Andrei Arevalo

February 13th, 2021

Foundations of Programming, Python

Assignment06

<https://github.com/Arevalohm123/IntroToProg-Python>

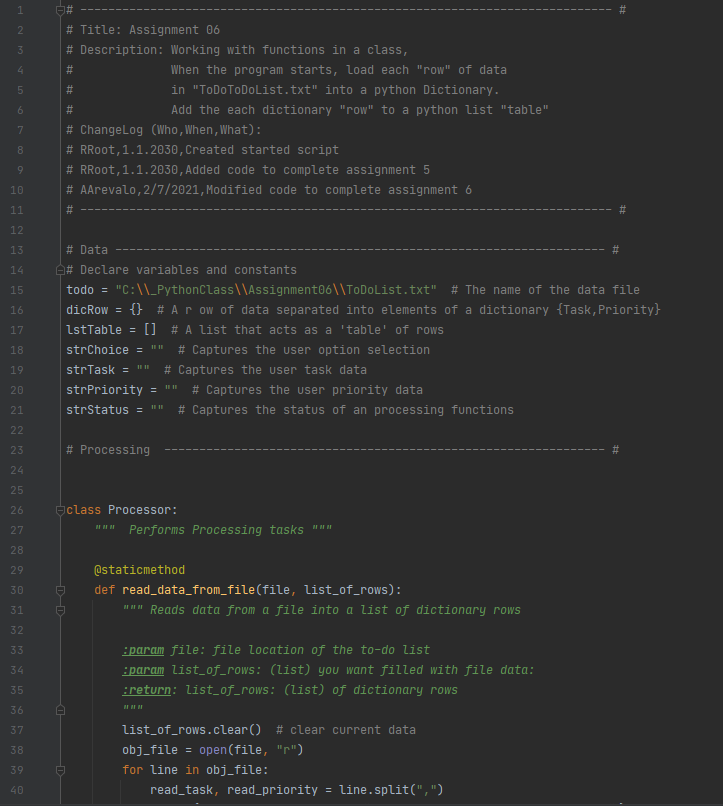
Modifying a To-Do List

# **Introduction**

In this assignment we are tasked with importing data from a to-do list and then manipulating the data several different ways including adding new tasks and deleting existing tasks. This is achieved using a while loop with 5 different options: show current data, add a new item, remove an existing item, save data to a file, and exiting the program. This code also builds on the ideas brought forth in the previous week with functions of different classes to reduce the length of the overall code. Overall this assignment was a good example of manipulating different types of data and working with a text file, and then taking those features and turning them into functions.

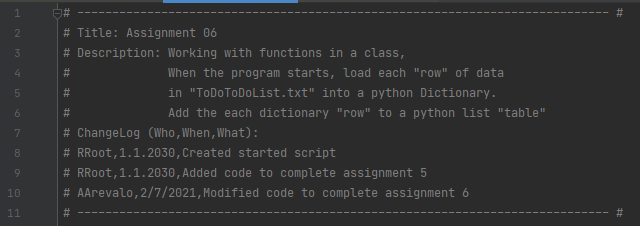
# **Creating the Script**

1. The purpose of this program is to capture and save tasks and priorities in a text file. This is building on previous weeks by adding additional features that allow the user to change the data from the text file and then resave it to a new file. The script allows for additions to the text file as well as showing the current data to the user. The main script is focused on a while loop that remains open until user exits the program. Before the while loop, the data is loaded into a table and variables are declared. In addition, this week builds on previous weeks by adding classes and functions, which goes beyond what was initially done in HW5. (figure 1)



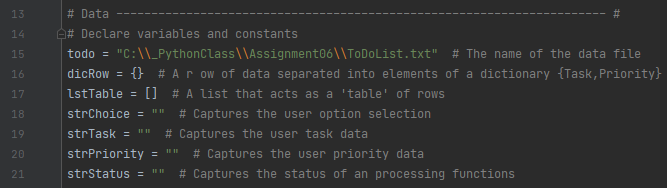
***Figure 1: The main script***

1. At the top of the script is the title block, which shows information about the script and keeps track of revision history in case the file needs to be revised in the future. The author and date are also included in this title block. In addition below the title block is a summary of the assignment with the task outlined. The main goal for this assignment was to continue what was accomplished in HW5 by turning many of the features in the HW script into classes and functions. (Figure 2)



