Lin Martin

September 6, 2021

IT FDN 110 A

Assignment 08

<https://linmartinn7.github.io/IntroToProg-Python-Mod08/>

# Working with Classes in Python

## Introduction

For Assignment 08, our class was tasked with applying our knowledge of working with classes to a script that was provided to us by Professor Root. In this script, we were exposed to working with a file that had already been created and having to work within the parameters established by that script. In this document, I will discuss how I worked on this assignment and completed each section.

## Updating the File

Due to the script file for Assignment 8 being one that was already made for us to use, I immediately went into the header of the file and added my information to the change log. I continued to update the header as I worked on the file, documenting the progress I made for other programmers to see. I also elected to add numbers to the existing ***TODO*** tasks and ***TODO*** tasks I added so I could more thoroughly document what I had worked on.

# ------------------------------------------------------------------------ #  
# Title: Assignment 08  
# Description: Working with classes  
  
# ChangeLog (Who,When,What):  
# RRoot,1.1.2030,Created started script  
# RRoot,1.1.2030,Added pseudo-code to start assignment 8  
# LMartin, 8.28.2021, Modified code for class and process data from file, added pseudocode for tasks - *TODO 1-2*# LMartin, 8.29.2021, Modified code to process data to a file - *TODO 3*# LMartin, 8.31.2021, Modified code by starting Presentation code - *TODO 4-8*# LMartin, 9.1.2021, Modified code by finishing Presentation code - *TODO 4-8*# LMartin, 9.3.2021, Modified code by completing Main Body code - *TODO 9*# LMartin, 9.4.2021, Modified code by finishing Main Body code - *TODO 9*# LMartin, 9.5.2021, Tested script and fixed errors with script - *TODO 1-9*# ------------------------------------------------------------------------ #

***Figure 01: Header in Assignment08\_LMartin.py file.***

## Preparing the Script

The first part of the script I needed to work with was setting up the classes for the data portion of the script. Classes are a methodology that allow programmers to group data and functions together. This is typically applied to environments that focus on working with processing data. Because this file was working with product data, I went ahead and set up the class with the four main sections the scripts include: Fields, Constructors, Properties, and Methods.

# -- Fields --  
 # Already defined above.  
  
# -- Constructors --  
 **def** \_\_init\_\_(self, product\_name = "", product\_price = ""):  
 # -- Attributes --  
 self.\_\_product\_name = product\_name  
 self.\_\_product\_price = product\_price  
  
# -- Properties --  
 # Product Name  
 @property  
 **def** product\_name(self):  
 **return** str(self.\_\_product\_name).title()  
  
 @product\_name.setter  
 **def** product\_name(self, value: str):  
 **if** value.isnumeric():  
 **raise** Exception ("ERROR - Product Name must not contain numbers. Try again.")  
 **else**:  
 self.\_\_product\_name = value  
  
 # Product Price  
 @property  
 **def** product\_price(self):  
 **return** float(self.\_\_product\_price)  
  
 @product\_price.setter  
 **def** product\_price(self, value: float):  
 **try**:  
 self.\_\_product\_price = float(value)  
 **except** ValueError:  
 **raise** Exception("ERROR - Product Price must contain numbers. Try again.")  
  
# -- Methods --  
 **def** \_\_str\_\_(self):  
 **return** self.product\_name + "," + str(self.product\_price)

***Figure 02: Functions in the Data section of the script.***

Next, I proceeded to the processing section of the script. Classes were used again in this area with each section being broken up between the two functions being performed to process the data. This included reading data from a file and saving data to a file.

