David Lammers

Dec 2, 2022

FavCookie.py

Assignment 07

[Github link](https://github.com/dlammers3/IntroToProg-Python-Mod07.git)

Favorite Cookie Survey

## Introduction

In this assignment we were asked to write a program that shows how pickling works and at the same time, capture any errors that may occur. The program I wrote is meant to be used as a survey to poll users on their favorite of three different cookies. It then stores their selection in a dictionary which is then pickled and saved in a .pickle file. Exception handling is employed to handle a fresh load of the program on a shared server or webpage.

## Writing the Script

I decided to write a script that would survey the user and store their input in a dictionary. In order to do this directly, I would have to employ pickling so that it could be saved to a *.pickle* or *.bin file*. Since this program is meant to be used as a survey tool, I wanted to be able to initialize the program with an empty dictionary and then overwrite it as users participated in the program. Short of creating the *.pickle* file with a separate program and saving it locally with the *FavCookie.py* script, I needed a way to initialize an empty dictionary and then never call it again. I also found that if the file was not pre-existing, that I would get a *FileNotFoundError.* I will come back to this issue shortly.

Assuming that the file is already present in a local folder, the data is read in using the *pickle.load()* method as seen in fig 1. Already this is much more convenient and straightforward than loading in a list of dictionary rows from a .txt file. Once the dictionary is read in, the user is prompted to answer a question that will be saved as the variable *cookie.* This answer is then used to build up the dictionary by counting up the value associated with each cookie key by 1 (fig 2).



Figure 1 Reading in the binary file

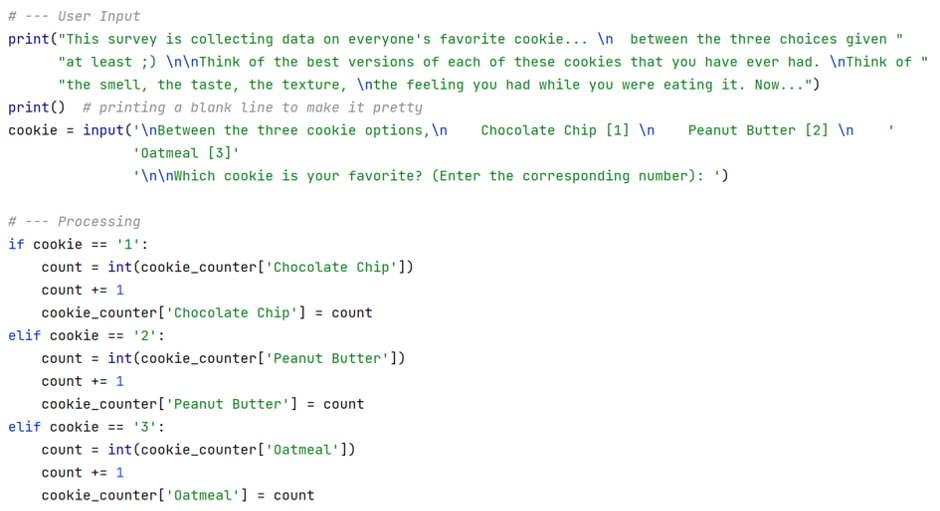


Figure 2 User input and counting logic

The collected data is then displayed back to the user with a little message in fig 3. Just as easily as the binary folder is read in, this dictionary is saved to the same binary file, overwriting the previous data, fig 4.

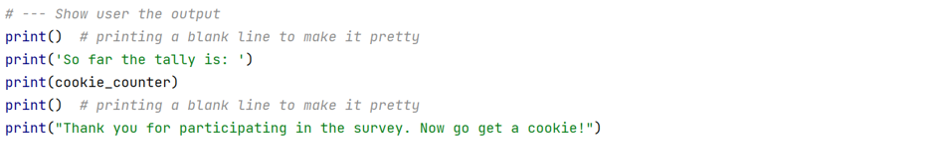


Figure 3 Closeout display message and current survey results



Figure 4 Saving the dictionary to a binary file

To verify that the data is stored as binary, we look at the *cookie.pickle* file with a text editor. We should expect to see gibberish with some slight indicators that does indeed belong to this programs output. Looking at figure 5, this is precisely what we see. The words, “Chocolate Chip”, “Peanut Butter”, and “Oatmeal” can be seen, along with a bunch of nonsensical characters. The important thing is that the computer can read it.

