Emily Brown

August 17, 2022

IT FDN 110 B Su 22: Foundations of Programming: Python

Assignment06

# **Assignment 06**

**Introduction**

The purpose of Assignment 06 is to modify an existing code to make it include more functions. I modified this script in PyCharm and utilized PyCharm and Terminal to ensure it runs properly. I am working on a Mac OS.

**Strategy**

In order to modify this code, I utilized the ability to store dictionaries within a list. To accomplish each subtask, I utilized for and while loops when needed. The script allows the user to see what they have included in the list, add or remove data, and save to a text file when finished making modifications. Once the user is ready to save their data, the script saves all list contents to a text file.

**Script**

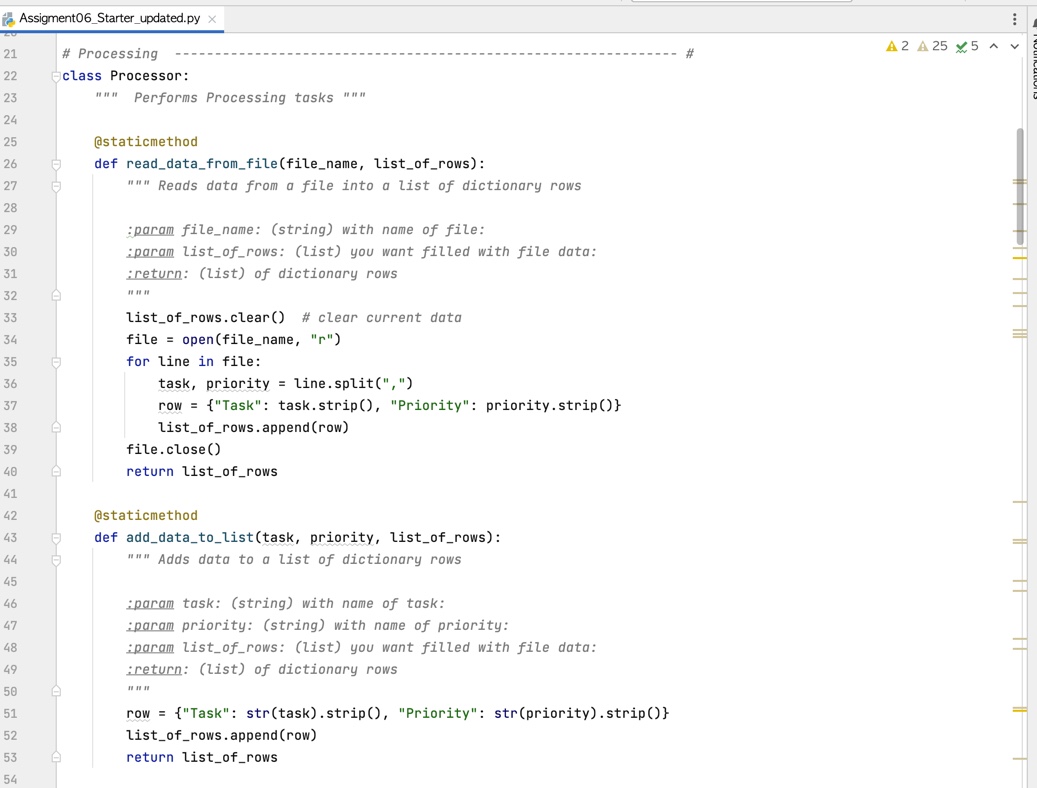
Below are screenshots from PyCharm of the script for Assignment 06. The first section of code is the change log and declaration of global variables. I did not make any changes to this section except adding my name to the change log.

Graphical user interface, text, application

Description automatically generated

*Figure 1: Header and Global Variables*

Below is the first portion of the processing section. The processing section contains multiple functions, but no functions are called yet. In this section, I modified the “add\_data\_to\_list” function to ensure the directive to append the “list\_of\_rows” was included.



*Figure 2: First portion of Processing section*

In this next section of code, the rest of the Processing section is displayed. I added the code to the “remove\_data\_from\_list” function.

Graphical user interface, text, application

Description automatically generated

*Figure 3: Second Portion of Processing Section*

Next, a new class of code begins. This section is the IO or Input/Output section. Nothing was changed in this section except adding menu options 1 and 5.

Graphical user interface, text, application, email

Description automatically generated

*Figure 4: First portion of IO Class*

The next section of code contains the rest of the Input/Output section. I added script to the “input\_task\_to\_remove” function.

