<Jingyin Chen (Seline)>

<2021-Aug-20>

<Foundations of Programming (Python)>

<Assignment 07>

### Step 6 - Document your knowledge

### GitHub link to this week’s assignment:

<https://github.com/SelineCCC/Assignment_07>

### Introduction

This week is tougher than I thought. I tried my best. I think, the hardest part is how to translate a concept into codes and break them down to the most singular element. Specifically, I tried to include as many build-in exceptions, but most of them are beyond my scope. And I had a hard time understanding ‘Modify the permanent data store to use binary data’, and still not sure whether I did the right thing (or close to it).

Topic 1: Error Handling

So far, I think the try-except error handling is handleable. Although I know that things will change when we dive into Raising Custom Errors or Custom Exception. I feel the try-except error handling is handleable because Spyder points to the specific line where the error occurs. Following these hints makes it easier to pinpoint where to insert the try-except error handling.

The harder part is to anticipate where the errors are. And sometimes when you think, ‘O, the error resulted from the user inputting a wrong datatype.’ But the error handling should be placed in the data processing section. Guessing the point is to find where the program stops, rather than thinking, ‘O, it’s an input error, so let me put it in the I/O part.’ Not sure whether that’s a universal issue. But it happened on my codes.

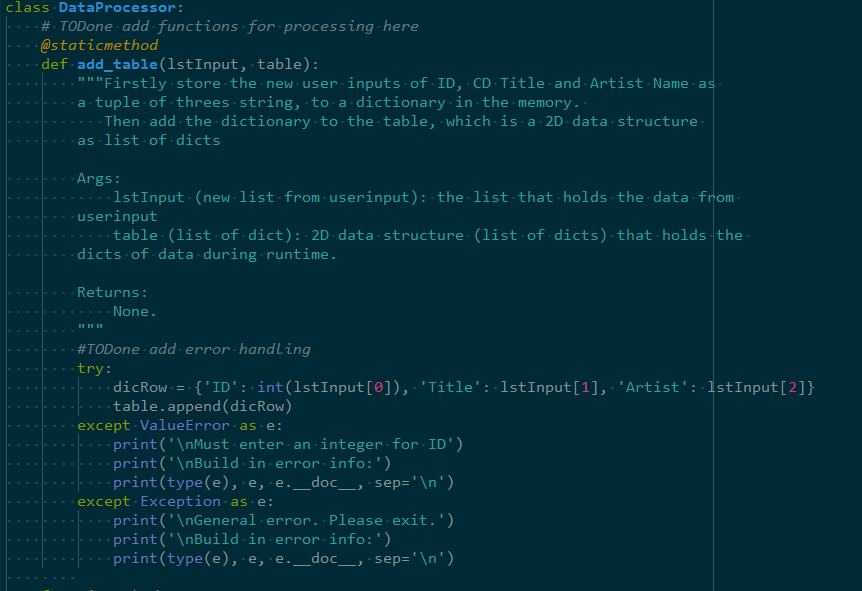


Figure – Error Handling: thought it should be placed in the I/O section but Nah.

Topic 2: Plain or binary, this is a question

I...still don’t think I got the ‘Modify the permanent data store to use binary data’ right.

All I can do is bring my super-cooperative user and ask her/him/them to figure out or guess what kind of text file we have to load, and don’t let the program stop if she/he/they got the wrong text file type.