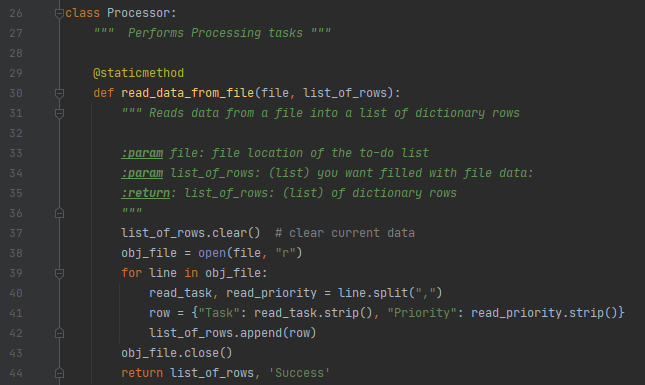
***Figure 2: Title block of the script***

1. At the beginning of the script, several variables are declared to show any future developers the naming convention I used for the different variables in the program. This allows for much easier reading in the future if someone else needs to change the script. (Figure 3)



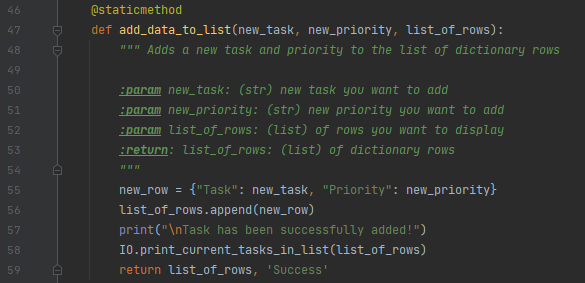
***Figure 3: Declaring variables to be used in the script***

1. The next part of the script introduces the first class of functions under “Processor”. These functions are CPU based functions that process data from the text file and perform other types of computations. The first of which is the read function which takes data from the text file and reads it into a list of rows. (Figure 4)



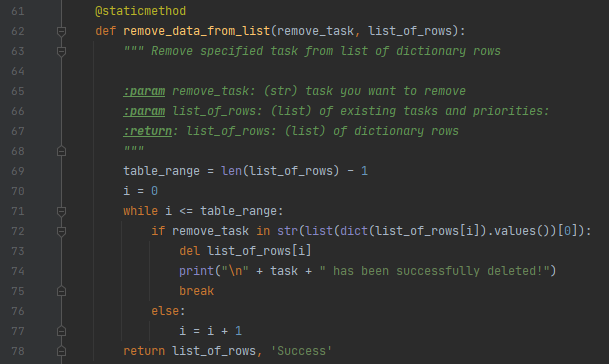
***Figure 4: Uploading the current tasks and priorities from the .txt file.***

1. The next function adds data to the list. This takes the inputs of new tasks and new priority and adds them to the list of rows. (Figure 5)



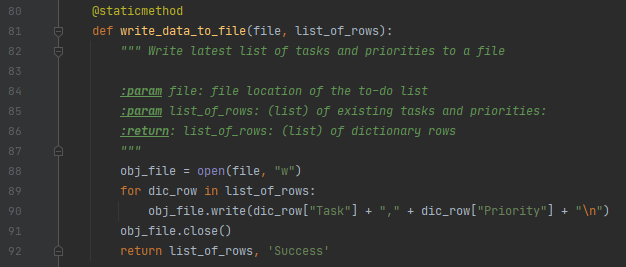
***Figure 5: Adding data to the list***

1. The next function is used to remove data from the list of rows. Similar to last week, the function searches the list of rows to find a match between the requested task to remove and the list of existing data. If there ends up being a match the data is deleted from the list. (Figure 6)



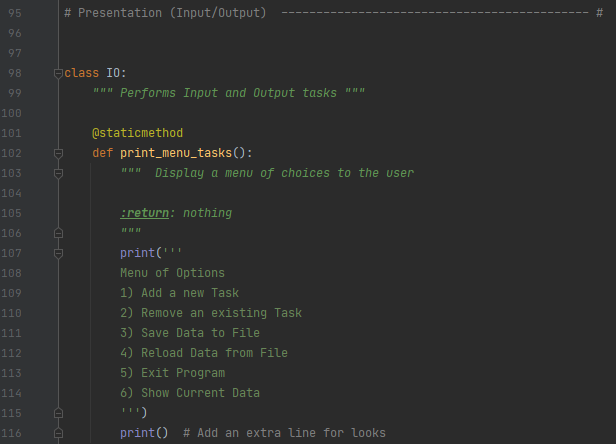
***Figure 6: Removing data from the list***

