Elisabeth Chapman

3/8/2021

Foundations of Programming Python

GitHub: https://github.com/echapman-2021/IntroToProg-Python-Mod08

Assignment 08

Classes and Object-Oriented Programming

In this project the concept of object-oriented programming was explored and a new vocabulary to describe the script was utilized. This concept again used classes to construct separate segments of the code, but the organization of the classes differed from those of our previous project. Elsewhere within the code, a few familiar concepts continued to be deployed, the use of docstrings continued to be used to describe the functions intended purpose, as well static methods

# Constructor and Field

**Figure 1:** A closer look at the init

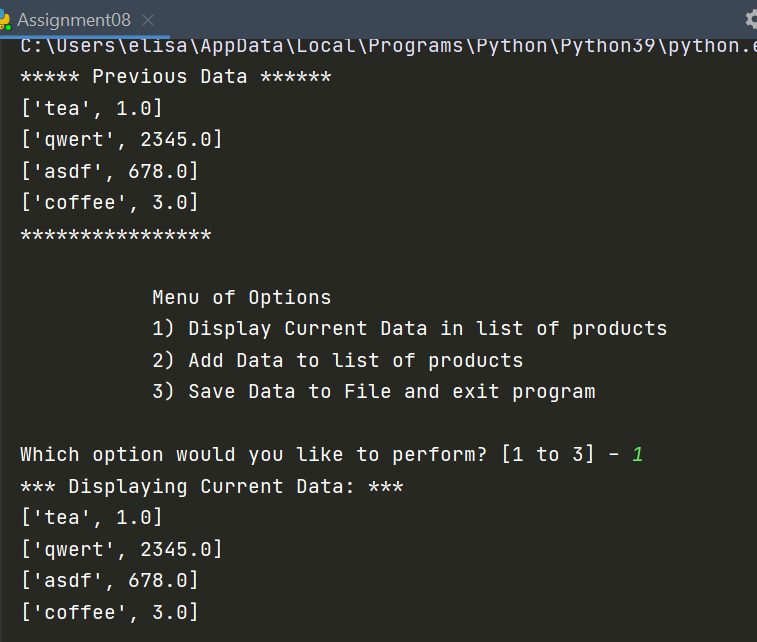
As seen in the code above, this script is different in a few ways from that of previous scripts. The variables of this class are now referred to as fields and the entire class shares its own encapsulated fields—which functions as an object class-specific global variable. Following the fields, is the class Constructor method, which is recognized by \_\_init\_\_ and the parameter self. The special method \_\_init\_\_ is a type of dunder method and stands for ‘initialize’—as in to initialize the Constructor. Moreover, constructors as a method are “used to initialize the object’s state”(geeksforgeeks.org). Each statement within the Constructor is activated by the creation of the object, meaning that anything within this method is automatically in effect as soon as the script runs. Slightly more ambiguous—“the keyword self represents the instance of a class and binds the attributes with the given arguments” (geeksforgeeks.com), thus any “instance method […] must have a [..] parameter called self by convention (Dawson, 221).

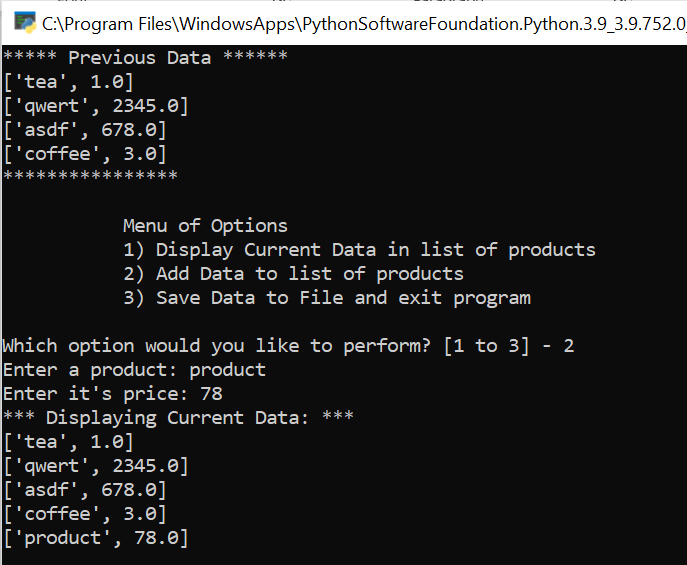
# Attributes

After the instance is established, you “access their instance attributes using **dot notation**” (Amos, realpython.com) which in this code was exemplified by self.StrProduct, which took advantage of both fields and the keyword self. Luckily, for the sake of the program, attributes are also mutable, so whatever attribute the object is given does not need to be considered permanent. Attributes, however, need to be accessed, and while they may be accessed outside of their class “usually, you want to avoid directly accessing an object’s attributes outside of the class definition” (Dawson,228).

# Conclusion

In Conclusion, the code runs because each class is called between the other classes. The object class Product provides information about the products being entered, the Processing class contains the operation-based code—i.e., the actions being done—and the IO class collects input and outputs the desired data. Although the object class introduces a new vocabulary and concept, in action it behaves similarly to older scripts because ultimately the program still functions by collecting data, storing its information, and outputting the data when asked, only now with greater efficiency and potential for reuse.

**Figure 2:** The code as it executes in PyCharm

**Figure 3:** The code executing in Command Prompt

**References**

* Dawson, Michael, *Python Programming*, Third Edition, Course Technology, a part of Congage Learning, 2010
* nikhilaggarwal3, *\_\_init\_\_ in Python*, geeksforgeeks.com, Nov 26 2019, <https://www.geeksforgeeks.org/__init__-in-python/>, Accessed Mar 7 2021
* David Amos, *Object-Oriented Programming (OOP) in Python 3*,realpython.com,Jul 06, 2020,<https://realpython.com/python3-object-oriented-programming/>, Accessed Mar 8 2021