Sherin Soman

September 04, 2021

Assignment 08

GitHub Repository: <https://github.com/SherinJoel/IntroToProg-Python-Mod08>

**Create a script using object-oriented programming techniques**

**Introduction**

This Assignment gives a basic overview of Classes and Exception Handling in python. In this Assignment, I explained the steps I have done to create a script using custom Functions, Classes and Exception Handling. I start my program by loading the data in a text file called 'programs.txt' into a python list of class objects. The script continues to display a menu of choices to the user until the user ask to save and exit the program. I used a printed "menu" to guide the user through this process.

## Python Classes and Objects

Python is an “object-oriented programming language”. Everything in Python is an object such as integers, lists, dictionaries, functions and so on. Every object has a type, and the object types are created using classes.

**Classes have:**

* Data attributes: Define what is needed to create an instance of a class and can be accessed using the dot (.) operator.
* Methods: Define how to interact with the instances of a class

Methods are just like functions, but they belong to a particular class.

## Creating a class

Classes are created by keyword 'class'.

*class Student:*

*pass*

**Class Object**

An Object is an instance of a Class. A class is like a blueprint while an instance is a copy of the class with actual values.

**Declaring Objects (Also called instantiating a class)**

When an object of a class is created, the class is said to be instantiated. All the instances share the attributes and the behavior of the class. But the values of those attributes, i.e., the state are unique for each object. A single class may have any number of instances.

*stud = Student ()*

**The Self Parameter**

Class methods must have an extra first parameter 'self ' in the method definition. In Python, you identify which copy is referenced using the pronoun "self."

**Class and Instance Variables**

Instance variables are variables whose value is assigned inside a constructor or method with self-whereas class variables are variables whose value is assigned in the class.

**Static Methods**

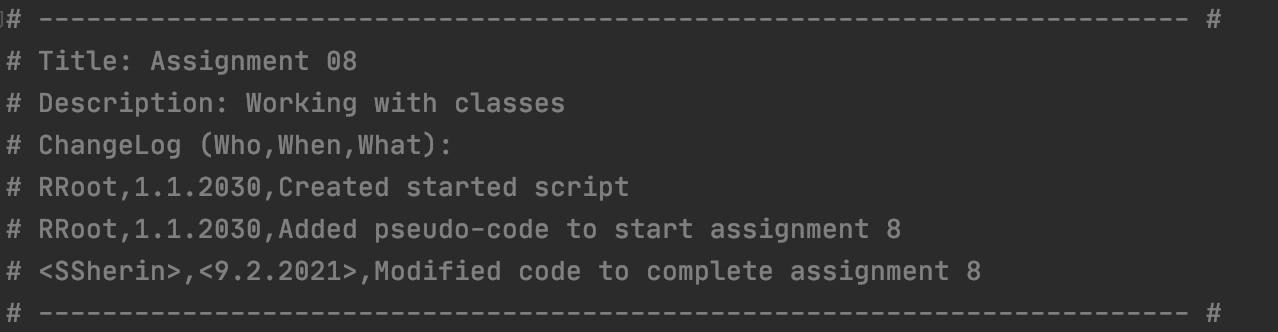
Classes can have both instance methods and static methods, but most will not. In general, when a class focuses on processing data, use "static" methods. However, when a class focuses on storing data, use "instance" methods (the ones with self).

**Create a new project in Pycharm**

To create a new Project in PyCharm, I created a sub-folder called Assignment 08 inside of the \_PythonClass folder and used \_PythonClass\Assignment08 as its location to create the new project. Within the project, I added the python file, "Assignment08-Starter.py.

**Code Explanation**

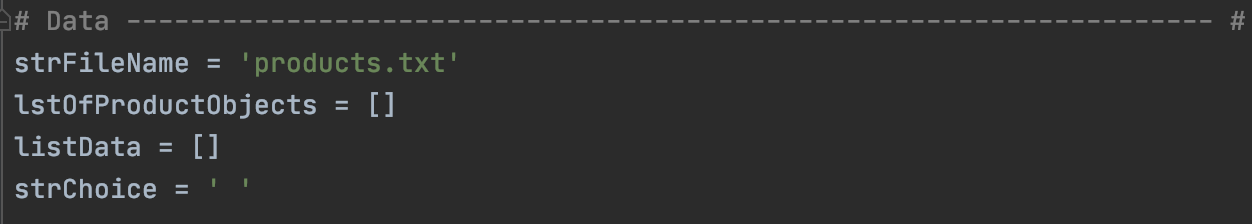
I started my script by adding the script's header.

*Figure 1: Script Header*

**Declaring variables and constants**

Variables are containers to store values. Variable declared outside of the function or in global scope is known as a global variable. This means that a global variable can be accessed inside or outside of the function.

A variable declared inside the function's body or in the local scope is known as a local variable.