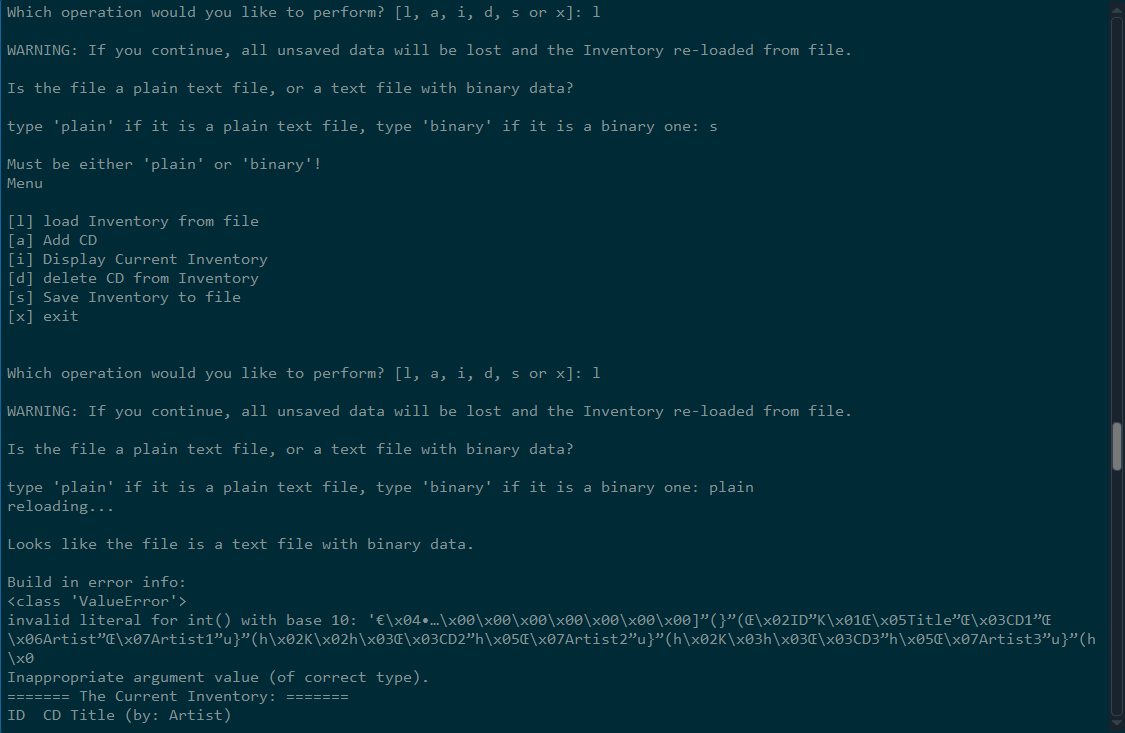


Figure – All I can do, is try to make the program non-stop till the user wanna exit

And I also had a hard time working with the pickle.load() for the Lab exercises because I forgot the teeny tiny note of this:

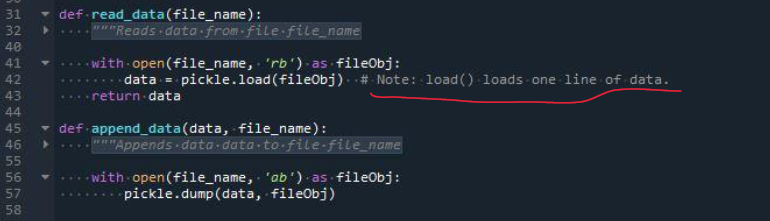


Figure – Did not grasp the pickle.load() at first.

(Alright it’s not exactly teeny tiny but it is in grey. And it is explained in the tooltip. And it was emphasized in the videos)

Then I just kept testing with the print() and realized line = dictionary of the 2D list in the assignment, as like the level/dimension of objects after the first breaking-down. Therefore, a for loop could solve it.

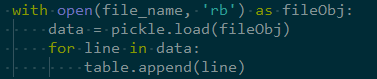


Figure – pickle.load() and for loop

As I am writing this document, I have the feeling that the Raising Custom Errors might be able to make the program to ‘load the plain text file or the text file with binary data, whichever one works, just load it’. As, just, whenever there’s an error comes up, go to the other direction?

Topic 3: Final Result

Below are how the final codes work in Spyder and the terminal:

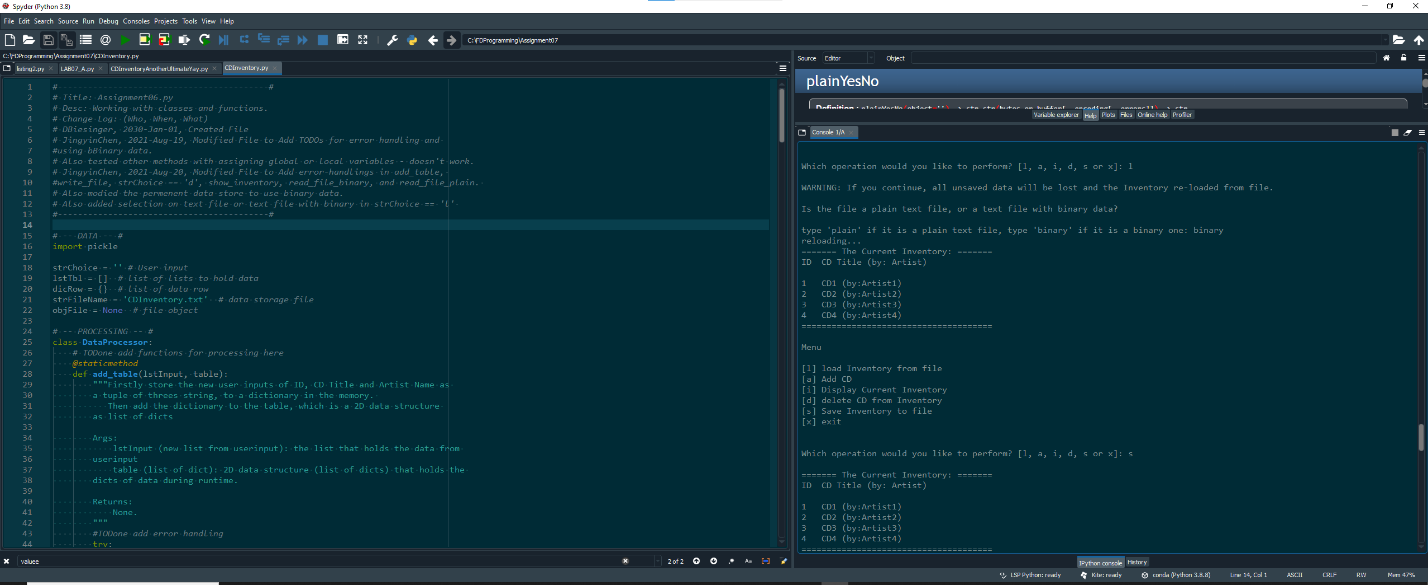


Figure - working in Spyder

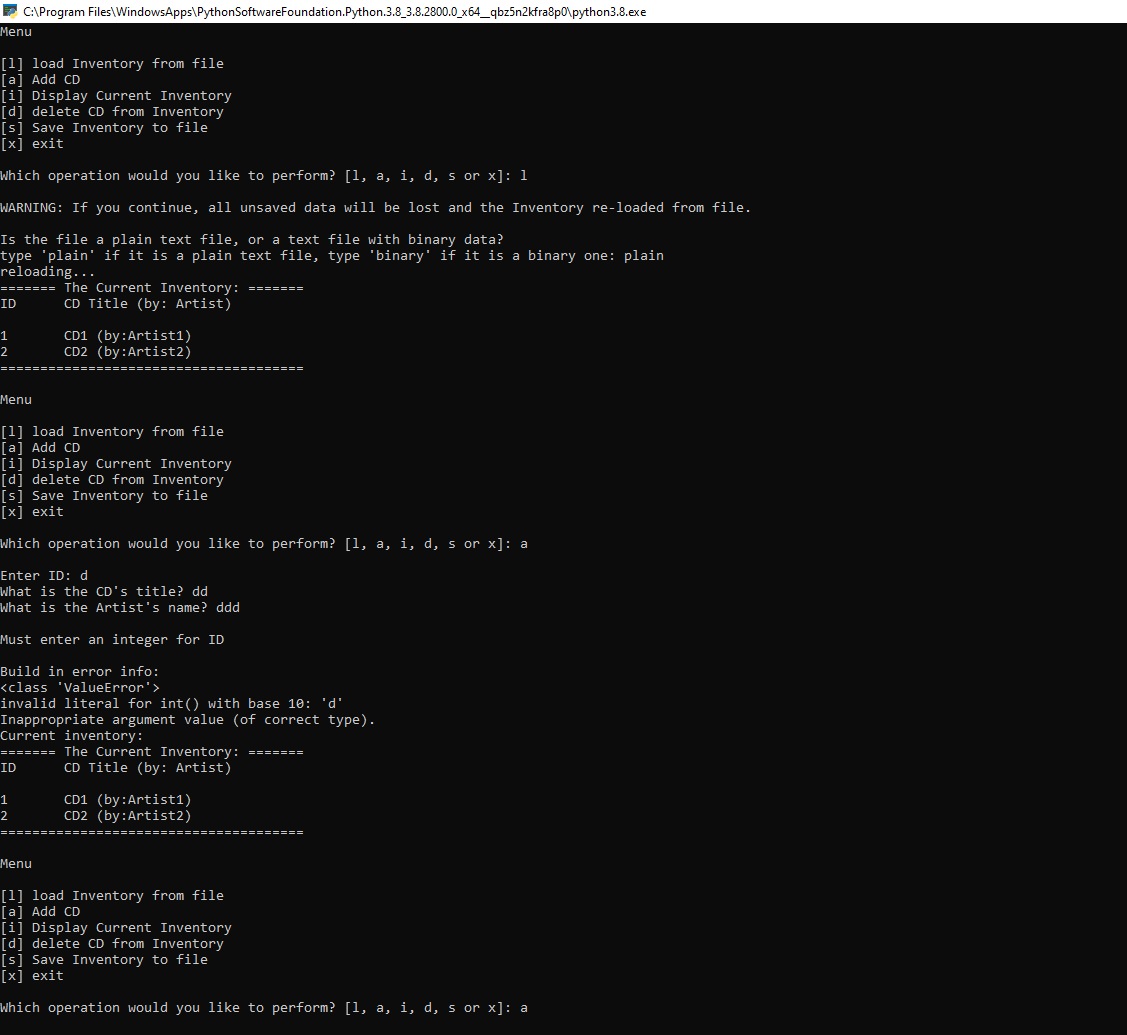


Figure - working in the terminal

Topic 4: Doubt

My doubt is how to have a smart and elegant way to let the program load file with a plain text file or a text file with binary data.

### Summary

This week is tough. But it is very satisfying to use as little code as possible to translate my thoughts/the assignment’s requirements. I think for this week’s knowledge, the concepts are okay to grasp, but it is really hard to integrate them into practice.