Santi Syjuco

03/13/2022

Foundation of Programming (Python)

Assignment\_07 – Knowledge Document

# Introduction

The assignment for this module included a mix of reading a chapter from the textbook, doing our own research for Exception Handling and Pickling topics, and to modify our code from last week’s Assignmnet06.

The coding task for this assignment was to modify our Assignment06 to include structured error handling for the following use cases: (1) Where there is user input (2) Where there is a change in data types being processed (3) when files are being processed. In addition to including structured error handling in our work, we also needed to use a binary file to capture the permanent data store.

# Steps Followed

## Step 1: Read Chapter 7 of the book

## Step 2: Research exception handling in Python

I was able to review the following external websites that also gave an overview of exception handling in Python. The following are the two that I liked:

1. [Python Exception Handling - GeeksforGeeks](https://www.geeksforgeeks.org/python-exception-handling/): I really liked this one because it gives you a high level overview of the 2 main kinds of errors you’ll run into in programming, exceptions and syntax errors. One it gives you that appreciation, the site then goes into an explanation of the types of errors and how they can be mitigated.
2. [Python - Exceptions Handling (tutorialspoint.com)](https://www.tutorialspoint.com/python/python_exceptions.htm) – I liked this resource because it starts off with a decent (but not overwhelming list of the kinds of exception errors that a user would go into, and then subsequently goes into how to solve for these.

The following I did not feel was a good resource for beginning programmers.

[Exception Handling — Python 3.10.2 documentation](https://docs.python.org/3/c-api/exceptions.html) – This website open its paragraph with multiple terminologies that I I got distracted with finding out their meaning, such as “POSIX” and “C API”. In the second paragraph it continues on with other terms such as “error indicator” and “…three object pointers”. I gave up reading after that point.

## Step 3: Research pickling in Python

I looked up the following sites to better understand Pickling for Python. They were satisfactory but I felt that the best resource was still the textbook. The biggest aha moment for me in the textbook was that the benefit of pickling was you would be able to save and load an entire file with just 1 to a few lines of code. In class it was also discussed that any data that was pickled will be unpickled in the exact same format. None of the following resources did these.

1. [How to Pickle: A Beginner's Guide to Pickling and Unpickling (pythoncentral.io)](https://www.pythoncentral.io/how-to-pickle-unpickle-tutorial/): This resource was fine in that it gives guidance on how to pickle and unpickle data but did not give the reader an overview of why pickling was important.
2. [What is Pickling in Python? (In-depth Guide) - Afternerd](https://www.afternerd.com/blog/python-pickle/): I appreciated from this resource the overview that pickling is just one o the many methods of storing data.
3. [Python Pickling (tutorialspoint.com)](https://www.tutorialspoint.com/python-pickling): This one doesn’t really give a good step by step of how to pickle but at least gives you the warning to NOT unpickle data from an untrusted source because it could pose a security threat.

## Step 4: Referenced LAB07-C

For the majority of the assignment I referenced LAB07-C as it gave me a good reference to remember how to do pickling from LAB7B while at the same time adding in the exception handling use cases.

## Step 5: Open Assignment06\_starter.py.

## Step 6: Started with Exception Handling

I added the following exception handling use cases:

1. Input\_cd: I added a check to make sure the user enters a number for the CD ID.
2. Menu choice ‘I’: I added a check for the show\_inventory function to let the user know to input a table in case inventory doesn’t show.
3. Menu choice ‘d’: I added a similar check to the one in menu choice ‘I’ to let the user know to input a table in case the user doesn’t see the current inventory from memory to delete a CD from.

## Step 7: Worked on the code to change the permanent store to use .dat instead.

I quickly realized after Step 6 that this was the more challenging portion of the assignment for me. The biggest challenge was ensuring that I was able to get the program running properly to be able to access the menu functions. I was able to reference the lecture from 03/11 where Dirk discussed the importance of having data being pickled first before they can be loaded. I actually had initially tried to just create a text file CDInventory.txt and just save it was CDInventory.dat manually, but this method did not work. It seems like the .dat file needs to be created by running the proper pickling syntax in code before it could work.

The next area I worked on was the load [l] function. This was to be equivalent of unpickling the data from the file. I remembered from the lesson that data will be unpickled to the same format that it was pickled in. Which reminded me that in this case, if I pickled the data , lstTbl, was a list, it would be unpickled again in the same format. So I then setup a function to read the data from the created CDInventory.dat file and assign the results now back to lstTbl. This way, the show\_inventory() could work as normal:



Lastly, I replaced the existing save function to now write to the CDInventory.dat file instead rather than a text file.

## Step 8: Added docustrings to my functions and commented on my code to make it easier for others to follow along.

I deleted all the old TODOs and TO Done comments. I then added new TO Done comments to show the changes I made for this assignment.

## Step 9: Capture Code working on Spyder

Created a new start sequence since we will now be using a .DAT file instead

Text

Description automatically generated

Figure 1 New opening sequence to create the binary file

Text

Description automatically generated

Figure Add CD option

Text

Description automatically generated

Figure Display current inventory

Text

Description automatically generated

Figure Delete Function

Graphical user interface, text

Description automatically generated

Figure Save Inventory to File

Text

Description automatically generated

Figure Load inventory

Text

Description automatically generated

Figure 7 Exit

## Step 9: Capture Code Working on Terminal

Text

Description automatically generated

Figure Program startup

Text

Description automatically generated

Figure Display option

Text

Description automatically generated

Figure 10 Add CD

Text

Description automatically generated

Figure Delete option

Text

Description automatically generated

Figure Save option

Text

Description automatically generated

Figure Load Option. See that Jars of Clay and BTS CDs are now back in memory.

Text

Description automatically generated

Figure 14 Exit

## Step 10: Uploaded python script and assignment to Github

[syjuco/Assignment\_07: Homework for Module\_07 for UW Foundations of Programming (Python) Course (github.com)](https://github.com/syjuco/Assignment_07)

# Summary

The most challenging parts of this assignment was pickling and unpickling data to be used by the user. Another layer of complexity that was added was ensuring that the code worked together with the other pieces of the code that initially only referenced data from the text file. There were likely other areas I missed in the assignment to still add more checks for file processing but I ran out of time to check for all the locations. I opted to focus on ensuring the program would at least be resistant to user input error and to ensure that the program continued to run with a permanent file store in binary format.