George Biehl

September 6, 2020

IT FDN 110 A Su 20: Foundations of Programming: Python

Assignment09

# Working with Modules

## Introduction

In this assignment we are given three data classes – each containing a number of modules. The goal is to use a harness to test each module and, finally, to combine use of the modules in a single, main script. As I worked through the various testing stages, I saved the script with a meaningful name for each version.

## Beginning a Harness File

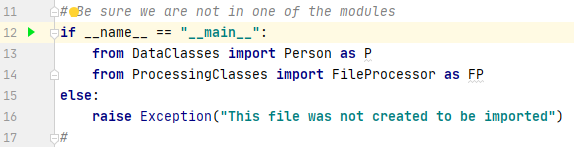
My harness file began by testing that it was running in the main script. I also imported the Person and FileProcessor classes, giving them aliases of “P” and “FP”. The aliases are not necessary but, if well-chosen, they shorten the code lines with no loss of clarity. as shown in Figure 1.

Figure 1: Testing to assure code is running in the main script and to import two classes

## Testing the Person and FileProcessor classes with PersonData.txt

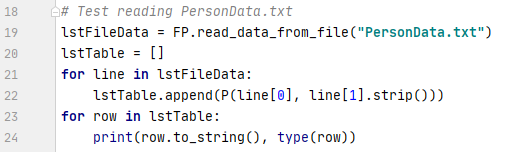
I imported code from the listings adding only a line to initialize lstTable (Figure 2). Running that script read and printed the PersonData.txt file including the data type that was created in memory.

Figure 2: Script for reading and printing the PersonData.txt file

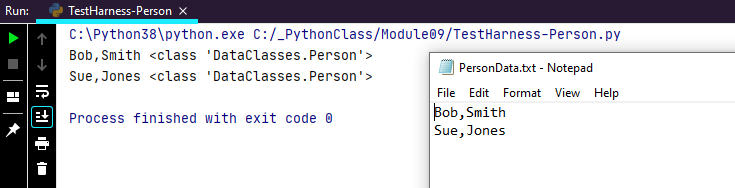
Figure 3 shows the output after read\_data\_from\_file ran and the PersonData.txt file itself.

Figure 3: Output from lines 19 – 24

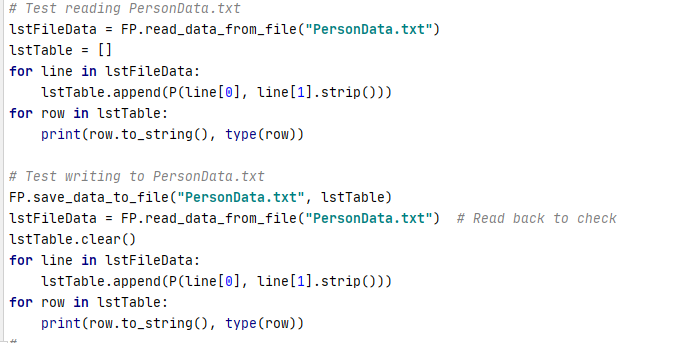
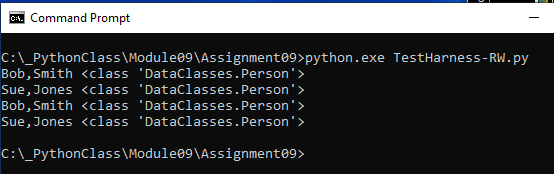
Next, I tested the save\_data\_to\_file method. This was one step more complicated. The script (shown in Figure 4) has 4 lines of outpout – two from that original read process described above and two from the read back after writing to file - a “round trip.” Figure 5 shows the output running from a command line.

Figure 4: Script to test save\_data\_to\_file method

Figure 5: Note that the first two lines are repeated showing the data was unchanged

## Testing the Employee Class with EmployeeData.txt

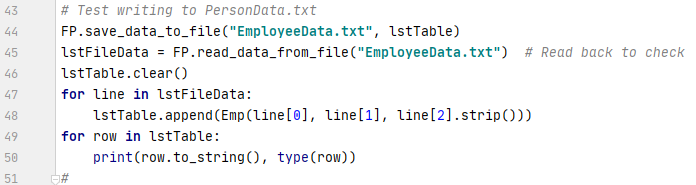
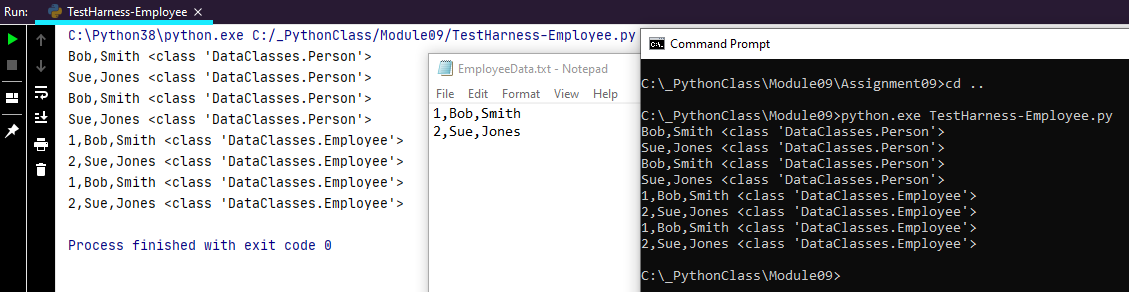
Testing the Employee class was done via a very similar process. For example, shown in Figure 6 is the script used to write and later read back from file the employee data.