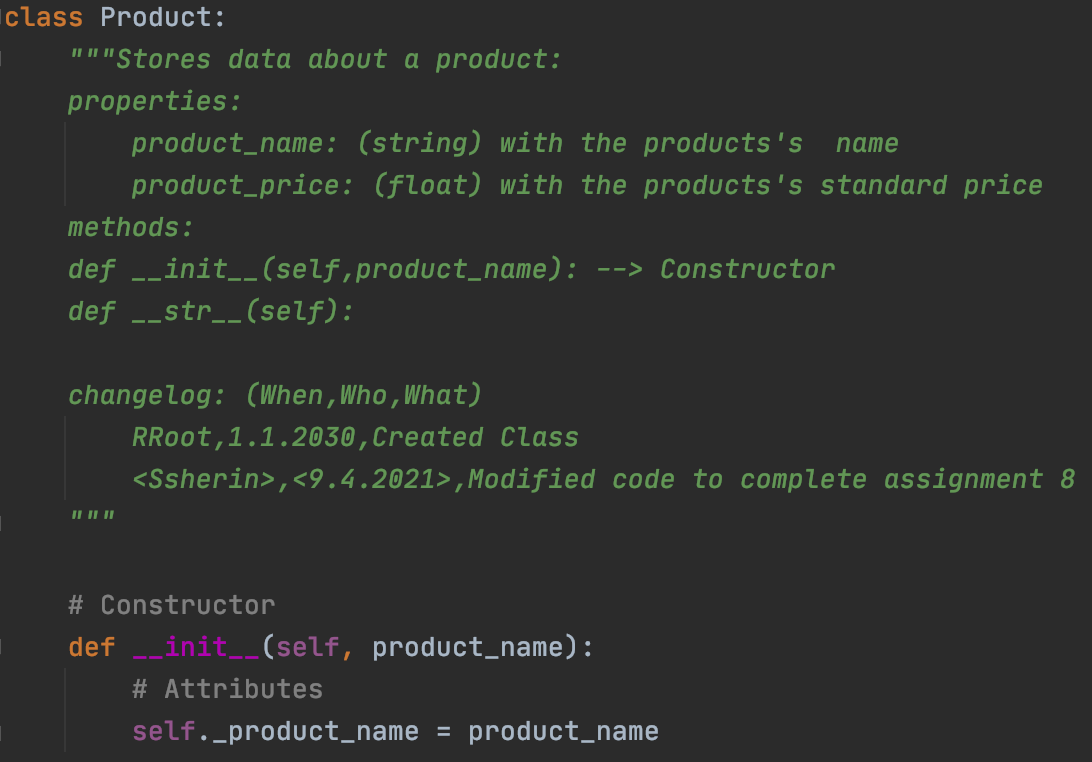
*Figure 2: Declaring Variables and Constants*

**Storing data about a product – class Product**

In this script, the product attributes, constructor, getters and setters are defined in a Class Product.

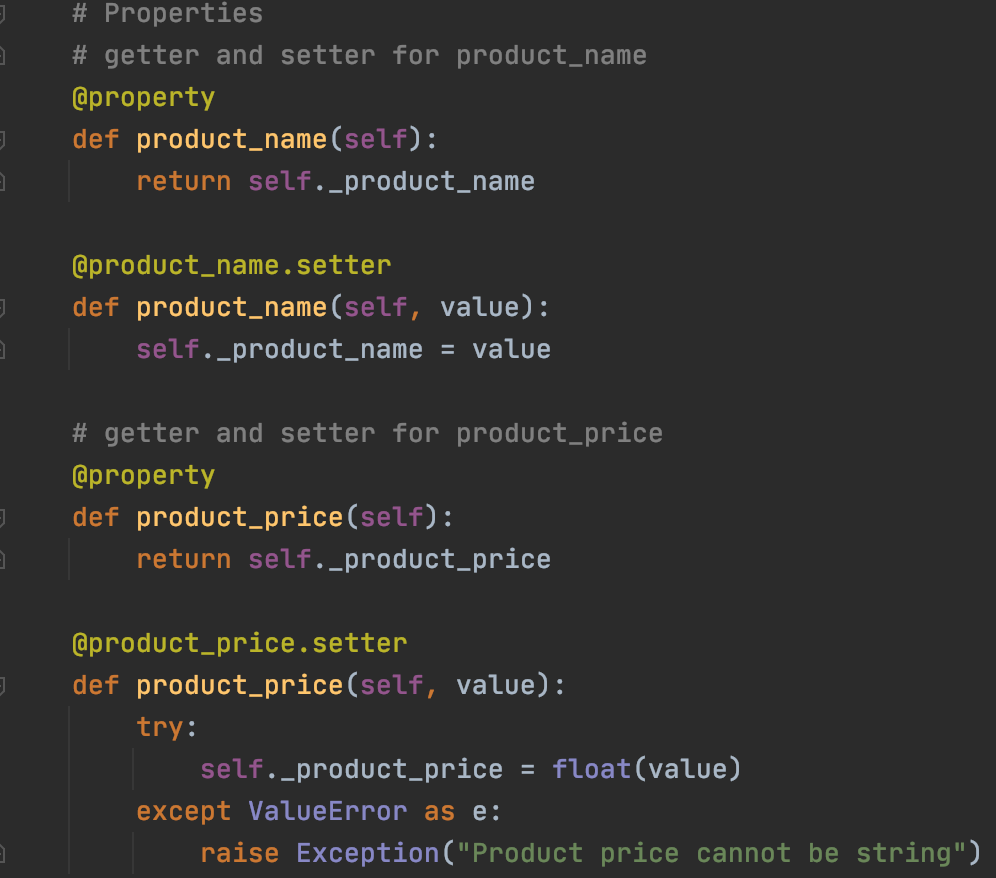
**\_\_init\_\_ Method**

The \_\_init\_\_ is a special function that is automatically executed when an instance of class is created. It is also called class constructor. The parameters of the init function represent the data attributes of a class. The method is useful to do any initialization you want to do with your object.

*Figure 3: Defining the instance attributes using the constructor method.*

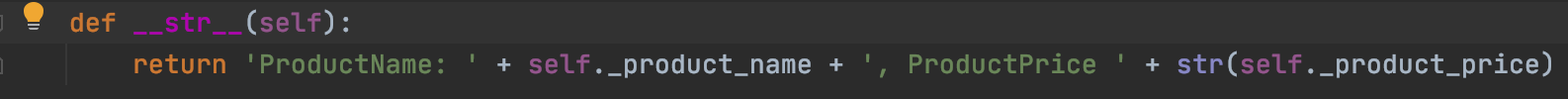
**Getter and Setter Properties**

Maintaining data encapsulation is the key objective of using getters and setters in object-oriented programs. One method to create getter and setter functions is by using @property decorators to accomplish the actions of getters and setters. One of the built-in decorators is python @property.

*Figure 4: Getters and Setters for product name and price*

**The \_\_str\_\_() Method**

This method returns the string representation of the object. This method is called when print () or str() function is invoked on an object. This method must return the String object. If we don’t implement \_\_str\_\_() function for a class, then built-in object implementation is used that actually calls \_\_repr\_\_() function which returns the object representation in string format.

