Ryan Clabots

March 6, 2022

Foundations of Programming (Python)

Assignment 08: Working with Classes

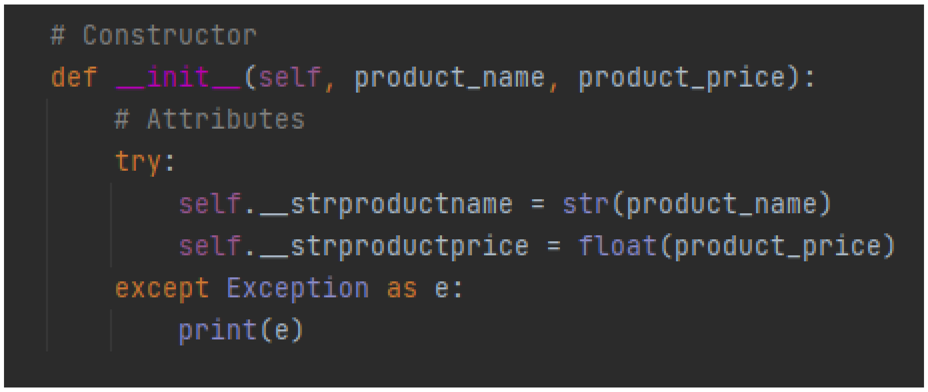
Github Repository Link: Lazyboy369/IntroToProg-Python-Mod08 (github.com)

Task: Demonstrate examples of pickling data as well as error handling

Introduction

This week we started working on how python handles classes. The script below