Arianna Morgan

December 4, 2022

Foundations of Programming: Python

Assignment 08

Object-Oriented Programming

Introduction

In this module, we learned about object-oriented programming: specifically, we learned about classes and objects, constructors, fields, attributes, and methods. Classes are the high-level construct, which sets out the information that an object holds and its functionality. Objects are instances of the class. Constructors are a method that instantiates an object and defines the attributes of the object, which are the variables that hold data for the object. Fields are data stores within a class. This week, we recreated last week’s script to use a CD class that allows for the creation of a new object for each CD the user wants to add to the inventory.

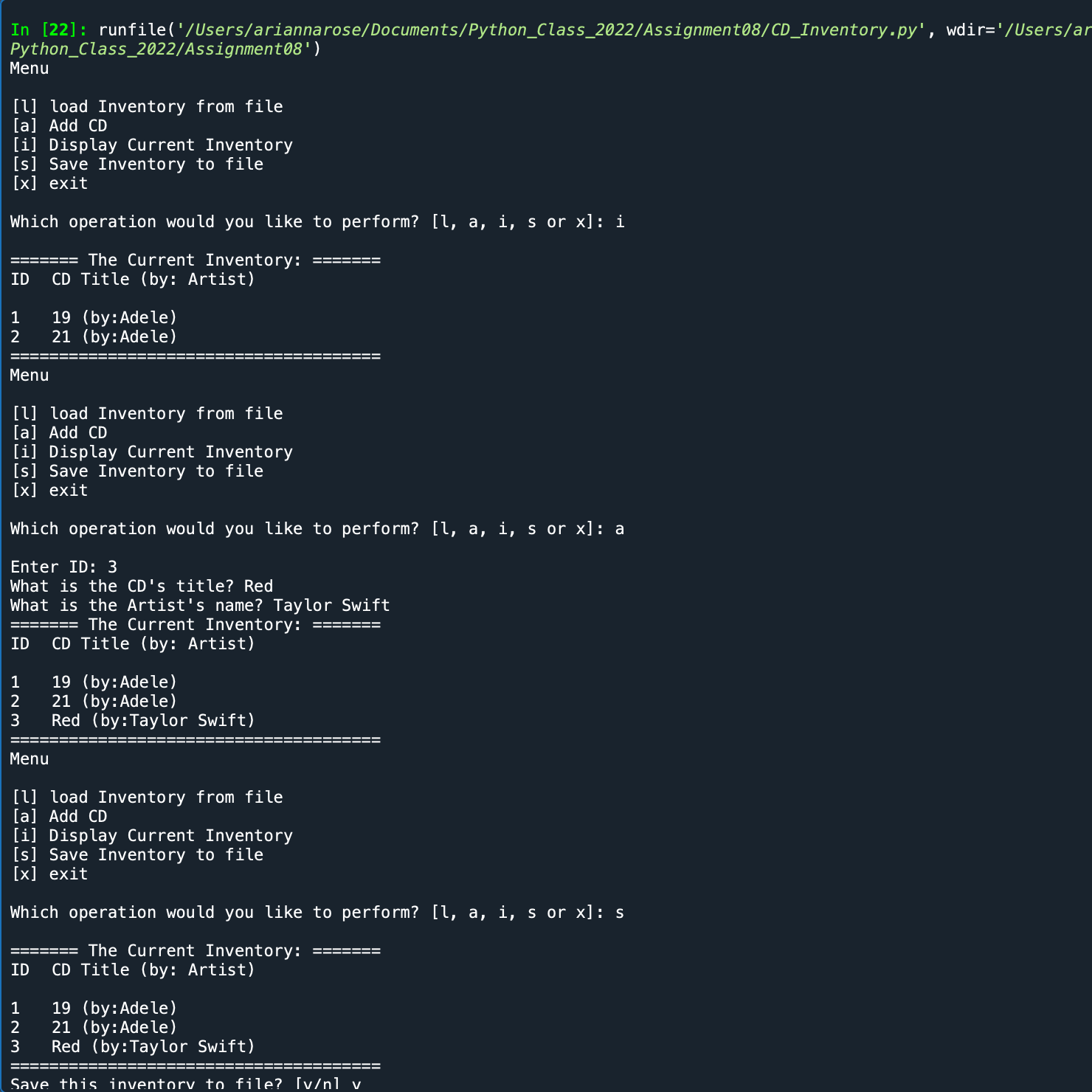
Details

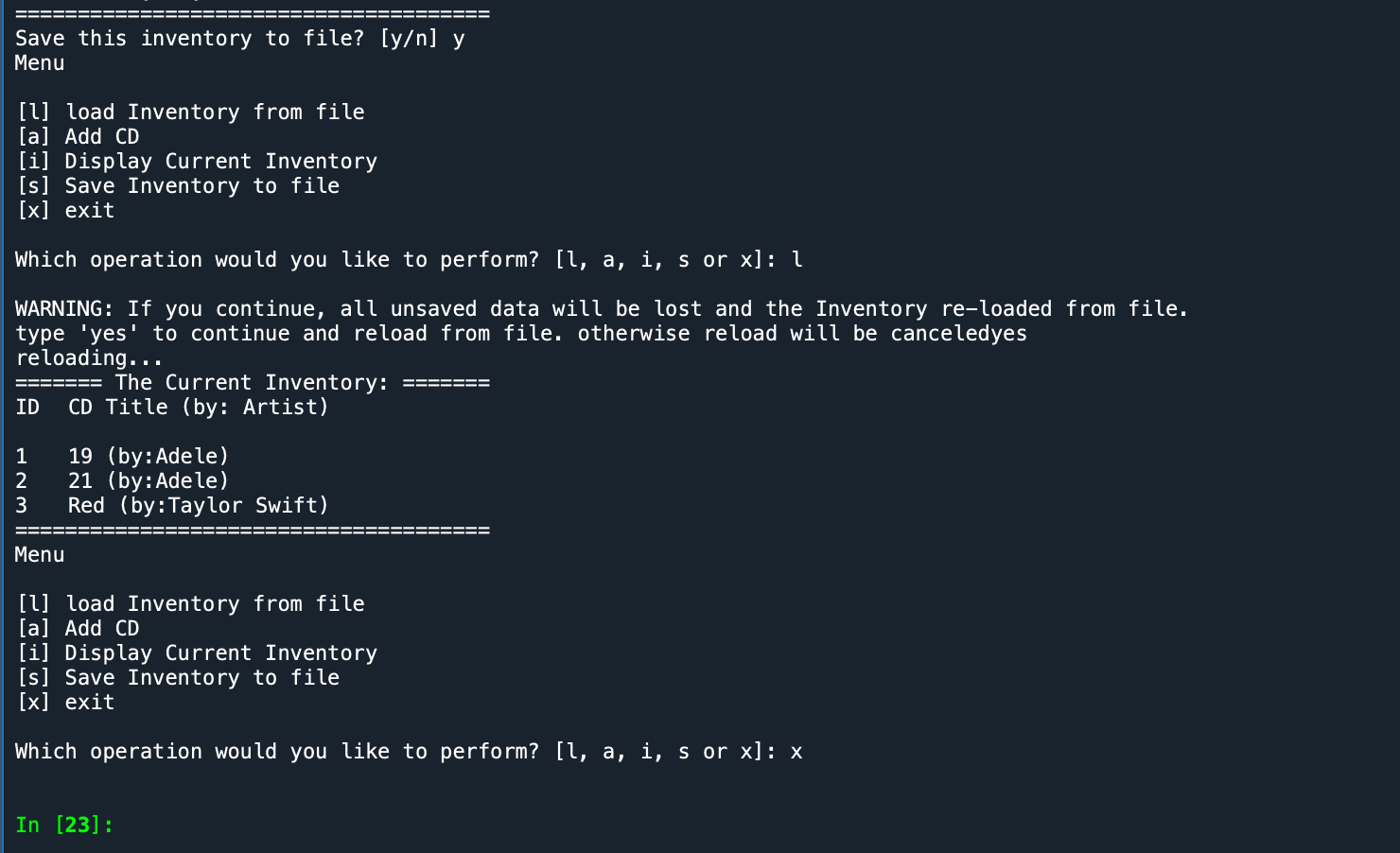
I was able to use much of my script from last week for this assignment (updating class and variable names as needed). The main change is the addition of the class CD. I added the constructor, which has self as its first parameter and then the CD data fields as the next three. I created attributes for these three data fields (ID, title, and artist) and a final attribute of dicRow, which creates a dictionary row for each object that can be added to the CD list in the main code. I was also able to improve the doc string for the IO section to list the methods that I have defined with the IO class.