*Figure 5: Defining the \_\_str\_\_ method*

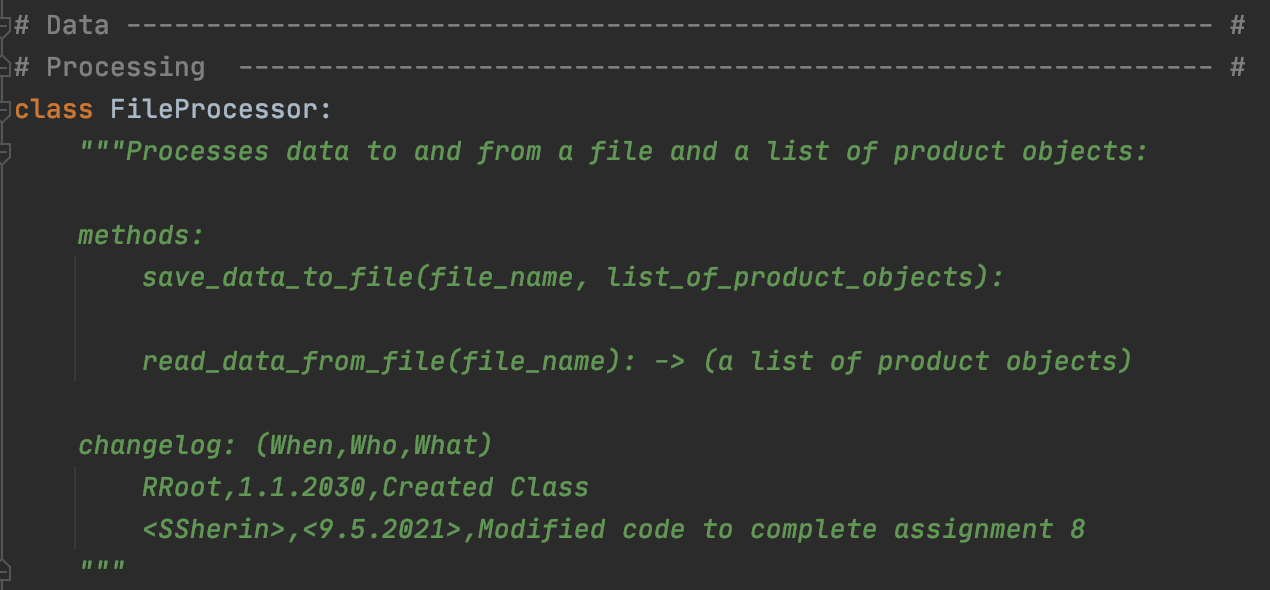
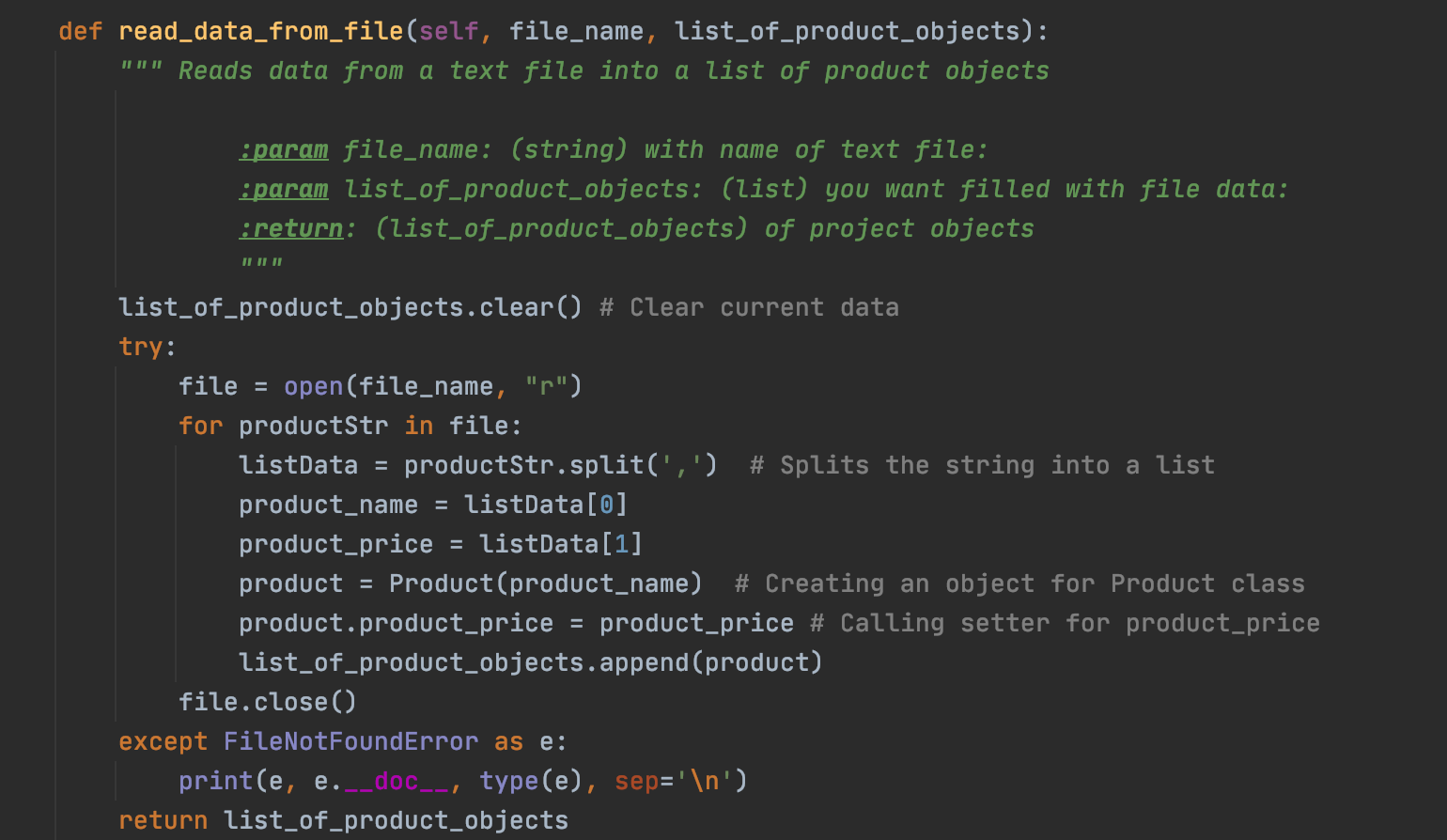
**Performing File Processing Tasks – class FileProcessor**

In this script, file tasks such as reading and writing to a file are defined in a Class FileProcessor.