Figure 6: Script for testing that EmployeeData.txt is correctly written and read

In the output shown in Figure 7 one can see the contents of the EmployeeData.txt file, the PyCharm output, and the command line output. The first four lines are created by the Person class tests already discussed. The fifth and sixth lines are the result of reading and printing the EmployeeData.txt file. The final two lines are caused by the script in Figure 6 which writes and then reads back the file contents.

Figure 7: EmployeeData.txt file and output from PyCharm and command line.

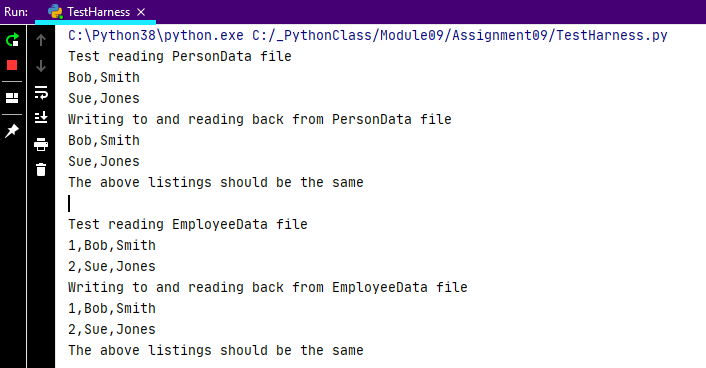
I now knew that the data classes and file reads/writes worked well. I improved the print statements so that the output more clearly showed what was being tested as shown in Figure 8.

Figure 8: Improved clarity for the output of the first tests

## Demonstrating the EmployeeIO Class

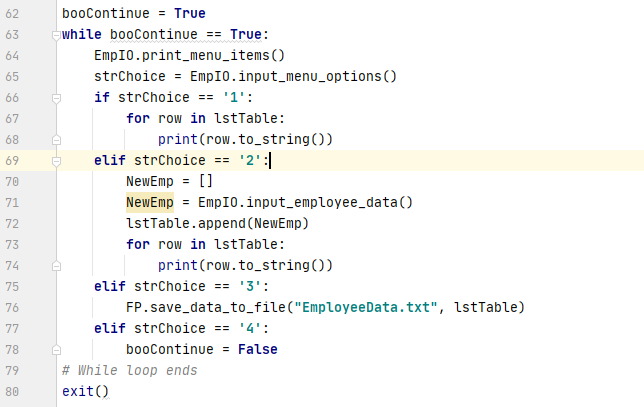
It remained only to add the menu along with code for handling each menu option. We’ve done this quite often in this course so the process was very straight-forward as shown in Figure 9.

Figure 9: While loop to present and respond to menu choices

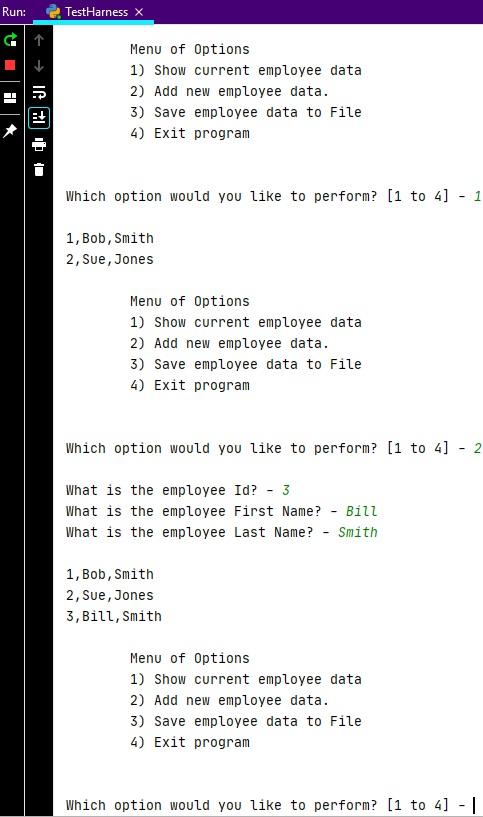
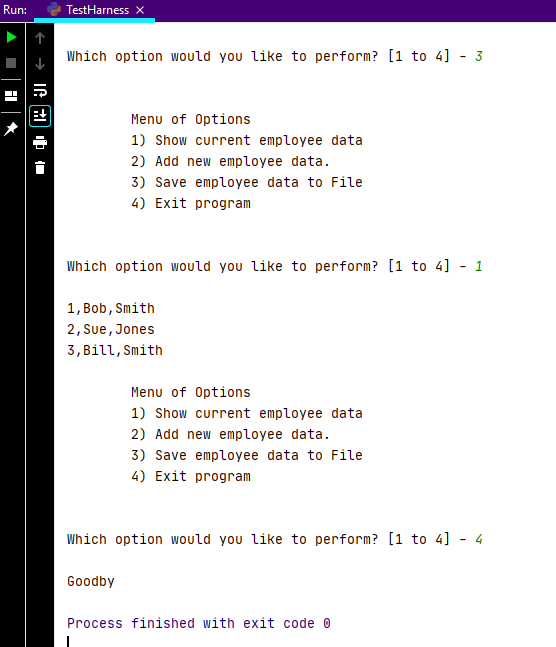
Figure 10 demonstrates each of the menu option in order running under PyCharm – reading the file, adding a new employee, saving the updated list, reading again from the file, and exiting.

