Anna Paulson

May 17, 2020

Foundations of Programming: Python

Assignment 05

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

Python Script to Prioritize Tasks

# Introduction

In this paper I will discuss the steps I took to complete Assignment 05 for the class Foundations of Programming: Python. The assignment involves adding to a start script in order to write a script that creates a list for a user to select choices from in order to add and prioritize or remove tasks in a text file. Furthermore, the organization of the file would be achieved by reading and writing files into dictionaries and lists.

# Assessing the Starter Script

I opened the starter script and took a look at what kinds of steps would need to be taken in order to complete the assignment. The starter script provided specific guidance as to how to complete the script. This allowed me to determine the framework necessary to complete the assignment. The starter script outlined the menu for the user to select from in order to alter the “todolist.txt” file that contained the list of tasks and priorities. When selected by the user, the options would allow them to manipulate the table of data in the todolist.txt file with dictionary rows. The script needs to both read and write data in the file.

# Completing the Script

I began by defining the variables like I have done in past assignments. I then continued to write the loop script in order to achieve the desired result of prompting the user to select a choice from the menu provided in the starter script.

Per the starter script, the data in the text file needed to be in dictionary rows stored in a list. Figure 1 shows the script opening the text file, reading through each row of data in the file, storing the rows of data into a dictionary row, and finally storing new dictionary rows into a list array.

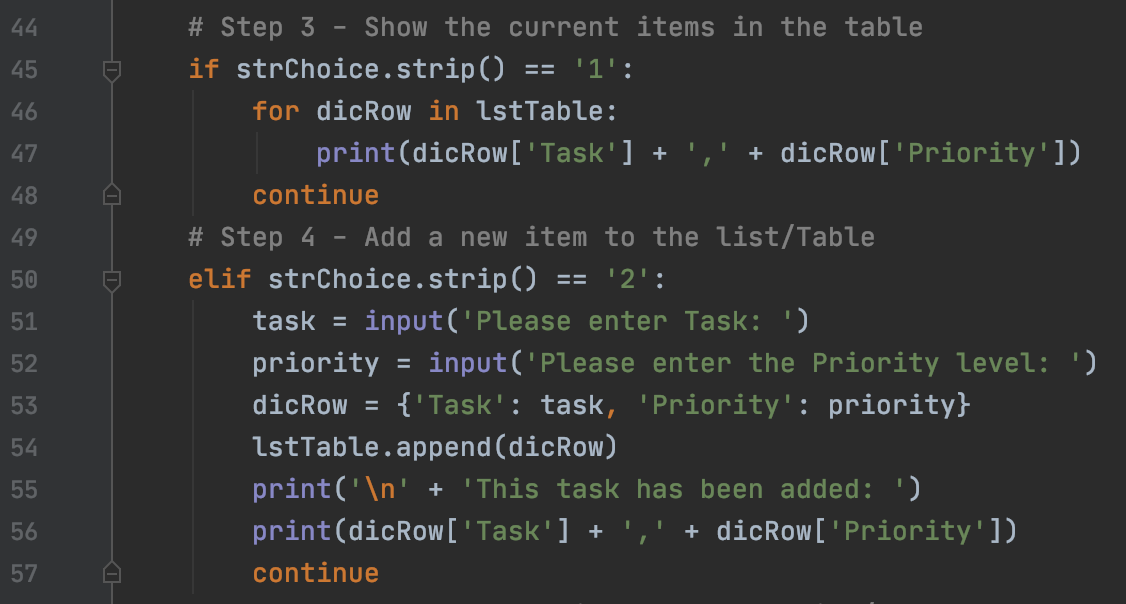
A screenshot of a cell phone

Description automatically generated

***Figure 1. Processing Rows, Lists, and Dictionaries***

I then continued writing the script for the ‘Input/Output’ steps and prompts. The first option functioned to display any data currently in the text file. This is done by using a loop to cycle through and finally display the values in the dictionary rows stored in the text file’s table. This begins the use of the strip() function throughout the program in order to strip any invisible carriage returns and spaces off the beginning and end of strings.

The second option prompted the user to input data for their task and priority and append the table with their inputs. The script to achieve the first and second option can be seen below in Figure 2.



***Figure 2. User input Options 1 and 2***

The third option shown in Figure 3 prompted the user to identify a task to remove and then append the table by deleting the desired task. I also included script so that if the user entered a task that was not already present in the text file, they would be shown what is actually on their current list of tasks.

# A screenshot of a cell phone Description automatically generated

***Figure 3. Option to Remove a Task***

The script “if itemRemoved is None:” allows it to continue with an alternate action if the user’s input is not currently in the text file’s table. The remove() functions allows the script to delete the user inputted item from the table as shown in Figure 3.

The final options on the menu allow the user to save their data in the text file and finally end the program. The script to achieve these tasks is shown below in Figure 4.

A screenshot of a cell phone

Description automatically generated

***Figure 4. Saving the File and Ending the Program***

The script opens the file, loops and writes the tasks and priorities to the file, and closes the text file. The print() function is used to communicate the data being saved to the user. Figure 4 also shows the program’s exit and a friendly ‘Thank you’ printed to the user.

# Running the Script in PyCharm

Figure 5 shows the script running in PyCharm and responding to the user’s inputs.

A screenshot of a cell phone

Description automatically generated

***Figure 5. Script Running in PyCharm***

Figure 5 shows the script successfully adding the user’s input and then displaying it back to them when prompted.

# Running the Script in Terminal

In Figure 6, the user adds an item to the list and then tried to remove an item that is not on their list. The new item is successfully added. When they enter the item that is not on their list, they are reminded that it is not on their list.

A screenshot of a social media post

Description automatically generated

***Figure 6. Script Running in Terminal***

As shown in Figure 5 and 6, the script successfully ran in both PyCharm and Terminal. It’s time to upload the work to Github.com so that the script can be shared.

# Summary

The “Separation of Concerns” or specifically in this case, adding data to a file vs. adding data to memory is critical in Assignment05 so that data is properly appended and saved in a text file. The distinction means that data will be properly saved when the user thinks it is being saved. The script created a list for a user to select choices from in order to add and prioritize or remove tasks in a text. Proper indexing within the table in the text file, use of dictionaries, and adding to existing scripts lead to success in creating a functioning script to prioritize tasks.