***Reading data from a file into a list of Product objects***

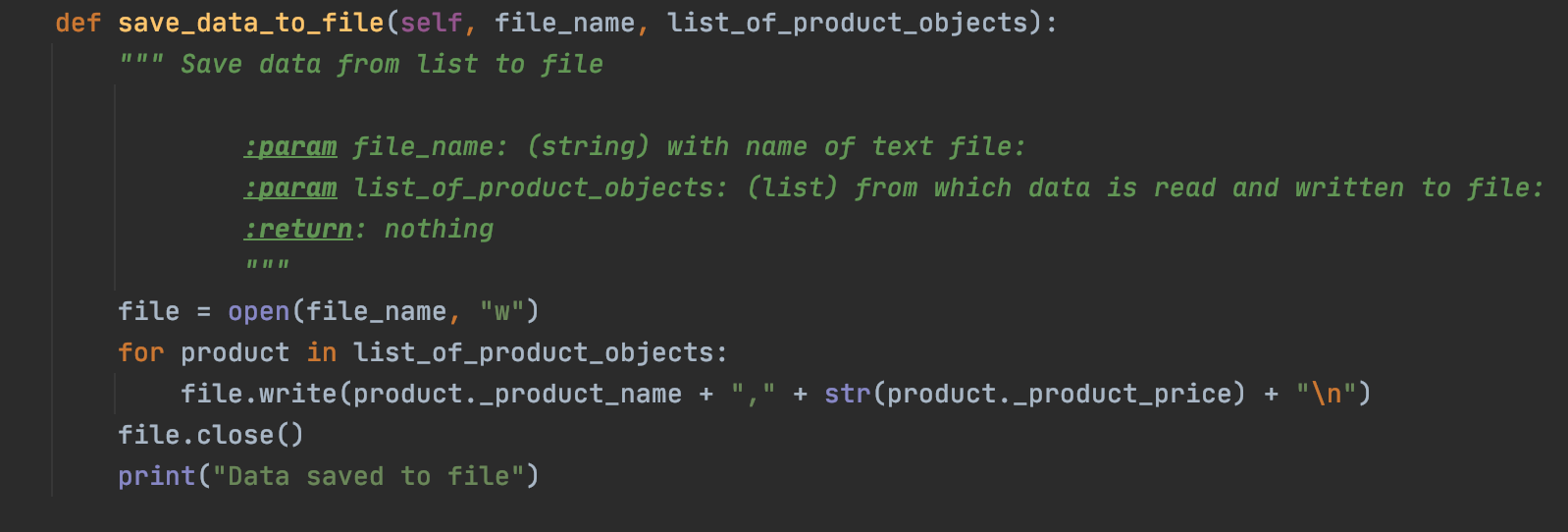
This task is defined by a function which returns the list of product objects. To read data from a file, the file is opened in “r” mode. Using a for loop, each row(string) in the text file is read and splits at the specified separator (comma) using the split () method into a list and is stored in the list variable listData. The value of product\_name is set by creating an instance of Product class. Product\_price is set by calling the setter method for product price. These values are then appended to the list as class objects using the append (). The file is closed using the close ().

If the file to be read is not found, an exception FileNotFoundError is handled.

*Figure 6: Reading data from a file*

***Write data to file***

The file is opened in write mode and the elements from the list are written to the file.

