Mercedes Gonzalez Gonzalez

August 16th, 2020

IT FDN 110 A

Assignment 06

GitHub Repository: <https://github.com/mglezglez/IntroToProg-Python-Mod06>

GitHub Website: <https://mglezglez.github.io/IntroToProg-Python-Mod06/>

WORKING WITH LISTS, DICTIONARIES & CLASSES IN PYTHON:

# Introduction

This week, as part of Module 6 in Foundation of Programming (Python) course, I learned how to work with functions in Python. Functions can be seen as a “wrapper” of Python statements that accomplish a specific task and that can be invoked multiple time from the main program contributing to code re-use. Functions can be declared in the global namespace in a module or they can be part of a class. When they are created as part of a class, functions are typically called methods. There are instance, class and static methods. In this assignment, I have modified a Python script using PyCharm that works with two classes for processing the data and interacting with the user via I/O operations. Both classes contain multiple static methods, each of them, implements a simple tasks and it does it well. Then these methods are invoked from the main program by calling them by their names. The next sections contain a step-by-step description of the code and the tests done to validate it works as expected.

# Modifying the script template

In order to implement this program, I re-used a script template provided for this assignment and I proceeded to write the code in each section as indicated in the comments in order to get the main program to work.

Essentially, the script template came with two classes: Processor and IO. The Processor class implements all the methods required to read and write data to a file and update a list of dictionaries containing all the tasks and their respective priorities. The IO class is responsible for the Presentation layer, that is, the input/output operations that presents data to the user and obtains data from the user. In other works, the IO class contains methods that print the menu, ask the user to select an option in the menu, ask for the name and priority of the new task to add, ask for the name of the task that the user wants to remove, ask the user whether they want to continue or ask the user to answer yes/no to a question.

The first task was to implement the pending methods in the Processor class. The methods implementation shows up in the next image with the code I added. Those are ***add\_data\_to\_list()***, ***remove\_data\_from\_list()*** and ***write\_data\_to\_file().***

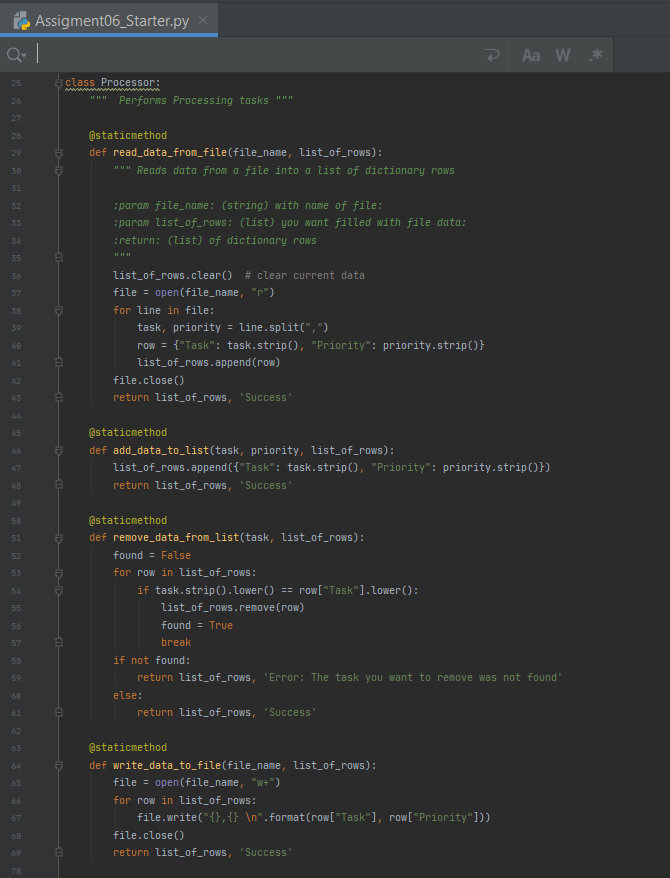


Figure 1. Processor methods implementation

The second task was related with the IO class. There were two static methods in the IO class that needed to be implemented. Those were input\_new\_task\_and\_priority() and input\_task\_to\_remove(). The former is expected to ask the user for a new task name and its priority and the latter asks the user for the name of a task to be removed. The implementation for both methods can be found in the picture below.