was filled in from a starter script where a list of products is created.

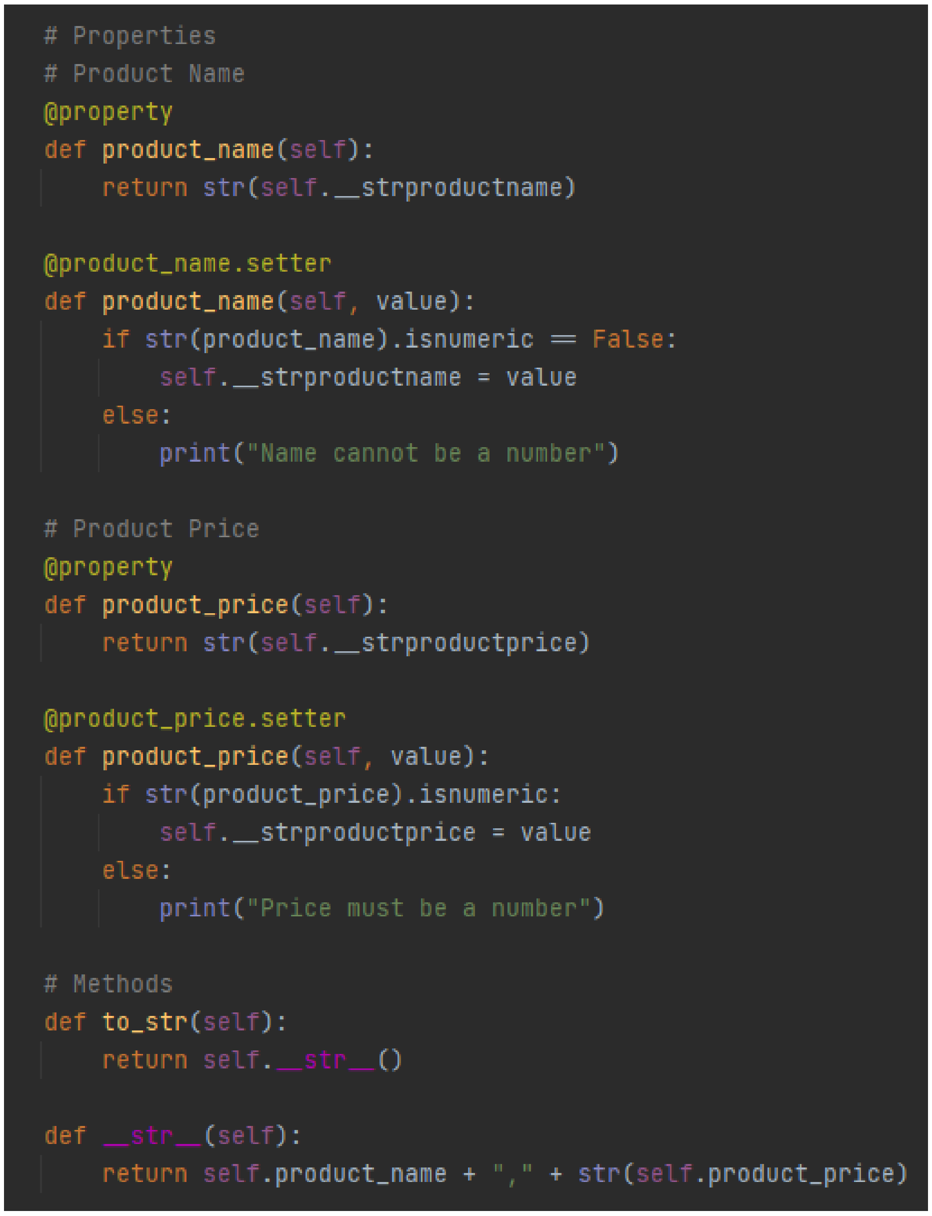
Writing the Script



```
# Constructor
def __init__(self, product_name, product_price):
    # Attributes
    try:
        self.__strproductname = str(product_name)
        self.__strproductprice = float(product_price)
    except Exception as e:
        print(e)
```