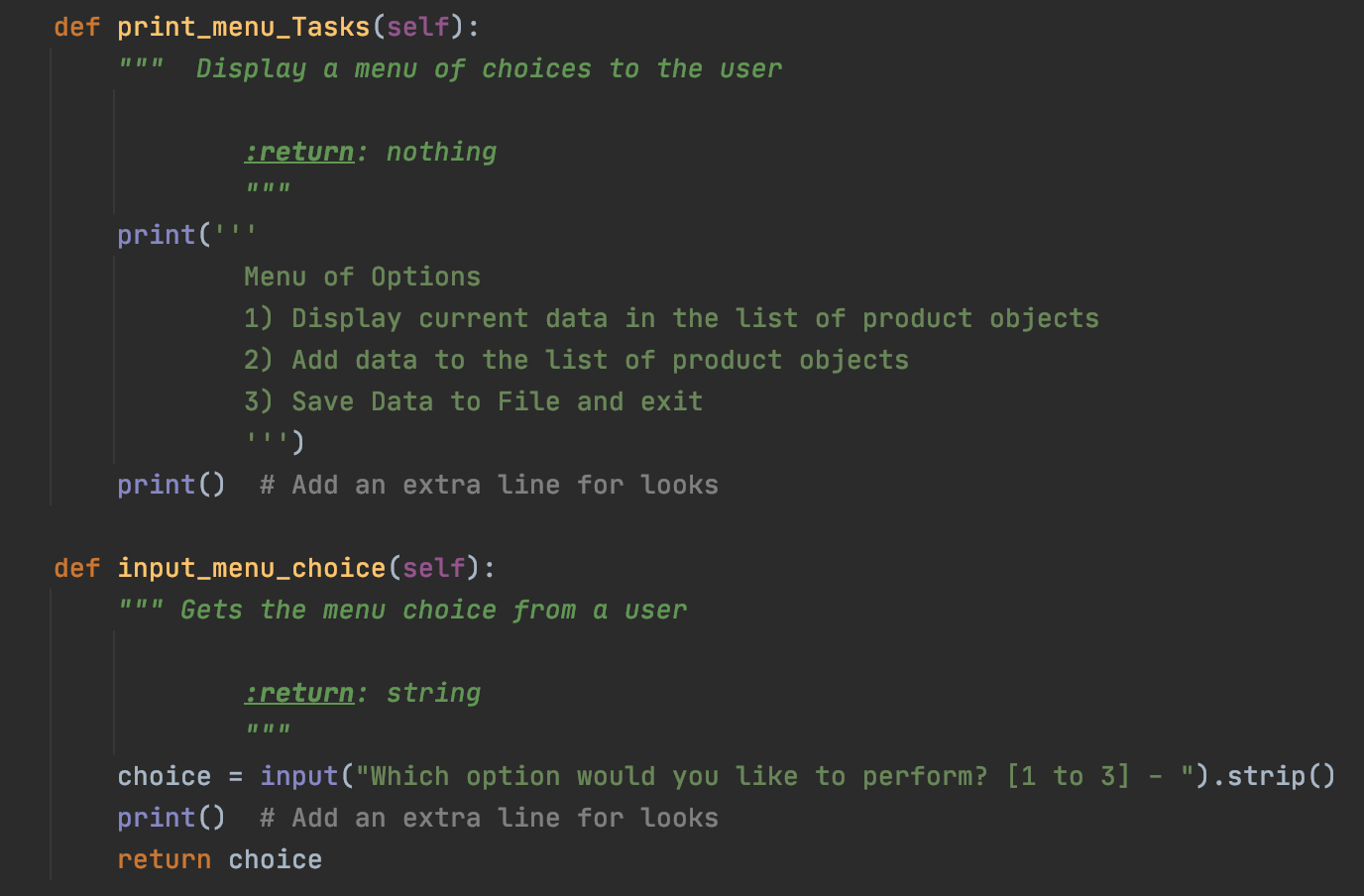
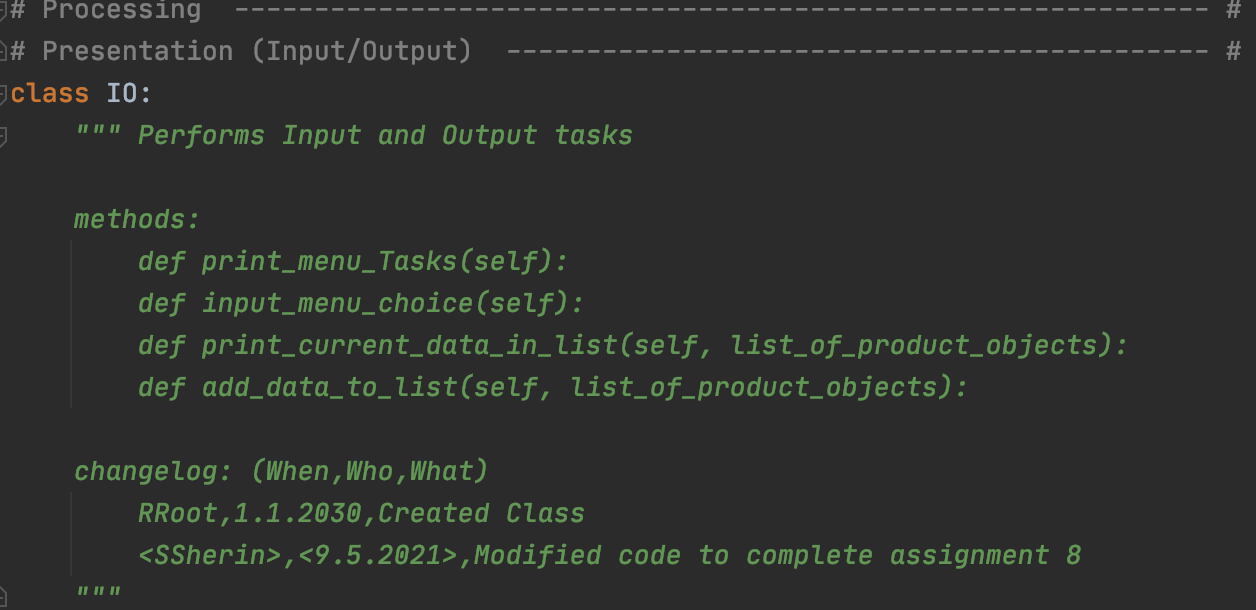
*Figure 7: Writing Data to a File*

**Performing Input and Output Tasks – class IO**

The input received from the user and the output displayed are defined in a class IO.

