Paula Shoup

November 27, 2019

Foundations of Programming: Python

Assignment 08

https://github.com/eldonsdata/IntroToProg-Python-Mod08

Classes & Objects

Introduction

This week the course reviewed classes and objects, specifically what they are, and how to properly use them in Python. Digitalocean.com defines classes and objects as:

Class — A blueprint created by a programmer for an object. This defines a set of attributes that will characterize any object that is instantiated from this class.

Object — An instance of a class. This is the realized version of the class, where the class is manifested in the program.[1]

Python is an object-oriented language which relies on the class-object structure for programmers to create template-based code that is easier to reuse, keep organized, and maintain. These facets save time for programmers not only in crafting code, but in debugging.

Code

To practice with using classes and object, we used a similar script theme that has been used throughout the course which involves providing a menu of options to the user that allows for information display, adding data, saving data, and exiting the program. This particular script prompts the user to enter a product and associated price.

The program is broken into two parts: classes and the main script to call the functions within the classes. The classes are broken into section based on attribute. The first section manages the variables of product and price, the second section processes the data by writing user input to file, and the last section handles input/output by displaying the option menu, receiving user choice, user input on product/price, and displays data.

Each class is constructed by providing a name for the class such as “IO” (for input/output), and constructor (def), properties, and methods. Below is an example for a portion of the product class:

```
class Product:
    def __init__(self, product_name: str, product_price: float):
        try:
            self.__product_name = str(product_name)
            self.__product_price = float(product_price)
        except Exception as e:
            raise Exception("Error setting initial values: \n" + str(e))
```

[1] Tagliaferri, Lisa. How to Construct Classes and Define Objects in Python. 17 March 2017. https://www.digitalocean.com/community/tutorials/how-to-construct-classes-and-define-objects-in-python-3

```
    @property
    def product_name(self):
        return str(self.__product_name)
```

After all the classes and functions within the classes were created. Testing was conducted to see if the code worked properly. Below is the program being run in PyCharm.

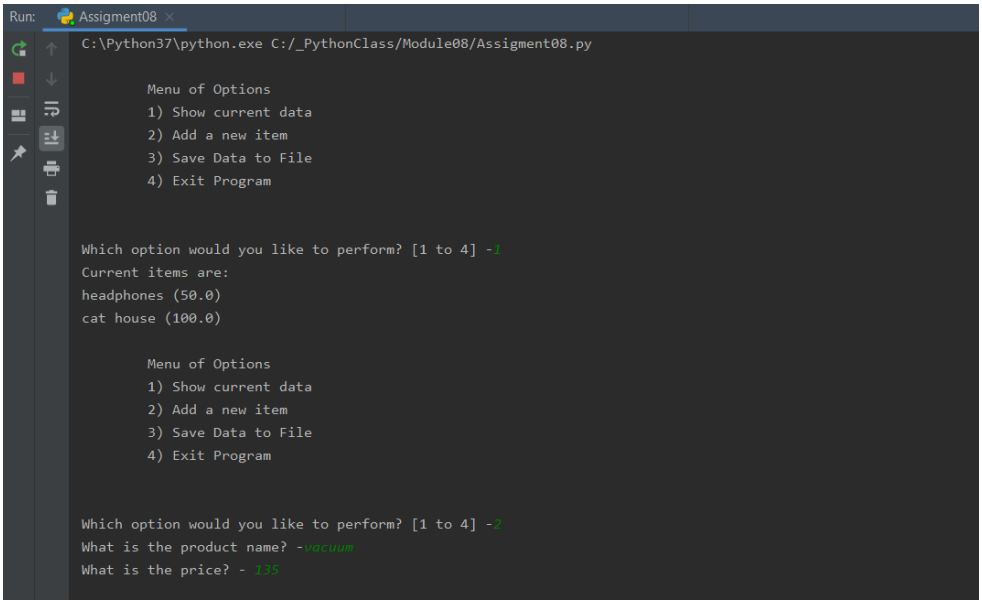

Figure 1. Screenshot of script run in PyCharm

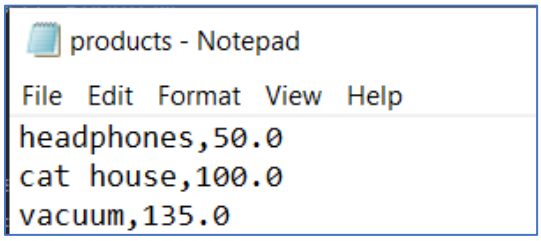

Figure 2. Screenshot of program output saved to the “products.txt” file.

