Nick Culbert

2021 Mar 16

IT FDN 110 A Wi 21

Assignment 09

The Final CD Inventory

# Introduction

In this assignment, I modified Dirk’s existing code to add a new object for a CD’s tracks and allow the program to save to multiple files.

I do not think, however, that I have fully grasped the power or flow of object-oriented programming, and most of the time spent for this assignment was keeping my head from reeling while trying to keep track of all the different modules.

# Research

I did not do any additional research beyond watching the class meetings and studying the recommended materials; since this module focused more on the organizational mindset than on syntax, I did not find it necessary to perform random google searches to fix syntactical issues.

# Method

Before diving into the code writing, I wanted to take a step back and really think about the method I planned to use to tackle the code. In previous assignments, the biggest thing I struggled with once we started getting into object-oriented programming was keeping track of the program’s flow. It felt like I was constantly jumping around the code.

So, I decided to use Dirk’s method of reading down the main file and tackling tasks in a linear fashion. This way, even if I went back-and-forth between the various .py files, the main script would act as a centering foundation. This saved a significant amount of headache, but I still found that the majority of the ‘coding’ time was spent going back-and-forth between all the modules. I think the *object* aspect has just not clicked for me yet.

The remainder of this knowledge document briefly discusses changes/additions made in each class.

# Data Classes

I added the properties as indicated to the Track class and created getter and setter methods for each property. I also added the two methods to return the track’s details in different formats for displaying and saving to file. To do all this, I mostly copied and pasted from the CD class while changing field and method names where appropriate.

I added a property to the CD class which is a list of the CD’s tracks and added a getter method for this property. Since the manipulation of the list of tracks is not done within this class, I did not need to add a setter method.

I then added methods to append a new track to the CD and remove a track. This was very simple since I could just use the append() and del functions, respectively.

# IO Classes

In the save\_inventory method, I added an argument to include the list of tracks that would run throughout the program. This way, I could write both the list of CD objects and list of Track objects to their respective files. This did not seem like the most elegant solution, and I am sure there is a better way to do it, but under the time crunch it is what I could come up with.

I kept the number of arguments in the load\_inventory method to the list of file names. I also added the seek() function as a double-check.

The remainder of the methods in these classes were pre-written.

# Processing Classes

In the select\_cd method, in order to raise the exception that the method’s description indicates, I used a boolean. In the for loop, the row becomes the cd object, which is what allows me to reference its ID property. I struggled in this assignment and the last to wrap my head around referencing properties rather than iterating through a list, since an object’s properties seem like a list to me (I accidentally used the syntax row[0] at first, before remembering this).

In the add\_track method, I added error handling to raise the exception as indicated in the description. Then, I created a new Track object and referenced the add\_track method.

# Running the test harness

Figures 1 and 2 are screenshots of the test harness running in Spyder.

Text

Description automatically generated

Figure - Running the test harness

Text

Description automatically generated

Figure - Running the test harness CONT.