***Displaying a menu of choices to the user and getting the user input choice***

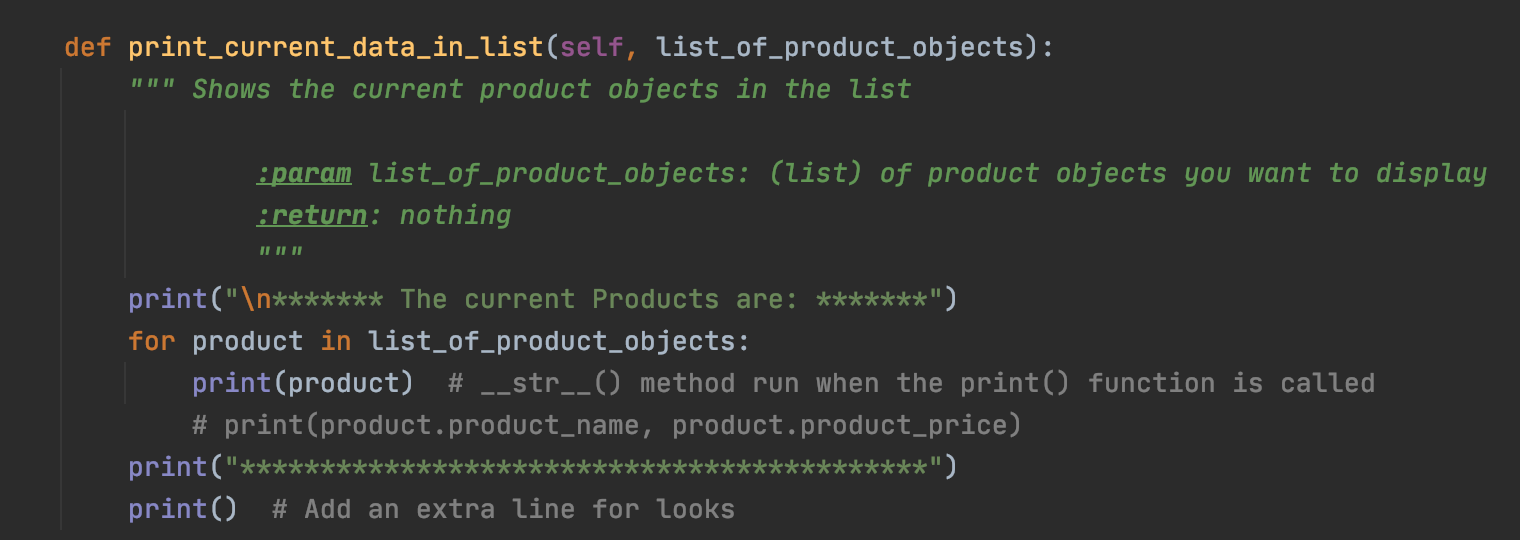
Menu choices are displayed to the user using the function def print\_menu\_Tasks. The function input\_menu\_choice gets the user input choice of which task to be performed and returns the choice.



*Figure 8: Displaying menu choices and getting user input choice*

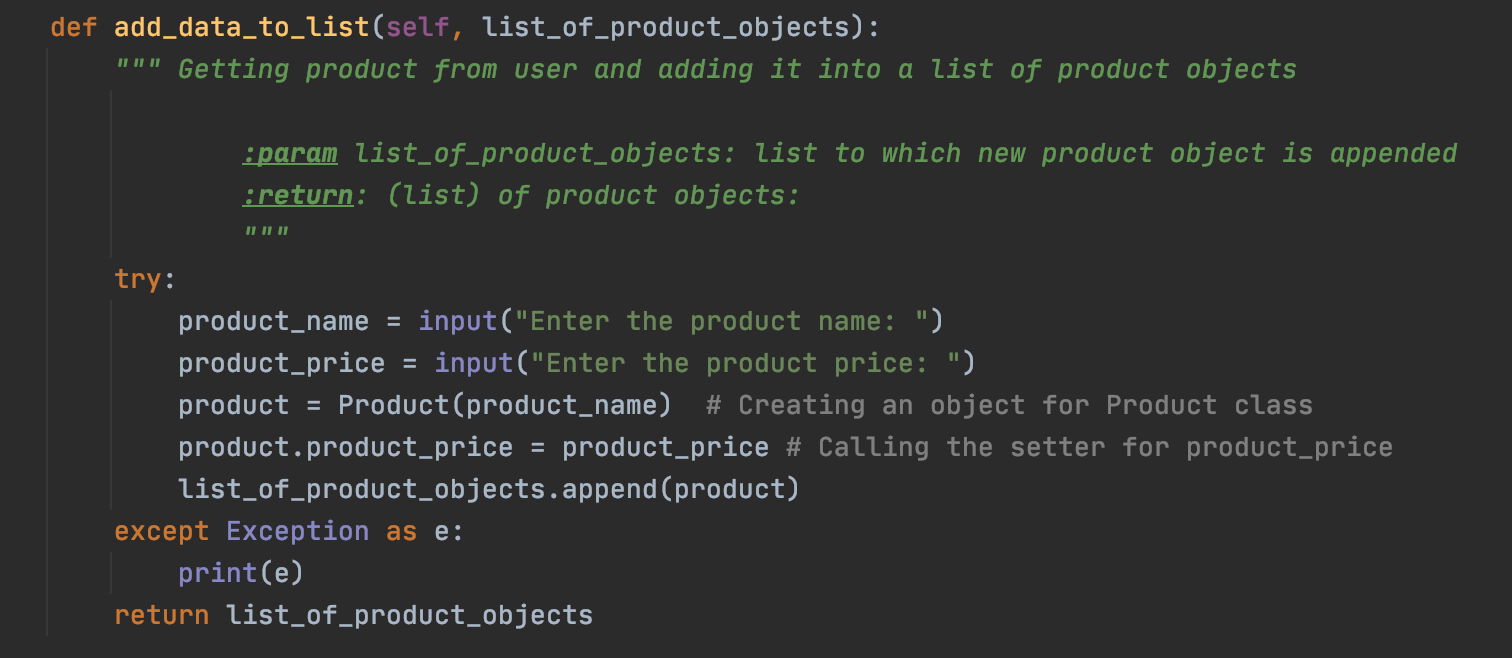
***Displaying current data in the list of product objects***

Using a for loop, each product object in the list is printed.The \_\_str\_\_() method is run when the print() is called.

