Kristin Temperly

03/01/2022

Intro to Programming (Python)

Assignment 07

GitHub URL: https://github.com/xkkt0700/https-github.com-xkkt0700-IntroToProg-Python-Mod07

Pickling & Structured Error Handling

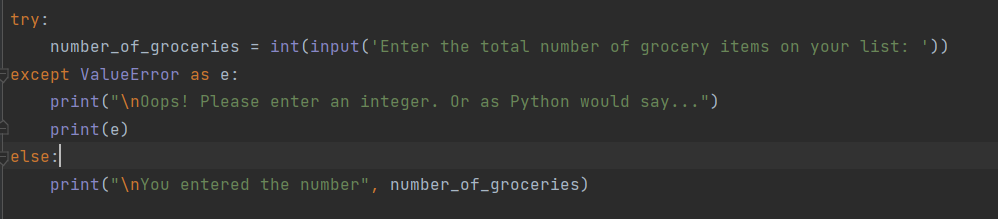
**Introduction**

**In module 7, we were asked to research and demonstrate Python pickling and structured error handling. In this document, I will explain the steps I took to explore and test these concepts. In addition to Randal’s usual lecture and the required reading in our textbook, we were also required to use external Python resouces found on the internet.**

**Structured Error Handling**

**When Python encounters an error, the script is stopped and an error message is displayed. When this happens, an exception is raised. Porgrammers use error handling exceptions to manage Python errors more gracefully instead of allowing the program to abruptly crash. The rudimentary way to handle this is with the try/except block, but Python also allows for more customized error handling.**

**I found an informative explanation of Python structured error handling at a popular website called The Python Guru (**<https://thepythonguru.com/python-exception-handling/>). This external reference illustrated, in great detail, many different examples. The first few examples were simple, but they grew more complex. I followed their examples and began my assignment this week by testing if the user of my script indeed enters an integer when prompted.

**Listing 1. Example of Try Except Else**

The [try](https://docs.python.org/3/reference/compound_stmts.html#try) statement in my code works as follows.

* First, the try clause (the statement(s) between the [try](https://docs.python.org/3/reference/compound_stmts.html#try) and [except](https://docs.python.org/3/reference/compound_stmts.html#except) keywords) is executed.
* If no exception occurs, the except clause is skipped and execution of the [try](https://docs.python.org/3/reference/compound_stmts.html#try) statement is finished.
* If an exception occurs during execution of the [try](https://docs.python.org/3/reference/compound_stmts.html#try) clause, the rest of the clause is skipped. Then, if its type matches the exception named after the [except](https://docs.python.org/3/reference/compound_stmts.html#except) keyword, the except clause is executed, and then execution continues after the try/except block.
* If an exception occurs which does not match the exception named in the except clause, it would be passed on to outer [try](https://docs.python.org/3/reference/compound_stmts.html#try) statements. If no handler is found, it is an unhandled exception and execution stops with a message from Python.

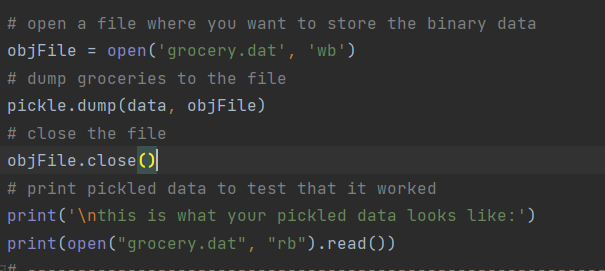
**Let’s Pickle!**

**The concept of pickling is actually quite simple. Instead of writing actual characters to a file, a pickled object is stored as a binary file. We can pickle a variety of objects, such as: dictionaries, lists, tuples, strings and numbers. Picking begins by importing the pickle module.**

**I chose a blog called JournalDev** <https://www.journaldev.com/> **as an external reference because it seemed to be one of the less common Python reference websites, and I deemed it to be trust worthy. I wanted to find a website or external reference that other students wouldn’t be as inclined to use. Additionally, the examples in the refernced blog were very similar to what Randal taught us in lecture.**

**Pickling Data and Writing It To a File**

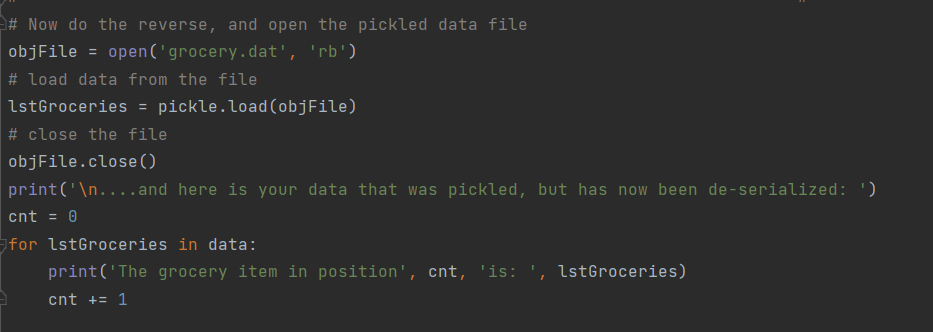
**In my assignment illustrated below, I followed the examples in the external blog but changed a few concepts to make it my own. Essentially, I opened a new binary file named grocery.dat for writing by passing wb as the file access mode. wb stands for write binary. I then use the dump() function to move the data into the .dat file. After closing the file, I finally show the user what the pickled data looks like to indeed prove that pickling was successful.**



**Listing 2. Pickling Data and Writing It To a File**

**Reading Data from a File and Unpickling It**

**To restore the pickled data, we use the load() function. To protect your computer from malicious acivity, it’s best not to unpickle data from an untrusted source. The pickling module does all the work to de-serialize the data upon loading. Next I close the file, and print the contents to prove to the user that the data is now in a format that humans understand. I had to use a simple counter in order to print all of the data in the file.**



**Listing 3. Reading Data from a File and Unpickling It**

**Summary**

Exception handling makes code more robust and also helps prevent potential failures that can cause a program to crash. It is a best practice to include exception handling, especially when working with the presentation code layer. The Python pickling module allows data storage in binary mode, which is valuable when tranporting data.

**I found the assignment this week was a bit easier than prior weeks. A breath of fresh air! Maybe Randal is giving us break before the rigor of week 8. Upon completion of my script, I realized that I probably should’ve created functions to exemplify pickling, which would’ve made the code more re-usable and adheres to best practices.**