Figure 1 – Initializing of the constructor and Attributes

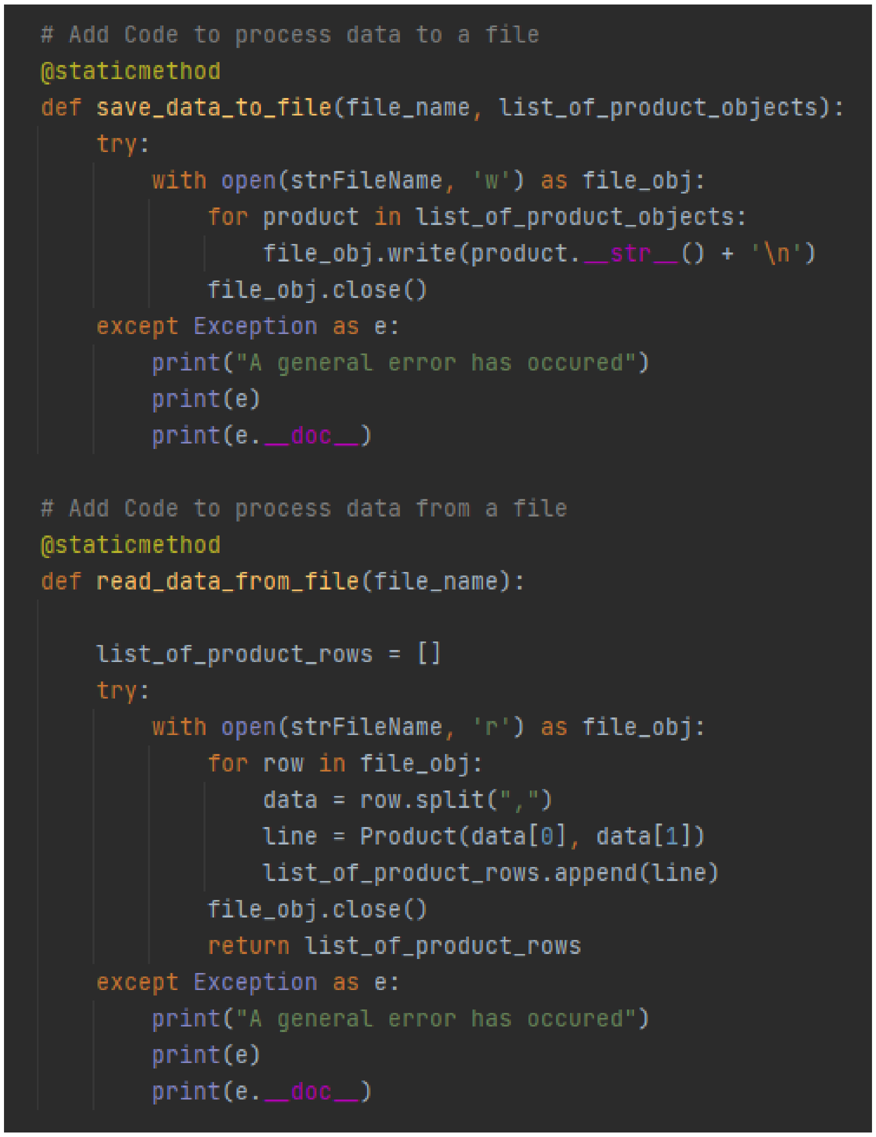
At the start of the script, the class is created along with an initializing constructor with an attribute for the product name and product price. Double underscore is used to make the attributes private.



```
# Properties
# Product Name
@property
def product_name(self):
    return str(self.__strproductname)

@product_name.setter
def product_name(self, value):
    if str(product_name).isnumeric == False:
        self.__strproductname = value
    else:
        print("Name cannot be a number")

# Product Price
@property
def product_price(self):
    return str(self.__strproductprice)

@product_price.setter
def product_price(self, value):
    if str(product_price).isnumeric:
        self.__strproductprice = value
    else:
        print("Price must be a number")

# Methods
def to_str(self):
    return self.__str__()

def __str__(self):
    return self.product_name + "," + str(self.product_price)
```

