ISTM6200 – Python Programming With Database Applications

Fall 2018 Semester Project: Part 1 – Text Processing

**Introduction to the Semester Project**

Over the course of the semester we will learn a wide variety of features in the Python programming language that can be used in the manipulation of data. To demonstrate your mastery of these features each student will be required to write a programs to read data from a file, analyze it, store it in a database and do further analysis on it. For Part 1 of the project, you will write functions to read a data file, count characters, words, lines, and some more advanced functionality. I will provide a graphical user interface which you will modify to invoke the functions and some test data that you can use to test your software.

**Academic Integrity Issues**

This project is both individual and collaborative, as not all students will possess the same talent with regard to programming. I expect stronger students to make some effort to aid weaker students. However, there is a fine line between getting legitimate help and just copying someone else’s code -which is cheating and is a violation of Academic Integrity. So to keep this in the realm of helpful collaboration (which benefits both the helper and the person being helped) and to keep it out of the realm of cheating, please consider the following.

1. Acknowledge anyone from whom you received help in comments at the beginning of the file.
2. If you provided help for anyone, acknowledge that as well.
3. You MUST use the GUI provided for the current semester as I make minor modifications from one semester to the next and use those modifications to discourage cheating.
4. Since it is nearly impossible for two programmers to write identical code on an application program of any complexity, I will look for unexpected commonality between programs. This includes:
   1. Using the same variable names
   2. Containing the same errors
   3. Having common but unusual stylistic features

All of this can be avoided by following a couple simple rules. It is OK to ask questions. It is OK to view somebody’s code as they explain a principle. However, **do not share code in any way**. That is, do not cut and paste segments of someone else’s code or type in segments of someone else’s code. Whatever you do to get help with a problem, when you type in your code, do not use any thing that did not come from your own head.

There are great temptations for students to “borrow” projects from previous semester in order to make the project a little easier for them. However, this not only interferes with your learning, but it is a very risky strategy. Requirements will be changed slightly from one semester to the next and some of these changes will be rather subtle. Any projects which contain features from a previous semester will automatically be downgraded as this could not happen in any way other than “borrowing” a project from a preceding semester. If the similarities are too obvious, I may be forced to submit the case to the Office Academic Integrity to see if there is enough evidence to bring a case. I sincerely hope this does not happen. As a final disincentive to “borrowing” code, I reserve the right to ask you to explain any segment of code in the project that you turn in. If you cannot explain it, I will assume that you copied it from somewhere.

In addition, I should mention that the project serves a dual purpose. The first of which, I just explained. The second of which is that there is no better way to learn a programming language than to use it to write an application. On the quizzes and exams, there will certainly be some questions that can be answered from memorization. However, there will be others that require an understanding of the language. If you do not develop your own project, it is unlikely that you will be able answer questions of this second type. So, if you do your work yourself, then you will do the necessary learning. If you do not, you probably won’t.

**Part 1: Text Processing**

Your program should be able read a file, count the characters, lines, words and occurrences of a particular word, and provide some more advanced features as follows:

Create the Following Functions:

WhoAmI() – This function will display your name in the first line edit box so I can verify whose program I am working on.

CurrentSemester() – This function will display the current semester in the form XYYYY (e.g. F2017)

OpenFile() – I will type the name of the test file into the line edit box and press the button above the box. If the file does not exist, print “File Not Found” in the line edit box. Otherwise open the file and print “File Opened” in the line edit box.

CountCharacters() – This function will count the number of alpha characters (a-z and A-Z) in the file. Tabs, line feeds and non-alpha characters should not be counted.

CountLines() – This function will count the non-empty lines in the file.

UniqueWords() – This function will count the number of unique words in the file. Note: Be careful because the word “fox” occurring in a sentence and the word “fox.”, occurring at the end of a sentence, are the same word.

AvgCharsInWords(char) – This function determines the average characters per word. Punctuation does not count. A hyphenated word is one word but the hyphen does not count as a character.

Quit() – This function will quit the application.

A graphical user interface will be provided so that the functions can be invoked by pressing buttons on the interface.

Here are some important tips:

We will be covering similar functions in class. As soon as you know how to do a given function you should do it, or at least a rough working version of it. Doing little bits, as you go along, is way, way easier than waiting until the project is due and trying to get it done in crunch mode. In addition, writing your own functions and getting them to work together is way, way easier that attempting to modify someone else’s code from a previous semester.

I will be testing the project using Python(x,y) version 2.7 on a Windows 10 machine. If you do your development on a difference platform (Mac, Unix, Linux, etc) make sure that it runs on the lab computers using Python 2.7. If you program crashes on my machine I will not debug it. You will just loose the points.

I will enter the file name as a local file in the current directory. Make sure that your program does not use a pathname as that will cause it to crash on my machine.

Projects will be graded on a curve where the best projects will receive full points and projects with some errors or limitations will receive fewer points. If you cannot do the full set of functions, do whatever you can as the worst that will happen is you will lose a few points for incomplete or erroneous functions.

The point of the project is to learn programming not to get points or a grade. Nobody will ever care how many points you got on the project. But, when you open your mouth and reveal serious limitations in your understanding of programming, they will draw conclusions about you.

Don’t underestimate the amount of time you will need to complete this. However, programming when you have lots of time can be a lot of fun while programming done under the pressure of a deadline can be your worst nightmare.

**Turning it in**

Your functions and graphical user interface should all be in a single file with your name followed by a 1 as the file name and a py or pyw extension. (For example, if I were turning it in, my file would be Artz1.py) If you think there is somebody else in the class with your last name, then add your first initial (e.g. ArtzJ.py). If you think there might be somebody else in the class with your first and last name, then add as much as you need to make it unique. However, make sure it begins with your last name so I can keep them straight. Email your file to me by the beginning of class on Week 9. Late projects will be accepted, but will lose points for being late. Please don’t send me multiple copies of the project unless you are willing to loose points.