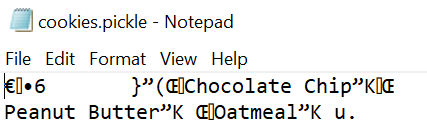


Figure 5 Reading the pickled data in Notepad

Now, revisiting the error handling. Since this survey should start with an empty dictionary, but also needs to read in the previous dictionary (in order to keep the tally going), we run into a problem. The binary read-in method does not like a non-existent file if you are asking it to read-in a file. You get the error, FileNotFoundError. To remedy this, I used a try statement to initially read-in the file. If there is no file, then the exception clause will trigger and the reinitialization code will write an empty dictionary to the file. After this takes place the code jumps back into the normal routine as if the file had existed from the beginning. I chose to just have the exception clause automatically remedy the error rather than pointing it out the to user. This way, even if no file was there to read in, it would automatically create one and read it in without the user even knowing, see figure 6.



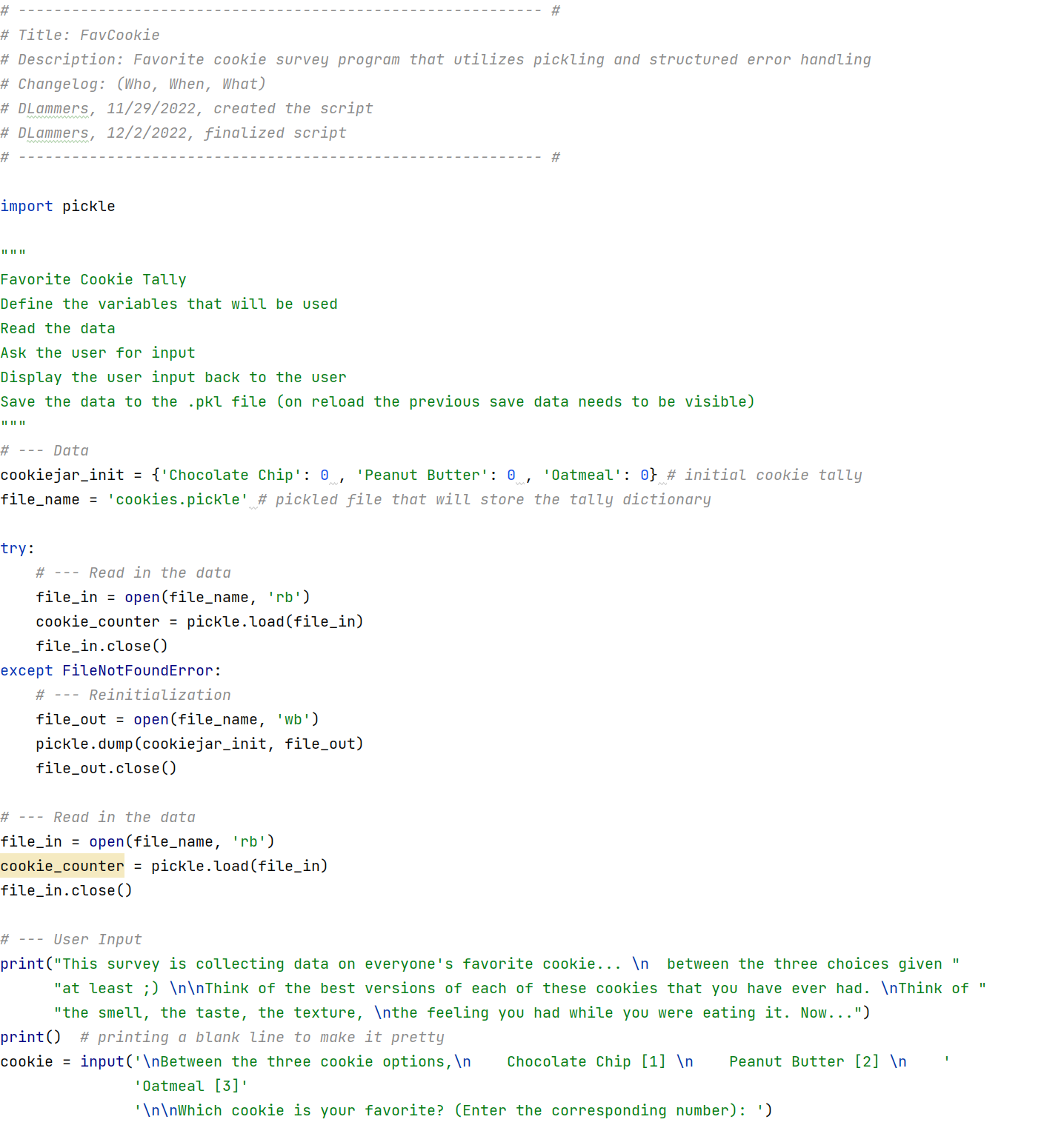
Figure 6 Exception handling of FileNotFoundError

