Kyle Biondich

8/15/2023

[IT FDN 110 A](https://canvas.uw.edu/courses/1655585)

Assignment 06

<https://github.com/kbiondo/IntroToProg-Python-Mod06>

ToDo Funs with Functions

# Introduction

Week 6 of the course introduced classes and functions and how to use the return element when passing data between functions. The following paragraphs outline the methods that were used to read a text file into a python script, capture user input from a menu, and either add a new item to a list, read the total list, or write the list back to an external file.

# Intended Outcome

The intended outcome of this assignment is to initially load a text file that consists of tasks and priorities if it exists and store the contents of that text file in a list and provide the ability to:

1. Add new tasks and their priorities to the list
2. Remove a task and its priority from the list
3. Save the contents of the list to a file
4. Exit the program.

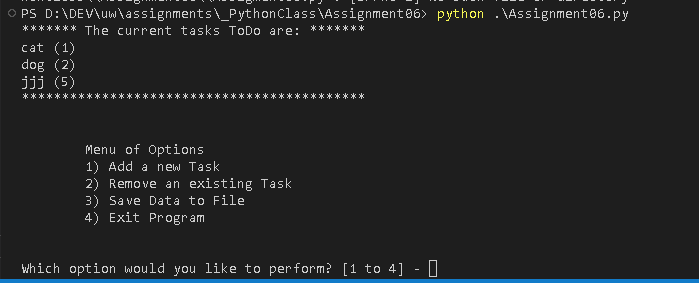


Figure 1: Intended Outcome: Assignment06.py Menu

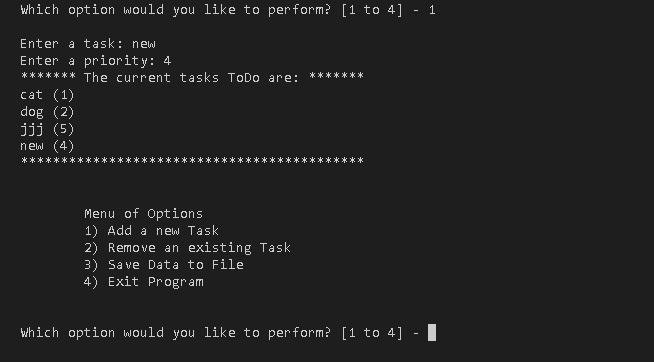


Figure 2: Intended Outcome: Assignment06.py Menu 1

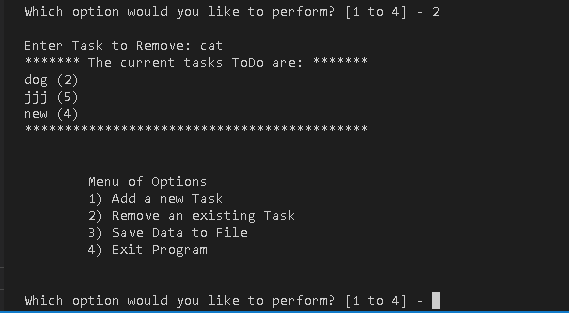


Figure 3: Intended Outcome: Assignment06.py Menu 2

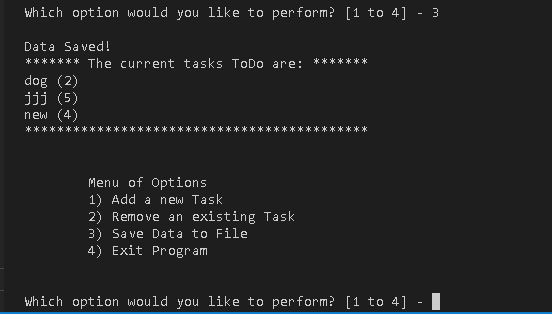


Figure 4: Intended Outcome: Assignment06.py Menu 3

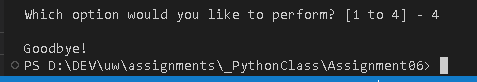


Figure 5: Intended Outcome: Assignment06.py Menu 4

**Declare Variables and constants**

The starter assignment python file contained a bunch of starting variables and constants, as seen in figure 6.

