Noah Cooper
9/9/24
IT FDN 110 B Su 24: Foundations Of Programming: Python
Assignment 07
https://github.com/DrBones2/IntroToProg-Python

Assignment 07 - Classes and Objects

Intro

This module is focused on the concepts of classes and objects and how to use these to organize code. In addition, we learned about constructors, as well as how to utilize data validation and inherited code.

Learning Fundamentals

The first topic of this module was examining the structure of functions and classes. Classes are used to organize code, and to give functions their own specific space in memory, which helps prevent conflicts with two different statements writing to the same variables. Having different classes for different tasks is useful to understand what the goal of a function is. For example, a script may contain classes for FileProcessor and IO, which handle file reading/writing and user input respectively.

Constructors are functions that are called at the start of a class that helps define default values for the variables used. If no constructors are defined, Python will automatically create an invisible default constructor with no values. It is wise to set default values for our classes to prevent errors if the variables are accidentally called sooner than intended. The “self” keyword allows us to easily assign these values, and to have them loaded alongside the classes when they are loaded into memory first.

Data validation is used to further prevent unnecessary errors. One way to help with this task is by utilizing private attributes. This can be done by using two underscores before an attribute to mark it as private. This signals that the attribute should not be modified outside of the object’s class.

Inherited code refers to when a child object derives its behavior and data from its parent class. This is useful in an object-oriented programming environment to make quick new iterations based on a pre-existing class.

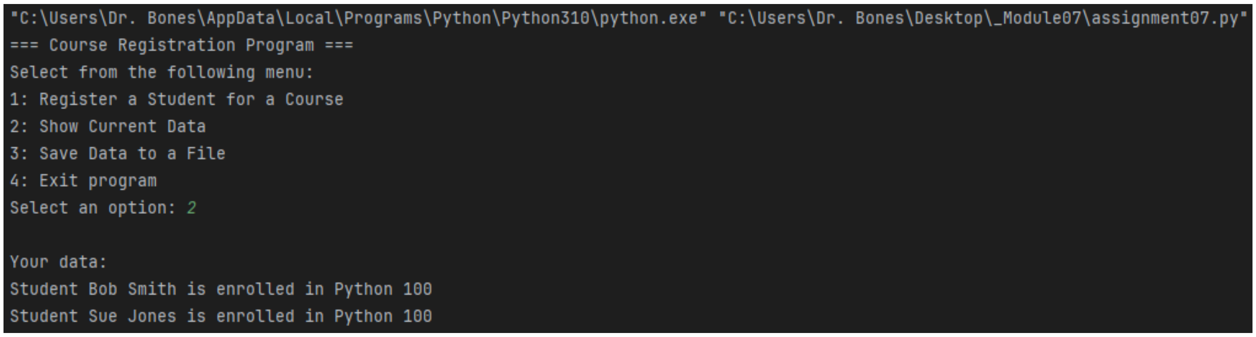
Assignment

Building on the base of previous assignments, this module’s assignment sees improvements to the functions that define the student’s name and course name.

Two new classes are created, Student and Person. The Student class contains functions that will help validate the values input by the user, and ensure that there are no invalid values being input. For example, if the user attempts to input a blank value for the course name, then the function will result in an error message being displayed back to the user, specifying they cannot enter a blank value.

The functions that stored the inputted data have been updated to use these functions from these new classes. This allows us to seamlessly integrate data validation into our later functions without needing to have data validation in each area the variables are used.

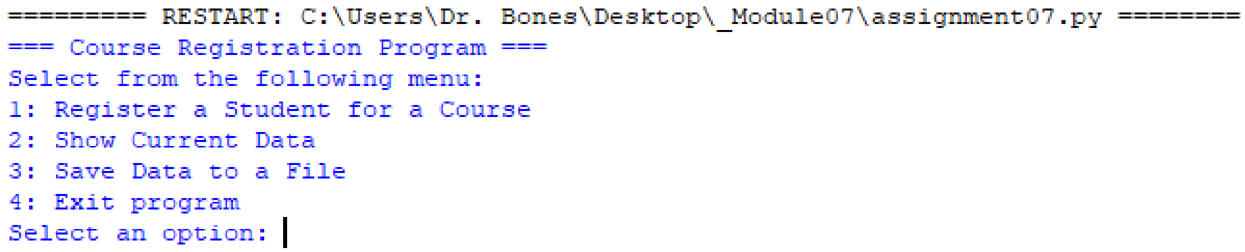
Lastly, the formatting of the script was improved to help readability when outputting information to the user.



```
"C:\Users\Dr. Bones\AppData\Local\Programs\Python\Python310\python.exe" "C:\Users\Dr. Bones\Desktop\_Module07\assignment07.py"
=== Course Registration Program ===
Select from the following menu:
1: Register a Student for a Course
2: Show Current Data
3: Save Data to a File
4: Exit program
Select an option: 2

Your data:
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
```

Figure 1: Script running in PyCharm



```
========= RESTART: C:\Users\Dr. Bones\Desktop\_Module07\assignment07.py ========
=== Course Registration Program ===
Select from the following menu:
1: Register a Student for a Course
2: Show Current Data
3: Save Data to a File
4: Exit program
Select an option: 
```

Figure 2: Script running in IDLE.

Conclusion

This module helped prepare us for being able to use classes effectively, and taught us the techniques as to how developers sort their functions effectively.