*Figure 9: Displaying current product objects in List*

***Adding new product to List***

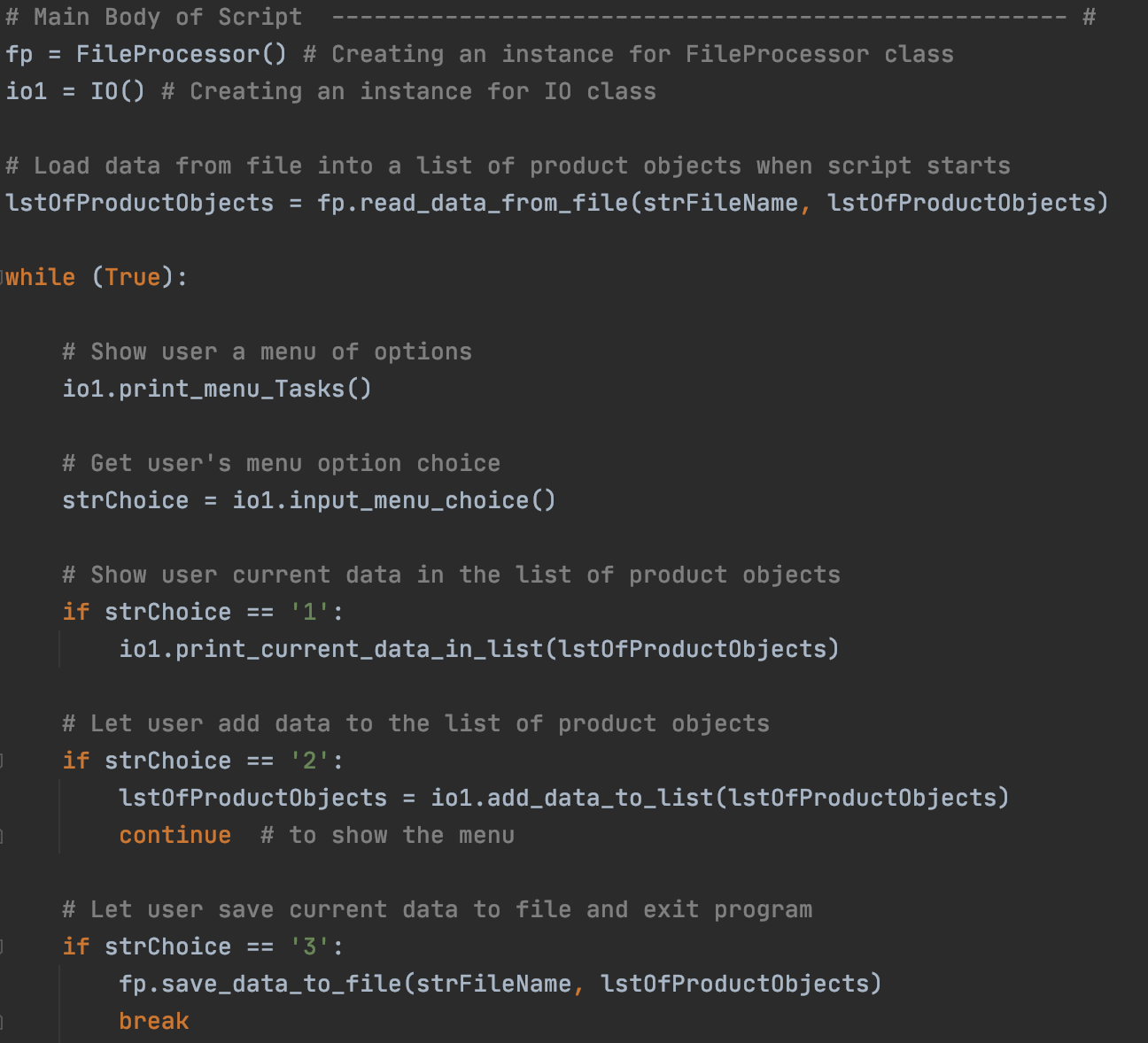
This function gets the new product and price from the user and returns the list after appending the data. An error is handled if the price entered is not a number.

*Figure 10: Function adding new task and priority to the list*

**Calling functions in the main body of the script**

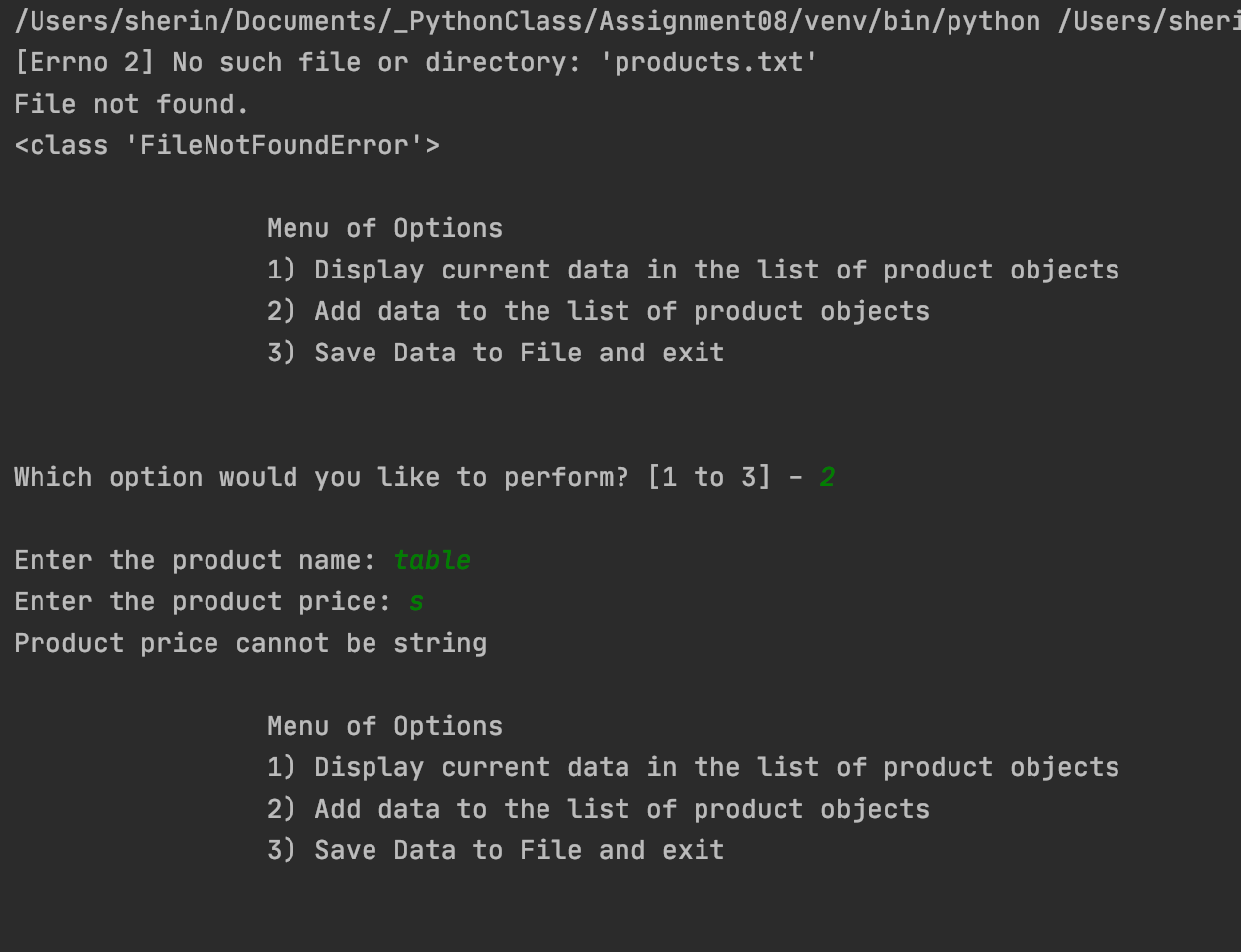
As the program starts, the data from the file is loaded into the list of product objects by calling the function read\_data\_from\_file in the Class FileProcessor.