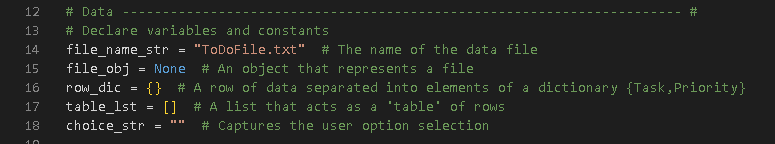


Figure 6: Variables and Constants

## Step 1 – Processing Class – Read Data from file

In this step, the function read\_data\_from\_file is defined. It starts with using the open function with the append argument to ensure a ‘TodoFile.txt’ exists, then uses the open function again with the ‘read’ argument to read the contents of the file, if any. It then loops through the file with a for loop and puts any contents of the file first into a dictionary row, then into a list of dictionary rows.

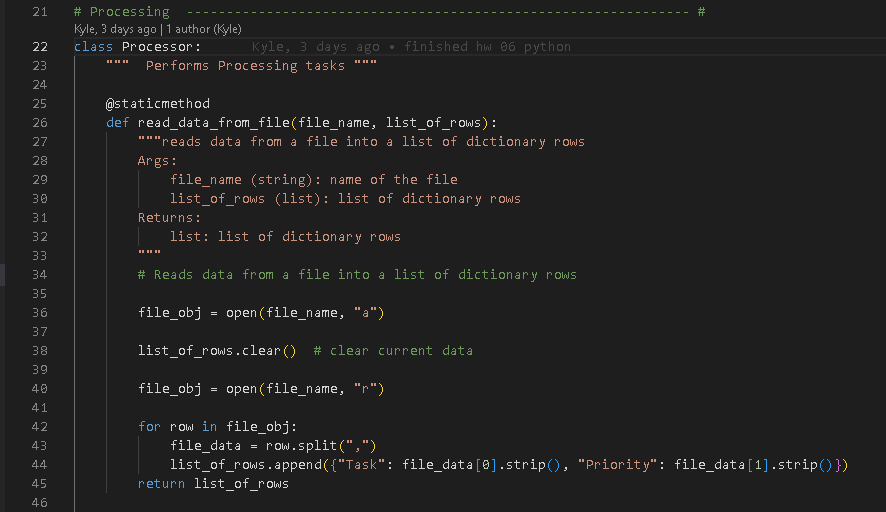


Figure 7: Load a File into a dictionary list

# Step 2 – Add data to list

Next, the starter file contained the ‘add\_data\_to\_list’ function. In this function I added a counting variable along with a while loop to check if the data passed into the function was already in the list. Only if the task name wasn’t found would it be added to the list.

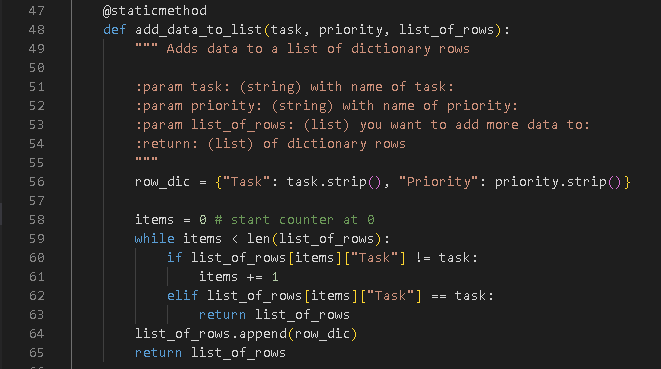


Figure 8: Add data to list

# Step 3 – Remove Data from List

The next function is the ‘remove\_data\_from\_list’ function. Similar to the previous function, I use a counting variable and a while loop to check each item in the list against the item that was captured from the user. If the item in the list is found, it is removed from the list.

