Jessica Wray
05 December 2021
Foundations of Programming, Python
Assignment 08
GitHub URL: https://github.com/jkwray/IntroToProg-Python-Mod08

Using Classes and Objects in Python

Introduction

This week's assignment dives into the larger programming concept of object-oriented programming (OOP). We have used classes in past assignments but hadn't really dug into the differences between classes and objects nor the different elements that can be part of a class. The script we wrote this week allowed us to practice using these concepts and helped us gain a greater understanding of them.

Creating the Script

I began with the starter script that was provided and did a first pass so I knew what was working with. I started from the beginning and built out the Product class using examples from this week's notes. I also added in a custom definition for converting to a string that would make it easier to print out the list of documents later on as seen below.

```python
class Product:
    """Stores data about a product:

    properties:
        product_name: (string) with the products's  name
        product_price: (float) with the products's standard price
    methods:
    changelog: (When,Who,What)
        RRoot,1.1.2030,Created Class
        JWray,12.05.2021,Modified code to complete assignment 8
    """

    # -- Constructor --
    def __init__(self, product_name, product_price):
        # -- Attributes --
        self.__product_name = product_name
        self.__product_price = '%.2f' % float(product_price)

    # -- Properties --
    # Product Name
    @property
    def product_name(self):
        return self.__product_name

    @product_name.setter
    def product_name(self, value):
        self.__product_name = value

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

    @product_price.setter
    def product_price(self, value):
        self.__product_price = value

    # -- Methods --
    def __str__(self):
        return self.product_name + ', ' + self.product_price
```

Figure 1: Product Class

Following that, I worked my way down the code and filled in the missing code. After every method, I tested it in the main script to make sure it worked before moving onto the next. You can see the rest of the class code in the attached script file.

Finally, I took out my test code in the main script block and wrote the code that called the class methods based on what the user chose from the menu. I included it in a try/except block to catch any errors. Below is the main 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()

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

        if choice_str.strip() == '1':  # Show current product list
            IO.output_current_product_list(lstOfProductObjects)
            continue

        elif choice_str == '2':  # Add new product to list
            new_product = IO.input_new_product()
            lstOfProductObjects.append(new_product)
            print('New product successfully added!')

            # print current product list to show new one added
            IO.output_current_product_list(lstOfProductObjects)
            continue

        elif choice_str == '3':  # Save data and exit
            FileProcessor.save_data_to_file(strFileName, lstOfProductObjects)
            print('Good-bye!')
            break

        else:
            print('"', choice_str, '"', 'is not a valid menu choice.')
            print()
            continue

