Sara Tupponce

05/17/2021

IT FDN 110 A Sp21: Foundations of Programming: Python

Assignment 06

GitHub URL: <https://ssara-tt.github.io/IntroToProg-Python-Mod06/>

To-Do List Script with Functions

## Introduction

The goal for this assignment was to modify a starter script in order to improve the previous script for managing a to-do list. Most of what the user would see would remain the same. Most of the changes would occur behind the scenes, where the commands would be grouped into self-contained functions.

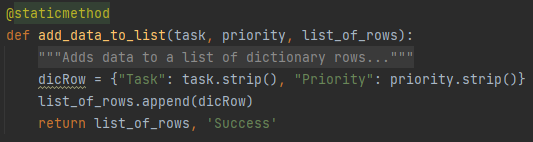
## Creating Python Script

Set up

Just as in the previous assignment, a starter script was used as a starting point. It already had declared most of the variables, many of the functions, and called some of the functions in the body of the script. There were two groupings of functions: the Processing layer functions and the Input/Output layer functions. The initial set up was completed by filling in the header.

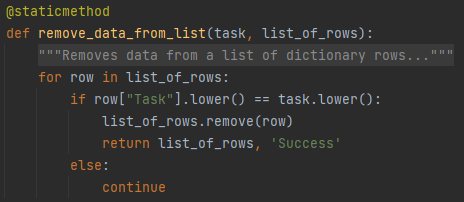
Processing Layer

The first section of code that had to be added to was the Processing layer, where the Processing class was. The function “add\_data\_to\_list” was for adding the new task and priority from the user into the existing task list. The new data was captured as a dictionary and appended to the existing list of rows that made up the table. The code can be seen below in Figure 1: add\_data\_to\_list Function.



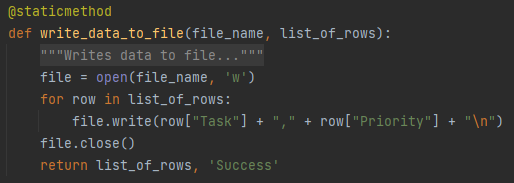
**Figure 1: add\_data\_to\_list Function**

The function “remove\_data\_from\_list” was for removing a task selected by the user from the existing task list. For every row of data in the existing list, a comparison was made using an “if” statement to see if the user defined task was in the list. The row that contained that task was deleted using the “remove()” method. The code can be seen below in Figure 2: remove\_data\_from\_list Function.



**Figure 2: remove\_data\_from\_list Function**

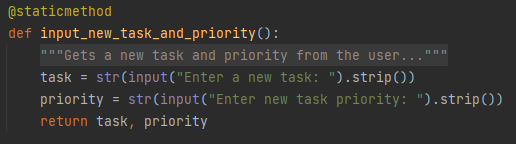
The function “write\_data\_to\_file” was for writing the tasks and priorities that were currently saved in the table to the designated file listed as a parameter. For every row of data currently in the table, the data was written to the file using a key since the data was saved in a dictionary. The file was then closed using the “close()” method. The code can be seen below in Figure 3: write\_data\_to\_file Function.



**Figure 3: write\_data\_to\_file Function**

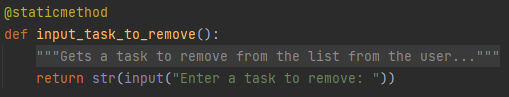
Input/Output Layer

In the Input/Output layer was where the IO class was defined. There were already a few functions defined in this section. The first function that required more code to be added to it was “input\_new\_task\_and\_priority”. This function captured the user input for what new task to add and what the corresponding task priority is. The code can be seen below in Figure 4: input\_new\_task\_and\_priority Function.



**Figure 4: input\_new\_task\_and\_priority Function**

The function “input\_task\_to\_remove” captured the user input for what existing task to remove from the list. The code can be seen below in Figure 5: input\_task\_to\_remove Function.