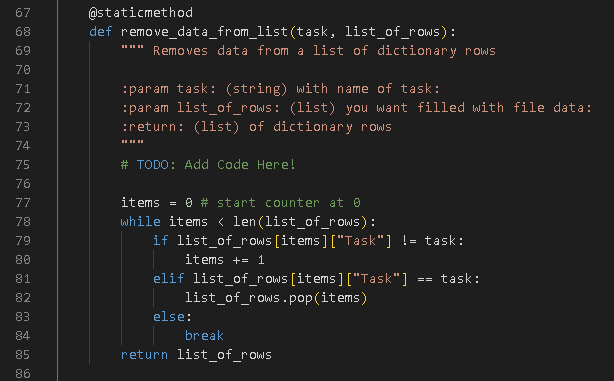


Figure 9: Remove data from list

# Step 4 – Write data to file

The next function ‘write\_data\_to\_file’ calls the open function with the ‘write’ argument and writes the contents of the list to the file.

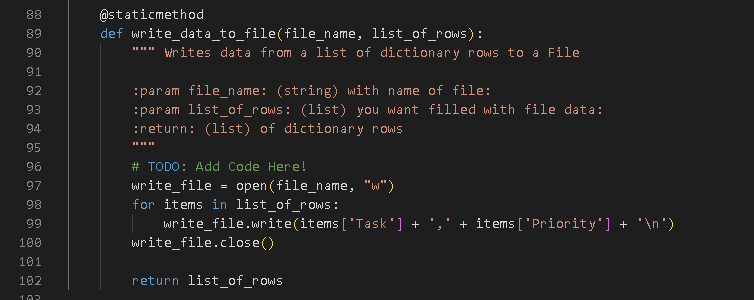


Figure 10: Write list to file

# Step 5 – Class IO

The next part of the script that was provided contains the input and output processing items. Here are the functions for displaying the menu to the user, capturing the users’ choice, outputting the current list items, inputting a new list item, and removing a list item.

