Andrew Moy

August 21, 2022

IT FDN 110 A Su 22

Assignment 7 (Module 7)

https://github.com/andrewrmoy/Assignment\_07

CDInventory.py program again with creation of error code messaging and modification of the data storage for binary data

Introduction

For this week’s assignment, we were tasked to modify our CDInventory program again, this time asked to take the error/exception handling feature we learned about in class and add it into our program to handle any unexpected interruptions that would normally completely stop our program from running. In addition, we were also asked to utilize the pickling function innately available in Python and use it in our same inventory program instead of having to write functions to separate, write and store data into a text file of our choosing.

We first went over both exception handling and pickling in class, but we were encouraged to do further research on both topics to get a better understanding of them. Outside of the course’s description of these topics, I went googling for a few extra references that reiterated what we were already exposed to, so I do not believe there was anything majorly new I found out there on the world wide web.

Creating the Program

Using that new knowledge in combination with the CDInventory program we already made modifications to last week, I first prioritized the conversion of our read and save functionality into binary data. First, we had to initialize the pickle module at the beginning of our program, using the import pickle function. I then went to my save\_CD function in Data Processing and converted my previous file opening functionality into the with option to cover opening, writing, and closing the file. Because we were now dealing with binary data, instead of w for ‘write’, we had to use wb for ‘write binary’ and then added the pickle.dump function after it to save our specified *list of tables* into the CDInventory.txt file.

With the same idea in mind, I went to my read\_file function in File Processing and wrote a pickle.load function to pull the binary data from our CDInventory.txt file using rb for ‘read binary’. At times while testing the program, I realized that I would get exception errors if the text file were completely blank, so I ended up having to add a handling statement to cover that for the program to load properly. Once my binary data conversion was done, I turned to adding catches for the most common sense issues, such as not inputting the correct ID or integer numbers or filename mismatches.

Summary:

This assignment was mostly straightforward after working out the kinks in the binary data conversion.