I updated my add\_inventory\_menu function, where the user provides inputs to add a new CD, to return an object from the CD class instead of the individual pieces of CD data. This allowed for a simple addition of the CD to the CD inventory table in the main code instead of requiring its own function. For the new CD, I appended the dicRow attribute to my CD list. With my load inventory function that I pulled in from last week, I ran into an issue when I tried to add a CD to the list when starting the script without a file to load from. I realized that I was only returning something in that function if it successfully loaded data from the file, so when it didn’t have anything to load, it returned nothing. Since I was assigning the return value to my table variable, I was making it a None type when it failed to load, which then didn’t let me append anything to it since it was no longer a List type. I fixed this by having the load inventory function return an empty list in the exception cases.

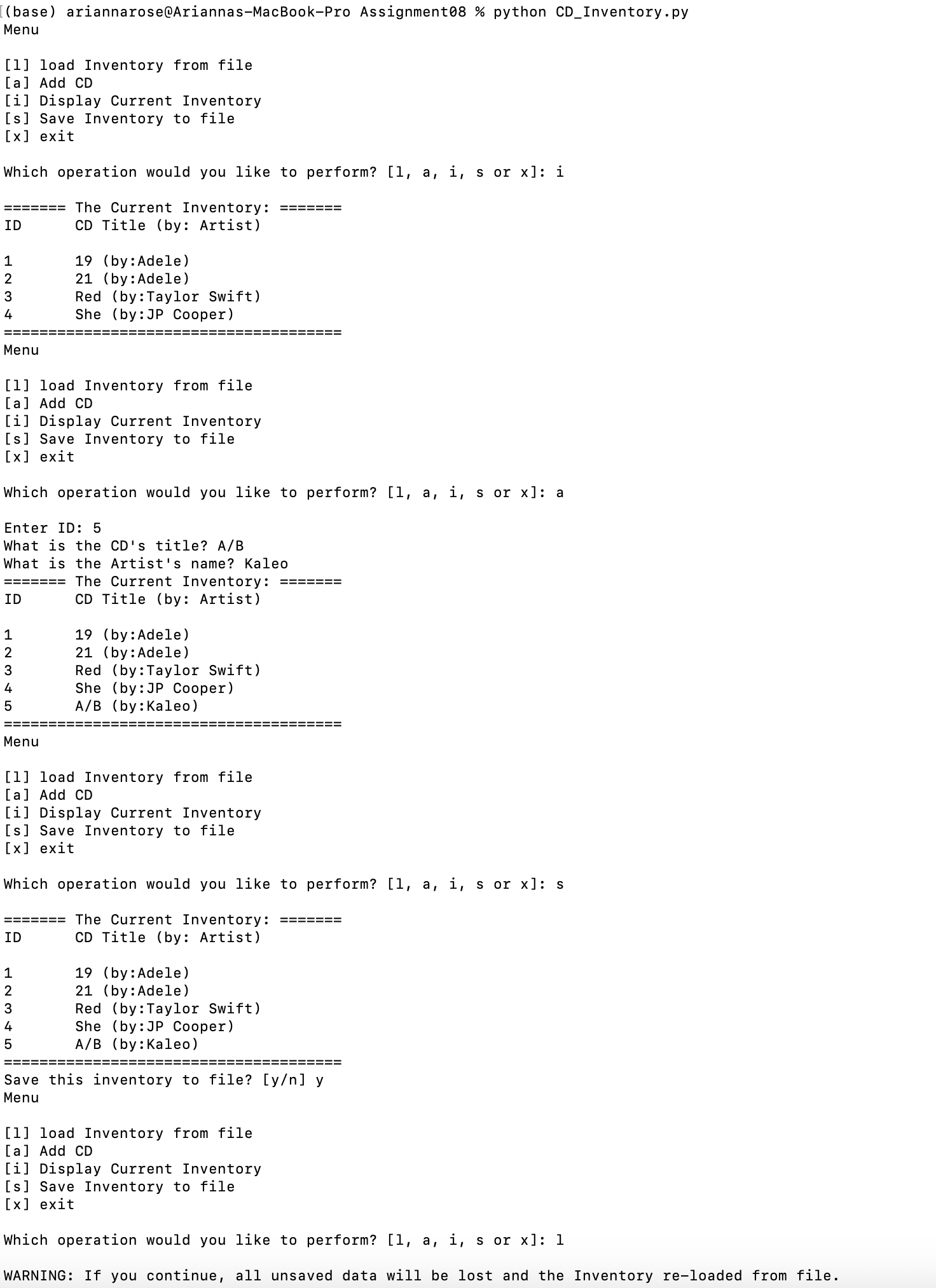
Below are screenshots of my code working:

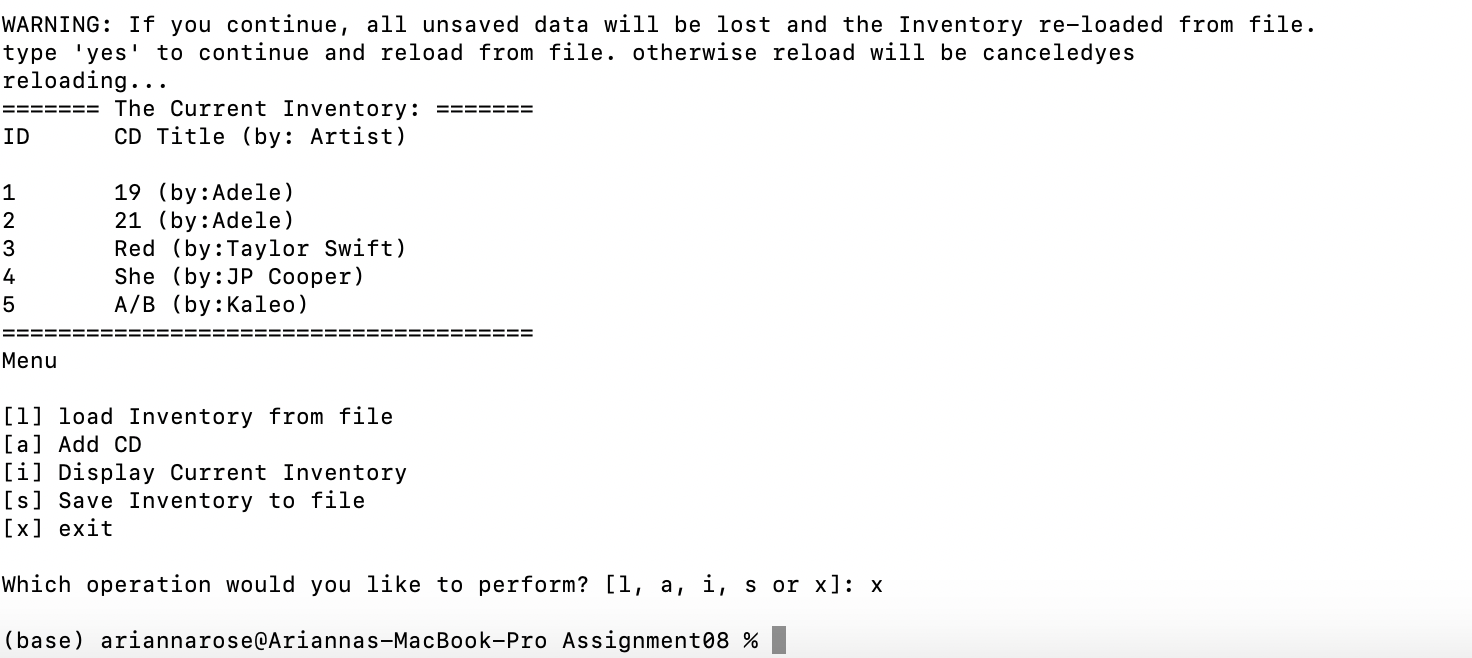
In Spyder:





In Terminal:





Summary

This assignment was lighter lifting on the coding side compared to prior assignments since we were able to leverage a lot of work from prior weeks, but there was a lot of mental processing required to grasp this week’s topic of object-oriented programming. It’s clear we are only scratching the surface of the functionality available to us with these concepts, so I am looking forward to digging into it more. This assignment was helpful in providing a concrete example that was a little more complex and functional than, say, the dog example in this week’s online reading, but still easy to understand.