Kristen Burke

Aug 24, 2021

IT FDN 110A

Assignment 07

GitHub: <https://github.com/kristenrb/IntroToProg-Python-Mod07>

The Pickled Exception

# Introduction

This assignment asked for a program that would demonstrate pickling and error handling. This project was somewhat open-ended, with no starting material provided and instructions to do more research. The assignment was coded in PyCharm.

# Functions and Concepts Used

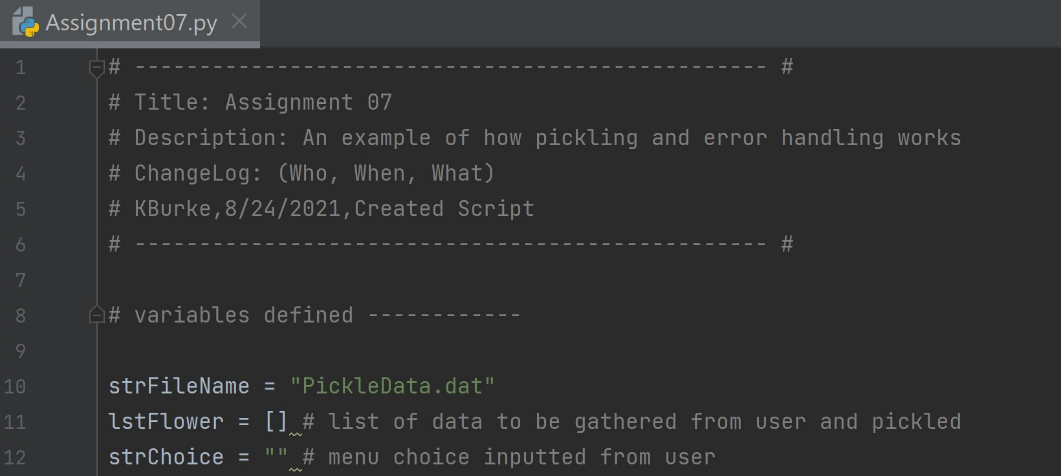
Pickling is when data is stored into a binary file instead of a text file. This allows for larger chunks of data to be stored in a smaller space. When a file is unpickled, the data will be reverted back to its usual form. The main functions used for pickling are **dump()** and **load()**, respectively for putting data into the file and loading it back out.

This module also explored error handling in python. Sometimes, a user inputs some bad data and causes the script to stop and throw a confusing error message. Or a file read by the program has no data left and stops the program instead of just moving to the next step. Using a **try-except** block of code can help get around these situations and keep the program running. Python has standard exceptions but custom ones can also be created.

# Creating and editing the Program

Since this program had to be created from scratch, I decided to keep it as simple as possible to demonstrate the key concepts accurately. I pulled from my work on the previous project to create a simple program that asks the user for some data, pickles that data, and then unpickles it into a list.

As usual, I started with a header and defining some variables (Figure 1).

Figure 1: Script header and variables defined.

Next, I defined my functions for pickling and unpickling the data (Figure 2). Here is also where I realized I would need to use a try-except block. The load function only pulls one row of data from the binary file so it would need to be run over and over again. But eventually, the file will run out of data to unpickle and throw an end-of-file error. An exception here allows the script to keep moving with whatever it did pull from the file.

