James Henderson

8/17/2022

IT FDN 110 B

Assignment06

https://hendej25.github.io/IntroToProg-Python-Mod06/

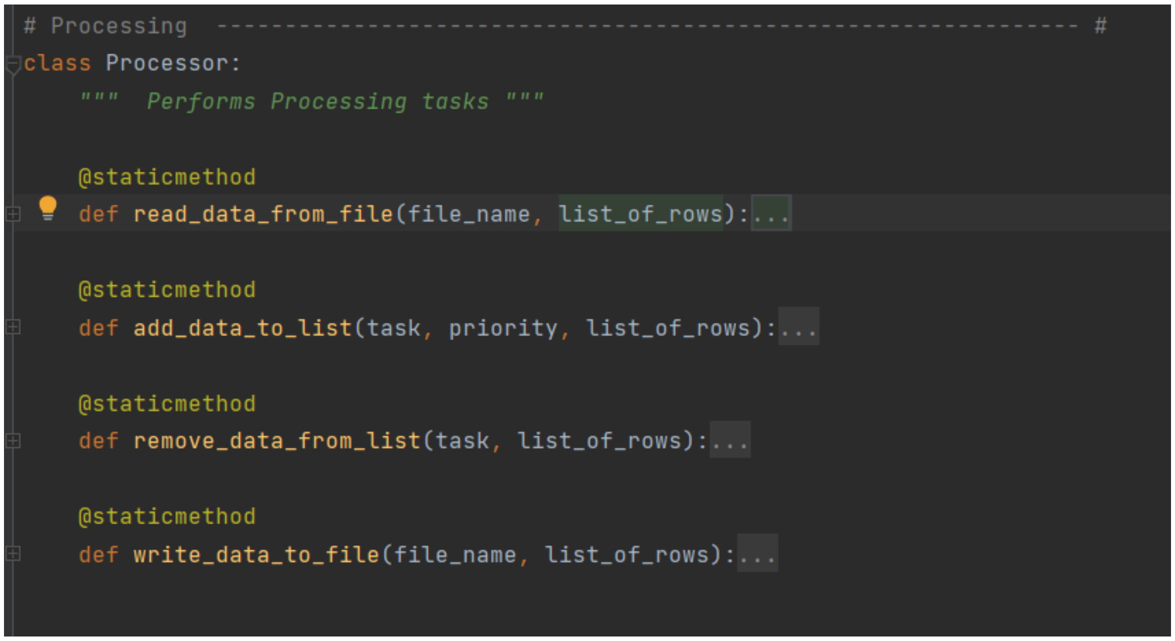
# Assignment 6 – Task List Script

## Introduction

For this assignment, I worked to modify an existing script similar to the one that we completed for Assignment 5. However, instead of using a simple start-to-finish approach, the script I modified for this Assignment uses functions to accomplish similar tasks. The use of functions allows for the code to be separated more cleanly into processing, I/O (there is one Class for each of these, each with several functions) and "main code body" sections.

## Task List: Processing

To accomplish the goals of this assignment, I added code to make the script run correctly. First, the program reads in existing task data from a text file in the same folder as the script, "ToDoFile.txt". As shown in ***Figure 1***, all of the functions that make up the processing layer of the script are contained within a class named "Processor". The 4 static methods within this class provide functionality to read / write data in and out of a text file (storage), as well as add or remove a task from a list.



