Rudolph Carvajal

August 12, 2020

Foundations of Programming: Python

Assignment\_05

CD Inventory: Lists and Dictionaries

# Introduction

In this document I will discuss what I had to learn this week to re-write the CD Inventory program using Dictionaries instead of Lists. I will discuss Dictionaries and Lists and explain how they are different (and similar). I will also write about using structured error handling so that your program will not shut down when a user enters an incorrect input. Additionally, I will discuss the concept of loading data from a file into a memory. Lastly, I will discuss the most difficult part of writing my script this week which was figuring out how to delete an entire dictionary (or row of a List).

# Dictionaries

A dictionary stores data in pairs using a key:value structure. This structure is very common and used extensively by companies such as Amazon, Google and Twitter. With a dictionary, when you provide a key, you obtain a value back. In Python you create a dictionary using curly brackets {}. However to access data from a dictionary you use [] brackets. For example, if I created a dictionary *channels = {‘ESPN’ : ‘206’}* and I wanted to get the value of ESPN then I would enter channels[ESPN] and get back 206. You can also use get() to return a value and ensure you get some result back even if the key you entered isn’t in the dictionary.

Similar to Lists, Dictionaries are mutable and thus elements within a dictionary can be replaced or deleted. In terms of when to use Dictionaries vs Lists, I found that one should use dictionaries whenever the data has a key-value relationship.

# Structured Error Handling

I found structured error handling very helpful to use in my program. It is frustrating to the user to get kicked out of the program if he/she happens to make a mistake and enter an incorrect value. Error handling can be included in your code to ensure your user inputs a correct value. In my CDInventory script, I include the try:except concept to ensure the user inputs an integer when entering in an ID. I also found it helpful to take notes, documenting what parts of my program were frustrating me when I would test it. This can be done directly within the code.

# Loading Data from a File into Memory

In order to work with data from a file, it must first be loaded into memory. To do this, I used the strip() method which returns a copy of the string by removing both the leading and trailing characters then I needed to create dictionary by indexing the data from the file.

# Deleting an Entire Row (or Dictionary) of List

I found this to be the most difficult part of this week’s programming but it was good to struggle through it order to really grasp the concept. In order to delete a CD from inventory I asked the user to input a CD Title and then I cycled through each row in the CD Inventory List using a *for loop* using the len method to find the number of entries in inventory and then range so the for loop knew how many times to cycle through. I then used an *if* condition to see if the CD was include each row, or dictionary, and thus had to use .values() to check the values of the dictionary. If the CD was included the entire row/dictionary would be deleted using del lstTbl[row].

I also included error handling to let the user know if his/her input was not included in inventory. I did this by comparing the new inventory size to the original size and if they were equal then my program must not have found the CD and deleted a row.

# The CD Inventory Program: Revisited

By beginning with the starter code, I revised the script to include a 2D list of dictionary table. Below are snapshots that first show the data my text file had at the beginning, the script running in spyder then a snapshot of the textfile after.