Figure 2 – Properties and methods for the Product class

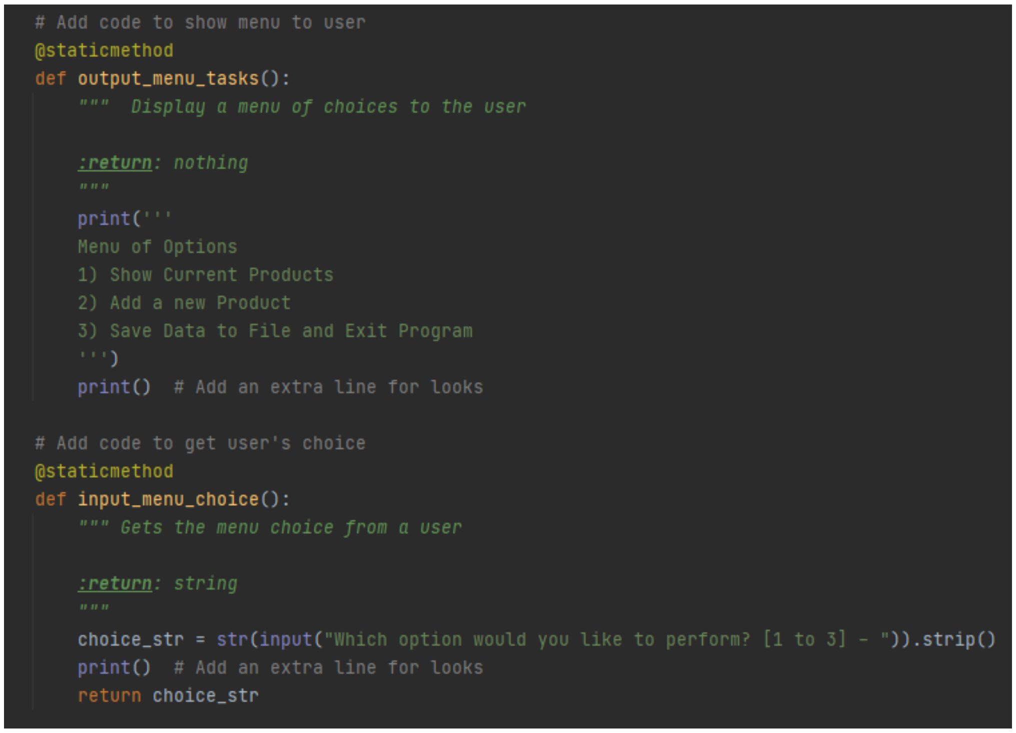
Getter and setter properties are then created for the product name and price. A custom string method is used to format the output code later on in the script.



```python
# Add Code to process data to a file
@staticmethod
def save_data_to_file(file_name, list_of_product_objects):
    try:
        with open(strFileName, 'w') as file_obj:
            for product in list_of_product_objects:
                file_obj.write(product.__str__() + '\n')
            file_obj.close()
    except Exception as e:
        print("A general error has occured")
        print(e)
        print(e.__doc__)

# Add Code to process data from a file
@staticmethod
def read_data_from_file(file_name):

    list_of_product_rows = []
    try:
        with open(strFileName, 'r') as file_obj:
            for row in file_obj:
                data = row.split(",")
                line = Product(data[0], data[1])
                list_of_product_rows.append(line)
            file_obj.close()
            return list_of_product_rows
    except Exception as e:
        print("A general error has occured")
        print(e)
        print(e.__doc__)
```