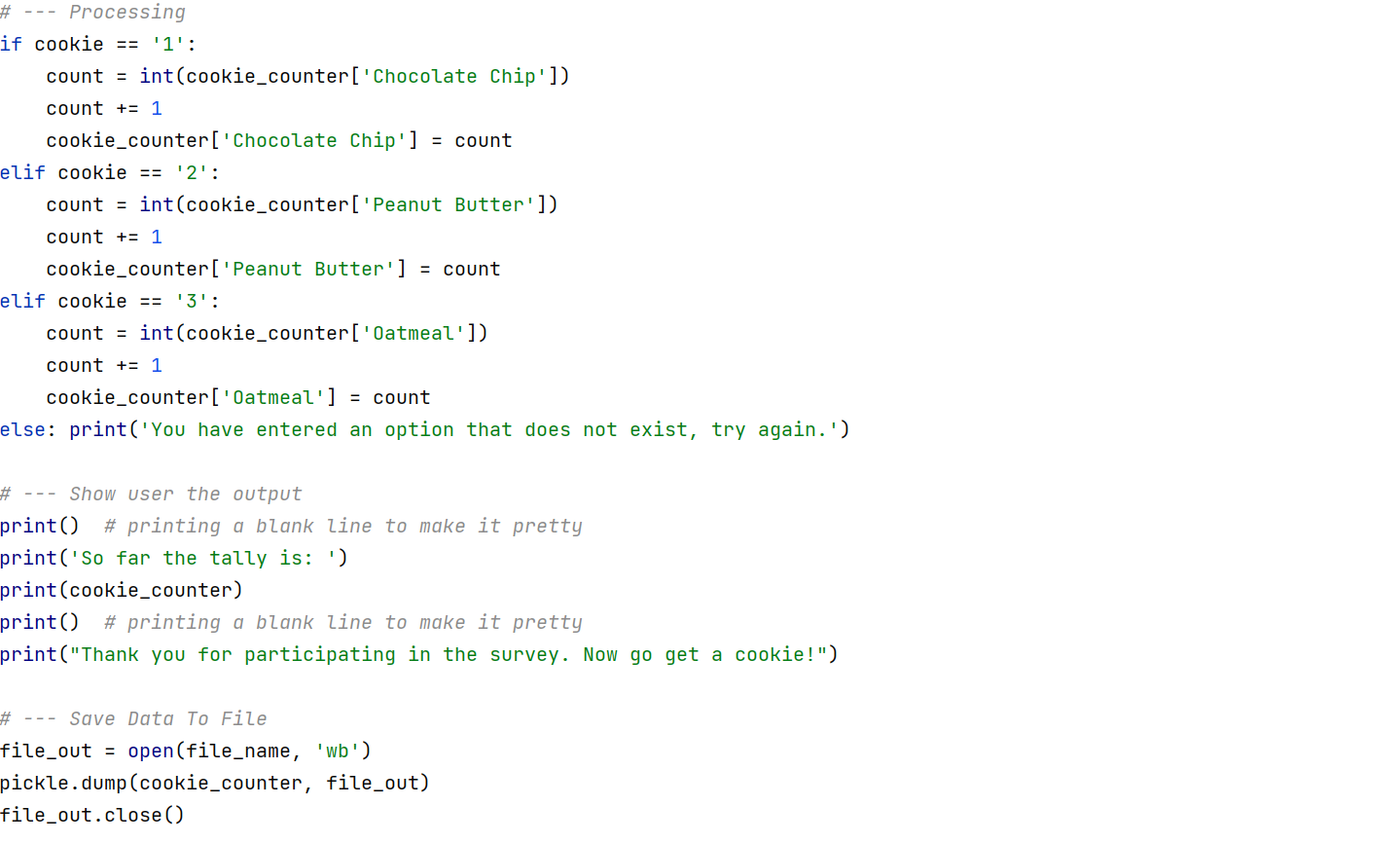
## Summary

We were asked to show how pickling and structured error handling works, and I believe that this program achieves both. Both of these new tools are very useful and will help me make more robust programs in the future. I did find a lot of useful information on the web, particularly [*https://www.datacamp.com/tutorial/pickle-python-tutorial*](https://www.datacamp.com/tutorial/pickle-python-tutorial)for pickling and [*https://www.datacamp.com/tutorial/exception-handling-python*](https://www.datacamp.com/tutorial/exception-handling-python)for exception handling. I have used Datacamp in the past to learn Python and I find their tutorials very informative but not cumbersome like others out there. The exception handling tutorial is very useful because it explains what errors cannot be ignored, or more importantly, which ones you should not code in to ignore, like the OutOfMemoryError because it is irrecoverable, meaning that the program will not recover even if it is handled.

