output.

The second part involves writing an interactive program to examine ferry departure delays using a data set of ferry schedules from BC Ferries. The program will interactively prompt the user for input which will affect the the information that the program outputs.

The overall goal of this assignment is to familiarize yourself with programming in Python. There are two parts to this assignment and Sections 2, and 3 below contain `Specifications` for each of the two programs. Section 4 describes the (new) `Constraints` you should consider in your implementation, and Section 5 describes the `Testing` component. `What you should Submit` is outlined in Section 6 and the `Evaluation` scheme is given in Section 7.

Your code is expected to run without warnings **in the course lab (ECS 342)** using Python 3.4.3.

2 Part A

The first part of the assignment involves writing a program to solve the same problem as Assignments 1 and 2 except using Python 3. It includes optional arguments to sort and print the words of the same length, exactly as in Assignment 1 and 2. You **cannot** assume that the arguments will be run in this order.

```
$ python3 word_count.py --sort --print-words --infile <input_file>
```

Recall that there are four possible options to run the program with these arguments:

`$ python3 word_count.py --infile <input_file>` or,
`$ python3 word_count.py --sort --infile <input_file>` or,
`$ python3 word_count.py --print-words --infile <input_file>` or,
`$ python3 word_count.py --sort --print-words --infile <input_file>`

The same test files will be used during the evaluation so please review the expected output formatting.

3 Part B

The second part of the assignment is to write an interactive Python program, contained in a source file called `ferry_delays.py`, which calculates ferry delays using one or more CSV files containing ferry schedule data from BC ferries. The program must run, with no errors, using the following commands:

```
$ python3 ferry_delays.py <dataset1.csv> [dataset2.csv, ..., datasetN.csv]
```

where `dataset1.csv - datasetN.csv` are the data files from BC Ferries.

3.1 Data Set

There are 3 data files in CSV format included with the assignment using the naming convention `bc_ferries_route1_MON2017.csv`, where MON is is the month for the data. Your program should take a list of one or more file names at the command line that contain bc ferry data (You can assume that the files loaded are in the correct CSV format and have the correct headers).

3.2 User Input

The program is required to prompt the user for inputs to select what information is displayed to the user. You must also validate the data upon input. If the data is invalid notify the user of their error and prompt them to enter their selections again. Follow the steps below:

- Prompt the user if they would like to calculate delay statistics for a Tsawwassen or Swartz Bay. The user should enter **t** for Tsawwassen, or **s** for Swartz Bay.
- Prompt the user to select which month they would like to calculate the average for by having the user enter a number from 1 to 12.

The user should also be able to enter **q** for quit at either prompt, and the program should continue to run (ask for input after results are provided) until **q** is entered.

3.3 Output

Once the user has entered their desired selections, you should output delay averages based on their choices. Output the `name of the terminal` followed by the line `Average delay for MON: AVG`, where MON is the 3 letter month abbreviation and AVG is the average delay rounded to 2 decimal places. If there is no data for the specified month output `No delay data for MON`, where MON is the specified month. Consider the following:

- The departure `delay` is calculated by taking the difference between scheduled departure and actual departure. When calculating delay, you can assume that the no ships will be delayed by a day (i.e. delays are only within the same day). A negative departure delay indicates the ship left early, and a positive delay indicates the ship left late.
- Use the following three letter month abbreviations when printing month output: `Jan, Feb, Mar, Apr, Jun, Jul, Aug, Sep, Oct, Nov, Dec`
- To make sure our automated testing works properly, the printing of results should be framed between the keywords `RESULTS` and `END` (in all caps).
- The example formatting must be followed exactly due to the automatic testing. Submissions that do not follow the output spec, will fail

Here are three examples of user input/output with the required formatting:

1. When the user inputs:
```
t
5
```

The expected output is:

```
RESULTS
Tsawwassen:
    Average delay for May: 4.76
END
```

2. When the user inputs:
```
s
6
```

The expected output is:

```
RESULTS
Swartz Bay:
    No delay data for Jun
END
```

3. When the user inputs:
```
s
2
```

The expected output is:

```
RESULTS
Swartz Bay:
    Average delay for May: 4.09
END
```

NOTE: The average delay in these examples are fictional and do not reflect the data in the files provided

4 Constraints

You may only use use python3 modules that are installed on ECS 342 or linux.csc.uvic.ca. If an python3 module is not installed, you may not use that module in your code.

You may assume that all test files will be in the same CSV format provided with the assignment.

5 Test Inputs

You should test all of your programs with a variety of test inputs, covering as many different use cases as possible, not just the test input provided. You should ensure that your programs handle error cases (such as files which do not exist)

appropriately and do not produce errors on valid inputs. Since thorough testing is an integral part of the software engineering process, you will be provided test input.

Provided For You

For this assignment, you will be provided a folder containing 3 test input files used to evaluate assignment 3 in a zip file tests.zip available on Connex in the assignment description. You will also find in the same Connex folder an archived file `output.zip` containing 3 input and corresponding output files you can use for testing, together with a README.md file for description of the expected input and corresponding output files and how to run the tests.

(HINT: Use `diff` to compare your results with the provided expected output)

6 What you must submit

- Python source-code name `word_count.py` which contains your solution for Part A of assignment #3.
- Python source-code name `ferry_delays.py` which contains your solution for Part B of assignment #3.
- Ensure your work is **committed** to your local repository in the provided **a3** folder **and pushed** to the remote **before the due date/time**. (You may keep extra files used during development within the repository.)

7 Evaluation

The teaching staff will primarily mark solutions based on the input files provided for this assignment, as well as additional input files. Students must adhere to the command execution and output formatting outlined in this assignment. For each assignment, some students will be randomly selected to demo their code to the course markers. Each student will have this opportunity to demo at least one assignment during the semester. Sign-up procedure will be discussed in class.

In addition to automated testing, your code will be evaluated based on:

- Proper error handling
- Good coding practices (i.e. good variable/function naming, use of functions when appropriate, limited globals, etc.)

Our grading scheme is relatively simple.

- "A" grade: A submission completing ALL requirements of the assignment with good code quality and all tests pass. The `ferry_delays.py` and `word_count.py` programs runs without any problems.
- "B" grade: A submission that completes part A & B of the assignment and some or all tests pass. The `ferry_delays` and `word_count.py` programs runs without any problems.
- "C" grade: A submission that completes part A of the assignment and all tests pass. The `word_count.py` programs runs with some problems.
- "D" grade: A serious attempt at completing requirements for the assignment. The `word_count.py` program compiles and runs with some problems.
- "F" grade: Either no submission given (**or did not attend demo**); submission represents very little work or understanding of the assignment.