|  |  |
| --- | --- |
| **Reading Data from File** | # Read Data From File  @staticmethod  **def** read\_data\_from\_file(file\_name):  *""" Reads data from text file and places it into dictionary rows.* ***:param*** *file\_name: (string) of file name.* ***:return****: (list) of dictionary rows with product information.  """* list\_of\_product\_objects = []  **try**:  **if** os.path.exists(file\_name):  objFile = open(file\_name, "r")  **for** line **in** objFile:  data = line.split(",")  row = Product(data[0], data [1])  list\_of\_product\_objects.append(row)  objFile.close()  **except** FileNotFoundError **as** e:  print("ERROR - Cannot locate file.")  input("Press ENTER key to return to the main menu.")  **else**:  print("Data successfully loaded.")  objFile.close()  **return** list\_of\_product\_objects |
| **Saving Data to File** | # Save Data To File  @staticmethod  **def** save\_data\_to\_file(file\_name, list\_of\_product\_objects):  *""" Saves data to text file.* ***:param*** *file\_name: (string) of file name.* ***:param*** *list\_of\_product\_objects: (list) of product data inputted.* ***:return****: (list) of dictionary rows with product information.  """* **try**:  objFile = open(file\_name, "w")  **for** product **in** list\_of\_product\_objects:  objFile.write(product.\_\_str\_\_() + "\n")  objFile.close()  **except** Exception **as** e:  print("ERROR - Data was not saved to file correctly.") |

***Figure 03: Functions in the Processing section of the script.***

The next step in the script was to add code into the presentation section. This section of the script handled what the user was going to see in the actual program once it was running. This included the menu of options the user needed to interact with, obtaining the user’s selection from the menu, displaying current products, pressing enter to continue within the script, selecting yes or no for saving the file, and adding new products.

|  |  |
| --- | --- |
| **Menu of Options** | @staticmethod **def** print\_menu\_list():  *""" Displays Choice Menu to user."""* print('''  ~~~ Products Menu Options ~~~  1) Display Current Products  2) Add New Product  3) Save Data to File  4) Exit Program  ''') |
| **Menu Selection** | @staticmethod **def** input\_menu\_choice():  *""" Obtains user's menu selection.* ***:return****: (string) of menu input value.  """* menu\_choice = str(input("Which option would you like to do? [1-4] - "))  **return** menu\_choice |
| **Current Products** | @staticmethod **def** display\_current\_products(list\_of\_rows):  *""" Shows user the current products.* ***:param*** *list\_of\_rows: (list) of rows displayed.* ***:return****: N/A  """* print("~~~ Current Products ~~~")  **for** row **in** list\_of\_rows:  print(row.product\_name + "," + str(row.product\_price)) |
| **Pressing Enter to Continue** | @staticmethod **def** input\_press\_to\_continue(optional\_message=""):  *""" Have user press enter to get the script to continue.* ***:param*** *optional\_message: (string) An optional message displayed.* ***:return****: N/A  """* print(optional\_message)  input("Press [Enter] to continue.") |
| **Selecting Yes or No for Saving File** | @staticmethod **def** select\_yes\_or\_no():  *""" Have user select yes or no to save file.* ***:return****: (string) of selected option."""* **return** str(input(message)).strip().lower() |
| **Adding New Products** | @staticmethod **def** add\_new\_products():  *""" Adds new products and prices to the list.* ***:return****: (list) of product information.  """* **try**:  prod\_name = str(input("Enter a new product name [using letters only] : ")).strip()  prod\_price = float(input("Enter the project price [using numbers only]: ").strip())  prod\_1 = Product(product\_name=prod\_name, product\_price=prod\_price)  **return** prod\_1  **except** Exception **as** e:  print(e)  **return** prod\_1 |

***Figure 04: Code within Presentation section of the script.***

Lastly, the main body of the script needed to be completed. This section was what would have the script for running the user’s commands and was critical for the script to work correctly. I added options for each selection of the menu, along with a section of code that would appear once the script was launched letting the user know the database was being opened. I ended this section with an exception handling clause to generate an error if something went wrong with the script.

lstofProductObjects = FileProcessor.read\_data\_from\_file(strFileName)  
print("Opening database...")  
# IO.print\_menu\_list()  
  
**try**:  
 **while** (**True**):  
 IO.print\_menu\_list()  
 strChoice = IO.input\_menu\_choice()  
  
 **if** strChoice.strip() == "1":  
 IO.display\_current\_products(lstofProductObjects)  
 IO.input\_press\_to\_continue()  
  