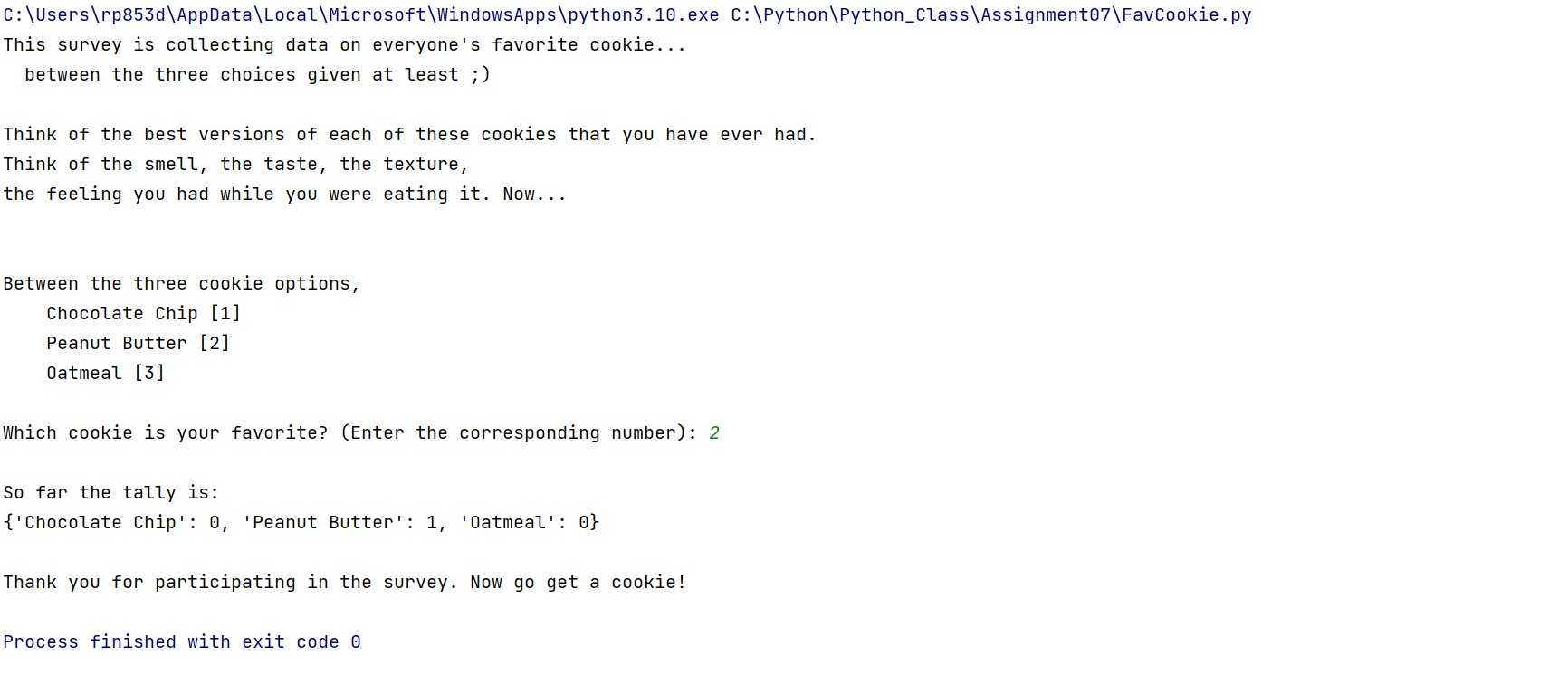
The program in its entirety can be found in the appendix, along with examples of it running in PyCharm and the Command Console.