Figure 3 – FileProcessing functions

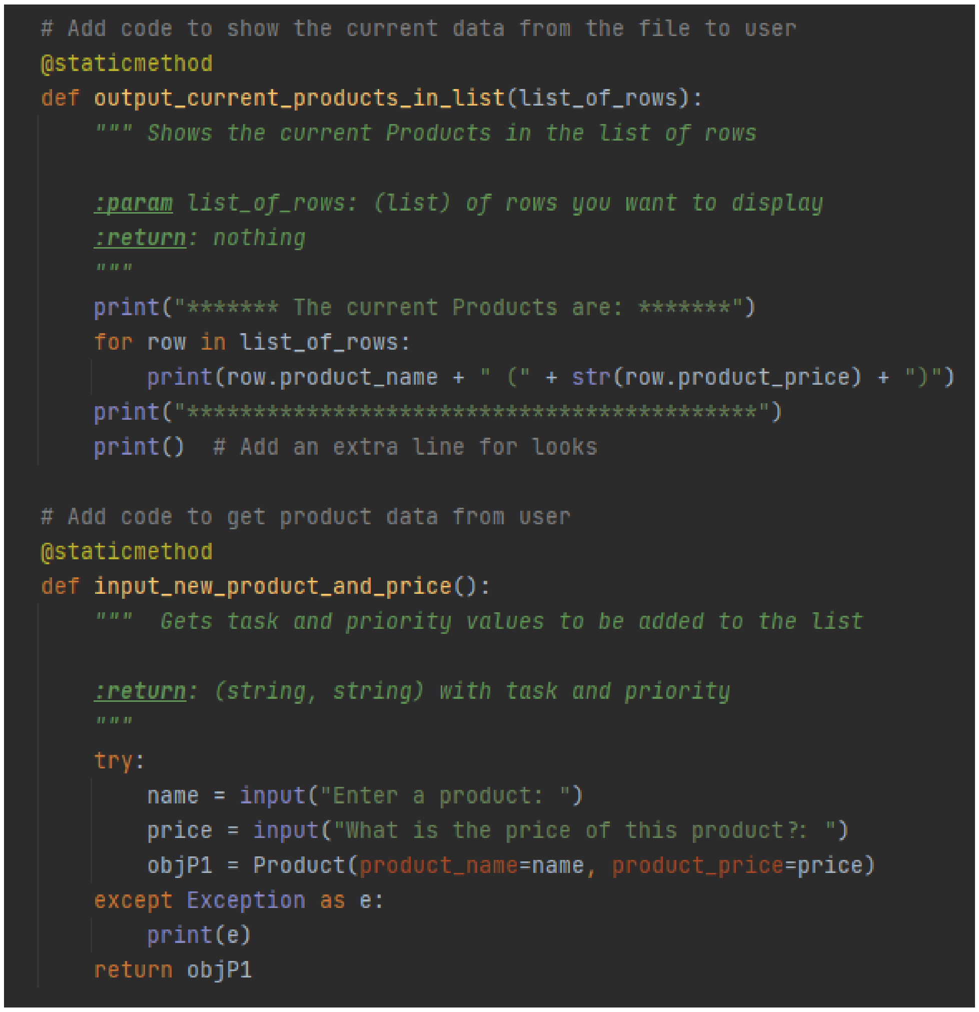
A new class for saving and reading data from a text file. The save data function is established with the inputs being the text file and the list of objects. The for loop loops through the list of objects and writes it to the text file using the custom string method from figure 2.



```
# Add code to show menu to user
@staticmethod
def output_menu_tasks():
    """  Display a menu of choices to the user

    :return: nothing
    """
    print('''
    Menu of Options
    1) Show Current Products
    2) Add a new Product
    3) Save Data to File and Exit Program
    ''')
    print()  # Add an extra line for looks

# Add code to get user's choice
@staticmethod
def input_menu_choice():
    """ Gets the menu choice from a user

    :return: string
    """
    choice_str = str(input("Which option would you like to perform? [1 to 3] - ")).strip()
    print()  # Add an extra line for looks
    return choice_str
```

