Name: Jocelyn Lum
Date: February 26, 2024
Course: IT FDN 110 B Wi 24: Foundations Of Programming: Python
GitHub URL: https://github.com/jlum2022/IntroToProg-Python-Mod07/ **(updated 3-18-2024)**

Assignment 07 – Classes and Objects

Introduction

In this module, we learned how to create and use classes to manage data.

A few takeaways from A07

- A class is a template or blueprint; objects are instances of the class. Classes which inherit from another class can add more properties and methods not found in the parent class. My simple analogy is this: A parent class is like the base model of a car with only the basic features. Classes which inherit from the base class are like the same model car with different trim packages. The child/derived classes "inherit" all the parent's properties and methods, and can be extended beyond the parent template, just as a higher model of the car will add or override features in the base model car such as upgraded upholstery, moon roof or a nicer sound system. There is much more to classes than this analogy can illustrate but this is the basic concept.

- For Assignment 07, we demonstrated classes by first creating a Person class which has attributes of first and last name. We showed inheritance by creating a Student class which inherits from Person, and extends it with an additional property for the student's course name.

- The concepts of constructors, "self", inheritance, properties, methods, getters and setters were familiar to me, but the Python syntax was new and different. For example, I learned that the class attributes are defined within the __init__ constructor. At first glance it looked like the variable declarations were missing from the constructor but that is not the case. Without the explicit declarations, the resulting code is more compact and flexible.

- Along with other dunder methods that were mentioned, the ability to define a custom **__str__()** method is very cool. This method returns a human-friendly string representation of the object that you can easily print. This is very helpful for providing an easy view of the composition of an object.

- I love the ability to define docstrings – how convenient and helpful it is to see a descriptive summary of a function pop up while debugging.

- It's best to name variables with informative names in order to reduce confusion between similar names. This will be helpful for anyone who reads your code in the future or even for yourself. For example, one of the error messages I got was: *'Student' object has no attribute 'append'*. The bug was that I was attempting to call **append()** on *student* which is a Student object, rather than on *student_data* which is the list. There are a lot of occurrences of "student" in this program so if I recreated this program for myself, it would be helpful to rename *student_data* to *student_data_table* or rename *student* to *student_obj*.

- I debugged this assignment using both PyCharm and Visual Studio Code. Both are good development tools but I found PyCharm to be more intuitive. I appreciate the advice we were given in class to learn to use more than one IDE. It has been good practice to see different types of error messages and track down the cause of the error.

Summary

Creating classes helps us to develop code that is more modular, reusable, and maintainable. The Python code for creating and managing classes took some getting used to, but I became more comfortable with it by the end of the assignment. I'm sure I will be coming back to the notes in this assignment for future Python development.