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 CSV file using some of the concepts we are learning about this week. Then, you will defining, instantiating and searching Python class objects constructed from that same CSV file.

Start by downloading the **cars-sample35.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 path below to reflect your own environment!!
# also be sure that the code is properly indented after you paste it!

with open('yourpath/cars-sample35.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 CSV file you specify 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 CSV file from which it came.

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

['high', 'high', '2', '4', 'med', 'low', 'unacc']

These seven distinct values represent attributes of a single type of automobile. Specifically we have the following:

- Price
- Maintenance cost
- Number of doors
- Number of passengers
- Luggage capacity
- Safety rating
- Classification of vehicle

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 seven attributes from each line of the file and create seven distinct Python list objects comprised solely of the values you extracted for a given attribute. In other words, you should have lists of prices, maintenance costs, number of doors, etc. For example, the first five "luggage" values should be as follows:

   ['med', 'small', 'big', 'big', 'med']

   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 seven required list objects.

2. **(15 Points)** Your second task is to find the list index values of each automobile having a “**price**” rating of "**med**". Create a new list object with your result. ***HINT:*** *you can accomplish this task by searching the list of price values you created earlier*. Be sure to print your results.

3. **(15 Points)** Your third task is to find the "**number of passengers**" value for each auto having a "**price**" value of "**med**". 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 automobile having a “**price**” value of "**high**" and a “**maintenance**” value that ***is not*** "**low**". 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 **cars-sample35.txt** data.

5. **(20 Points)** Create a new Python class to store and manage the information contained within the **cars-sample35.t**xt file. Your new class should be named “CarData” and should **include a class variable** ‘CarCount’ that represents a counter indicative of the number of CarData objects that have been instantiated within your current Python session. The class must also contain **eight instance variables** including a unique ID for each vehicle (use the name ‘uniqueID’ for this instance variable) and seven additional instance variables corresponding to the seven different vehicle attributes contained within the **cars-sample35.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 “CarCount” class variable. The new instance should then increment the ‘CarCount’ class variable by 1 (after assigning its previous value to the ‘uniqueID’ instance variable as mentioned above.

6. **(20 Points)** Read the cars-sample35.txt file again and store its content into a list of CarData class objects, with each row within the file being loaded into a distinct CarData class object within that list. If you have defined your CarData 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 vehicle 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 CarData class when you read the corresponding row from the data file into the class object. Then, using your list of CarData objects, find the uniqueID value for each auto having 2 doors and a **luggage** value of "**big**". Create a new list to store your findings and be sure to print your results.

Be sure to include some commentary 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_).