Eric Hoyle

02-27-21

Foundations in Programming; Python

Assignment\_07

# Introduction

In this assignment we utilized Pythons Pickle method, data serialization, and structured error handling to further refine the CDInventory.py. Overall, the confluence of new ideas with the opportunity to further revise the CDInventory program provide an excellent opportunity to solidify my knowledge and add a few new features to my python toolset.

# Warning: Exceptional Behavior!

In this week’s module we took a look at exception handling in python to cover those circumstances when the pesky user enters unintended input and throws our function a curve ball. Between the modules, the book, and a little web browsing, implanting the exception handling was pretty straight-forward with a few *exceptions*.

In particular, the structure of the try and except lines and their placement plays an important role in how an error is handled and what segment of your code you want to encapsulate in the error trap. In addition to exceptions for file processing and FileNotFound errors I focused on value errors that arise when a non-numeric value is entered as an ID in the add\_cd function. Adding exception handling for this type of error was easy enough but had unintended consequences. When the error is encountered a fitting statement was printed but downstream errors then arose.

Text

Description automatically generated

Figure 1- Exceptions on top of exceptions

Because the function returned an empty list, everything relying on that list also encountered an error. After trying unsuccessfully to get the function to return to the input line within the function when an error was encountered, I finally opted to wrap the entire main in a try/except block to catch these types of perpetuated errors.

I then went back and found that I could put the ID entry into a while loop to give the user repeated opportunities to enter a number for the ID input. But, that too got nauseating after looping through the function a dozen times. So, I then implemented a counter in the function and give the user three chances to get it right before pointing out their personal flaws and returning them to the main menu to start over.

Text

Description automatically generated

Figure 2- Successful error handling for ID input

This worked pretty well and I was so proud of it I decided to use the same approach in the del\_choice function where the same error potential exists.

For exception handling In the FileProcessor class, I didn’t include options to repeat the attempt as I did with data input. I was tempted to create an additional option to create a new file as a response to the exception but felt it was out of scope for the time-being. However, I did find it useful to issue warnings to the user when the expected file could not be found and that as a result data could not be loaded or stored.

Text

Description automatically generated

Figure 3- Exception handling in the FileProcessor class.

I found this video to be a helpful explanation of the process <https://www.programiz.com/python-programming/exception-handling>[[1]](#footnote-1)

# Das Gherkins!

You know what is better than having to do a bunch of tedious processing of your data to convert it to a string and package it nicely to be exported to a .txt file? Pickling! What a time saver. This simplified much of the writing and reading process we have been using so far and condenses the whole process into a couple of lines. Along with the with/as statement which elegantly allows the author to open, write, and then close the file in one fell swoop, this saves a lot of coding. The only trouble I encountered here was realizing that the pickle methods require the file *object* and not the file itself. It took me getting a few errors before I was able to identify that small but crucial detail.

The other mistake I made was the false assumption that if I pickled a table that it would come back a table. I was naïve in overlooking the fact that I was actually still only exporting a list of dictionaries and not a true 2D table. Because of this oversight, it took me a little time to realize I needed to unpack my list again to be able to display the inventory correctly. Once I figured that out it was pretty smooth sailing.

I found this video to be helpful for me to understand pickling in Python. It was also useful in that it spends a little time talking about serialization and the uses and reasons for converting data to binary which I might have overlooked otherwise.

<https://pythonprogramming.net/python-pickle-module-save-objects-serialization/>[[2]](#footnote-2)

# Summary

In this week’s assignment I felt like I got my feet back under me again and was able to not only complete the assignment but also toy around with a few ideas to get a better understanding of how the pieces fit together. I find that while I still feel I know so little it is becoming more apparent to me that, in fact, I have learned a lot. This became most apparent to me when troubleshooting the first few obstacles I encountered with the pickle method and with constructing the error handling in this program. The success, and relative to a couple of weeks ago, speed with which I was able to recover from my errors felt like to me like proof that I have progressed considerably from where I was only 10 days ago.

I look forward to the lessons to come and continue to find the material and application of knowledge engaging.

Text

Description automatically generated

Figure 4- Functional in Spyder

Text

Description automatically generatedFigure 5- Functional in Terminal

# Appendix

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

A picture containing text

Description automatically generated

1. Viewed on 2-27-21 [↑](#footnote-ref-1)
2. Viewed on 2-27-21 [↑](#footnote-ref-2)