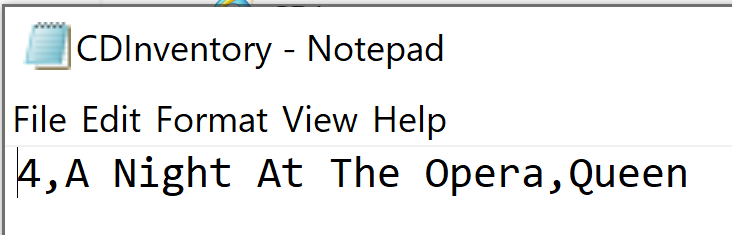


Figure - Text file with initial data

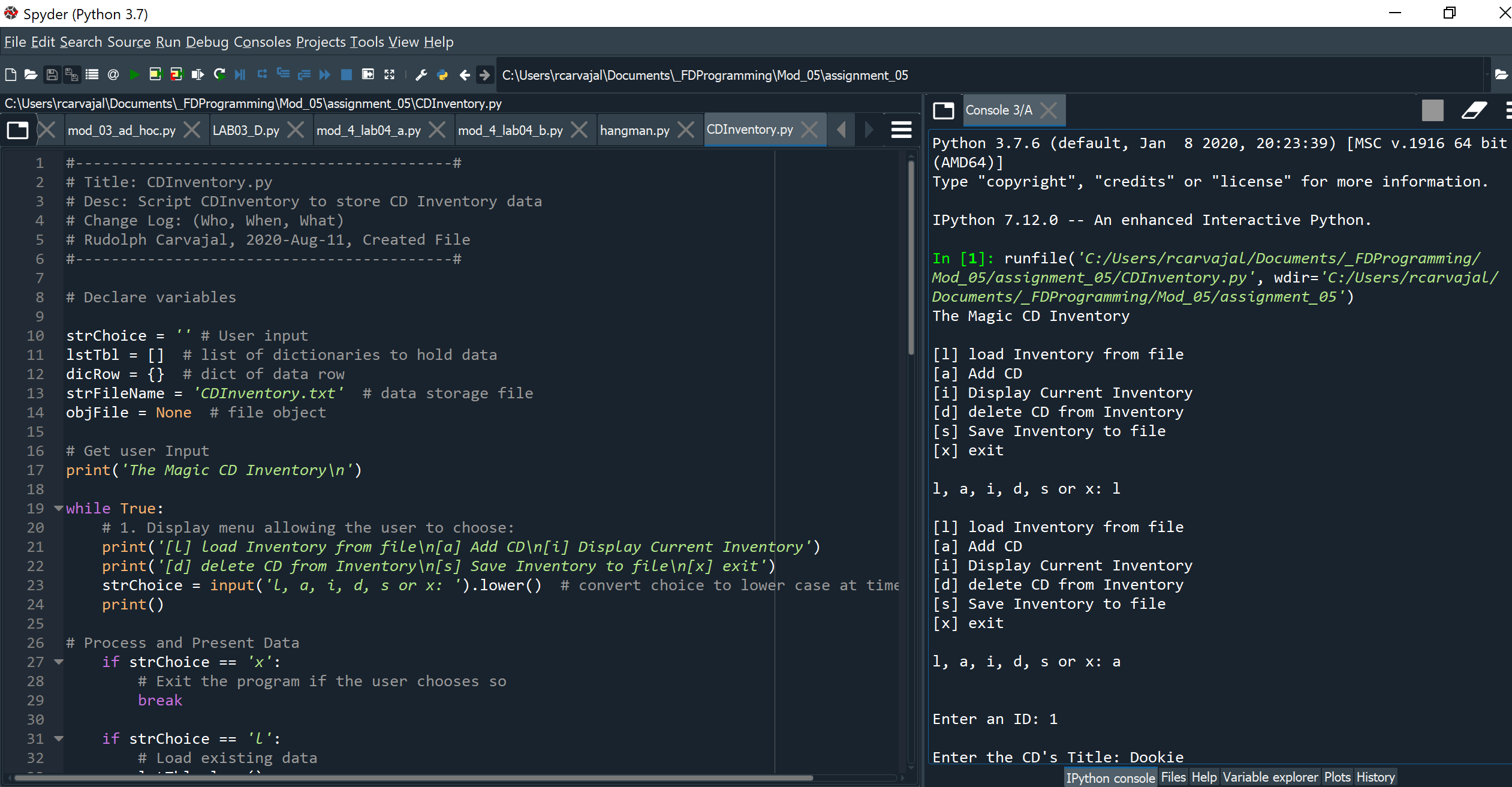


Figure - snapshot #1 of CDInventory script running

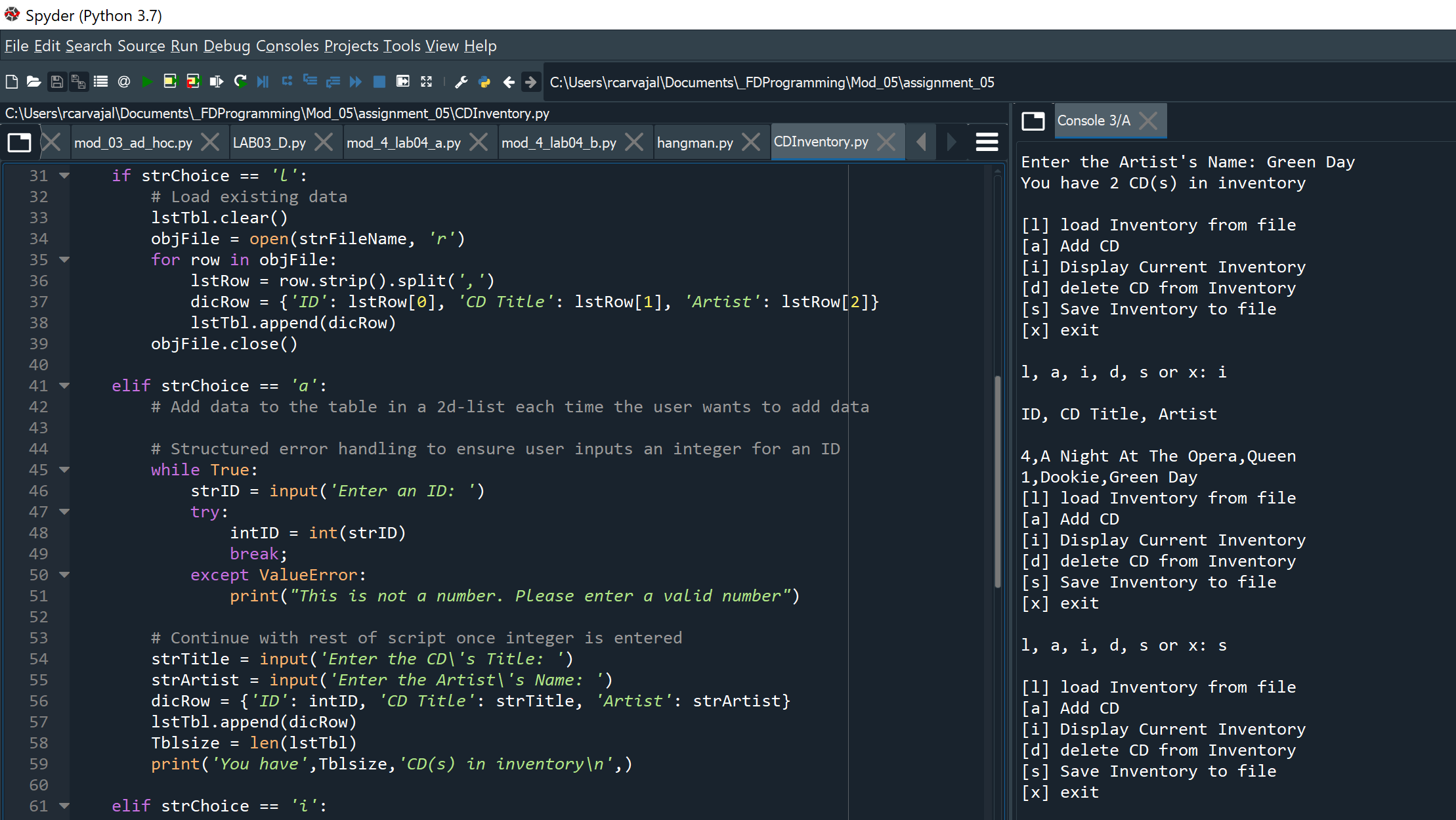


Figure - snapshot #2 of CDInventory script running

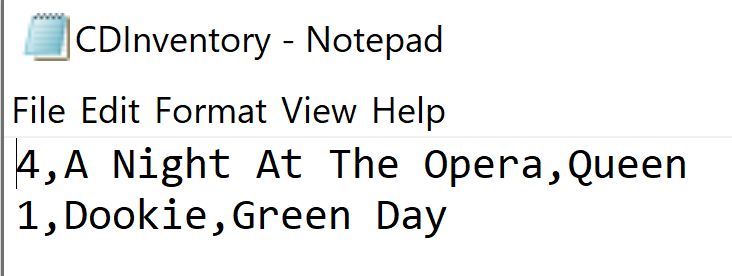


Figure - Text file output after running script

And finally, below is a snapshot from the terminal window with the script running:

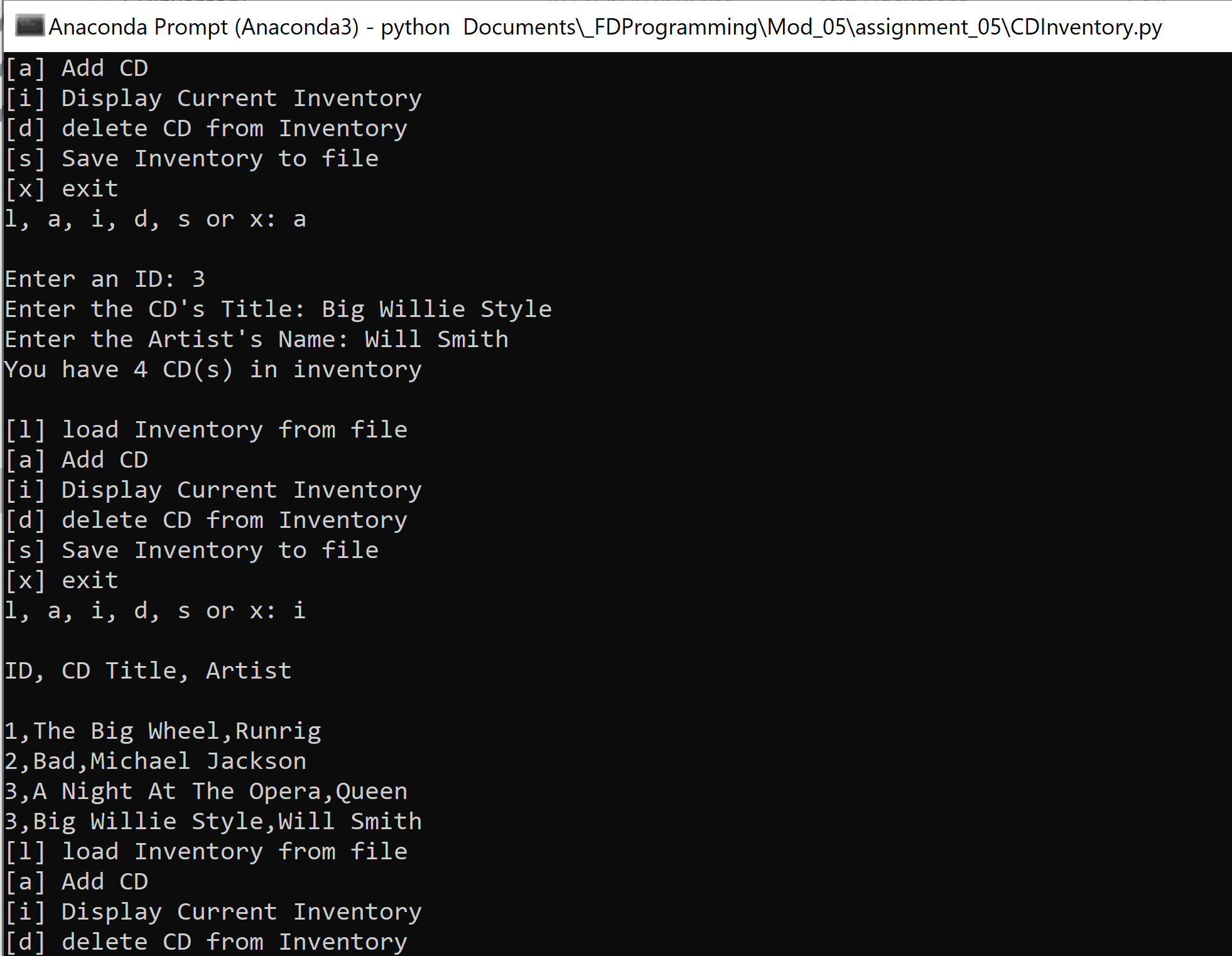


Figure - CDInventory script running in terminal window

# Summary

This week was a lot but it was filled with a lot of learning. I must have ran the code over 50 times as I was writing parts of the script. I have found that when I struggle the most, those are the concepts that stick. It was also helpful to perform a Google search whenever I was stuck. It’s amazing how much documentation is out there on python. Anyways, this week we learned about Dictionaries and how to load data from a file and work with it in memory. I also found reading about separation of concerns in this week’s module to be very helpful given the importance of keeping your code neat and readable. To wrap up the week, we created a repository in Github and used it to post our assignment so that others can review it. Here is the link to my Assignment 5 repository: <https://github.com/rudycarvajal/Assignment_05>