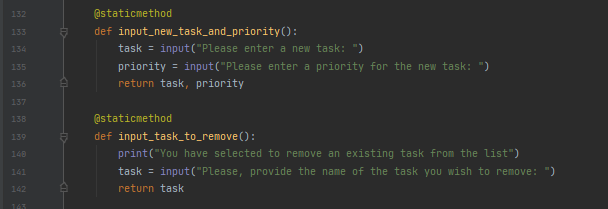


Figure 2. IO methods implementation

Finally, the main program contained some sections that needed to be implemented. Those sections where the places from where the methods implemented in the Processor and IO class were expected to be called in order to complete the needed tasks of adding new tasks, deleting a certain task, saving data to a file and reloading data form a file to the list of dictionaries in memory.

As can be seen, the code in these methods is not that different from the code in the previous assignment, so even though this assignment is different, it is mainly based on a new way to structure and organize the code, but the logic implemented is very similar.

The following picture contains the statements added to be able to implement the pending logic in the main program.

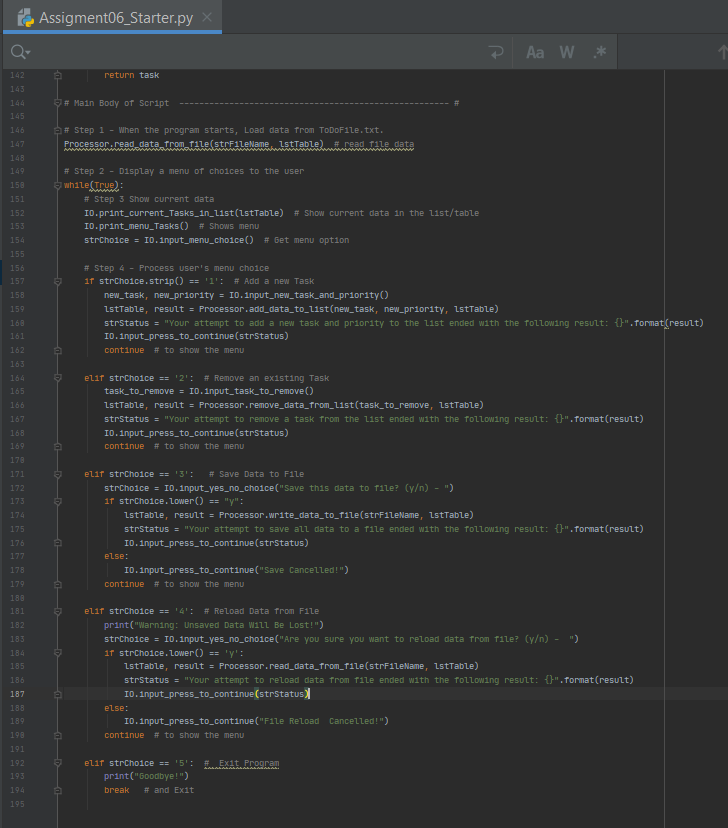


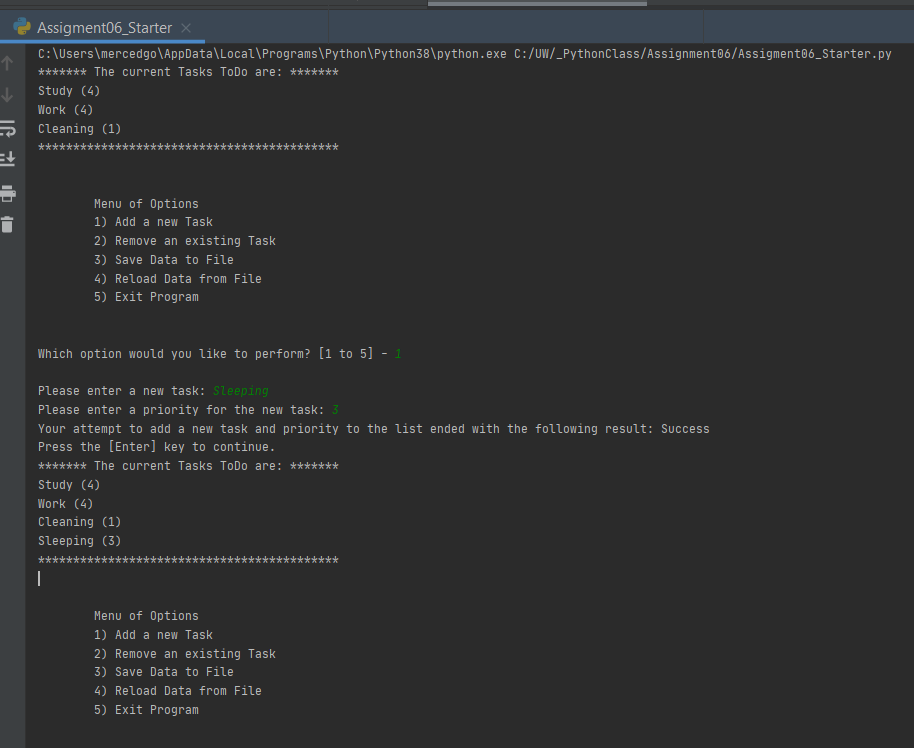
Figure 3. Main program implementation

# Testing the script

The following section contains screenshots from the program while running.

The initial content of the file is loaded into a list of dictionaries and is displayed on top of the menu of options.

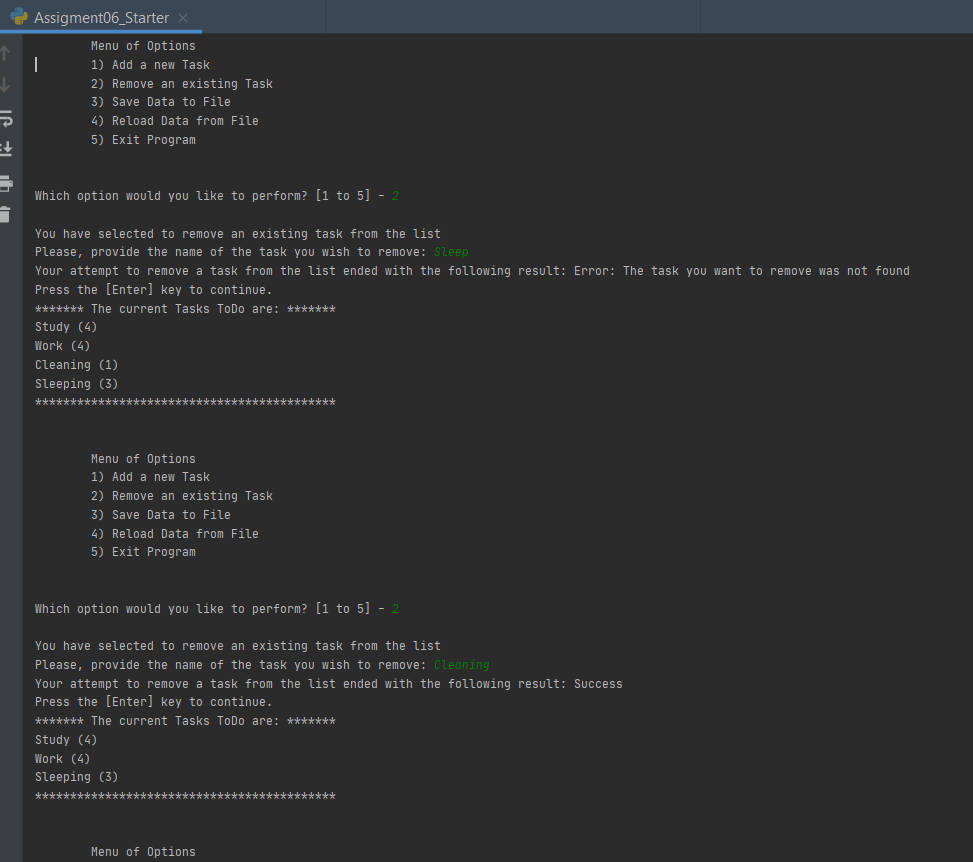
Test #1: Adding a new task/priority to the list of dictionaries.