**Figure 5: input\_task\_to\_remove Function**

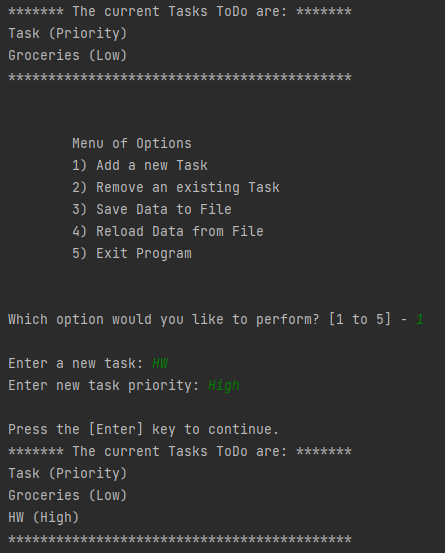
Main Body of Script

The main body of the script of where the functions that were created in the other sections were actually called using the applicable parameters. Based on the menu choice made by the user, certain functions were called to perform the action. The code can be seen below in Figure 6: Main Body.



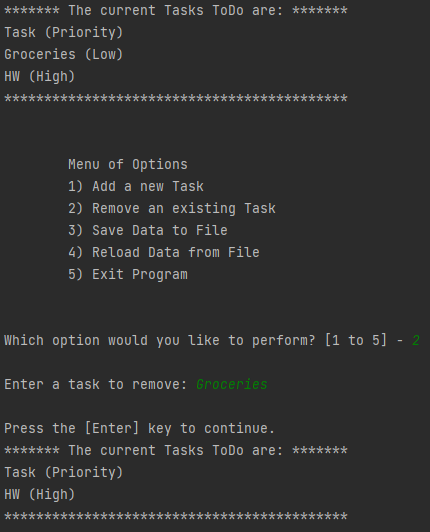
**Figure 6: Main Body**

The fist menu choice was to allow the user to add a new task and priority to the list. It used the functions “IO.input\_new\_task\_and\_priority” to ask the user for the new task and priority, and “Processor.add\_data\_to\_list” to add the new data to the list. The results of running this section in PyCharm can be seen below in Figure 7: PyCharm – Choice 1.



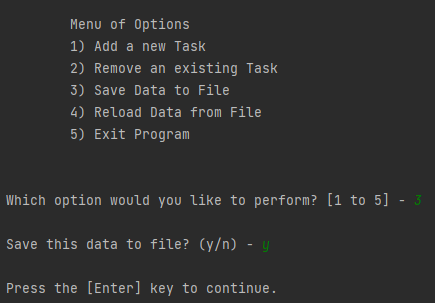
**Figure 7: PyCharm – Choice 1**

The second menu choice was to allow the user to remove an existing task from the list. It used the functions “IO.input \_task\_to\_remove” to ask the user for the task to delete, and “Processor.remove\_data\_from\_list” to remove the data. The results of running this section in PyCharm can be seen below in Figure 8: PyCharm – Choice 2.



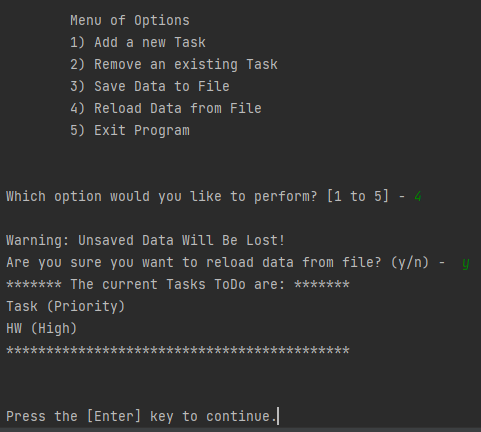
**Figure 8: PyCharm – Choice 2**

The third menu choice was to allow the user to save the data currently in the list to the text file. It used the function “Processor.write\_data\_to\_file” to write the data in the list to the file if the user approved. The results of running this section in PyCharm can be seen below in Figure 9: PyCharm – Choice 3.



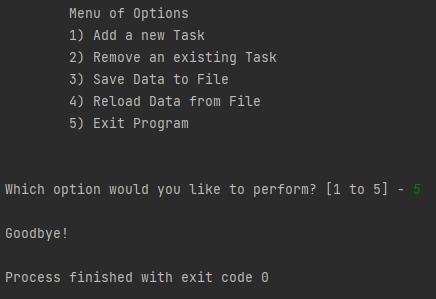
**Figure 8: PyCharm – Choice 2**

The fourth menu choice was to allow the user to reload the data currently in the text file. It used the functions “Processor.read\_data\_from\_file” to grab the data in the file if the user approved, and “IO.print\_current\_Tasks\_in\_list” to show the imported data. The results of running this section in PyCharm can be seen below in Figure 10: PyCharm – Choice 4.



