COMP 1012 Fall 2017 Assignment 2

Due Date: Friday, June 22, 2018, 11:59 PM

Material Covered

- list, tuples and dictionaries
- reading input from text (.txt, .csv) files
- built-in functions and list functions
- functions from math

Notes:

- Name your script file as follows: <LastName><FirstName>A2Q1.py. For example, LiJaneA2Q1.py is a valid name for a student named Jane Li. If you wish to add a version number to your file, you may add it to the end of the file name. For example, SmithRobA2Q1V2.py is a valid name for Rob Smith.

  Spyder automatically adds the .py part (called the extension) to your file name, so you don't have to type that part when saving the file. However, please ensure that your file name DOES include the .py extension, since our marking software looks for it.
- Name your output file as follows: <LastName><FirstName>A2Q1output.txt. For example, LiJaneA2Q1output.txt is a valid name for a student named Jane Li.

  In order to prevent Spyder from automatically adding the .py extension to your output file name, you must type the complete file name, including the .txt extension, when saving your output file.
- Follow the posted programming standards to avoid losing marks. Check your script for adherence to the programming standards by using CheckStandardsV2.py, which will show you exactly where your script does not comply. We will use the same script to assign marks for your assignment, so you have NO EXCUSE for losing these marks.
- You must complete the ***Blanket Honesty Declaration*** checklist in order to submit your assignment. This applies to all assignments in COMP 1012.
- To submit the assignment follow the instructions on the course website carefully. You will upload both script and output files, via the course website. We will demonstrate the assignment hand-in procedure in lectures. There will be a period of about a week before the due date when you can submit your assignment. ***Do not be late!*** If you try to submit your assignment after the late submission deadline, you will get a message indicating that the deadline has passed.

Question 1—Statistical analysis of large datasets [60 marks]

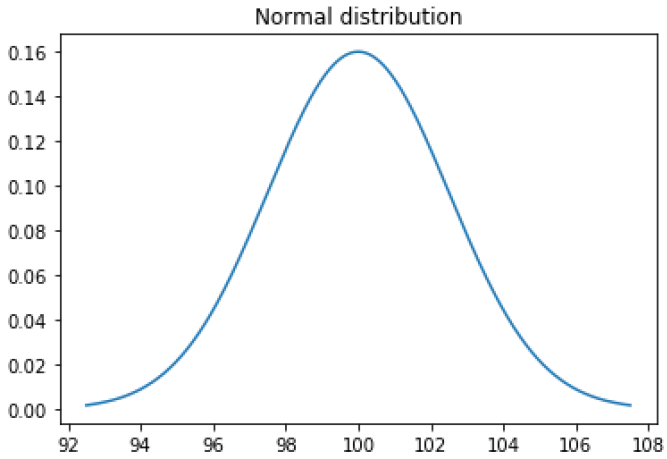
Description

One of our motivations for using Python is the ability to process large datasets efficiently. When those datasets are stored in text files (.txt or .csv files), we can read that data directly from our script. Storing the data into a data structure such as a list permits us to calculate statistical results like mean and standard deviation.

The mean (average) is one measure of the central tendency[1] of a probability distribution. An intuitive description of central tendency is "the tendency of data to cluster around some central value". We are familiar with the mean, but in this assignment, we will also calculate some other measures of central tendency, the median and mode.

Mean, Standard Deviation, Max, Min, Counts

The normal distribution is a common continuous probability distribution often used in the natural and social sciences. A plot of the probability density function (pdf) of a normal distribution has the familiar bell shape. Since the pdf represents probabilities, the area under the pdf curve is 1. The plot to the right was created using Python's plotting module, which we will use later in the course.



In the case of a normal distribution, the symmetry and shape of the pdf plot tells us that the mean, median and mode all occur at the same value (100 in this example).

The standard deviation is a measure that tells us how the values of a distribution are spread out from the mean. In the case of the normal distribution, a small standard deviation produces a 'narrow' bell shape, meaning most of the values are found within a small range around the mean. A large standard deviation produces a plot that is wider and lower.

In order to calculate standard deviation, we calculate the average of the squared differences of each measurement from the mean and take the square root of the average. We square the differences so that positive and negative differences don't cancel each other out. In order to calculate these differences, we must know the mean of the data, so we can't calculate the differences as we read the data. We must store the data in a data structure, calculate the mean and then the differences, by subtracting the mean from each value.

The equation for the population standard deviation σ (Greek letter sigma) for a population of measurements (x_0, x_1, … x_{n-1}) with mean μ (Greek letter mu) is:

$$\sigma = \sqrt{\frac{\sum_{i=0}^{n-1}(x_i - \mu)^2}{n}}$$

What to do

For Part 1, you will read input from a csv file (Grades.csv). This file contains tabular data that represents midterm and final marks of the students. The first row is a header record (row), with headings for the columns. Prompt the user for the file name, and open this file for reading.