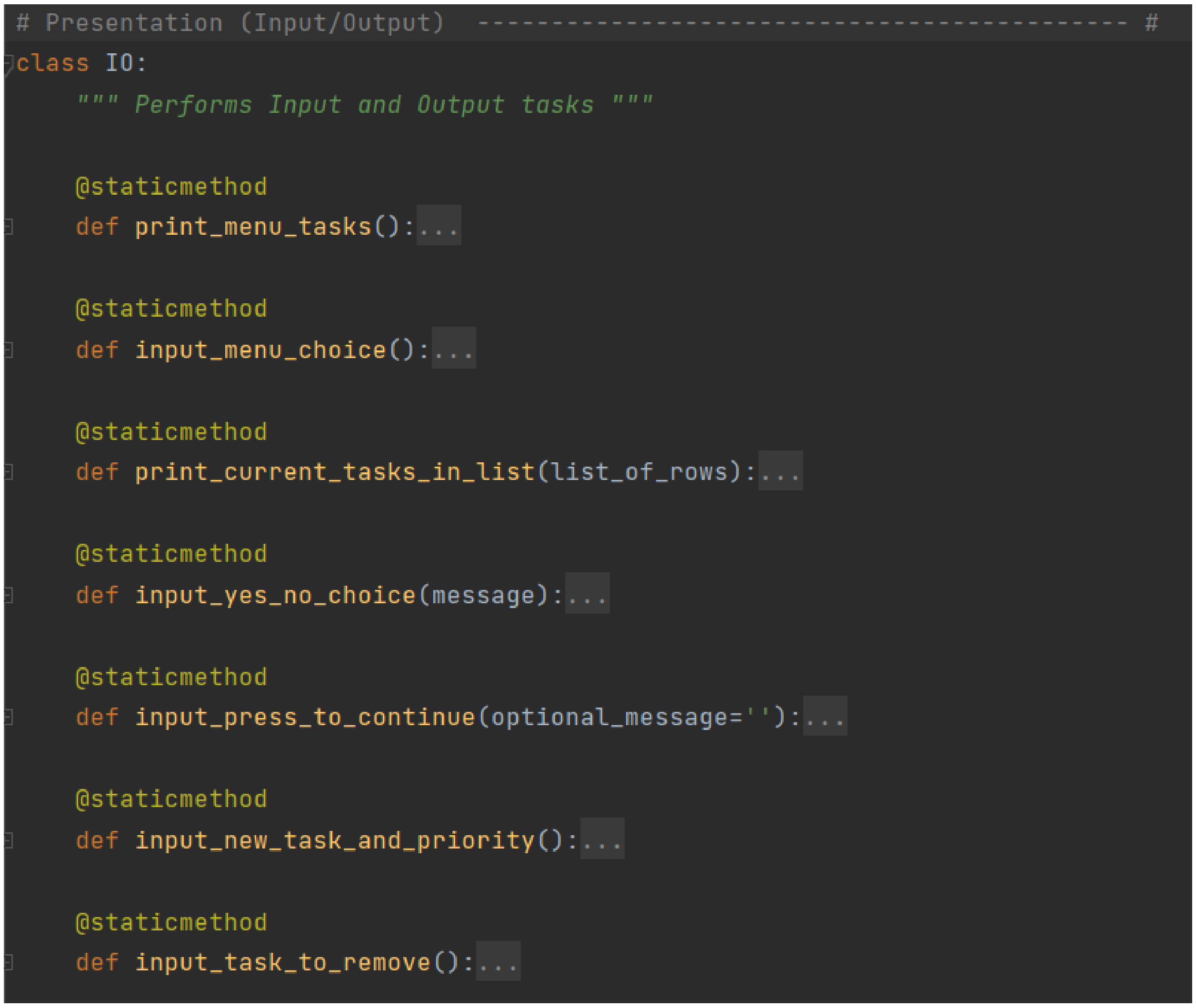
```
# Processing  ---------------------------------------------------------- #
class Processor:
    """  Performs Processing tasks """

    @staticmethod
    def read_data_from_file(file_name, list_of_rows):...

    @staticmethod
    def add_data_to_list(task, priority, list_of_rows):...

    @staticmethod
    def remove_data_from_list(task, list_of_rows):...

    @staticmethod
    def write_data_to_file(file_name, list_of_rows):...
```

***Figure 1 – The processing layer of the script is organized into the Processor class***

## Task List: Input / Output

As I mentioned in the introduction section, the script for this assignment has been separated into 3 distinct layers. The Input / Output functions that display data to the user, or take in input from the user, are organized into a Class named “IO”. There are seven functions in all – of these, 2 just display data or choices to the user, while 5 are interactive in that they require some type of input from the user.

The interactive functions (names beginning with “input_”) return the user input via one or more variables. Within the main code body, these returned values are then handed off to one of the Processor-class functions as needed.