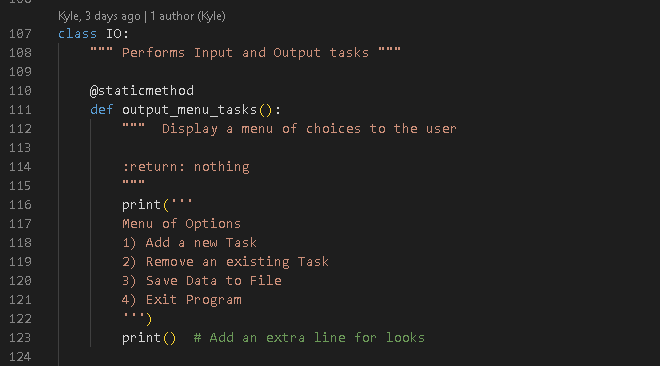


Figure 11: Menu Choices

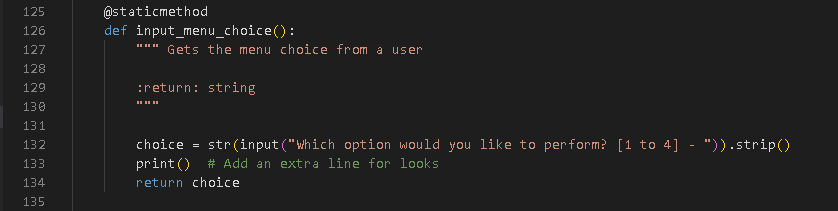


Figure 12: Input Menu Choice

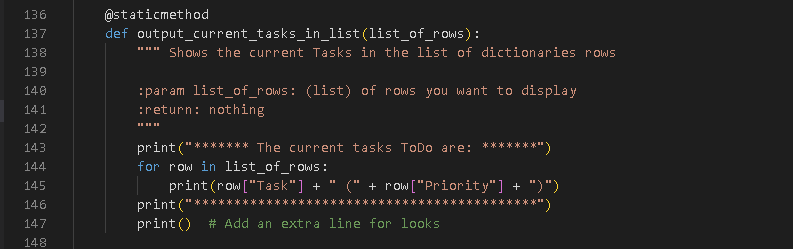


Figure 13: Current tasks in list

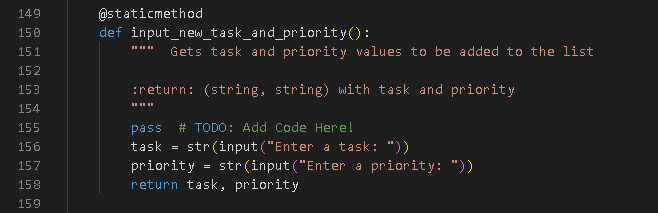


Figure 14: Input new Task and Priority Item

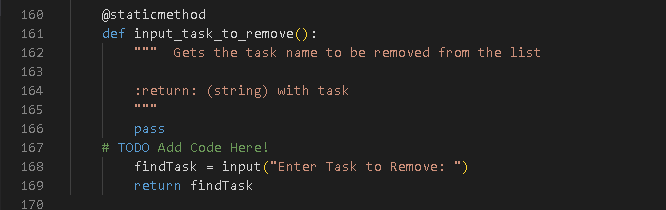


Figure 15: Input task to remove

# Step 6 – Main Body of the Script

This next section that was provided is the main script section, where the logic for running the script is contained. The script uses a while loop to keep displaying a list of the options a user can run. If the user chooses 1, it will run the part of the script that pertains to adding a new task and priority. If the user chooses 2, it will run the part of the script that pertains to removing an item form the list. If the user chooses 3, it will save the contents of the list to the ‘ToDoList.txt’ file, and if the user chooses 4, the script will exit.

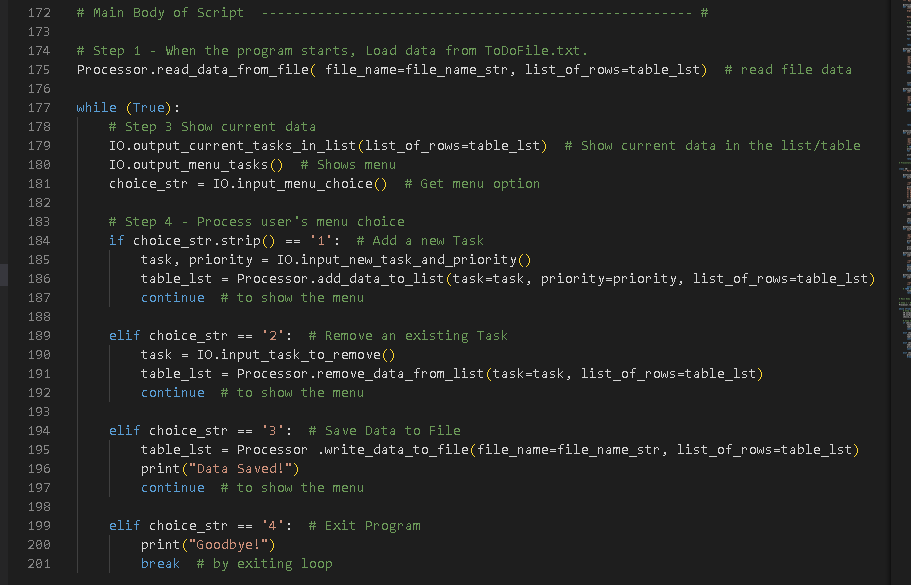


Figure 16: Main body of the Script

# Observations

This was a rather difficult assignment. It took me many attempts to get the logic right on the processing steps of adding and removing tasks to the list. I eventually got the while loops to work, but it took a long time going through the debug method to eventually get it right. I also had issues with returning the correct outputs from one function to another. Again, the debug process was instrumental here.

# Summary

In summary, utilizing all the resources provided to the class and the online lecture, this paper outlines all the steps that were taken to create a python script that results in a successful execution of the intended outcome (Figure 1). Following the steps outlined above will allow for the audience to recreate the presented result.