Minor Assignment 5

ENGINEER 1D04

Dr. Spencer Smith

McMaster University, Fall 2013

Revised: February 18, 2014

Please use AutoMarker (`automarker.mcmaster.ca`) and Avenue to acquire, test, package and submit your assignment. The procedure for submitting assignments is summarized on Avenue, with additional details provided by AutoMarker. **Please frequently back up your work by creating a submission package in AutoMarker**. This will provide a chance to recover your work in the event of an equipment failure.

Background

The purpose of this assignment is to create a program that can process a file containing your courses and the grades you have received and then display the results graphically. Design, implement, and test a program that satisfies the requirements below. Submit your program on Avenue to Learn.

Design, implement, and test a program that satisfies the requirements below.

****IMPORTANT!!!**:** This assignment will be run through an automated testing program to be graded. Function syntax in your program must be **exactly** as specified, including spelling, capitalization, and the order of function parameters. Failure to precisely follow the requirements below will result in a **significant loss of marks**.

Requirements

1. The program does three things:

    a. It asks the user for the name n of the input file in `main()`.

    b. `main()` calls `processFile` and `outputGUI`.

    c. It displays the two results produced by `processFile` using `outputGUI`.

2. The program includes a function named `processFile` that satisfies the following requirements:

    a. The function takes a filename as input (not the file content) in which each line contains the name of a course, the number of units in the course, and the course grade in the range of 0–12. That is, the file stores a finite sequence $(c_0, u_0, g_0), (c_1, u_1, g_1), \ldots, (c_{n-1}, u_{n-1}, g_{n-1})$ An example file would look something like:

```
CHEM 1E03       3   9
ENGINEER 1D04   4  12
ENGINEER 1P03   3  10
HISTORY 1B03    3   8
MATH 1ZA3       3  11
PHYSICS 1D03    3   9
```

Each element is separated by tab characters.

b. The function given for U computes the total number of units.

$$U = \sum_{i=0}^{n-1} u_i$$

c. The function given for A computes the cumulative grade average.

$$A = \frac{\sum_{i=0}^{n-1} u_i g_i}{U}$$

d. The function returns values U and A as output.

3. The program has a function `outputGUI` that provides a graphical user interface (GUI) for output that satisfies the following requirements:

   a. The function takes the total number of units (U) and the cumulative grade average (A) as inputs.
   b. The output GUI is a `GraphWin` window W entitled `Grade Processor Results`.
   c. U and A are displayed with appropriate labels on W.

Design and Implementation Instructions

1. You may use the methods provided by Zelle's `graphics` module to create your output GUI.

2. You should use your creativity in deciding how to display U and A on the output GUI.

3. The program requires very little besides the function definitions.

4. The program does not write anything to standard output. That is, the program does not print anything to the shell. The program also should not invoke `main()`.

5. The program is written in Python in a module, NOT in the Python Shell. To create a new module in IDLE, go to File → New Window. You must save this file with a .py extension. For more information on submitting your program, click the "AutoMarker Instructions" button above.

6. Your name, MacID, student number, and the date are given in comments at the top of your Python (.py) file before your program.
7. Your answers to the design questions and the test plan (see below) are given in comments at the bottom of your Python (.py) file after your program.
8. Your program MUST have valid Python syntax and it must run without errors. Ensure that your program runs properly by running it before you submit.
9. You must sign out with a TA or IAI after you have submitted your lab at the submission station. Failure to do so could result in a zero.

Design Question

1. Can all changes to how the file is processed be made without modifying any part of the program except for the `processFile` function? Why would this be desirable.
2. Do the parameter names matter? Do they have to be called A and U?
3. What if main() is invoked before the other two functions' definitions? i.e:

```
def main():
    block of code
    processFile(fName)

main()

def processFile( name):
    block of code
```

Test Plan

Produce a test plan for each test case `i` in the following form:

```
Test:  i
Input:  [n]
Expected Output:  [U,A]
```

You should have enough test cases to adequately support the argument that your code is correct. Your test cases should cover as many different classes of input cases as possible, including boundary cases. Your test plan should include case(s) where your expected output is a failure.

Preparation for Major 3: Writing a Library

Write a Python *library*, saved as `gradeCalcLib.py`, that can be used to support software that deals with grade calculation.

1. The library should include the `processFile` function described above.
2. The library should have a function, `plotGrades` that takes the same file input as `processFile` and produces a bar graph of grades.
3. **The program requires very little besides the function definitions. There is no `main`. The program does not read anything from standard input or write anything to standard output. That is, the program does not interact with the user who invokes it.**

Then try to modify your library and create a separate module with a `main` function that gives you a sophisticated grade processor by:

1. Having the program produce additional results such as the highest and lowest score obtained.
2. Using the GUI for handling both input and output.