CSCI 1133, Spring 2015 | Assigned: Friday April 17, 2015
Programming Assignment 11 | Due: 11:55pm Thursday April 23, 2015

This is ***not*** a collaborative assignment; you must design, implement and test the solution(s) on your own. You may not consult or work with anyone other than the course instructor or TAs. In addition, you may not include solutions or portions of solutions obtained from any source other than those provided in class. Obtaining or *providing* solutions to any homework problems for this class is considered academic misconduct. If you are not sure what this means, consult the class syllabus or discuss it with the course instructor.

This assignment requires writing a single Python module that must be submitted online *prior* to the due date/time. Late submissions will not be accepted. Name your module: `hw11.py` and submit it using the appropriate homework submission link on the Moodle website.

The total point value for programming assignments will be awarded for solutions that are *complete*, *correct*, and *well constructed*. A *"well constructed"* program entails good design, appropriate comments and general readability (descriptive names for variables and procedures, appropriate use of blank space, etc.). The following will result in a score reduction equal to a percentage of the total possible points:

- Incorrectly named/submitted source file (10%)
- Constraints not followed (40%)
- Failure to execute due to syntax errors (30%)

Note that your work will be graded using, and must function correctly with, the current version of Python 3 on CSE Labs UNIX machines. If you complete your programming assignment using a different system, it is your responsibility to ensure your programs work on CSELabs machines *prior* to submitting them.

A. (40 *points*) **Inheritance**

This is a multi-part problem in which you will construct several object classes and a short demonstration program[1]. Submit the entire program as your solution. Be sure to thoroughly test each part before moving on to the next.

A used car dealership maintains an inventory of several types and models of vehicles. There are three kinds of vehicles: Cars, Trucks and SUVs. Regardless the type, the dealership maintains the following information for every vehicle:

- Make
- Model
- Year
- Mileage
- Price

Additional information is maintained for each individual vehicle depending on its type.

For Cars:	Number of doors (2 or 4)
For Trucks:	Drive type (2-wheel drive or 4-wheel drive)
For SUVs:	Passenger Capacity

Part 1

Construct a base class named `Vehicle` to maintain the common vehicle data. The class should include a constructor that will initialize all 5 instance variables and separate accessor and mutator methods for each data attribute.

---

[1] *adapted from T. Gaddis, Starting out With Python, 2nd ed.*

Part 2

Construct three additional classes named `Car`, `Truck` and `SUV` to represent Cars, Trucks, and SUVs respectively. Each of these classes should be derived from the `Vehicle` class and extend it by adding the attributes unique to the type of vehicle. Each class should provide a constructor to initialize its attribute(s) as well as the attributes of the parent class. Provide accessor and mutator methods for each class to get and set the attributes for the particular class.

Part 3

Add a method named `Display` to each of the four classes to print out the individual vehicle information. For the `Vehicle` class, the `Display` method should print the following (one element per line):

```
Make:  vehicle_make
Year:  vehicle_year
Model: vehicle_model
Miles: vehicle_mileage
Price: vechicle_price
```

Each vehicle-type class (Car, Truck, SUV) should print out the information specific to its own type in addition to the vehicle information (Hint: use the superclass' `Display` method). Here is an example of the output for a `Car`:

```
Inventory unit: Car
Make:  Audi
Year:  2009
Model: A8
Miles: 40000
Price: 27300.00
Number of doors: 4
```

Part 4

Write another class named `Inventory` that will maintain a list of vehicles in inventory. The constructor should start with a Null list. Add the following methods to the `Inventory` class:

- `addVehicle(`*vehicle*`)` : A mutator that will add the *vehicle* object to the inventory list
- `Display( )` : An accessor that will print out the vehicle information for every vehicle in the inventory. Separate each vehicle's information with two blank lines

Part 5

Finally, write a function named `main` that will solicit inventory information from the console. The program should prompt the user to enter a vehicle type (car, truck or SUV) and then input all the information appropriate to that vehicle type. It should then construct an appropriate vehicle instance and add it to the inventory. This process should continue until the user indicates he/she is done entering vehicle data. The program should then print out the entire inventory by calling the inventory `Display` method.