Even though we already know how many columns are contained in this file, the script that you produce for reading this file must be ***generic***, which means it will work for a csv file with any number of columns and an initial header record. In order to accomplish this, we can't hard-code the data structures to work with a fixed number of columns; instead, we will use tuples and lists so that the data structures will adapt to the specifics of the input file.

For example, when reading the header record, we can't store the headings in separate variables because that would be hard-coding the number of columns. Instead, store the result in a tuple or list. The data structure will automatically contain the correct number of elements.

```
headings = tuple(infile.readline().strip().split(','))
```

Once you have read the header record, you know how many columns are contained in the file (by using len(headings)), and you can create your data structures to hold the data and statistics. You want to use lists to hold the data for each column, because you don't know how many data records are contained in the file, and you can append new data items to lists so they can grow to whatever size is required.

In order to make the data structures the correct size, use a for-loop to add appropriate initial values to your data structures. For example, the structure to hold the actual data will be a list of lists, so in your for-loop, append one empty list for each of the columns in the file. For your statistics data structures, you can append zeros.

Storing and using these data structures will require careful attention to subscripting, particularly for the list of lists storing the actual data. This list will require two subscripts (one for which list, one for which item in the list).

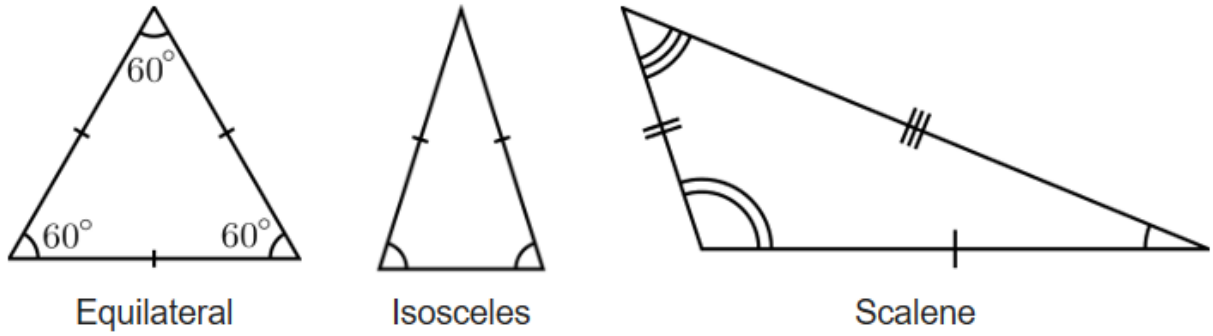
If the final marks of a student are less than 51 combining midterm and final term exams, the student failed in the exam. You also need to find how many students failed and passed in the exam. For this purpose, you need to add midterm and final marks for each student using a for loop and then find whether the student gets more than 50 or not.

```
Enter file name: Grades.txt
 Column Names |     Mean      | Std Deviation | Highest Score | Lowest Score
--------------+---------------+---------------+---------------+--------------
       Midterm|          22.25|          10.49|          40.00|          1.00
--------------+---------------+---------------+---------------+--------------
         Final|          39.12|           9.18|          60.00|          1.00
--------------+---------------+---------------+---------------+--------------

Total Number of students: 100
Passed in the exam: 81
Failed in the exam: 19


Programmed By Instructor: Sheikh Jubair
University of Manitoba
Computer Science
```

Question 2—Statistical analysis of Triangles [40 marks]

Description

A triangle is a polygon with three edges and three vertices[2]. So the triangle has three sides and based on the length of the side, it has three types: equilateral, isosceles and scalene.



Equilateral Triangle: The length of each side is equal.

Isosceles: The length of any two sides of the triangle are equal

Scalene: No sides of the triangle are equal.

Count Triangles

In this program, you need to count the triangles based on their category. You also need to find the count of sides that are equal. Remember, Isosceles can have three combination of sides that can be equal.

What to do

There is a file called 'Triangles.csv'. The file contains 3 sides of each triangle. The first column indicates first sides, second column is the second sides and the third column is the third sides. You need to

declare some count variables that can count the triangle number based on side length and triangle type. You should use the list and if-elif-else here.

```
     Column Names      |  Count
-----------------------+----------
s1 = s2 = s3           |      1446
s1 = s2 != s3          |       855
s1 != s2 = s3          |       674
s1 = s3 != s2          |        81
s1 != s2 != s3         |      6944
-----------------------+----------

     Column Names      |  Count
-----------------------+----------
Equilateral Triangles  |      1446
Isosceles Triangles    |      1610
Scalene Triangles      |      6944


Programmed By Instructor: Sheikh Jubair
University of Manitoba
Computer Science
```

Hand-in

You will hand in your program script file. Also hand in the output produced by your code that looks like the example. You can select your output in the IPython console window, then right-click and Copy (Raw Text) and paste it into a new document.

[2] https://en.wikipedia.org/wiki/Triangle