Assignment 08: Objects & Classes

# Introduction

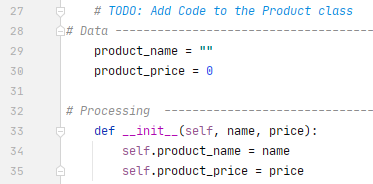
This week, students were introduced to organizing data or functions through the use of classes. We learned that when a class is focused on processing data, it is best to use it directly. However, if a class is used to store data, it is best to use it indirectly.

# Solution

Students were instructed to complete the starter file by filing in the remainder. Similar to past assignments, the user will open a file, read in the data, modify the list, save and close the file. The options they can choose is presented via a selection menu, accepting inputs (1-4) to perform the appropriate action. The script was ran successfully in PyCharm and Windows Command Processor.

# Class Product

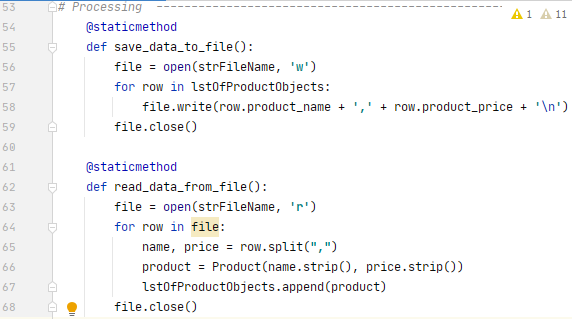
The product object is created by initially setting the product\_name and product\_price to blank and zero, respectively. These objected are defined starting in line 33 found in Figure 1.



*Figure 1*

# Processing

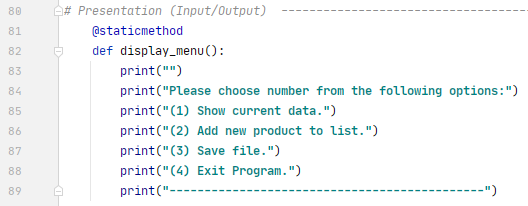
The processor class is responsible for reading and saving the data to a file. Saving the data involves a loop to iterate through each row of the list of products and writing the name and price to the file. After which, the file is closed. Reading the data from a file also uses a loop to iterate through each row in the file, splitting each row at a comma, and assigning each half to “name” and “price” and they are appended to the product objects list.



***Figure 2: Processing***

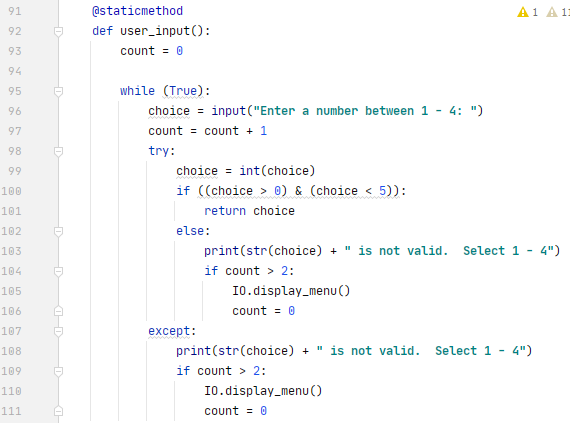
# Input/Output

Similar to past assignments, a user selection menu is presented to the user to prompt them for available options to select from.



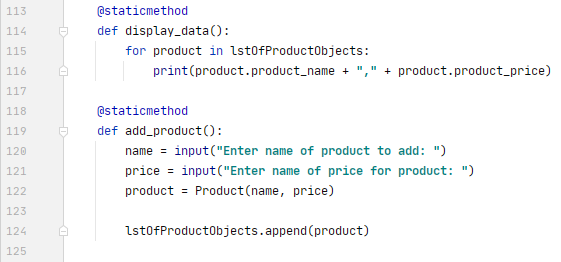
***Figure 3: Selection Menu***

User\_input utilized a counter to keep the menu available. I saw this technique from other’s in our Python course and decided to use. In past assignments, I used ‘continue’. Try-except was used for error catching and displaying further insight to the user if an invalid choice was made.



***Figure 4: User\_input***

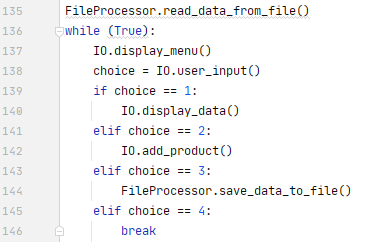
Display\_data utilized the listOfProductObjects from the above. A loop was used to iterate through each product and display the product and price. Add\_product appends the name and price inputs (together as “product”) to the existing product list (Figure 5).



***Figure 5: Display and Add data***

# Script Body

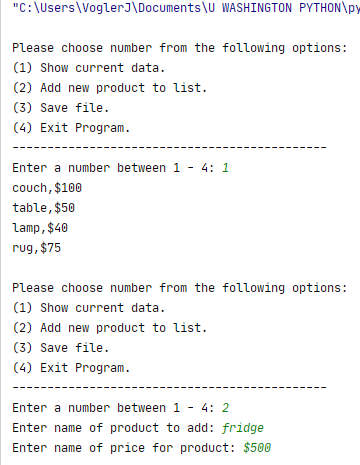
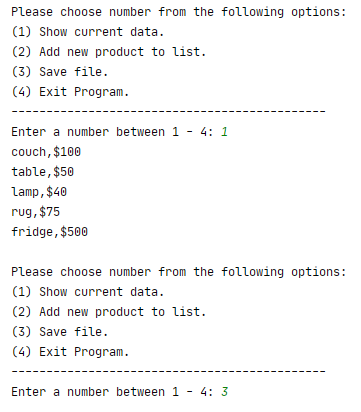
Figure 6 shows how the main body is simplified though the use of objects and functions. It is very clear what each user choice (1-4) is to perform.

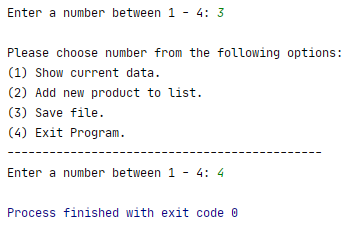
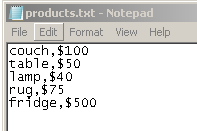


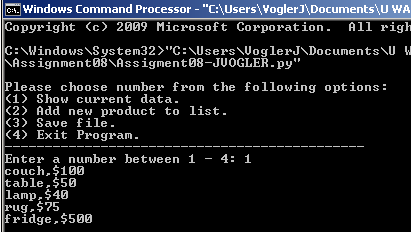
***Figure 6: User Selection and Functions***

# Summary and Running Program

The classes help to organize the code and makes it very readable and easy it to navigate when troubleshooting is required. Below demonstrates the code runs accurately on both Pycharm and Command Processor.



Demonstrating script runs as expected in Pycharm, validating saved file through running script via command processor and performing user menu selection #1: Show current data. We can see the new product (fridge, $500) is displayed.

# Command Processor and .txt