**Figure 10: PyCharm – Choice 4**

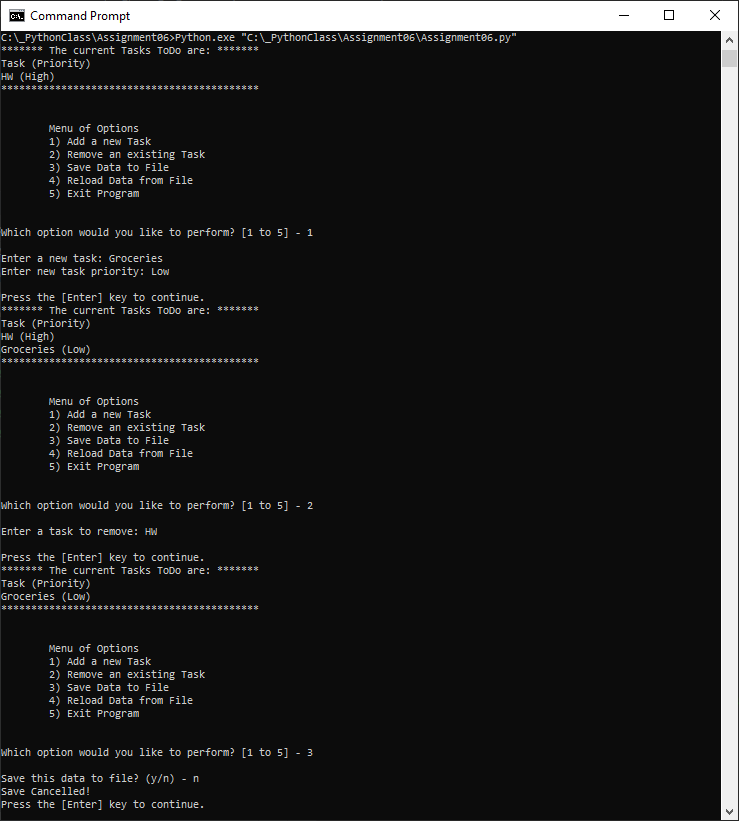
The fifth and last menu choice was to allow the user to exit the program. No functions were required for this section, only a “break” command. The results of running this section in PyCharm can be seen below in Figure 11: PyCharm – Choice 4.



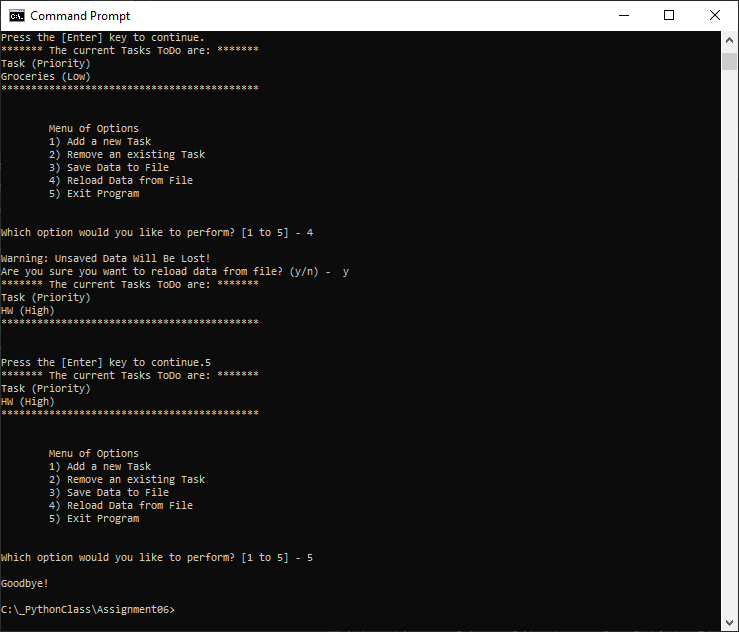
**Figure 11: PyCharm – Choice 5**

Test

Because incremental tests had been done along the way using PyCharm, the last test that had to be done was to run the script using the Command Window as well. Similarly to last assignment, the working directory had to be changed to the folder that contained both the assignment and the text file for the script to run. Figure 12: Command Window Running Script show the script being tested with the Command Window.



**Figure 12: Command Window Running Script**



**Figure 12: Command Window Running Script – cont.**

## Summary

This assignment showed how helpful functions could be for dividing the code into the different layers and into self-contained sections that could then be used in other programs. Although setting up the functions was more time consuming, reading the main body of the script was easier if you understood what each function was doing.