 **elif** strChoice.strip() == "2":  
 lstofProductObjects.append(IO.add\_new\_products())  
 **continue  
  
 elif** strChoice.strip() == "3":  
 strChoice = IO.select\_yes\_or\_no("Save data to file? Select YES [y] or NO [n]: ")  
 **if** strChoice.lower() == "y":  
 FileProcessor.save\_data\_to\_file(strFileName, lstofProductObjects)  
 IO.input\_press\_to\_continue()  
 **else**:  
 IO.input\_press\_to\_continue("Save cancelled.")  
 **continue  
  
 elif** strChoice.strip() == "4":  
 print("Exiting Program.")  
 **break  
  
 else**:  
 print("Please select options 1-4 only.")  
  
**except** Exception **as** e:  
 print("ERROR - Program did not run correctly.")  
 print(e)

***Figure 05: Main body of script.***

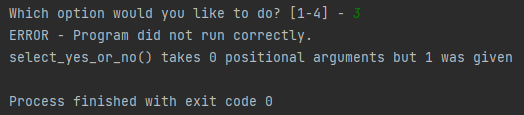
## Troubleshooting the Script

Once the script was written, I ran it in PyCharm to see how well it worked. I ran into my usual set of errors which were usually missing punctuation errors or typos. I was able to successfully fix multiple errors I discovered, including the sections to display the current products and adding new products to the script.

|  |  |
| --- | --- |
| **Option 1 - Displaying Products in Text File** | **Option 2 - Adding New Products** |
|  |  |

***Figure 06: Successful testing of Option 1 and Option 2 of the Python script.***

When it came to test the saving data to a file, I encountered errors with the ***select\_yes\_or\_no()*** function. Because this was an existing script we were tasked with modifying, I did my best to utilize the existing structure for storing functions and began to review the code I had added to the script.



***Figure 07: Option 3 saving file error message.***

There were two areas of the script that handled the ***select\_yes\_or\_no()*** function. I reviewed the code and added a choice for the **“n”** selection along with the **“y”** selection. This resulted in the same error being generated by the script once I ran it.

|  |  |
| --- | --- |
| **Original Version** |  |
| **Modified Version** |  |

***Figure 08: Attempt to fix Option 3’s saving file code.***

I attempted to troubleshoot this error for quite some time and continued to make no headway. Because I was unable to resolve this issue by modifying the script, I elected to remove the choice to confirm whether the user wanted to save the file. I went back into the stored functions and removed my ***select\_yes\_or\_no()*** clause and altered the script to automatically save the new data to the text file. I re-ran the script to test whether my new code was working successfully.

|  |  |
| --- | --- |
| **Modified Script - Save to File** | **elif** strChoice.strip() == "3":  FileProcessor.save\_data\_to\_file(strFileName, lstofProductObjects)  print("SUCCESS - Data was saved to text file.")  IO.input\_press\_to\_continue()  **continue** |
| **Testing Save to File Command in Script** |  |
| **Confirmation New Data was written to products.txt Successfully** |  |

***Figure 09: Fixing Option 3 and testing the script to verify data was saved to the file correctly.***

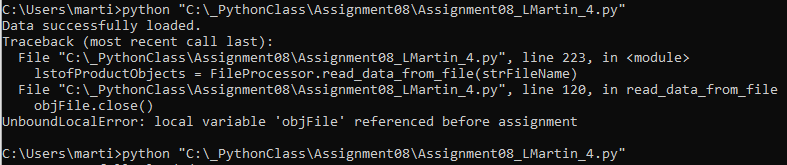
At this point, I re-ran the script to test how well it was working. I was able to add a new product with no errors and the script closed correctly. I even tested my error handling by inputting a wrong command, which it recognized.



***Figure 10: Successful test of altered script to save the data to products.txt file.***

## Testing the Script

Now that the errors were resolved with the script, I fully tested the script in PyCharm. I was able to successfully add items to the text file and verified that the information was saving correctly. I proceeded with testing the script in Windows Command Prompt. I immediately ran across an error with the script, which forced me to go back into PyCharm to figure out what I had done.