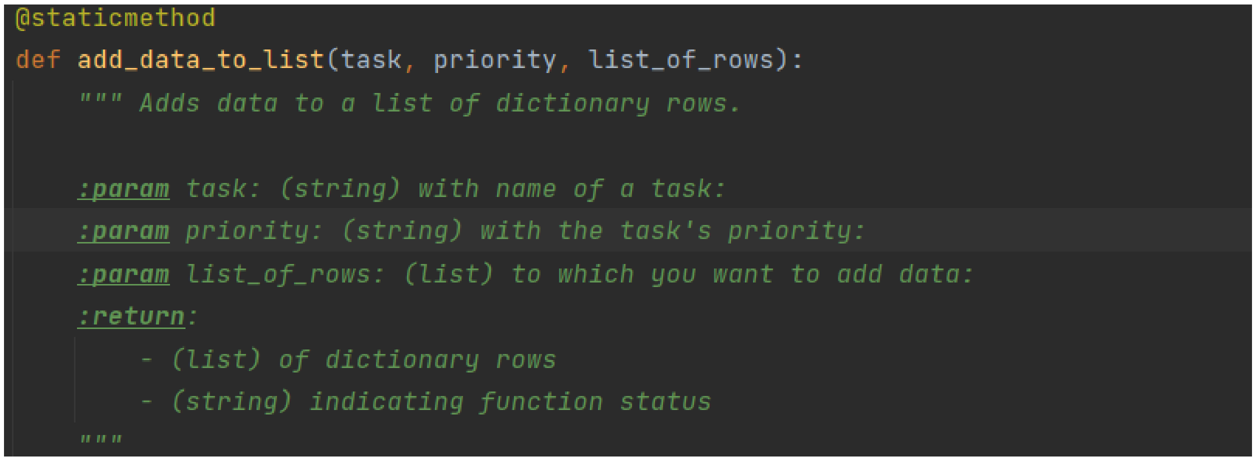
```
# Presentation (Input/Output)  -------------------------------------------- #
class IO:
    """ Performs Input and Output tasks """

    @staticmethod
    def print_menu_tasks():...

    @staticmethod
    def input_menu_choice():...

    @staticmethod
    def print_current_tasks_in_list(list_of_rows):...

    @staticmethod
    def input_yes_no_choice(message):...

    @staticmethod
    def input_press_to_continue(optional_message=''):...

    @staticmethod
    def input_new_task_and_priority():...

    @staticmethod
    def input_task_to_remove():...
```

***Figure 2 – There are no less than 7 different static methods for handling interaction with the user***

## Task List: Docstrings

For this assignment, I also learned how to use docstrings to document the general purpose and details of how to use the functions I’ve created or modified. I learned that while Python allows you to document multiple parameters along with their type, you must use plain text to denote the return values. When there are multiple values returned, it makes sense to add a little extra formatting to make it clear what will be returned by the function.

An example of this can be seen in the function I edited for adding data to the task list in the script's Processor class ***(Figure 3):***