Figure 4 – First half of the Input/Output functions

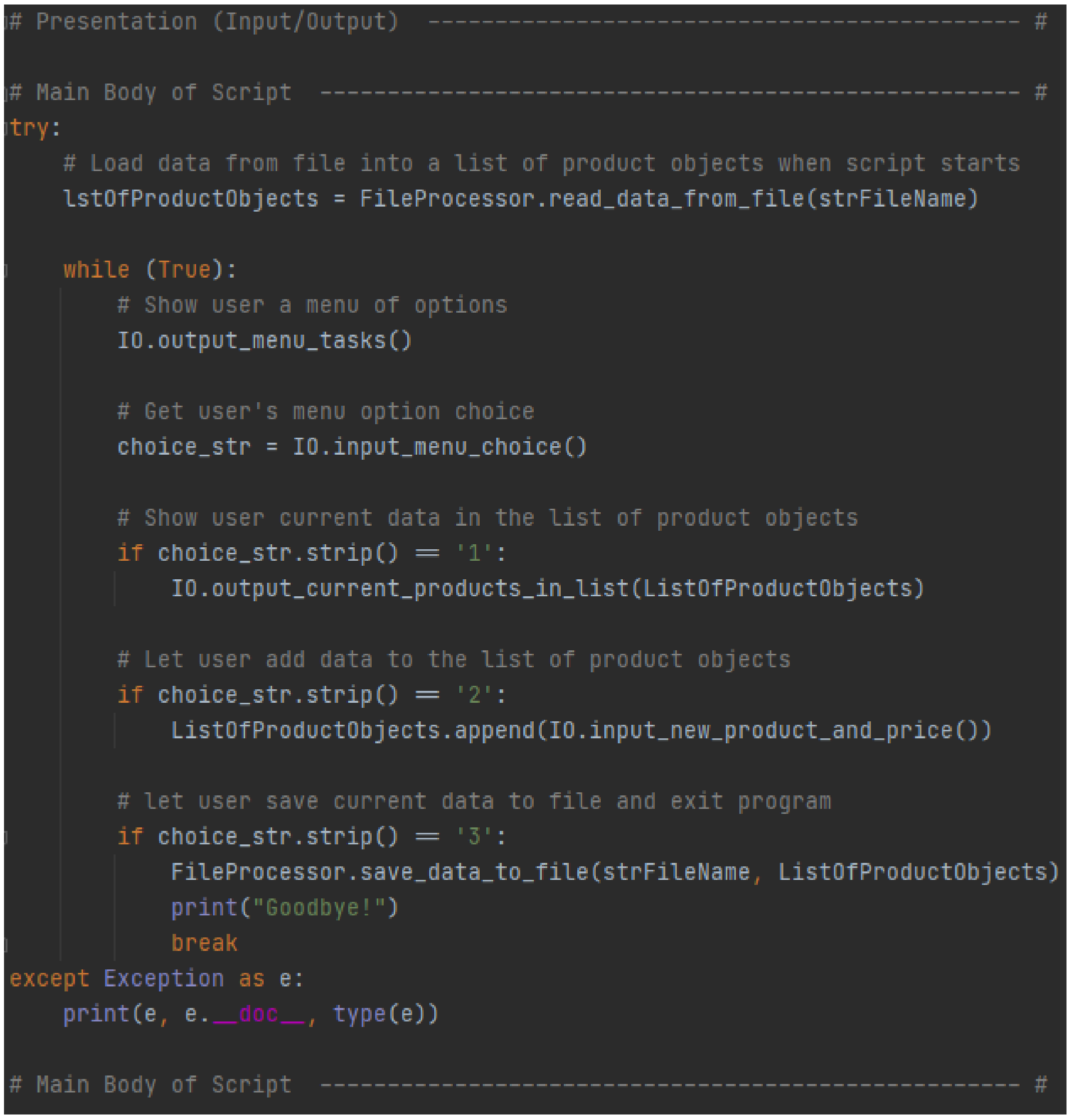
Another class is created for the inputs and outputs of the script. The output menu and menu input choice functions.



```python
# Add code to show the current data from the file to user
@staticmethod
def output_current_products_in_list(list_of_rows):
    """ Shows the current Products in the list of rows

    :param list_of_rows: (list) of rows you want to display
    :return: nothing
    """
    print("******* The current Products are: *******")
    for row in list_of_rows:
        print(row.product_name + " (" + str(row.product_price) + ")")
    print("*******************************************")
    print()  # Add an extra line for looks

# Add code to get product data from user
@staticmethod
def input_new_product_and_price():
    """  Gets task and priority values to be added to the list

    :return: (string, string) with task and priority
    """
    try:
        name = input("Enter a product: ")
        price = input("What is the price of this product?: ")
        objP1 = Product(product_name=name, product_price=price)
    except Exception as e:
        print(e)
    return objP1
```