except Exception as e:
    print(e)
    print()
```

Figure 2: Main body of script

Running the Script

To run the script in PyCharm I right clicked and chose “Run” from the dropdown menu. You can see the initial screen below.

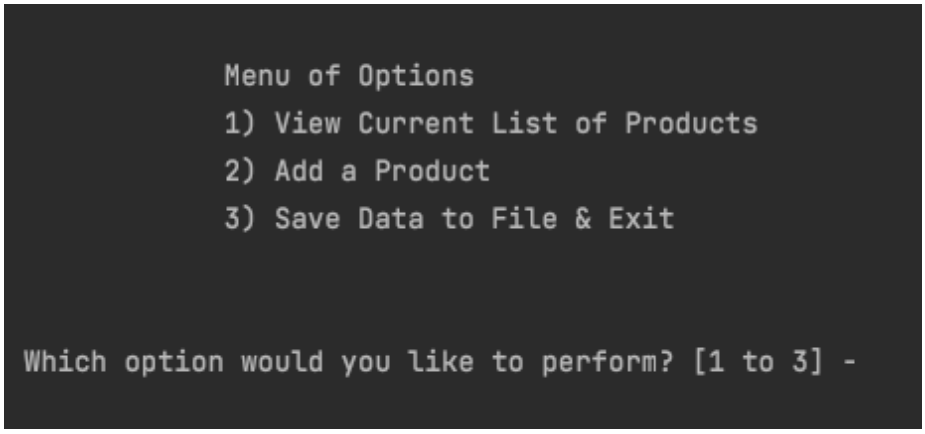

Figure 3: Start of the script in PyCharm

I started by viewing the current list of products by inputting a “1”. The script then showed the list of products that were currently stored in the text file by looping through the list of objects that was loaded at the start of the script.

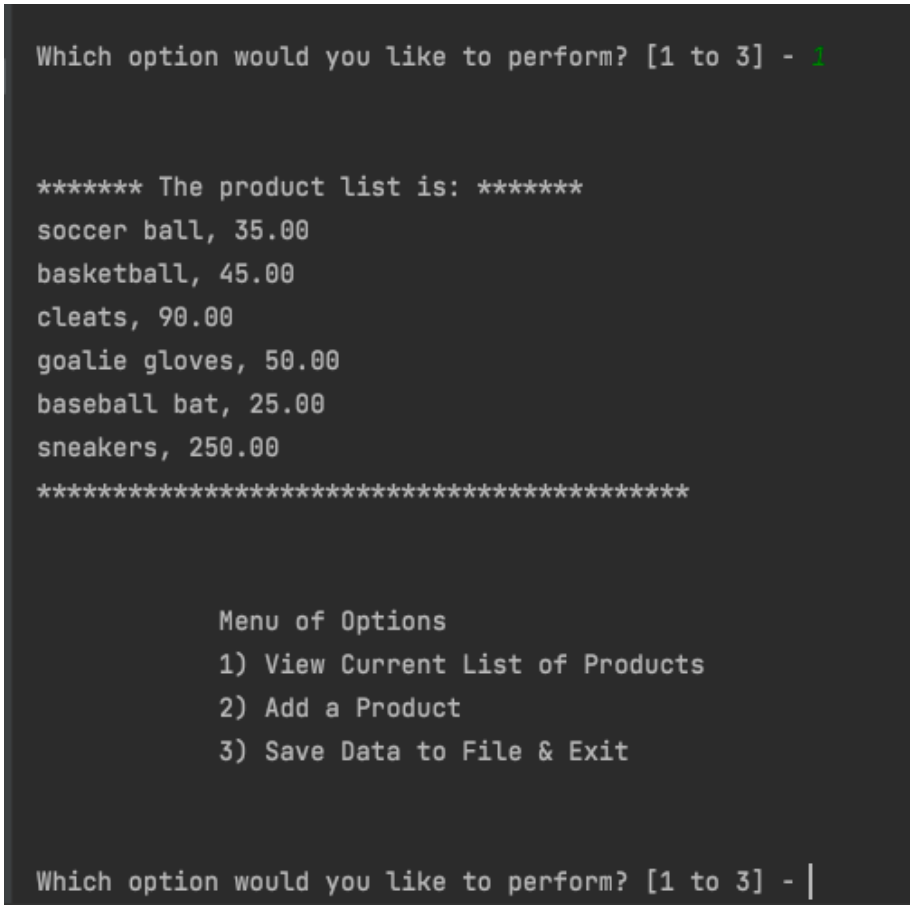

Figure 4: View of current list of products

Next, I added a product and the script appended that to the list and output the most current list of products as seen below.

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

Please enter product name: baseball
Please enter price: 15
New product successfully added!

******* The product list is: *******
soccer ball, 35.00
basketball, 45.00
cleats, 90.00
goalie gloves, 50.00
baseball bat, 25.00
sneakers, 250.00
baseball, 15.00
*****************************************
```

Figure 5: Adding a product

You can see above that the new product “baseball” and it’s price of “15” was added to the list. Finally, I chose “3” to save the data and exit the script.

```
            Menu of Options
            1) View Current List of Products
            2) Add a Product
            3) Save Data to File & Exit


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

Data has been saved!
Good-bye!

Process finished with exit code 0
```

Figure 6: Save & Exit

I checked the text file to verify that it now had the new item saved to it as seen below.

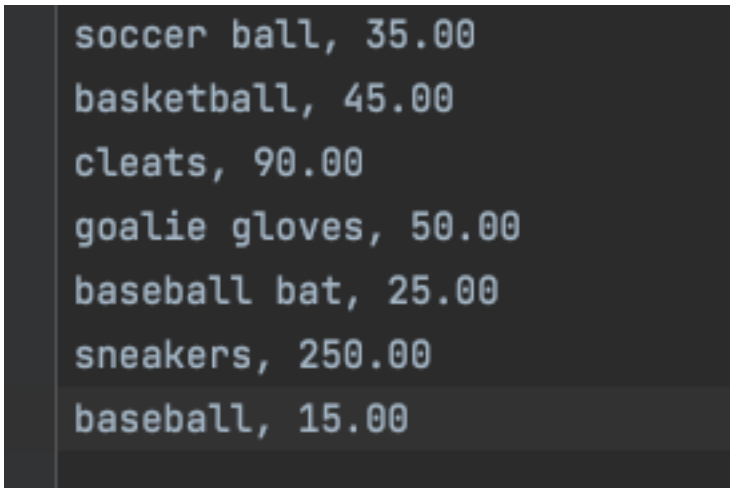

Figure 7: Updated text file

Finally, I ran the script in a terminal window to verify that the script executed as expected. You can see the results below.