**Graphical user interface, text, application, email

Description automatically generated**

*Figure 5: Middle section of Input/Output portion of code*

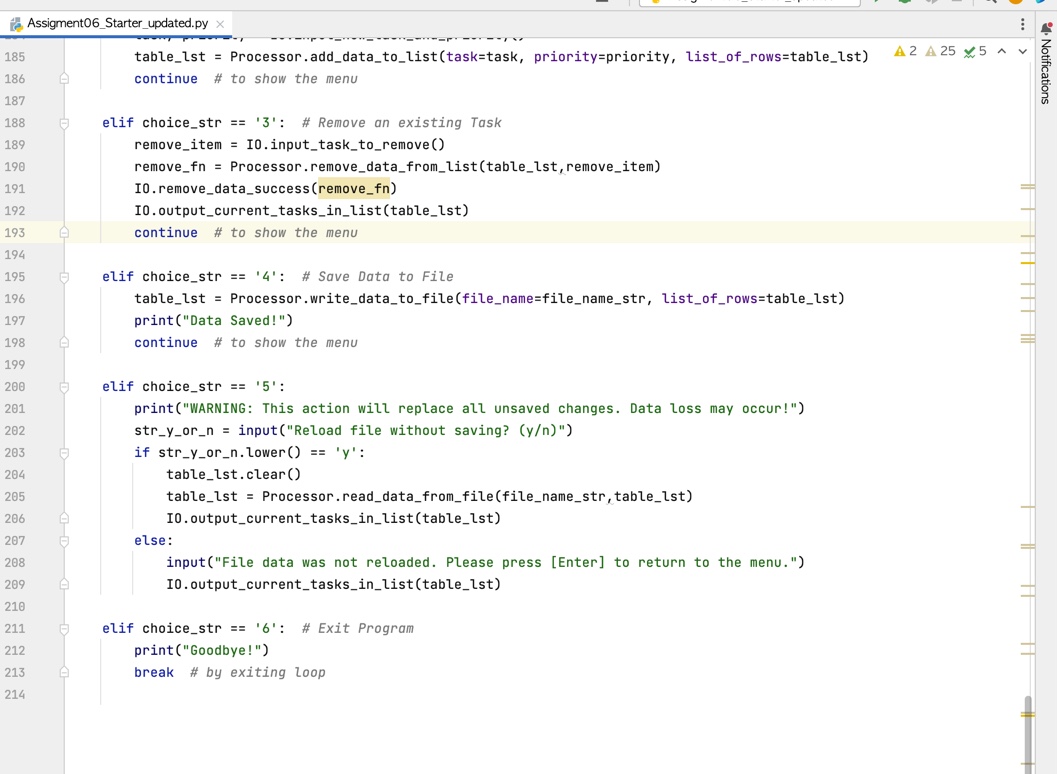
The next screenshot contains the final portion of the IO section and the beginning of the main body of the script where all functions are called. I added script to the “remove\_data\_success” function. I also created the section of code that gives directives if the user input from the menu is “1”.

*Graphical user interface, text, application, email

Description automatically generated*

*Figure 6: Beginning of Main Body of Script*

Finally, the last screenshot shows the rest of the script body. I wrote the code that runs for user input of “3” and “5”.



*Figure 7: Last Section of Main Body of Code*

**Results**

Below are screenshots of how the code presents itself when it is run in PyCharm. The first screenshot illustrates the text that existed in the text file ToDoList.txt before the code was run.

Graphical user interface, text, application

Description automatically generated

*Figure 8: Text file before running script*

The screenshot below is the first user interface that appears when the code is run. I selected option 1 to illustrate the existing content in the list. Note that it matches the content in the text file above. This is a result of the “Processing” section of the script.

*Graphical user interface, text, application, email

Description automatically generated*

*Figure 9: Initial user interface and Option 1*

Next, I selected option 2. The screenshot illustrates the second item that I added. The list of current tasks shows both items added.

Table

Description automatically generated

*Figure 10: Option 2 run successfully*

I chose Option 3 next. I removed Vacuum from the list.

Text, letter

Description automatically generated

*Figure 11: Option 3 run successfully*

Next, I chose Option 4 which saved this data to the text file. Then, I exited the program.

Graphical user interface, text, application

Description automatically generated

*Figure 12: Option 4*

See below for the text file after Option 4 was run.

Graphical user interface, text, application

Description automatically generated

*Figure 13: Text file after Option 4*

I rebooted the program and added a couple of items to the list via option 2. The screenshot below shows the items that I added.

Text, letter

Description automatically generated

*Figure 14: Added items*

Next, I chose option 5. Note that the new items added were lost since I chose option 5 without saving data.

Graphical user interface, text, application, email

Description automatically generated

*Figure 15: Option 5 with new items lost*

Finally, I chose option 6 to exit the program.

Graphical user interface, text

Description automatically generated

*Figure 16: Option 6*

The following screenshot illustrates the user interface when this code is run in Terminal.

Text

Description automatically generated

*Figure 17: Script in Terminal – Options 1, 2, and 4*

Below is a screenshot of the text file with the changes.

*Graphical user interface, text, application

Description automatically generated*

*Figure 18: Text file with changes from Terminal*

**Summary**

In summary, the Python script successfully achieves the objective of the assignment. It runs in both PyCharm and Terminal with the same results.