Stephanie Miller

November 15, 2022

IT FDN 100

Assignment 5

To Do List Script

# Intro

This week we started with a partially written code we then had to add to. Many of the concepts that we used this week are ones that we have previously learned and are simply expanding upon. The main new portion of this week’s code was the use of dictionaries.

# Creating the Script

As with every script we start with the overview of the document which includes a title, description, and change log. In the change log there are two entries. The first is for the person who created the initial code that we are working off. The second is me. So, I am documenting that I am the person making the changes to a premade document and what the changes I am making are. (Figure 1)

Text

Description automatically generated

**Figure 1: Overview of the Script**

The section of code is where we declare the variables and constants. For this script this section was written by the person who originally created the code. I used these when I was writing the rest of my code so that it would match up with the predetermined variable names. (Figure 2)

Text

Description automatically generated

**Figure 2: Declaring the Variables and Components**

In this section of the code, we are bringing in the current data that we have stored in our text file. We are taking this list and transforming it into a dictionary before adding it to our table. This will allow us to see the data currently in the text file. Going through the code the first line is the line we use to open the file we want to pull the data from. Here we use an r to convey that we want to read the data. The for loop is pulling the lines from the file, converting it to a dictionary, and then saving the data to the table. The last part of the code is closing the file. (Figure 3)

Text

Description automatically generated

**Figure 3: Pulling Data from the File into a Dictionary**

The section of code is the beginning of the while loop. The first part of the while loop is the menu of options that the user can choose from. The user will be able to choose the option they want to perform by entering that number into the next section in the code. This section askes the user to input the number of the action they want to perform. This section of code was all prewritten and provided by the person who created the code. (Figure 4)

Text

Description automatically generated

**Figure 4: Menu of Options for the User to Choose From**

The next step in the script is the first option in the menu. Here we want to display the current information that we have stored in the table. The if statement looks to see if the number entered is a 1. If this is true, then the statement “Your current data is:” will be printed. The for loop following the print statement will also be run. This will go through and display each row in the table. This is the same table that we loaded our text file data too. If there was no data in the file and we have not entered any data, then this will show no data. (Figure 5)

Text

Description automatically generated

**Figure 5: Code to Print Current Data in Table**

If a number 2 is entered by the user, then this section of code will be run. The code first checks to see that the user input a 2 if that is true then the print statement “Enter a Task and its Priority” will be displayed. The first user input will also be displayed asking the user to enter a task. Once the user has done this the second prompt asking for the user to enter the priority of the task will be displayed. Once the user has made their entries the data will then be added to a dictionary and saved to the table with the data we had previously pulled from the file and any other data we had entered. (Figure 6)

Text

Description automatically generated

**Figure 6: Code Where User Can Input New Data**

If a number 3 is entered by the user, then this section of code will be run. The code first checks to see that the user input a 3 if that is true. The user will then be asked to input the task they would like removed from the table. The code will then run through all the rows in the table, removing the one(s) that matched the entered task. If the row was found, then the statement “Row Removed” will be displayed. If the row was not found, then the statement “Row Not Found” will be printed. (Figure 7)

Text

Description automatically generated

**Figure 7: Code for User to Remove Task from File**

If a number 4 is entered by the user, then this section of code will be run. The code first checks to see that the user input a 4 if that is true, then the next line in this code will open the file we want to write our code too. In this case we want to use the write feature as we pulled all current data in the file to the table we are then saving. So, we want to write over everything to prevent duplicates or keeping lines we wanted deleted. Next the code will cycle through each line in the table and write it to the file. Next line of code will close the file. Lastly, the statement “Data Saved to File” will be printed so the user knows it worked. (Figure 8)

Text

Description automatically generated

**Figure 8: Code Used to Save Data to File**

If a 5 is entered by the user, then this section of code will be run. The code first checks to see that the user input a 5 if that is true, then the code will print a done statement, so the user knows that the script has ended. The code that ends the script is the break statement on the last line. (Figure 9)

Text

Description automatically generated

**Figure 9: Code to Close the Script**

# Testing the Script

## Running in PyCharm

When the code is first run in PyCharm the menu of options and request for the user to input the number of the option they would like to perform will be shown. (Figure 10)

Text

Description automatically generated

**Figure 10: Menu of Options and User Input Request**

When the user inputs a 1 then the current data contained in the table is displayed. Currently there is one line that I pulled in from the file and that is shown in the below screen shot. (Figure 11)

Graphical user interface, text

Description automatically generated

**Figure 11: Displaying Current Data in Table**

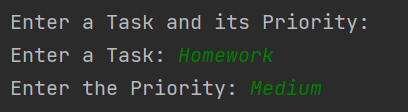
After the data is displayed you will be taken back to the home screen. This time I will select a two, this will allow us to enter new data into the table. (Figure 12)