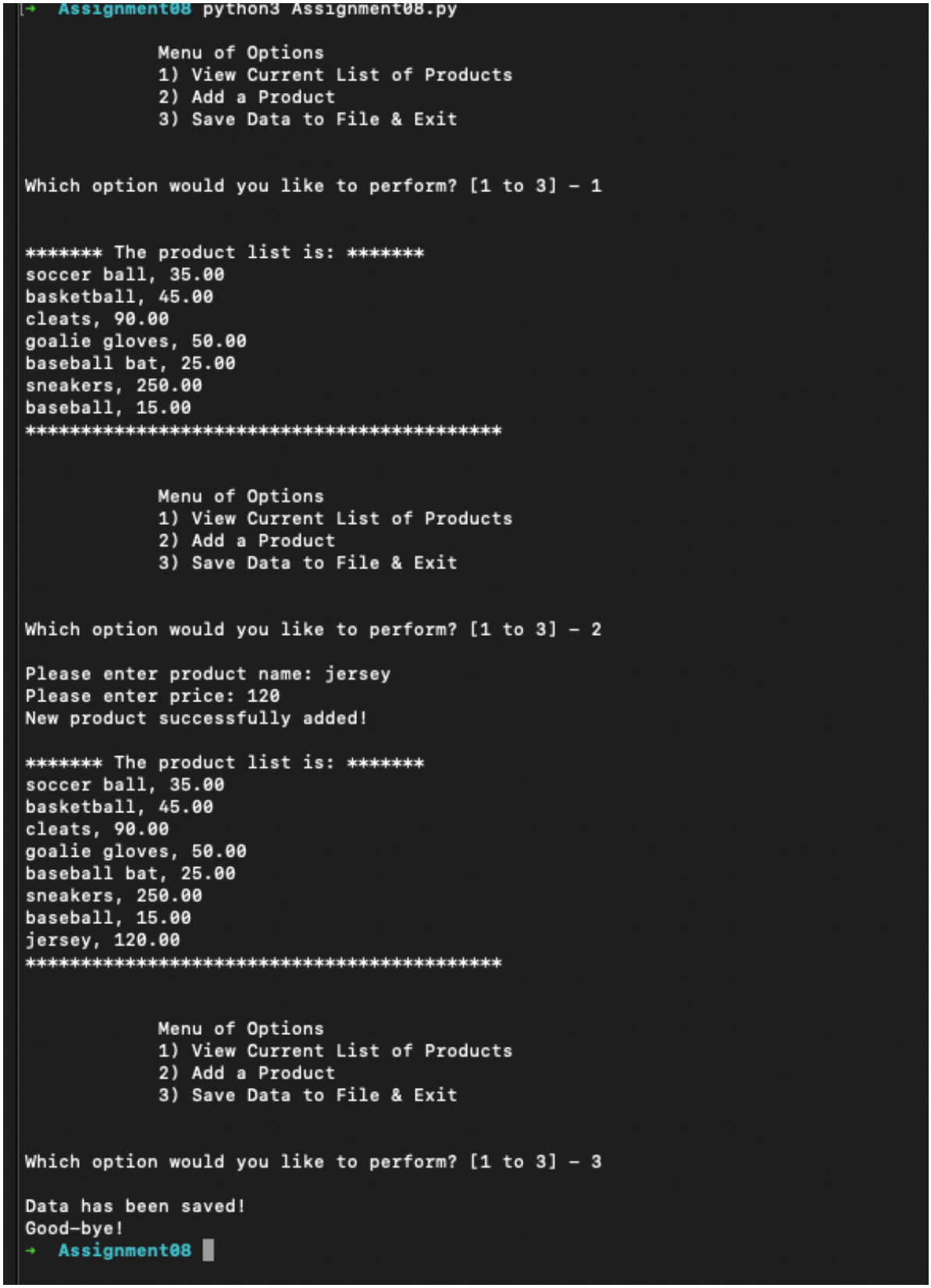

Figure 8: Output in Terminal

Conclusion

After completing this assignment, I feel my understanding of how to use classes, properties, and methods is solid. I like how clean the main script is as a result of encapsulating the code in classes and objects. It makes the code easier to follow and update moving forward.