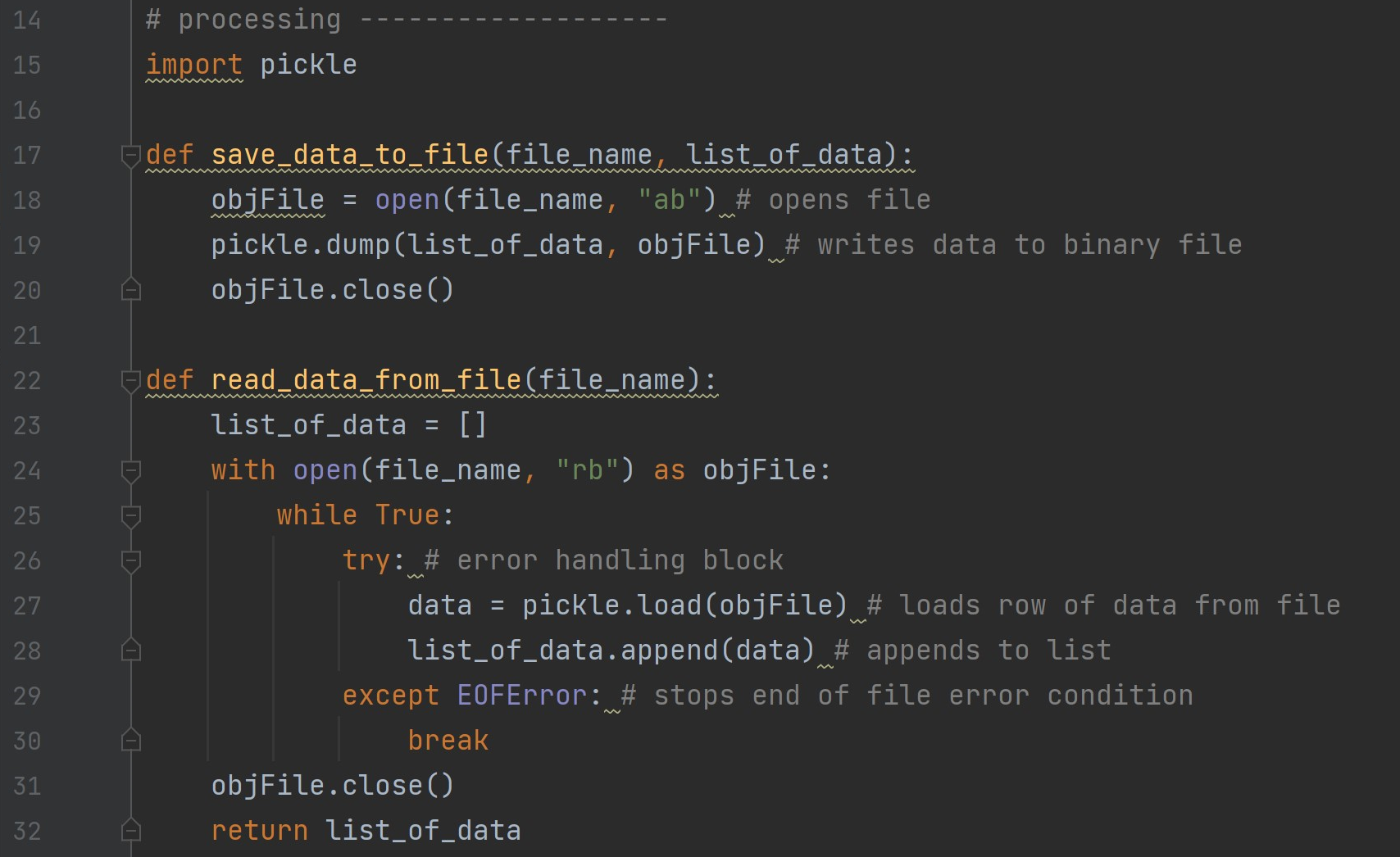


Figure 2: Pickling and unpickling functions.

I also defined some simple presentation functions (Figure 3).

