Rudolph Carvajal

September 07, 2020

Foundations of Programming: Python

Assignment\_08

CD Inventory: Classes and Objects

# Introduction

In this document I will discuss what I learned over the past couple of weeks being introduced to object oriented programming. I will briefly discuss the topics that I learned, acknowledge what I found challenging and conclude by going over the changes I made to the CD Inventory program that to incorporate Module 8 learnings.

# Classes

A class is like a blueprint for creating objects. Two things stuck with me over the week when learning about classes. Dirk’s example of a house blueprint – where a class is like a blueprint when a carpenter is building track homes. A blueprint can be used many times to build something which is similar to class. Also, I watched a video by CS Dojo on YouTube where the creator discussed building a website with robots that greet you – a class was used to build the robot objects, which had different characteristics but the same functionality (e.g. greeting you) and attributes (they each had a color, weight and name).

One of the main benefits that I can see with using classes is the reusability of the code. Going back to the robot example, if you wanted to build an app with robots that greeted users, then you can use the robot class without any additional coding. Additionally, classes help keep code organized and easy to read – keeping all the data members and methods in one place.

# Methods

Methods are similar to functions except they are associated with a class. Methods allow the programmer to define the functionality of the class. What allowed me to understand the concept of methods a bit better was to think of Lists and their related methods such as append, sort, extend, etc. The ability to append to a list is an example of using a method to do that.

The one thing that gave me trouble with methods and overall with classes and objects was the concept of *self*. The *self* keyword (not necessarily a keyword in Python but a convention) is used to to represent an instance of a given class. By including it in the method, it allows a class to hold multiple information for multiple objects by passing the object itself as the first argument to the corresponding function – the calling process is automatic while the receiving process is not which is why the parameter self is needed.

Private methods were also difficult for me to understand at first. I think it is difficult because so far in this class I have been the only person (prior to grading) using my program so it’s hard for me to understand why a private method is necessary. The best analogy I came across was comparing a private method to a car engine – creating private methods enable the programmer to only expose the items that warrant exposing vs showing the complexities under the hood (e.g. showing the steering wheel vs showing the machinery that makes the car turn).

# Properties

The getter and setter properties in Python allow access to instance attributes exactly as if they were public attributes but they act as “intermediaries” so that the data is not directly accessed. The class book used the example of a bank account as an object where you wouldn’t want another program to directly access a user’s account balance when performing a withdrawal. Instead you would want to have the account balance be a private attribute and create include rules behind the scenes when a withdrawal is being request. For example, if the withdrawal brought the user’s account balance to negative then the withdrawal would be canceled and the account balance would not be updated.

# The CDInventory program revisited

Based on this week’s learning, I made the following revisions to my CDInventory program.

* I created the class CD and changed the data store to be a list of CD objects
* I got rid of the add\_cd function as that was no longer needed as I added code to the main body that appended CD objects to a list
* I deleted the functionality to delete a CD as I have not learned to delete instantiated objects – I assume this was why the pseudocode did not include this

Below are snapshots of my program running in Spyder and the terminal window:

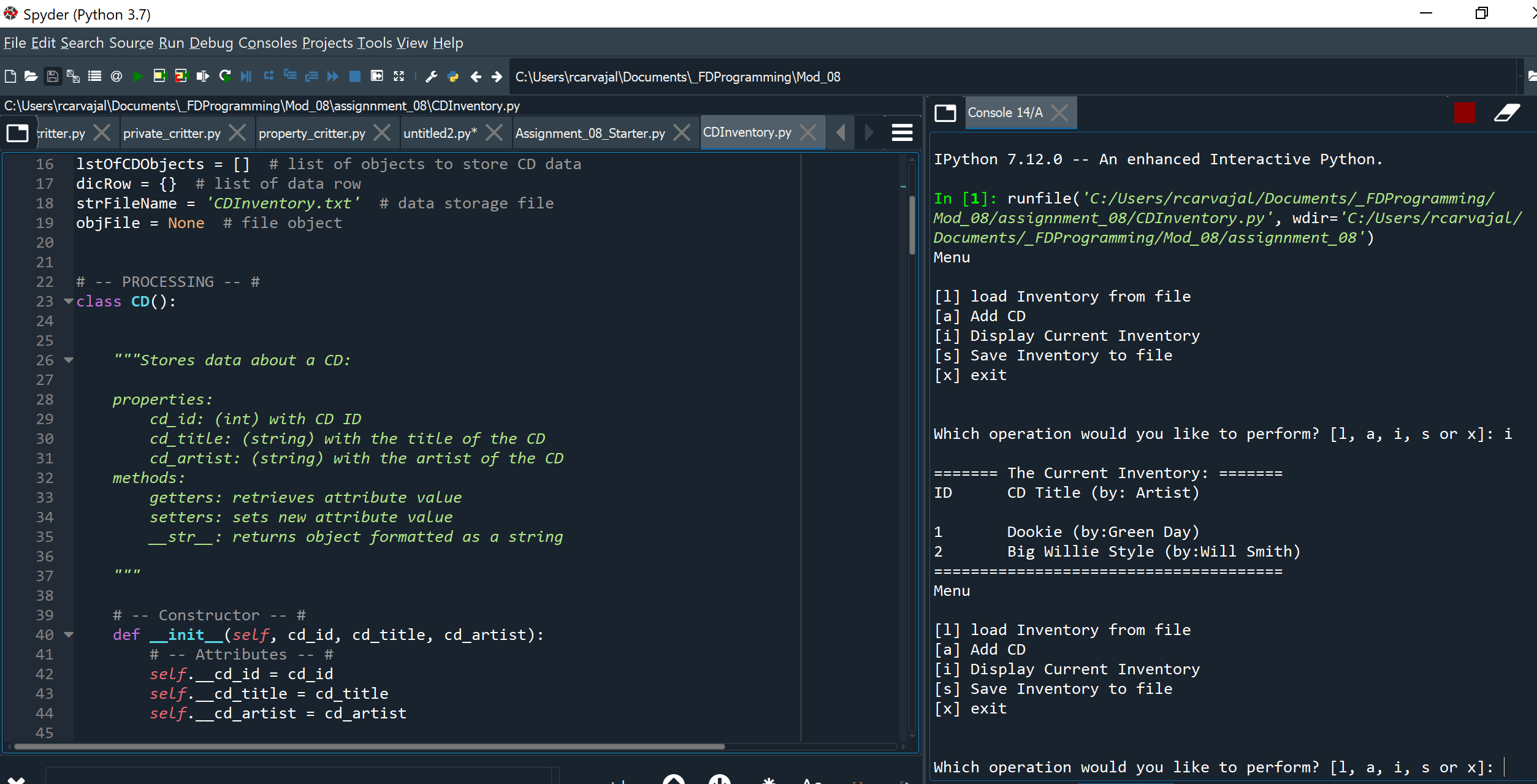


Figure - CDInventory program with list of CD Objects running in Spyder

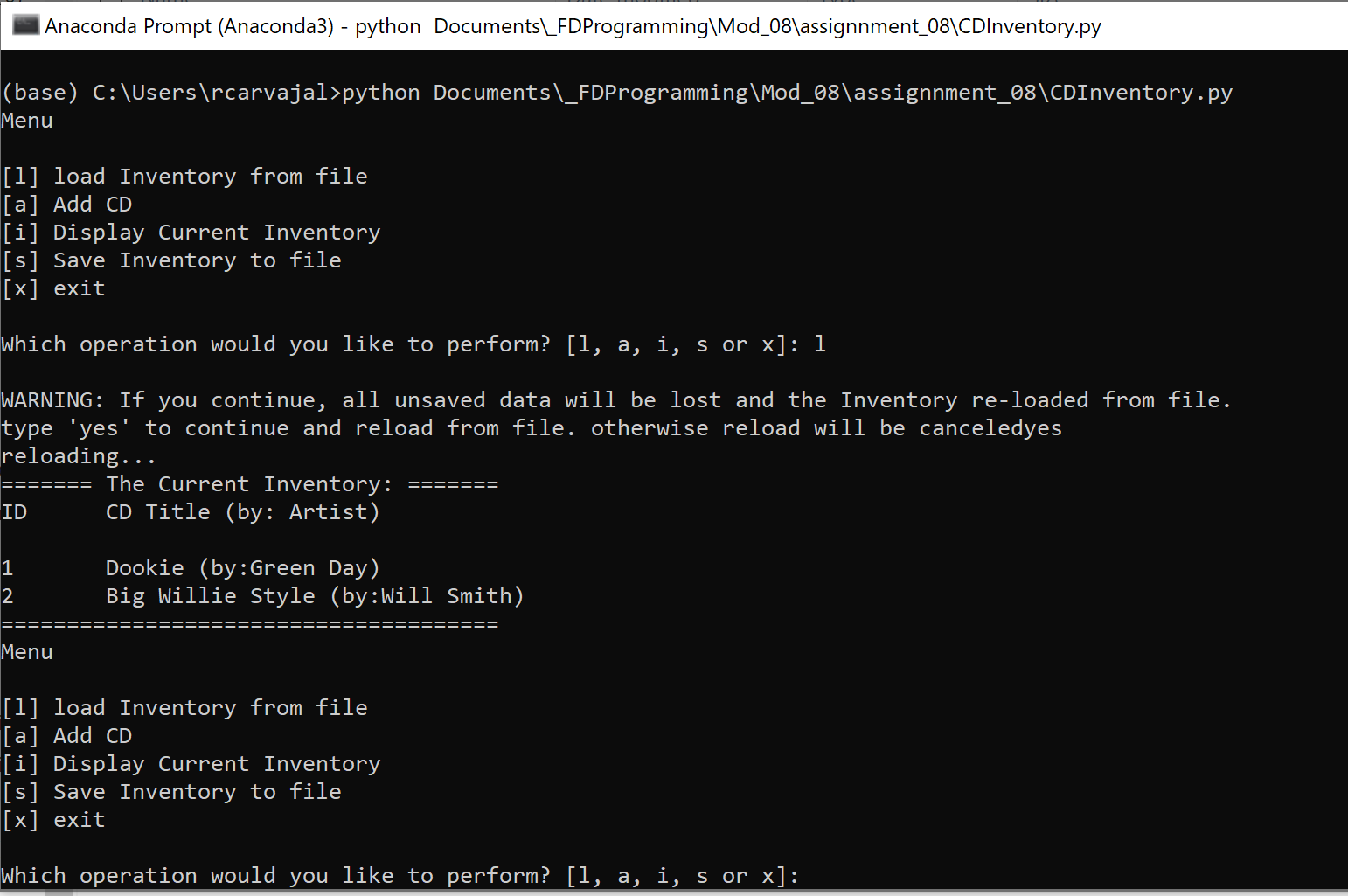


Figure - CDInventory program running in the terminal window

# Summary

This was the most challenging week of the class so far because I feel this was the first time brand new concepts were introduced. While I have never programmed in Python before, it was easier to understand variables, functions, for loops, etc because I have seen them before at work. The code wasn’t too challenging but understanding how and when to use Classes and Objects is still going to take some work though I am making progress. Even though this was a tough week, this concept really has drawn my interest and I plan to research this topic thoroughly. My code can be seen at my Github: <https://github.com/rudycarvajal/Assignment_08>