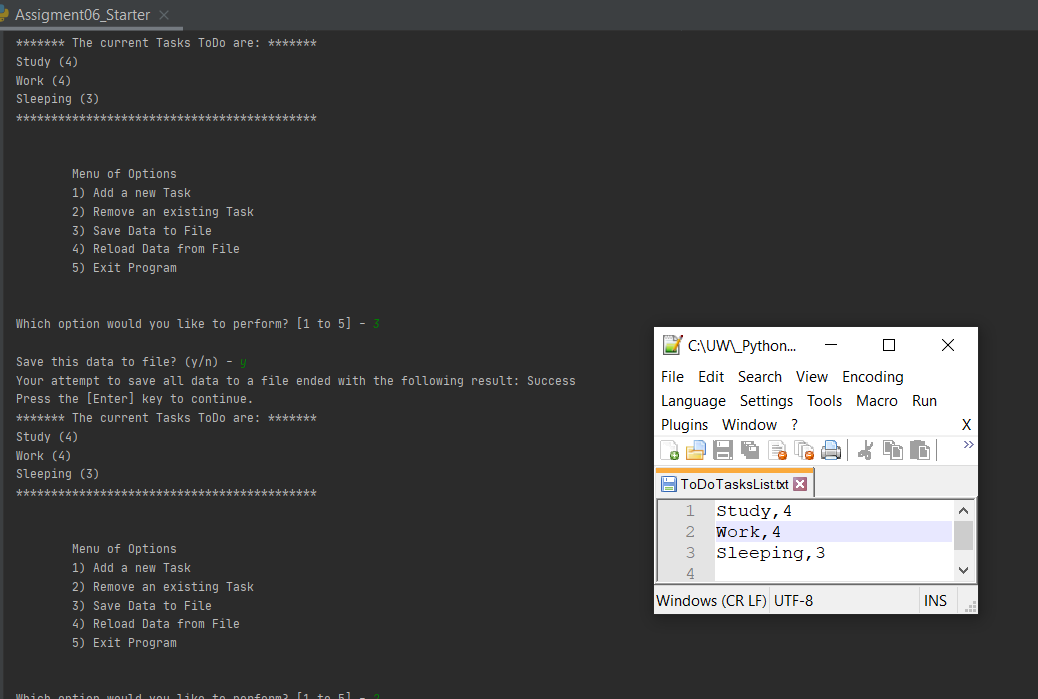
Test #2: Removing a task from the list

- The first attempt specifies a task that does not exist in the table (list of dictionaries). In this case, an error message is displayed on the screen to notify the user that the task was not found.

- The second attempt specifies a task that does exist in the table and therefore it is successfully removed.



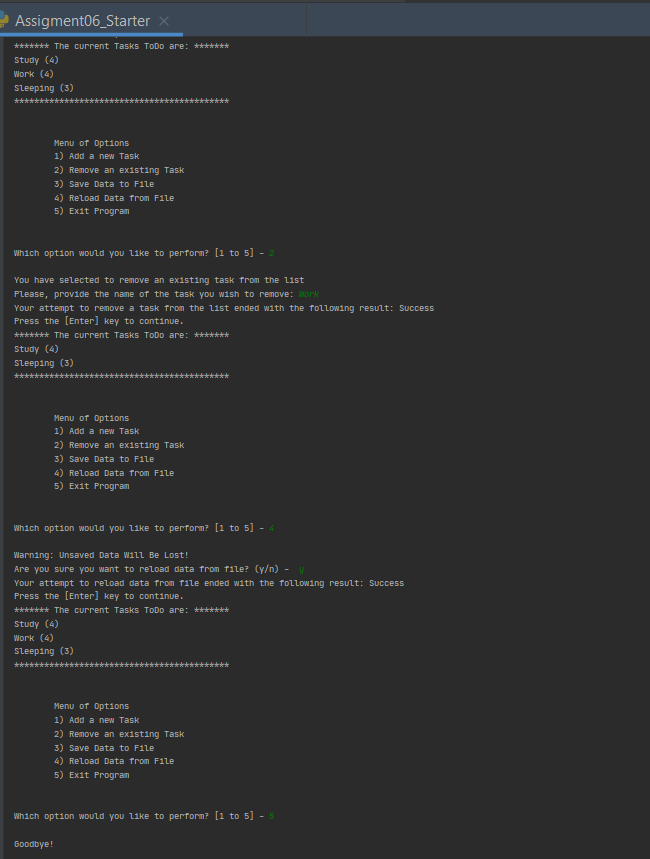
Test #3: Save Data to File



Test #4:

During test #4, I proceeded to remove a task from the list, and then reload the data from the file to shows how that element that had been previously deleted is recovered after the reload operation is completed successfully.

This is the last test conducted. Results can be seen in the next page.



# Summary

Classes and functions are more advanced Python elements that allow re-using code more effectively while creating scripts. Code reusability is especially important when developing professional projects where the number of lines of code can grow very quickly if planning and efficiency is not taken into consideration. This module opened the door to the work with these important Python constructs and I am excited and eager to learn more about it.