Shruthi Srivatsan

November 24, 2020

Foundations of Programming- Python

Assignment 06

Creating a Multi- Option Task List Script using Functions in Python

**Introduction:**

In this paper I will be discussing how to write a script that loads the data from the todofile.txt. Then it will prompt the user options through a menu where the user can select to add, delete, save, reload data or exit. The script uses the staticmethod that directly prompts the computer to do the commands. The new input will be programmed as dictionaries.

***Script Creating:***

First I used the python assignment 6 starter.py file to use the lay out and notes that was provided for this assignment. Using that we were asked to fill out the portions that were labeled “#TODO-add code here. Before creating the table of options, I was asked to write code that takes a file named “todolist.txt” and load the data from a text file and place it into a table.

First I had to declare the variables that would be used throughout this program. This would allow us to follow through with the script, everything is defined and program will be able to simply do the functions that are asked of it. I defined the following variables, shown in (Figure 1.)

Graphical user interface, text

Description automatically generated

Figure 1. Variables definitions

***StaticMethod – Read Data from File***

Then I had to write the code for each of the @staticmethod. I started off for the processing section of the code. The first @staticmethod had the code: def read\_data\_from\_file(file\_name, list\_of\_rows): on the following line. After that I had to write the following script (Figure 2.):

Text

Description automatically generated

Figure 2. Read data from file- staticmethod

This script basically opens the file\_name that is defined as the todofile.txt and reads it. It will loop and read the row as task, priority and append. Once that is done, it will show success and ask user to prompt or select next choice from the menu.

***StaticMethod- Add Data to List***

The next @staticmethod was used to write def add\_data\_to\_list (task, priority, list\_of\_rows). This was written to allow user to add data to the current file (Figure 3). The row = {“Task”:str(task).strip(), “Priority”: str(priority).strip()} and the list\_of\_row.append(row) will allow you to add the data to the file. Once data is added it will return the list of rows and display success.

Text, application, chat or text message

Description automatically generated

Figure 3. Add Data to List- StaticMethod

***StaticMethod- Remove Data from List***

The next @staticmethod was used to def remove\_data\_from\_list(task,list\_of\_rows). This would remove the data from the user input into a list as a dictionary row. So just like the other @staticmethods, we defined the param task, list\_of\_rows and return. Then I inputted the loop for row in list\_of\_rows which == strTask, it would remove a row if the task in table matches input with this script: list\_of\_rows.remove(row). Then once user enters the task that they want to remove it will print(“Row Removed Successfully) and print(list\_of\_rows). If the row is not found it has an else statement that will print(“Row is not found”). Once it successfully removes the row it will print success (Figure 4.)

Text

Description automatically generated

Figure 4. Remove data from list

***StaticMethod- Write Data to File***

The last @staticmethod I wrote was for def write\_data\_to\_file(file\_name, list\_of\_rows): this will write the data from the user input to add into the list as a dictionary row. This starts with strFile = open(file\_name, ‘w’). Then I listed the loop that will write in the strFile in the format mentioned in the line strFile.write(str(row[‘Task’])+ “,”+ str(row[“Priority’]+ “\n”)) which will write data to the text file. Then the rest of the script will close the file and print that it was saved (Figure 5.)

Text

Description automatically generated

Figure 5. Write data to file- staticmethod

***StaticMethod- Input New Task and Priority***

This starts the presentation of the input and output.

For this static method I added an input function for task and priority that will prompt the user to enter a new task and new priority to add on to the text file (Figure 6.)Text

Description automatically generated

Figure 6. Inputting New task and priority- staticmethod

***StaticMethod- Input Task to Remove***

This is to get a task input from the user to remove from the list. It will prompt the user to enter a task that the want removed and then return. (Figure 7.)

Text

Description automatically generated

Figure 7. Input task to remove- staticmethod

Now I moved on to the portion of the code that wanted to me [ENTER CODE HERE] for the presentation part of the script. Assignment 6 had everything laid out except for the portions of the text that prompted us to write code in order to complete that desired process. Until the static method for input new task and priority- the script was already filled out. This prewritten code section basically displays the menu that the user will select from and defines more staticmethods for menu choice, print current tasks in list, inputting yes or no choice, and inputting press to continue (Figure 8)

Text

Description automatically generated

Text

Description automatically generated

Figure 8. Pre-Written Code section

Now in the main body of the script, step 1, 2, and 3 are already written and require no additional code. Moving on to step 4- I wrote the following code in order to provide the user with the correct output when they select option 1 from the given menu. (Figure 9)

Graphical user interface, text

Description automatically generated

Figure 9. If User selects 1

If user selects 2, then it will ask user to input what they would like to remove and then list which row was removed (Figure 10)

A picture containing text

Description automatically generated

Figure 10. If User selects 2

If user selects 3, it will prompt user to select whether they want to save file or not “y” for yes or “n” for no and if they select yes it will save the entry to the file (Figure 11).

Text

Description automatically generated

Figure 11. If User selects 3

If user selects 4, it will reload the data that is on the file and let user know that anything unsaved will be lost. They will have to enter y for yes again to proceed and once they do it will reload the lsttable from the strFileName (that was defined). If they say or press anything else or n for no, it will say “File Reload Cancelled”. (Figure 12).

Text

Description automatically generated

Figure 12. If User selects 4

If user selects option 5, it will say goodbye! (Figure 13)



Figure 13. If User selects 5

**Running script in Pycharm and Terminal:**

You can run this code in two ways: on Pycharm (Figure 14) and on Terminal (Figure 15). Your text file should look like this if you open to see the current data stored in the file (Figure 16).

Text

Description automatically generated

Text

Description automatically generatedA picture containing shape

Description automatically generatedA picture containing text

Description automatically generatedA picture containing graphical user interface

Description automatically generated

Figure 14. Script run through Pycharm

Text

Description automatically generated

Text

Description automatically generated

Figure 15. Running it in Terminal

Graphical user interface, text

Description automatically generated

Figure 16. Text file