Text

Description automatically generated

**Figure 12: Returning to the Main Menu**

When the second option is selected, we have the option to add in new lines of data. The first thing to be printed tells the user what they need to do. They will then be asked to enter a task. Once they have done this the second user input will appear asking for them to enter the priority of the task. (Figure 13)



**Figure 13: Entering New Data to the Table**

Here I went back and selected a one again to show that the new line has been added to the table. (Figure 14)

Text

Description automatically generated

**Figure 14: Returning to the Main Menu and Displaying Data Currently in Table**

Again, after the data is displayed you will be taken back to the home screen. This time I will select a three, this will allow us to remove a row from the table. (Figure 15)

Text

Description automatically generated

**Figure 15:** **Returning to the Main Menu**

When the third option is selected, the user will be prompted to enter a task that they would like to have removed from the table. In this case I entered the task Clean and when entered a statement is printed telling me the line has been removed. If the line was not in the table, then we would see the statement “Row Not Found”. (Figure 16)

Text

Description automatically generated with medium confidence

**Figure 16: Removing Task from Table**

Again, after the data is displayed you will be taken back to the home screen. This time I will select a four, this will allow us to save our new data entries to the file. (Figure 17)

Text

Description automatically generated

**Figure 17: Returning to the Main Menu**

When the fourth option is selected, background code will run saving the data to the file. The user will only see the data save to file print out telling them the script has run successfully. (Figure 18)



**Figure 18: Statement Displayed After Data is Saved to File**

Again, after the data is displayed you will be taken back to the home screen. This time I will select a five, this will allow us to end the script. (Figure 18)

Text

Description automatically generated

**Figure 19: Returning to the Main Menu**

When the fifth option is selected, background code will run which stops the code from looping. After the loop has been broken the statement “done” is printed so the user knows it has finished. (Figure 20)



**Figure 20: Statement Printed After Breaking the Loop**

The last thing we can do is check in the file to make sure that everything worked as we expected. Since I removed the original row of code from the file, I only expect to have one row left, which is what I see in the file. (Figure 21)



**Figure 21: Data in the File**

## Running in Command Screen

When the code is first run in the Command Screen the menu of options and request for the user to input the number of the option they would like to perform will be shown. (Figure 22)

Text

Description automatically generated

**Figure 22: Menu of Options and User Input Request**

When the user inputs a 1 then the current data contained in the table is displayed. Currently there is one line that I pulled in from the file and that is shown in the below screen shot. (Figure 23)

Graphical user interface, text

Description automatically generated with medium confidence

**Figure 23: Displaying Current Data in Table**

After the data is displayed you will be taken back to the home screen. This time I will select a two, this will allow us to enter new data into the table. (Figure 24)

Text

Description automatically generated

**Figure 24: Returning to the Main Menu**

When the second option is selected, we have the option to add in new lines of data. The first thing to be printed tells the user what they need to do. They will then be asked to enter a task. Once they have done this the second user input will appear asking for them to enter the priority of the task. (Figure 25)

Text

Description automatically generated

**Figure 25: Entering New Data to the Table**

Here I went back and selected a one again to show that the new line has been added to the table. (Figure 26)

Text

Description automatically generated

**Figure 26: Returning to the Main Menu and Displaying Data Currently in Table**

Again, after the data is displayed you will be taken back to the home screen. This time I will select a three, this will allow us to remove a row from the table. (Figure 27)

Text

Description automatically generated

**Figure 27: Returning to the Main Menu**

When the third option is selected, the user will be prompted to enter a task that they would like to have removed from the table. In this case I entered the task Clean and when entered a statement is printed telling me the line has been removed. If the line was not in the table, then we would see the statement “Row Not Found”. (Figure 28)

Text

Description automatically generated

**Figure 28: Removing Task from Table**

Again, after the data is displayed you will be taken back to the home screen. This time I will select a four, this will allow us to save our new data entries to the file. (Figure 29)

Text

Description automatically generated

**Figure 29: Returning to the Main Menu**

When the fourth option is selected, background code will run saving the data to the file. The user will only see the data save to file print out telling them the script has run successfully. (Figure 30)



**Figure 30: Statement Displayed After Data is Saved to File**

Again, after the data is displayed you will be taken back to the home screen. This time I will select a five, this will allow us to end the script. (Figure 31)

Text

Description automatically generated

**Figure 31: Returning to the Main Menu**

This time when we enter 5 the screen will just close. I did not add an additional input statement to keep the screen open so you will not see the done statement like you do in PyCharm. The last thing we can do is check in the file to make sure that everything worked as we expected. Since I removed the original row of code from the file, I only expect to have one row left, which is what I see in the file. (Figure 32)

Logo

Description automatically generated with low confidence

**Figure 32: Data in the File**

# Summary

This week we had the challenge of finishing off another person’s code. This is valuable experience to have as this is something we will regularly deal with in a work environment. This week we also learned how to create a table of dictionaries and different ways we can use the data. We mainly focused on expanding upon what we learned last week. Instead of using lists we used dictionaries but used the same code foundations.