**Name: Stephanie Yong**

**Date: August 30, 2022**

**Course: Foundations of Programming: Python**

**Assignment: Document Your Knowledge – Module07 – Files and Exceptions**

**GitHubULR:** [**https://github.com/stepaidzzz/IntroToProg-Python-Mod07/**](https://github.com/stepaidzzz/IntroToProg-Python-Mod07/)

**Document Your Knowledge – Module07 – Files and Exceptions**

**Introduction**

In Module07 – Files and Exceptions, working with binary files and exceptions were introduced. On top of the lectures and Module07 programming notes, we are encouraged to research on both of these topics and to come up with a script on our own. To wrap up Assignment07, also need to try working with a more advanced GitHub pages.

**Pickling in Python**

Pickling in Python means converting a Python object hierarchy into a byte stream (i.e. a binary file or bytes-like object). While unpickling is the reversal of aforementioned process.

Some of the most common functions for pickling would be dump and load.

One of the biggest advantage of pickling is that it would significantly reduce the processing time depending on the object size.

General information of Python object serialization: <https://docs.python.org/3/library/pickle.html#example>

Three types of exception for pickle module:

<https://docs.python.org/3/library/pickle.html#pickle.PickleError>

Referencing YouTube videos:

Python Tutorial 14: Saving and Reading Data Files with Pickle

<https://www.youtube.com/watch?v=eWrTSBIQess>

Need to import pickle module in order to pickle and unpickle the script as shown in Figure #1:

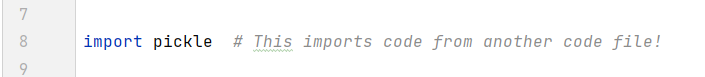


Figure #1

In Figure #2 and #3, “ab” and “rb” are the functions to instruct working on binary files:

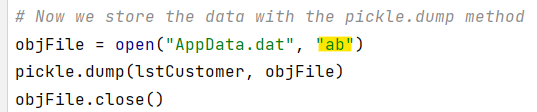


Figure #2

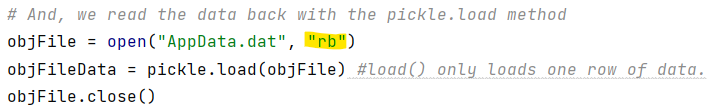


Figure #3

**Structured Error Handling**

During Assignment07, try-except statement has been introduced. The try-except statement is used to capture error and providing meaningful/customized error messages for the script users.

On the websites, there are plenty of example for ZeroDivisionError, however for Assignment07 that I have designed, the ZeroDivisionError is not quite applicable. Instead examples provided in Module07 programming notes under section “Raising Custom Errors” would be more applicable and relevant to my script as shown in Figure #4:

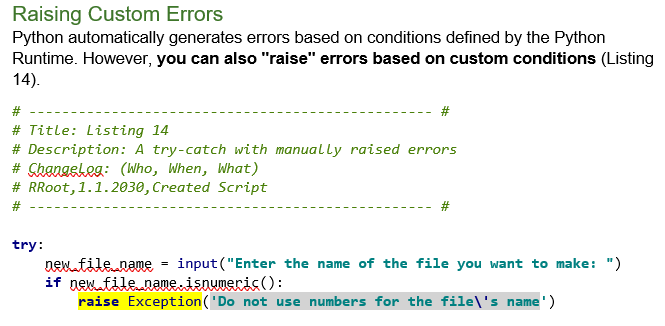


Figure #4

The example provided in the Module07 programming note is then further modified to fit my purpose as shown in Figure #5:

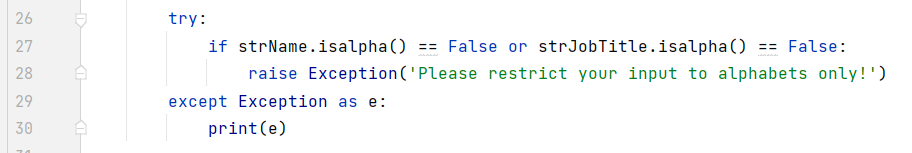


Figure #5

**Challenges for Assignment07**

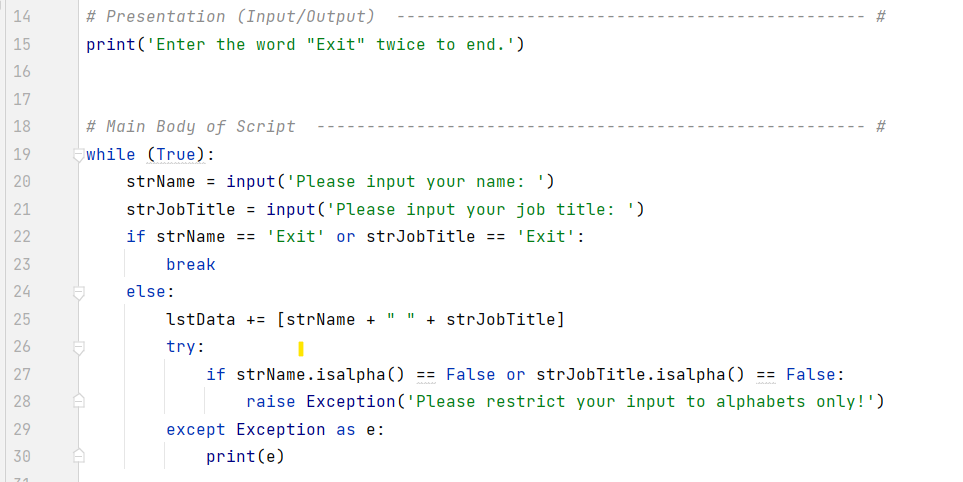
****

Figure #6

One of the major challenges for Assignment07 is to incorporate the Try-Except statement into the script. At the end, came to conclusion that the try-except statement should be put underneath the while loop and if-else statement as shown in Figure #6. Before coming to this conclusion, the try-except statement has been put in various different places and it did not work as planned.

Another challenge is figuring out the pickle.load function does not “print” the result. Later on noticed that a print statement is indeed needed in order for Python to “visualize” the converted binary file as shown in Figure #7.

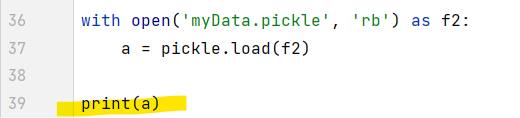
****

Figure #7

**Other Useful References for Assignment07:**

How to use OR operator in Python If Statement:

<https://pythonexamples.org/python-if-or/>

Python String Methods:

<https://www.w3schools.com/python/python_ref_string.asp>

The above link found useful in providing different functions for Python strings.

**Run and Test the Script under Python Environment**

As shown in Figure #7, the script yields a custom error message of “Please restrict your input to alphabets only!” if input contains anything non-alphabet.

Also, the “Exit” command is working. One “Exit” command in either the name or job title would terminate the script from further capturing inputs.

The script has also successfully read the data from a binary file at the very end as quoted within the square brackets.

Pickled file is also found under the designated location as shown in Figure #8 and the corresponding binary file contains information as shown in Figure #9.