***Figure 11: Error in Command Prompt referencing objFile variable.***

After deleting an extra ***objFile.close()*** clause in the ***read\_data\_from\_file*** function, I re-ran the script and the program opened correctly in Command Prompt. I tested the script to ensure it was working by viewing the existing data, which was empty due to the script re-directing to the local directory.

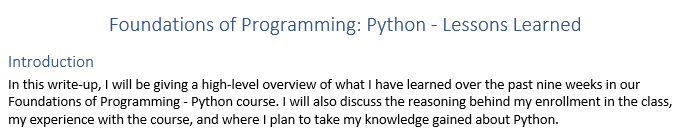
I proceeded with my testing and added a few items to the product list. After verifying that the data was captured correctly within the program, I used the option to save the data to a text file. I located the file in my local drive and opened it, confirming that the products and prices matched what had been inputted through Command Prompt (Figure 12). With this working, I exited the program to verify it was working, and Command Prompt exited the script with no issue. Now that the script was finished, I could proceed with the next steps of Assignment 08.

|  |  |
| --- | --- |
| **Viewing Added Products** | **Saving Data to Text File** |
|  |  |

***Figure 12: Adding new items in Command Prompt and saving them to a text file.***

## Lessons Learned Paper

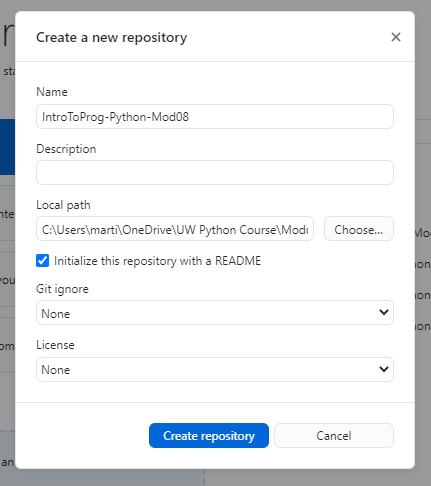
Due to the changes to our Final Assignment, I had to add one more document to Assignment 08’s tasks, the Lessons Learned Paper. This paper was to serve as our final write-up for the course, covering what we learned throughout the duration of the term. I wrote this paper in a separate file so it would be uploaded with my other assignment files on GitHub and through Course Canvas.



***Figure 13: Lessoned Learned paper for IT FND 110 A’s final assignment.***

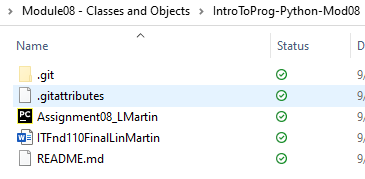
## Using GitHub Desktop

For Assignment 08, our class was tasked with creating a new GitHub repository to store our files for the assignment. However, for this week’s assignment we needed to use GitHub Desktop instead of the GitHub website. Installing GitHub Desktop from the GitHub website was fairly straightforward, and within minutes I had the program open on my computer and was creating a new repository for Assignment 08.