After drafting the code, I installed Desktop Github. The desktop interface basically allows a developer to push documents directly from their computer to Github, thereby bypassing the need to login on the web. Once installation was complete, I created a new repository for Mod08, uploaded my script, committed it to the master, published it to my repository, and confirmed that it was showing up on the web.

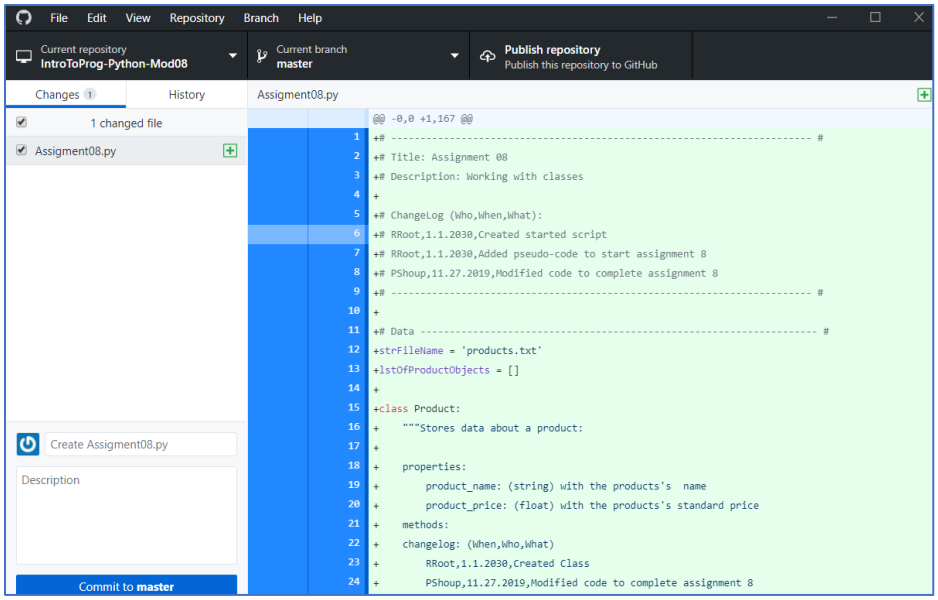

Figure 3. Screenshot of adding a file to Desktop Github.

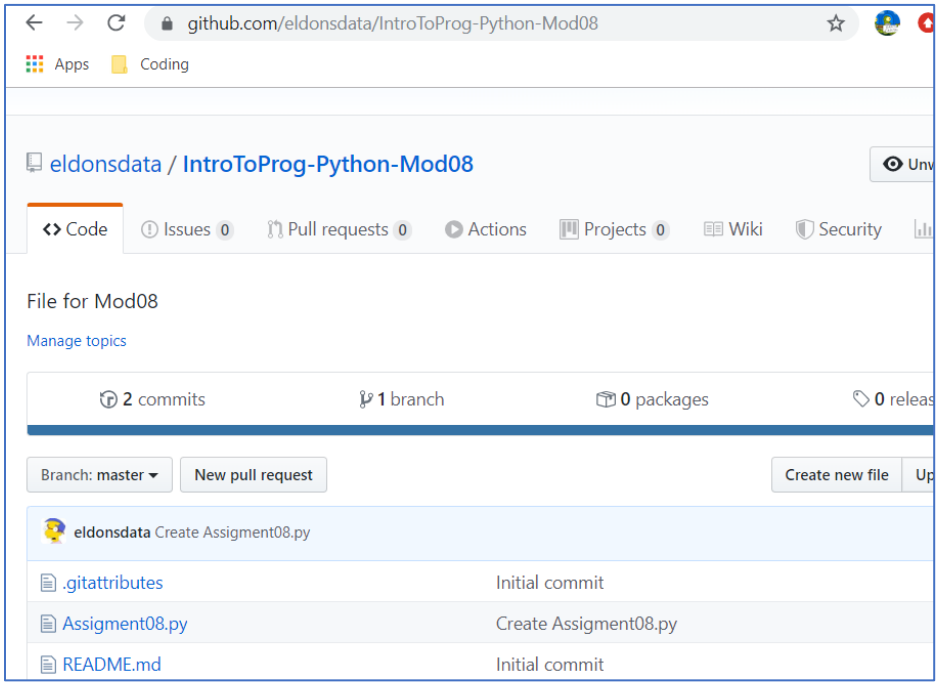

Figure 4. Screenshot of saved repository to Github.

Summary

Classes are used to organize data processes carried out by variables and functions. Objects are used to access the processes within classes. Python is an object-oriented programming language which relies on this class-object structure to create reusable, readable, and maintainable code. As the foundation for Python's structure, understanding classes and objects is a fundamental part of gaining an understanding of how to utilize this coding language.