Figure 5 – Second half of the Input/Output functions

The output current products in list function takes each row and uses the product name getter properties from the beginning before printing out the product instance of the product name and product price. The input new product function creates a name and price variable before saving it into the Product class.

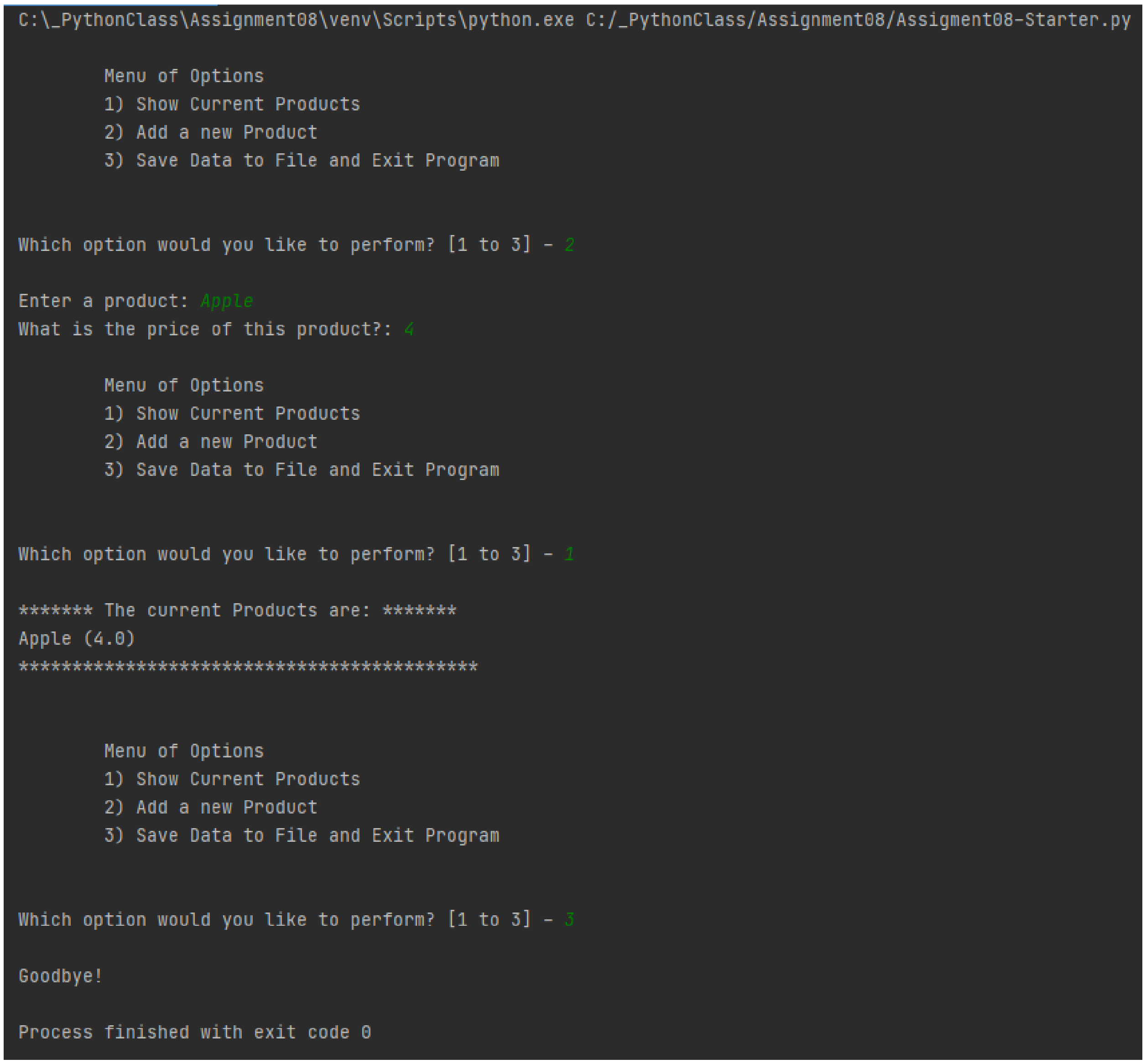


```python
# Presentation (Input/Output)  -------------------------------------------- #

# Main Body of Script  ---------------------------------------------------- #
try:
    # Load data from file into a list of product objects when script starts
    lstOfProductObjects = FileProcessor.read_data_from_file(strFileName)

    while (True):
        # Show user a menu of options
        IO.output_menu_tasks()

        # Get user's menu option choice
        choice_str = IO.input_menu_choice()

        # Show user current data in the list of product objects
        if choice_str.strip() == '1':
            IO.output_current_products_in_list(ListOfProductObjects)

        # Let user add data to the list of product objects
        if choice_str.strip() == '2':
            ListOfProductObjects.append(IO.input_new_product_and_price())

        # let user save current data to file and exit program
        if choice_str.strip() == '3':
            FileProcessor.save_data_to_file(strFileName, ListOfProductObjects)
            print("Goodbye!")
            break
except Exception as e:
    print(e, e.__doc__, type(e))

# Main Body of Script  ---------------------------------------------------- #
```