1. The next function goes through the current list of data and writes the list of rows to the text file. It does this by overwriting the text file and adding the data using the dictionary names. This can also be done using indices similar to calling cells within a matrix. (Figure 7)



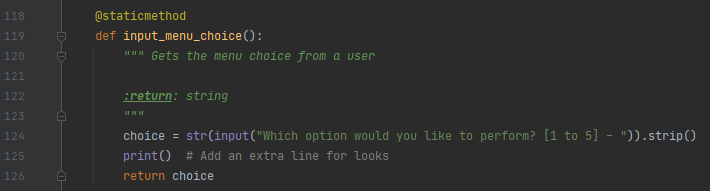
***Figure 7: Writing data to the file***

1. The next class of functions are IO functions, which perform all the input and output tasks within the main script. This is done through asking the user for input, which at times can be in the form of a new task to add or an existing task to remove. The first function is the “print menu” function which takes the menu of options and prints it for the user. (Figure 8)



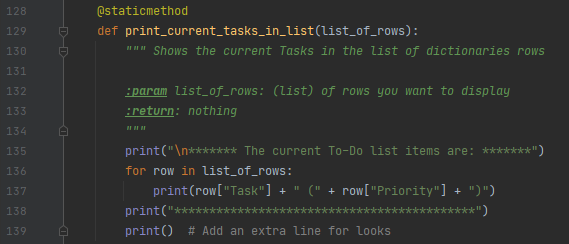
***Figure 8: Printing the menu for the user***

1. The next function in the input/output class of functions asks the user for which menu item to select. The user is prompted with the question of which option to select, and the output is that selection. (Figure 9)



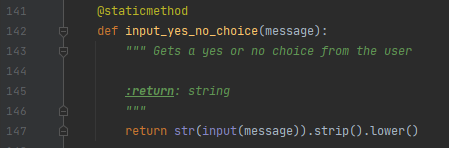
***Figure 9: Asking the user which option to perform***

1. The following function prints the current tasks in the list. This is done by going through the list of rows and print the task and priority within each of those rows. (Figure 10)



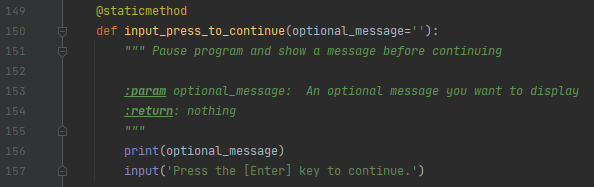
***Figure 10: Printing the current list of rows***

1. The next function simply asks the user a yes or no question, and the answer to that becomes the input for other functions such as saving the data or closing the program. (Figure 11)



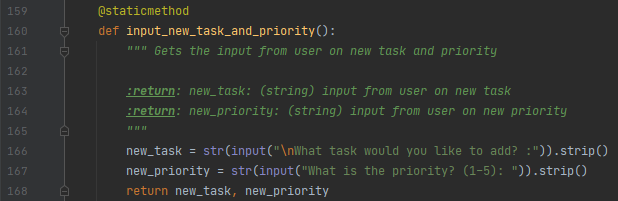
***Figure 11: Asking the user a yes or no question***

1. The next function asks the user to hit enter to continue. This function is used at the end of other selections as a method of separating the different options. (Figure 12)



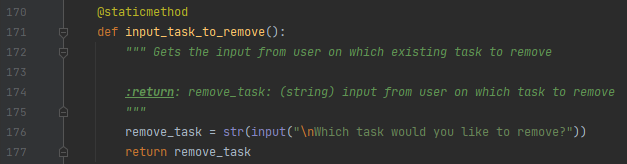
***Figure 12: “Press enter to continue”***

1. The next function asks the user which task and priority to add. This becomes the input to the processor function that adds the task and priority to the list of rows. (Figure 13)



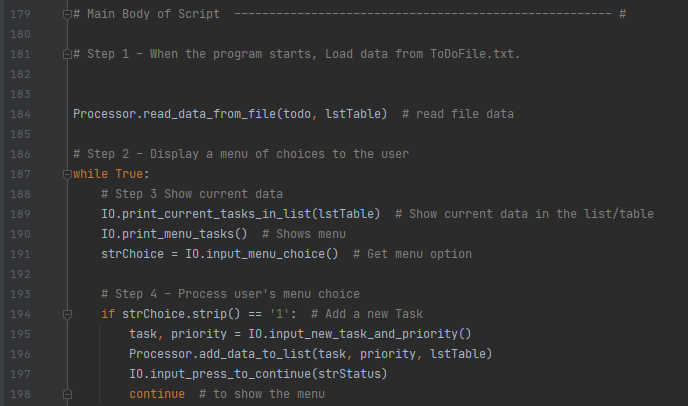
***Figure 13: Inputting the new task and priority***