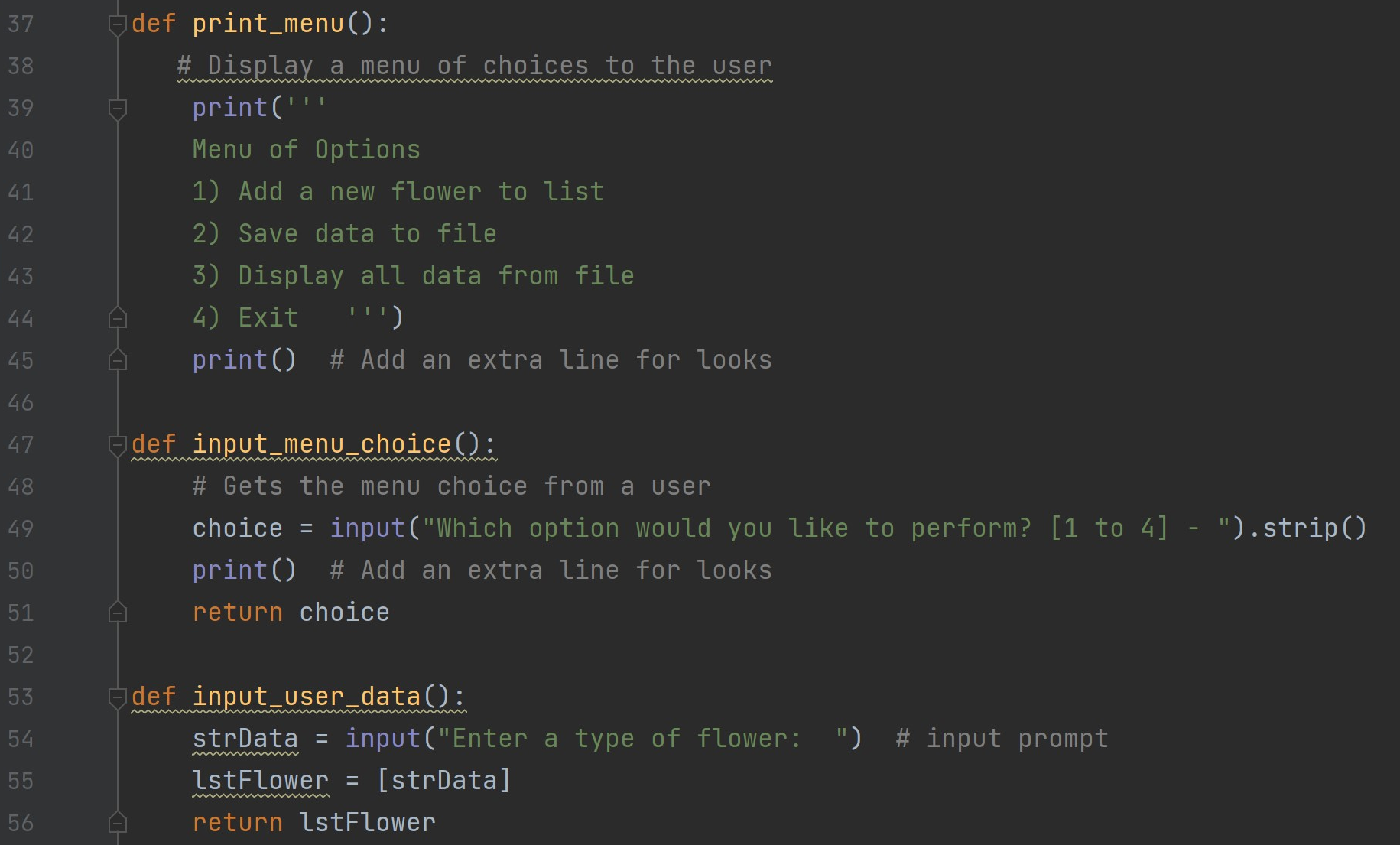


Figure 3: Defining presentation variables.

Finally, I constructed the main part of the script, calling the functions under their designated menu options (Figure 4).



Figure 4: Main body of script.

# Running the Program

I first ran through the program in PyCharm, verifying that each option does what it is supposed to do (Figure 5).

|  |
| --- |
|  |
|  |
|  |
| Figure 5: Testing script in PyCharm. Repeating elements such as the menu of options and beginning instructions omitted from some screenshots. |

The script was also tested from the command window (Figure 6).



Figure 6: Running the script in the command prompt window.

# Conclusion

While the script could definitely be more polished, I think it demonstrates an example of how to use the concepts in the module. In the future, I would probably find a way to organize the pickled data so that it is more readable when it is unpickled.