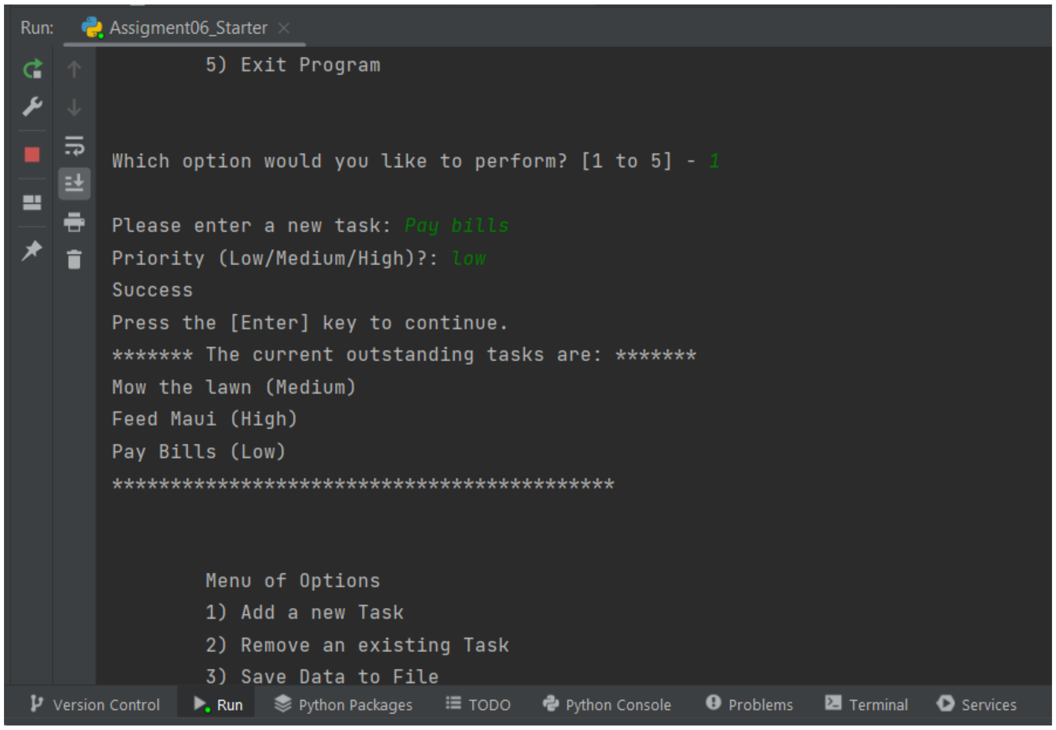
```
@staticmethod
def add_data_to_list(task, priority, list_of_rows):
    """ Adds data to a list of dictionary rows.

    :param task: (string) with name of a task:
    :param priority: (string) with the task's priority:
    :param list_of_rows: (list) to which you want to add data:
    :return:
        - (list) of dictionary rows
        - (string) indicating function status
    """
```

***Figure 3 – Adding a docstring that describes a function's 3 parameters and 2 return values***

## Task List: Running the Script

To test the script, I first ran the Assignment06_Starter.py file within PyCharm, using the latest available Python interpreter on my machine (Python 3.10).