## Appendix

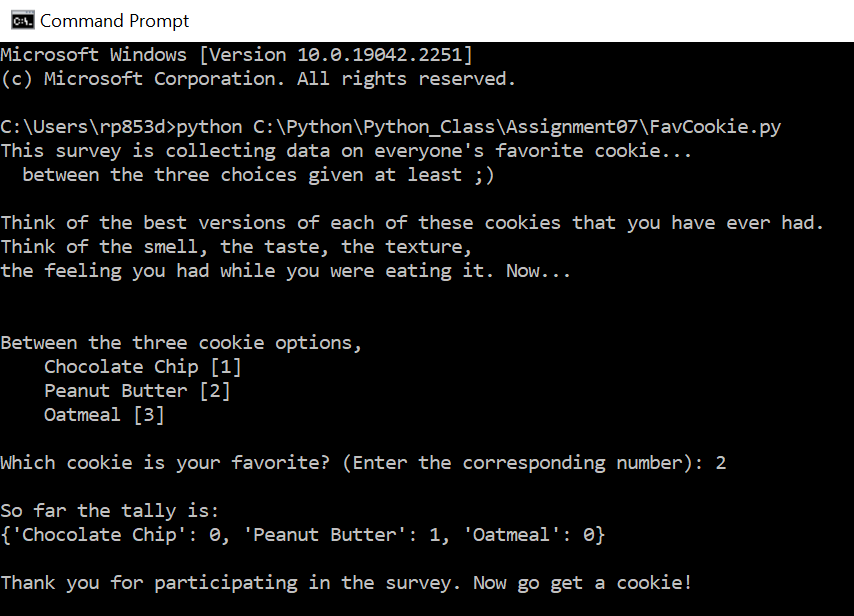




Appendix I FavCookie.py



Appendix II FavCookie.py running in PyCharm



Appendix III FavCookie.py running in the Command Console