AIM 5001 Module 3 Assignment (100 Points)

A crucial aspect of working effectively with lists in Python is understanding how Python applies its internal indexing structure to lists. For this assignment you will be creating, searching, and sorting lists you will create from the contents of a small text file containing data values that are separated by commas (also commonly referred to as a 'CSV' file (**C**omma **S**eparated **V**alues)) using some of the concepts we are learning about this week.

Start by downloading the **M3_Data40.txt** file to your local environment.

Then, open a new Jupyter Notebook and copy in the following small Python code snippet:

_____

```
import csv

# Be sure to update the file path contained within the 'open()' statement shown below
# to reflect the location of the file within your own computing environment!!
# Also be sure that the code is properly indented after you paste it!

with open('yourpath/M3_Data40.txt') as csvfile:
   readCSV = csv.reader(csvfile)
   for row in readCSV:
      # print each row as read by the csv.reader function
      print(row)
```

_____

This code will read the contents of a file containing data values that are separated by commas in a line-by-line manner and print the contents of each line in the form of a Python list. Each item within that list will contain the corresponding element of the comma-separated line within the data file from which it came.

Once you've run the code snippet, you will see that each line of the file contains six distinct values. For example, a print of the result of reading the first line will show the following:

['FALSE', '63', 'MALE', 'TYPICAL_ANGINA', '145', '233']

These six distinct values represent attributes of hospital patients who had complained of chest pain. Specifically we have the following:

- Diagnosis (i.e., was the patient diagnosed with heart disease?)
- Age
- Gender
- Type of chest pain
- Blood pressure reading
- Cholesterol level

Now that you are familiar with the content of the file, **complete the following 6 tasks**:

(Assignment Continues on Next Page)

1. **(15 Points)** Your first task is to read the file again and extract these six attributes from each line of the file and create six distinct Python list objects (one for each of the six attributes) comprised solely of the values you extracted for a given attribute. In other words, you should have separate lists for diagnosis, age, gender, etc. For example, the first five "chest pain type" values should be as follows:

   ['TYPICAL_ANGINA', 'ASYMPTOMATIC', 'ASYMPTOMATIC', 'NON_ANGINAL', 'ATYPICAL_ANGINA']

   One way to complete this task would be to use the code snippet from above as a starting point: you could replace the "print(row)" statement with whatever Python code you feel is necessary to create the required list objects. When finished, each of your list objects should contain 40 data values.

   Note that the lists you have created thus far are composed solely of character strings. However, three of the attributes (Age, Blood Pressure, and Cholesterol) are clearly meant to be used as integer data values. As such, before proceeding any further, convert the contents of each of the 'Age', 'Blood Pressure', and 'Cholesterol' lists to integer values

2. **(15 Points)** Your second task entails using the updated Cholesterol list to find the **list index** values of each hospital patient having a "**cholesterol**" reading **that is greater than 250**. Create a new list object with your result. ***HINT:*** *you can accomplish this task by searching the list of cholesterol values you created as part of the first task*. Be sure to print your results.

3. **(15 Points)** Your third task is to find the "**gender**" value for each hospital patient having a "**cholesterol**" value **that is greater than 250**. Create a new list to store your findings and be sure to print your results.

4. (1**5 Points**) Your fourth task is to find the index value for each hospital patient having an "**age**" value that is **greater than or equal to 50** and a "**pain type**" value that ***is not*** "**ATYPICAL_ANGINA**". Create a new list to store your findings and be sure to print your results.

Class Objects

As we have learned, Python's class objects allow us to manage related data in an "object oriented" fashion. For the remaining tasks in this assignment you will be constructing and applying your own Python class object to the **M3_Data40.txt** data.

5. **(20 Points)** Create a new Python class to store and manage the information contained within the **M3_Data40.txt** file. Your new class should be named "PatientData" and should **include a class variable** 'PatientCount' that represents a counter indicative of the number of PatientData objects that have been instantiated within your current Python session. The class must also contain **seven instance variables** including a unique ID for each patient (use the name 'uniqueID' for this instance variable) and six additional instance variables corresponding to the six different hospital patient attributes contained within the **M3_Data40.txt** file. Each time you create a new instance of the class the 'uniqueID' instance variable value for the new object should be assigned the existing value of the "PatientCount" class variable. The new instance should then increment the 'PatientCount' class variable by 1 (after assigning its previous value to the 'uniqueID' instance variable as mentioned above.

6. **(20 Points)** Read the **M3_Data40.txt** file again and store its content **into a list of PatientData class objects**, with each row within the file being loaded into a distinct PatientData class object within that list. If you have defined your

PatientData class correctly according to the specifications provided in **Problem 5**, each class object within your list should then contain the content of one row of the data file, with each hospital patient attribute value having been assigned to a distinct instance variable within the class object, as well as a distinct uniqueID value that was automatically assigned by the PatientData class when you read the corresponding row from the data file into the class object. Then, using your list of PatientData objects, find the uniqueID for each hospital patient having an '**age**' value of **less than 56** and a "**diagnosis"** value of "**TRUE**". Create a new list to store your findings and be sure to print your results.

Be sure to **include some commentary within formatted Markdown cells** explaining your approach to solving each of the individual problems. When you are finished, save the Jupyter Notebook containing your work and commentary and upload / submit it within the provided M3 Assignment Canvas submission portal. Be sure to save your Notebook using the following nomenclature: "**first initial_last name_M3_assn**" (e.g., J_Smith_M3_assn_).