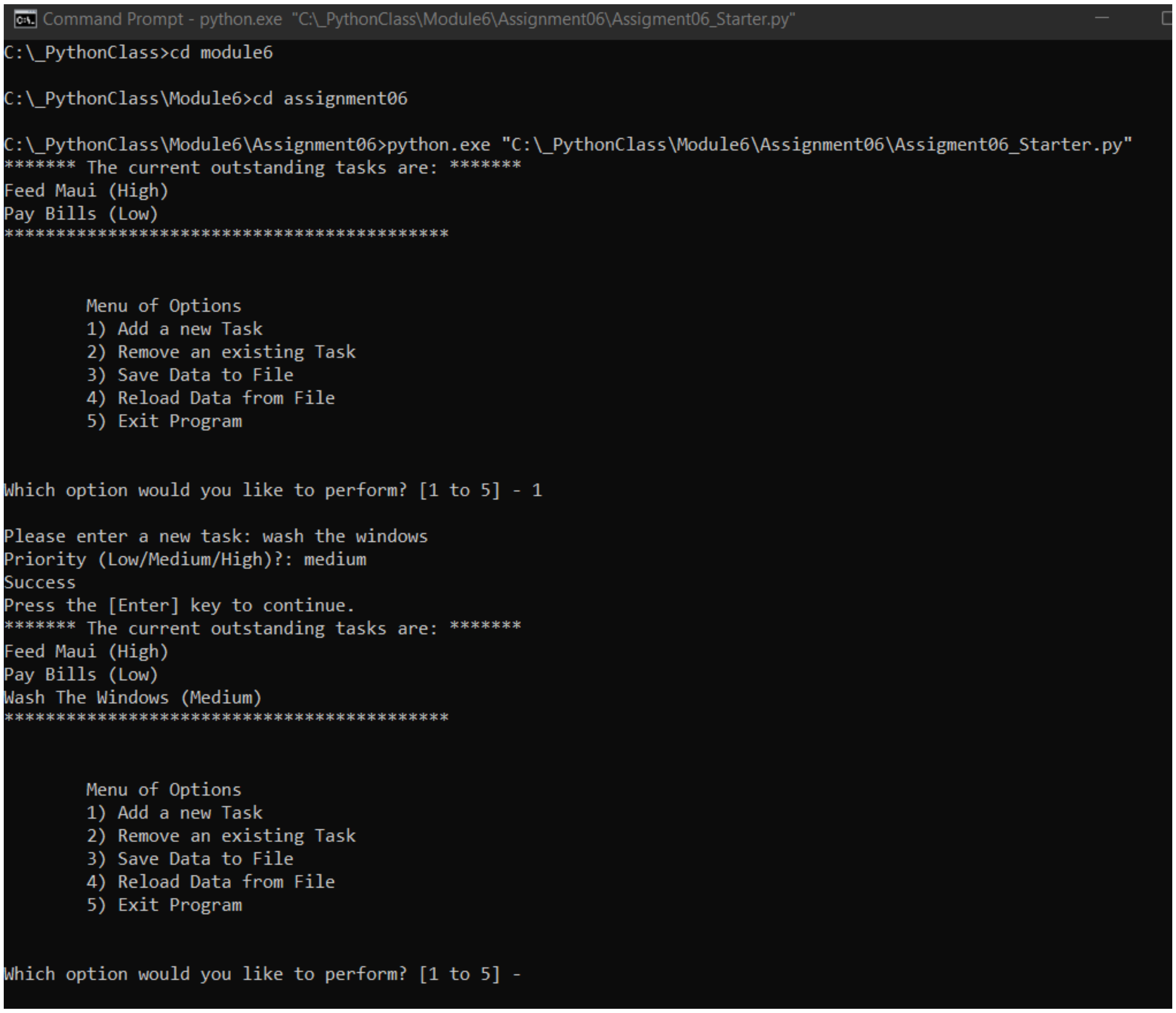
You can see part of the script's input / output in process in ***Figure 4*** below***.***



```
        5) Exit Program


Which option would you like to perform? [1 to 5] - 1

Please enter a new task: Pay bills
Priority (Low/Medium/High)?: low
Success
Press the [Enter] key to continue.
******* The current outstanding tasks are: *******
Mow the lawn (Medium)
Feed Maui (High)
Pay Bills (Low)
*******************************************


        Menu of Options
        1) Add a new Task
        2) Remove an existing Task
        3) Save Data to File
```

***Figure 4 – running the script in PyCharm***

I also ran the script from the Windows Command Prompt window (***Figure 5).*** I verified that data from a previous run of the script was correctly picked up at the start of the program.



```
Command Prompt - python.exe  "C:\_PythonClass\Module6\Assignment06\Assigment06_Starter.py"
C:\_PythonClass>cd module6

C:\_PythonClass\Module6>cd assignment06

C:\_PythonClass\Module6\Assignment06>python.exe "C:\_PythonClass\Module6\Assignment06\Assigment06_Starter.py"
******* The current outstanding tasks are: *******
Feed Maui (High)
Pay Bills (Low)
*******************************************



        Menu of Options
        1) Add a new Task
        2) Remove an existing Task
        3) Save Data to File
        4) Reload Data from File
        5) Exit Program


Which option would you like to perform? [1 to 5] - 1

Please enter a new task: wash the windows
Priority (Low/Medium/High)?: medium
Success
Press the [Enter] key to continue.
******* The current outstanding tasks are: *******
Feed Maui (High)
Pay Bills (Low)
Wash The Windows (Medium)
*******************************************



        Menu of Options
        1) Add a new Task
        2) Remove an existing Task
        3) Save Data to File
        4) Reload Data from File
        5) Exit Program


Which option would you like to perform? [1 to 5] -
```

***Figure 5 – running the script from the Windows command prompt***

## Summary

In this assignment, I edited an existing script to make it work with 3 layers – a processing layer (one class), interactive layer (one class), and a main code body. I made sure that the functions in the script were all appropriately documented with docstrings, and that the code I added was compartmentalized correctly.