Figure 6 – Main portion of the script where each of the functions are called

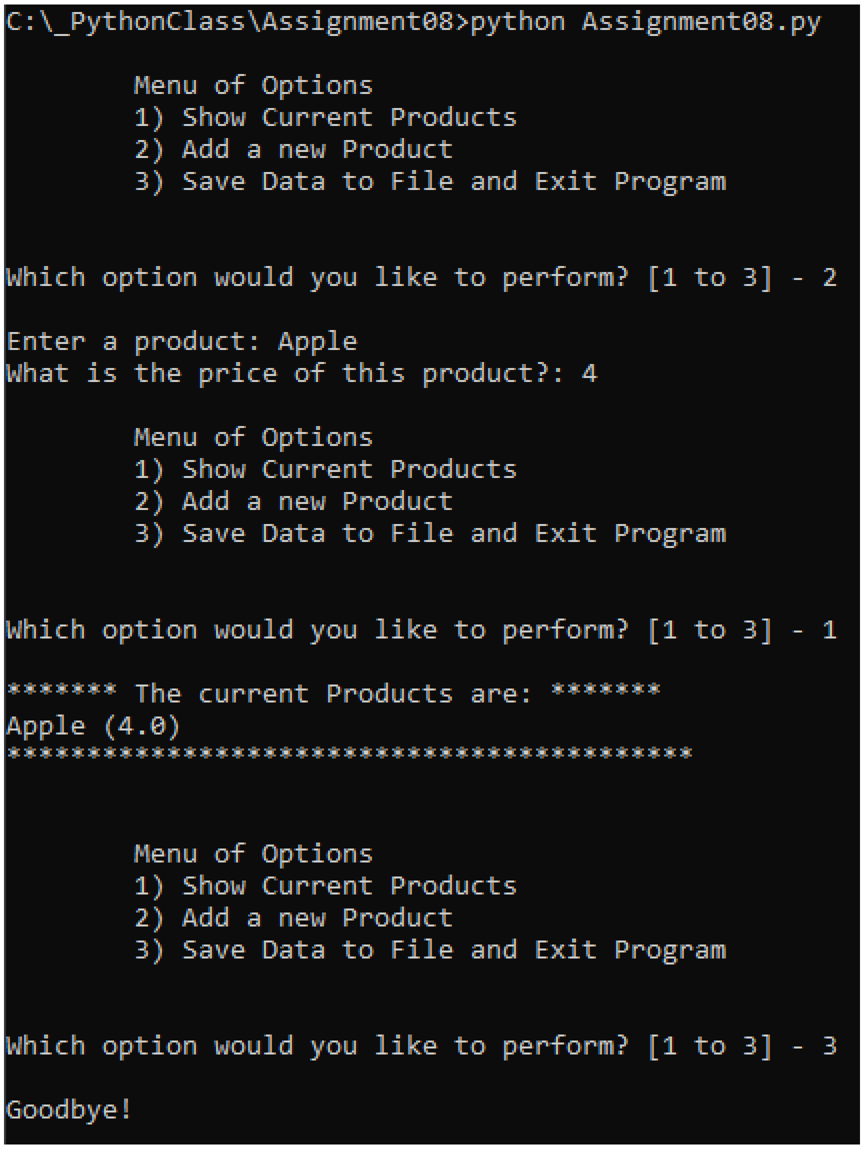
The last portion of the script is the main meat. It starts by reading data from the file and filling in the to the empty list created at the start. A while loop is opened and loops through the options of the menu.

Running the Code



```
C:\_PythonClass\Assignment08\venv\Scripts\python.exe C:/_PythonClass/Assignment08/Assigment08-Starter.py

        Menu of Options
        1) Show Current Products
        2) Add a new Product
        3) Save Data to File and Exit Program


Which option would you like to perform? [1 to 3] - 2

Enter a product: Apple
What is the price of this product?: 4

        Menu of Options
        1) Show Current Products
        2) Add a new Product
        3) Save Data to File and Exit Program


Which option would you like to perform? [1 to 3] - 1

******* The current Products are: *******
Apple (4.0)
*******************************************


        Menu of Options
        1) Show Current Products
        2) Add a new Product
        3) Save Data to File and Exit Program


Which option would you like to perform? [1 to 3] - 3

Goodbye!

Process finished with exit code 0
```

Figure 7 – Basic operation of the script



```
C:\_PythonClass\Assignment08>python Assignment08.py

        Menu of Options
        1) Show Current Products
        2) Add a new Product
        3) Save Data to File and Exit Program


Which option would you like to perform? [1 to 3] - 2

Enter a product: Apple
What is the price of this product?: 4

        Menu of Options
        1) Show Current Products
        2) Add a new Product
        3) Save Data to File and Exit Program


Which option would you like to perform? [1 to 3] - 1

******* The current Products are: *******
Apple (4.0)
*******************************************



        Menu of Options
        1) Show Current Products
        2) Add a new Product
        3) Save Data to File and Exit Program


Which option would you like to perform? [1 to 3] - 3

Goodbye!
```

Figure 8 – Script running in the CMD shell