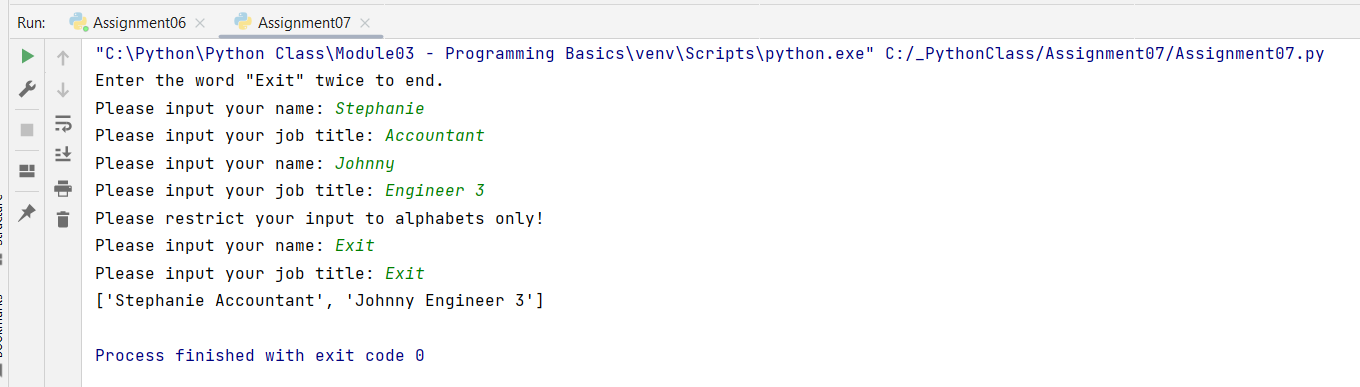
****

Figure #7

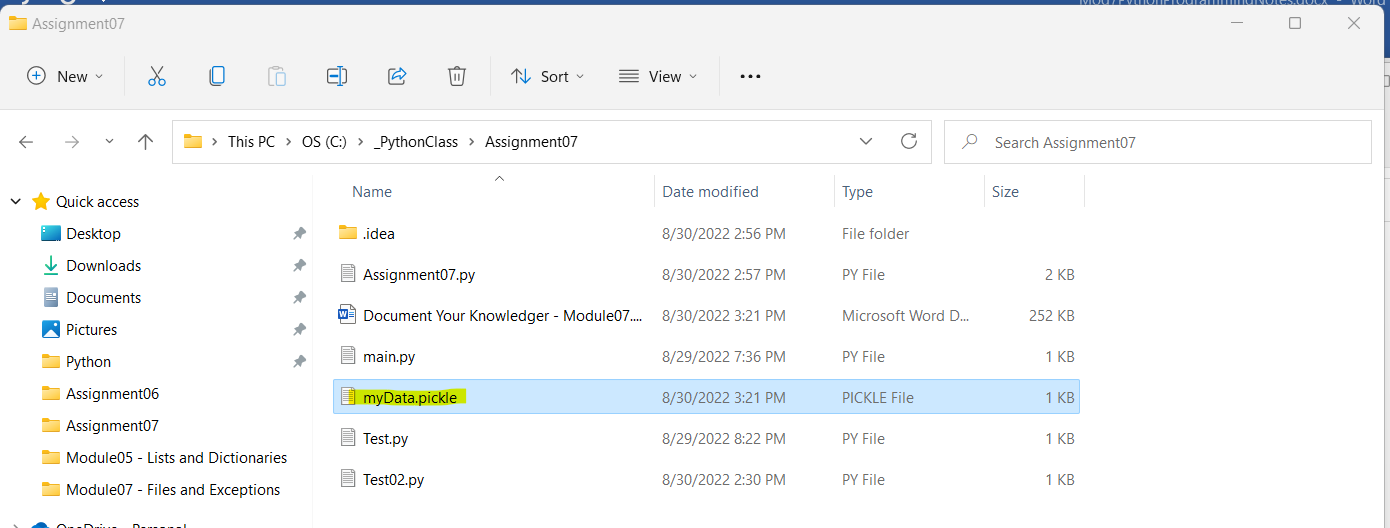


Figure #8

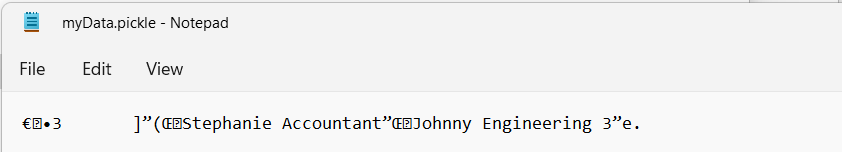
****

Figure #9

**Run and Test the Script under CMS Environment**

Tested the script under CMS environment and it yields result as planned as shown in Figure #10 below:

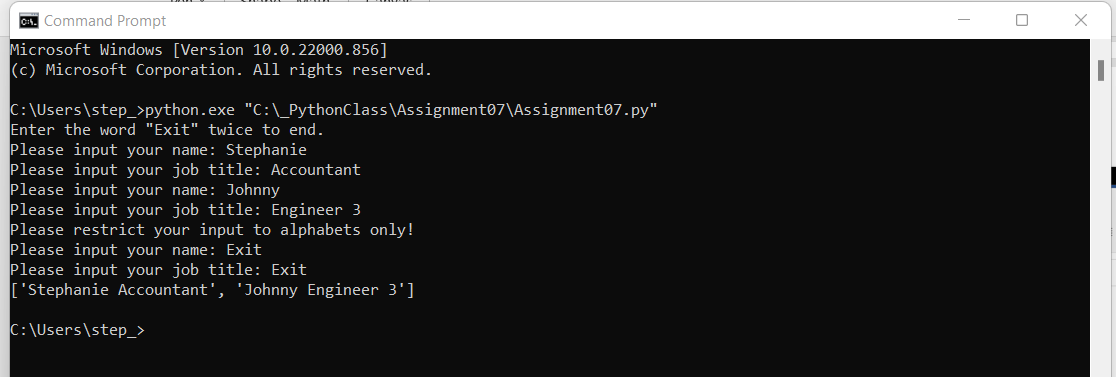
****

Figure #10

**Conclusion**

During Module07, we are given opportunity to work with binary files as well try-except statement. Since we have to come up with our own script from scratch, it was quite challenging and time-consuming in doing corresponding research.

One thing that I found fascinating about coding is that, even with the exact same script, putting lines in different orders/sections/hierarchy would have entirely changing the outcome. It is painful to test out millions of possibilities and yet it is so satisfying when you get the output anticipated.