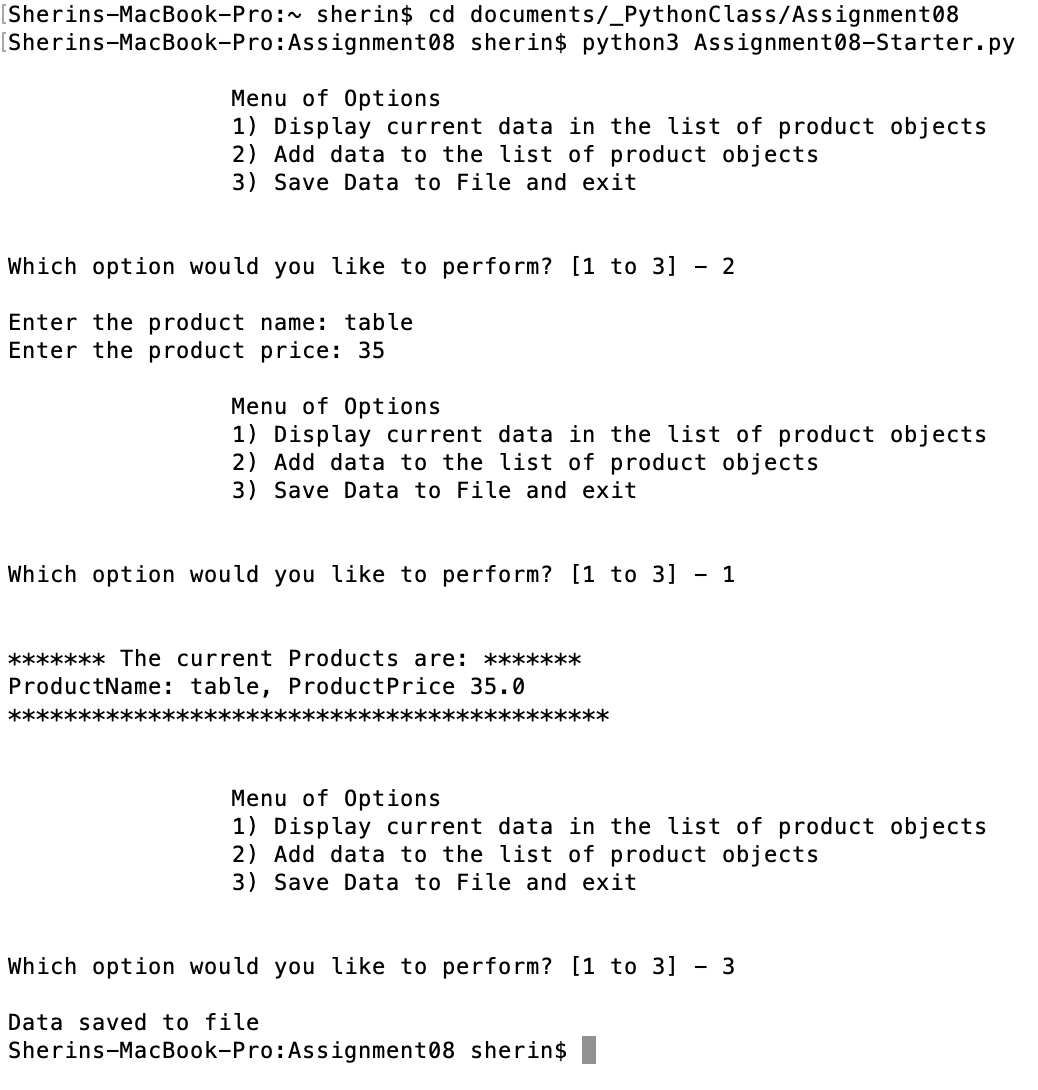
Then the menu choices are displayed, and the user input choice is entered.

Figure 11: Calling Functions to load data from file into a list of product objects and to display current data, add new product and to save to file and exit

***Running the Script***

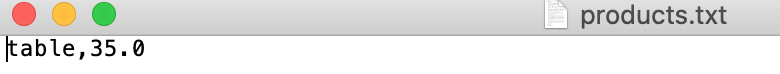
With the script created in its proper location, I run the script in both PyCharm (Figure 12) and an OS command/shell window (Figure 13).

Figure 12: Running the Script in Python Console

*Figure 13: A screenshot of the script running in Command Window.*

***Verifying the result***

Locate the text file and open it in a text editor



*Figure 14: Verifying that the file contains data*

***Summary***

In this assignment, I was able to write a python script using functions, lists, classes, and text files. The script is executed both in PyCharm and in command window to verify the results and the script ran as expected. Also, I Installed GitHub Desktop and logged into my GitHub account from GitHub Desktop.