Figure 10: Sample output from menu choices

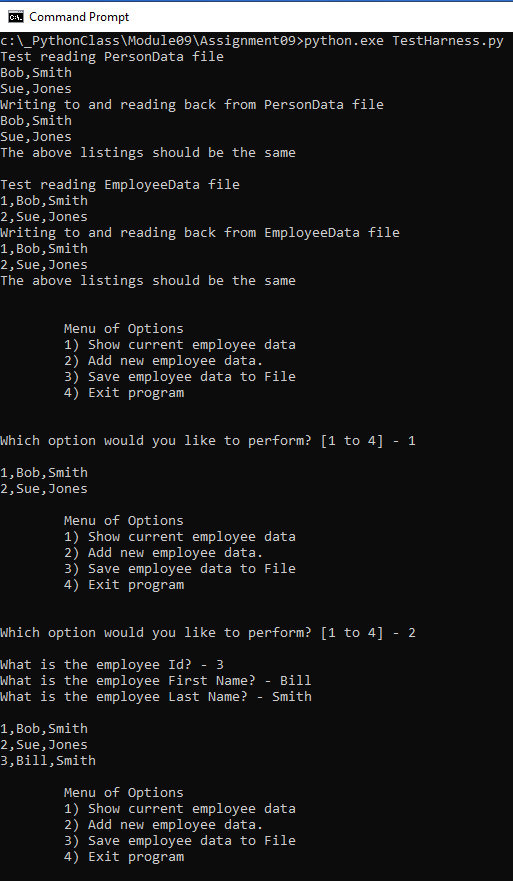
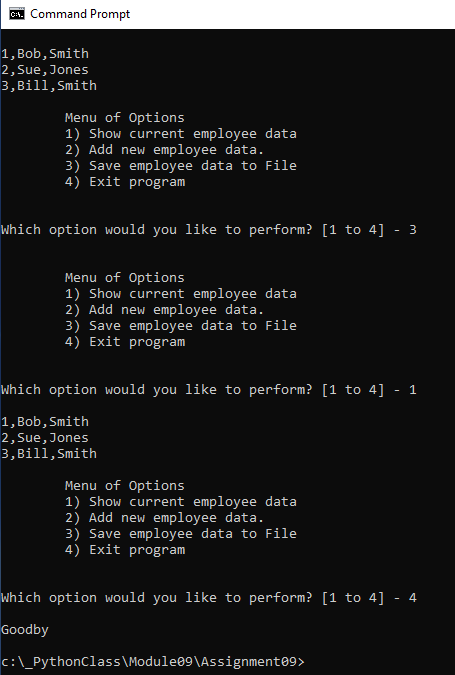
Figure 11 demonstrates each of the menu option in order running from the command prompt– reading the file, adding a new employee, saving the updated list, reading again from the file, and exiting.

Figure 11: Sample output from the command line

## Summary

Despite the fact that were essentially given all the code, it still took considerable time to work through all the details. I’m still not positive that I’ve demonstrated a harness properly. It would seem to me that in the end one needs to decide if this is actually a harness (for testing aspects of the program piece by piece) or whether it is actually a script for operating a simple employee record keeping system. If the latter, then I would think we would comment out lines 20-68.

I liked this week’s assignment which helped to clarify several issues. I think I have a much clearer picture of the importance of modules. Seemingly simple “printing to a physical printer” must involve many modules which pass the “to-be-printed-file” along from the word processor to modules supplied by the operating system, to others supplied by the printer manufacturer, etc.