1. The next function asks the user which task to remove. This becomes the input to the searching function that goes through the list of rows to find the existing task to remove. (Figure 14)



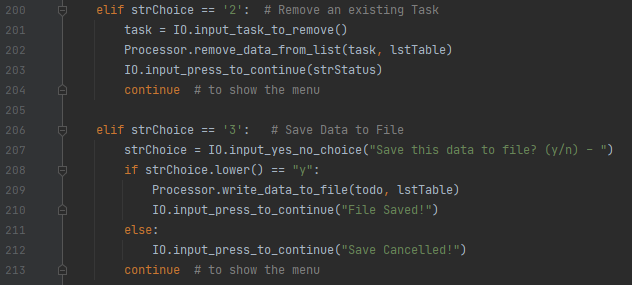
***Figure 14: Inputting the task to remove***

1. The next section starts by calling the function to read the data to the list of rows. The “while” loop is then started and then breaks down into the different menu options. If the user selects “1” the new task and priority are added to the table. (Figure 15)



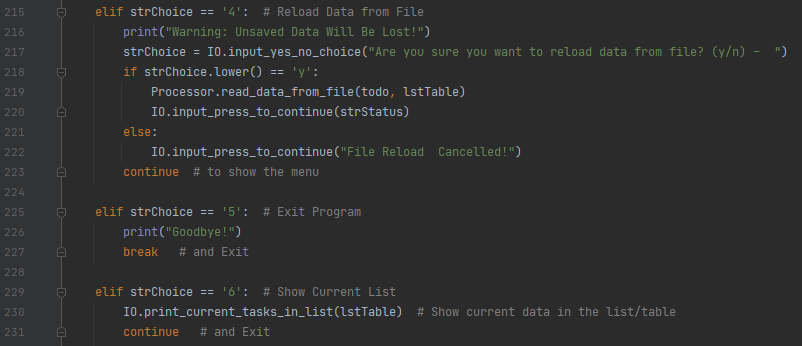
***Figure 15: Main script with first menu option***

1. The next part of the script shows the next two options in the menu which are removing a task and saving the data. Both options utilize functions created both in the processor and IO class. (Figure 16)



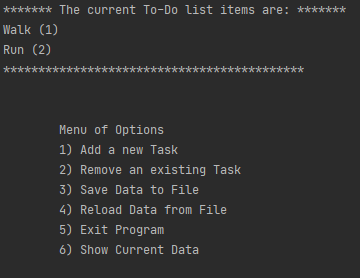
***Figure 16: Removing a task and saving the data***

1. The last bit of code shows the last 3 options in the menu which are reloading the data from the file, exiting the program, and an additional feature that I added which is showing the current list data. (Figure 17)



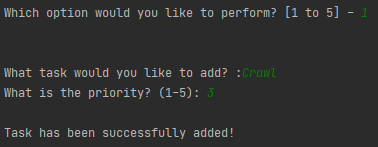
***Figure 17: Reloading the data, exiting the program, and showing the current list***

1. When running the program, the current list is immediately created and shown to the user along with the menu options. (Figure 18)



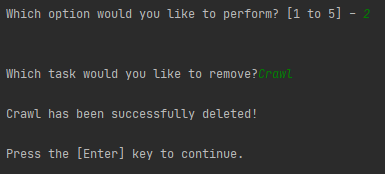
***Figure 18: Demo of the program showing the imported list and menu options***

1. Below is a demo of the program with using the first option in the menu. The green text is the data the user needs to input. Once the data is inputted a success message is displayed back to the user. (Figure 19)



***Figure 19: Demo of adding a task to the table***

1. The last bit of demo of the script shows the user selecting to remove a task from the list and a message being displayed back to the user that the task has been deleted. (Figure 20)



***Figure 20: Demo of removing a task from the list***

# **Summary**

In this assignment we are tasked with importing data from a to-do list and then manipulating the data several different ways including adding new tasks and deleting existing tasks. Expanding on the previous HW assignment, we created several functions for printing, reading, and writing the existing table. There were several times where I wanted to see what the latest data was on the table, and it was nice having a function to call that made the overall code shorter since I didn’t have to repeat the same line of code each time I wanted a printout of the table. This code was a good extension of what was accomplished in HW5 and turning many of the features of the previous assignment into functions.