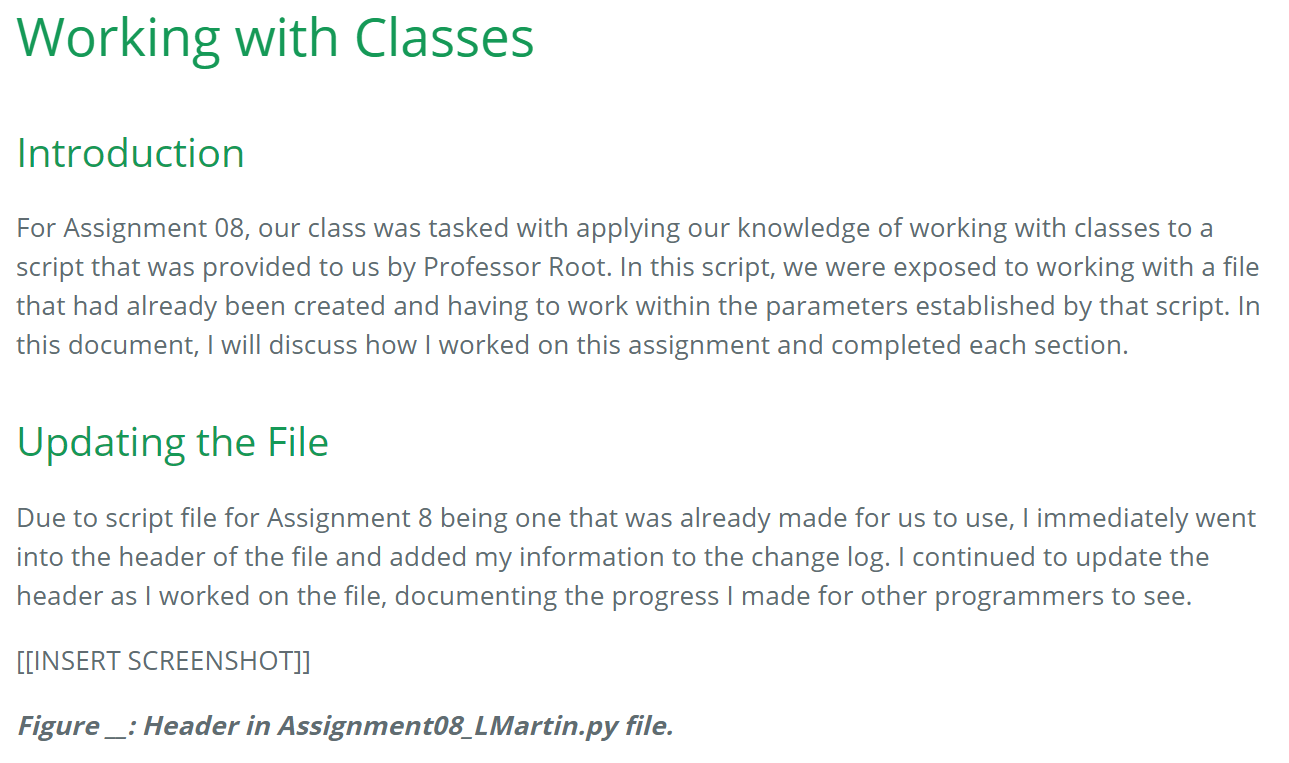
***Figure 14: Creating a new repository for Assignment 08 in GitHub Desktop.***

I copied my documents for Assignment 08 and placed them into the created folder on my PC by GitHub Desktop. This allowed them to be uploaded onto the site once I was ready to publish them.



***Figure 15: Uploading files to repository through GitHub Desktop.***

To verify that the files properly uploaded onto the GitHub website, I accessed my repository through the web. After verifying that the files had uploaded, I knew I wanted to go ahead and create a GitHub site with my Assignment 08 paper. I uploaded part of the paper to verify that it was working correctly as the main site (Figure 16). After finishing the write-up, I went back into the GitHub site and added the remaining sections of the paper, along with screenshots so the site would be complete.



***Figure 16: Initial GitHub site to verify that it had been successfully created.***

## Summary

I greatly appreciated Professor Root changing our final project to be Assignment 08. It allowed me to have more time to work with the script file, which ended up helping me grasp how to use functions and classes more thoroughly. I struggled with Assignment 06, so being able to re-visit the subject matter with extended time was greatly beneficial. I still don’t think I have quite mastered using functions as well as I would like, but I am quite proud of the progress I was able to make on this assignment. I struggled with certain sections and ended up having to change my plan for the ***select\_yes\_or\_no()*** function which was frustrating, but I was able to deliver a working product even with the change.

Even though creating a GitHub page was optional for this assignment, I decided to create one for the assignment. I felt it was important to continue using GitHub and the syntax for creating sites is easy for me to comprehend and is simpler than HTML. While creating GitHub sites does take an additional bit of time, due to having to save each screenshot as a photo on my PC for uploading and re-formatting my text, the final product is quite nice to look at. I have shared the sites I have created with friends and family, which helps them gain a sense of what it is I spend so many hours working on each week after work. GitHub is a great option for sharing work and showing my progress as a developer.