Hwai-Jiang Jong

May 10, 2021

Foundations of Programming: Python

Assignment 05

<https://github.com/uwp-h2021/IntroToProg-Python>

The To-Do List Script

# Introduction

In this report, I am documenting what I learned in Module 5 and the steps I took to create a script to perform Assignment05. It contains three main sections (excluding this introduction):

1. What I learned in Module 5;
2. Developing a script in PyCharm for Assignment05;
3. Summary

# What I Learned in Module 5

This section documents my key learning from reading and watching the resources provided in the Module 5 assignment. There may be duplicate pieces of information of an item I learned from multiple sources, so each learned item is mentioned only once in the following quoted sources.

1. Course video of module 5: I learned collection of data in lists and dictionaries, and how to load data from a file to memory and upload data from memory back to the file. The subscript of a dictionary using keys instead of numeric indices is a nice feature of Python which concept is not hard to grasp. The idea of “Separation of Concern” and use of Functions are important to drive organized and flexible programming. In a collaborative development environment, script templates are a key to standardizing the code format to make it easy for other programmers to read and understand. Error handling was lightly discussed, and I got the idea of the effort to make the script more user friendly for error handling. Finally, GitHub was introduced, and we will be using it to share the projects in this assignment.
2. Textbook: Chapter 5 discusses lists and dictionaries in details, but file reading/writing is not covered. For lists, I learned the difference between the *del* statement and the *.remove()* method on the elements in a list. For extracting an element from a two-dimensional table, one needs to use the double brackets to access it. For example, one should use *scores[2][0]* to access the first element of the third list within the table *“scores”*. For dictionaries, the discussion is all covered by the course video.
3. <https://www.afterhoursprogramming.com/tutorial/> (external site): I went to this website to learn lists, dictionaries, and reading file. One interesting thing, which is also mentioned in the textbook, is the use *in* operator with dictionary. Only the keys, but not values, of a dictionary can be checked against their existence in it. Figure 1 shows an example of a loop over the keys of a dictionary.

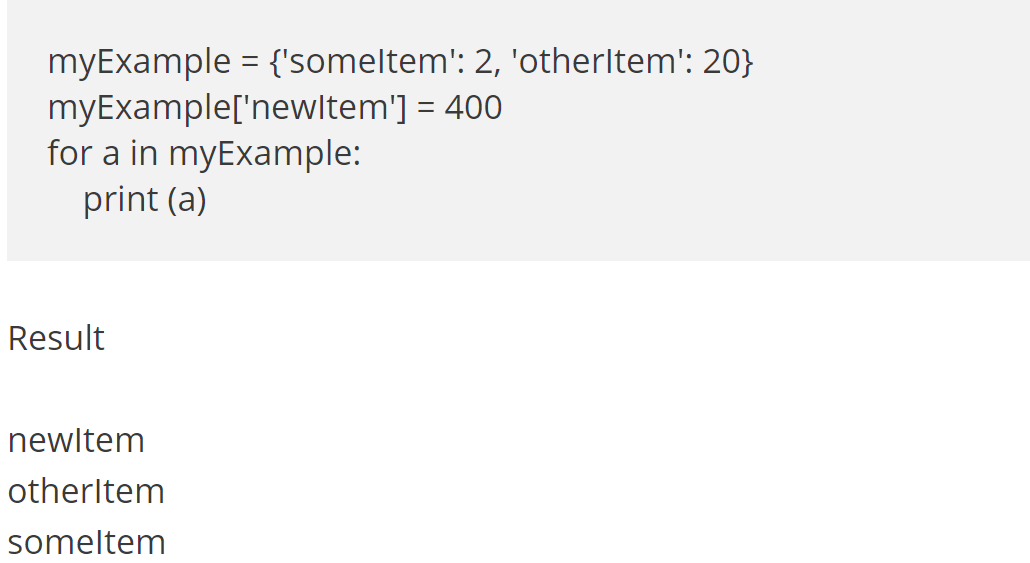


Figure Accessing the Keys in a Dictionary

1. <https://youtu.be/m0o0CkYsDzI> (external site): This website discusses opening a file, writing data into the file, and then reading the data back to memory. The content is already covered in the course video. One interesting thing was the method *.read()* reads the data in the file directly into a string. If the ‘\n’ exists at the end of each row in the file, then the string will carry them in it. As a result, when the string is printed, it will display data exactly the same as in the file. Another interesting method is the *.readline()* and .*readlines()* which allow reading data one line or multiple lines from the file.

# Writing To-Do List Script

## Script Construct Planning

Per assignment05, the script will load the existing data from “ToDoList.txt” on the hard drive into a list table, then ask users to choose between five different options concerning the data. The five options are

1. Show current data
2. Add a new item of task and priority
3. Remove an existing item of task and priority
4. Save the updated data into the file
5. Exit the program

I first wrote a pseudo code as the following.

*Open a file and read the ToDoList data into a list table, then close the file*

*Loop {*

*Print the menu of options for user to select*

*If option 1 is selected: Show current data*

*If option 2 is selected: Add a new item of task and priority to the table*

*If option 3 s selected: Remove an existing item of task and priority from the table*

*If option 4 is selected: Save the data into the file and close the file*

*If option 5 is selected: Exit the program*

*If selection is not option 1 - 5, continue the loop*

*}*

## Writing the Script in PyCharm

I launched PyCharm and opened a new project folder “C:\\_PythonClass\Assignment05\”. I loaded the provided template “Assignment05\_starter.py” into PyCharm and modified it to fulfill the assignment. I saved the script as the “Assignment05.py” as my final script file for submission.

This script was completed at incremental steps. I started with a text file “ToDoList.txt” in the working folder, with two items of task and priority. In the code, I created the variable definition to include all variables needed in this code. Then the data in the “ToDoList.txt” were loaded into a list table. The five options with their actions were implemented in a while loop based on their specific actions on the data. The script with its various sections is described in detail as follows.

1. Change History: Figure 2 shows the history of the steps I took to make this script perform as required.

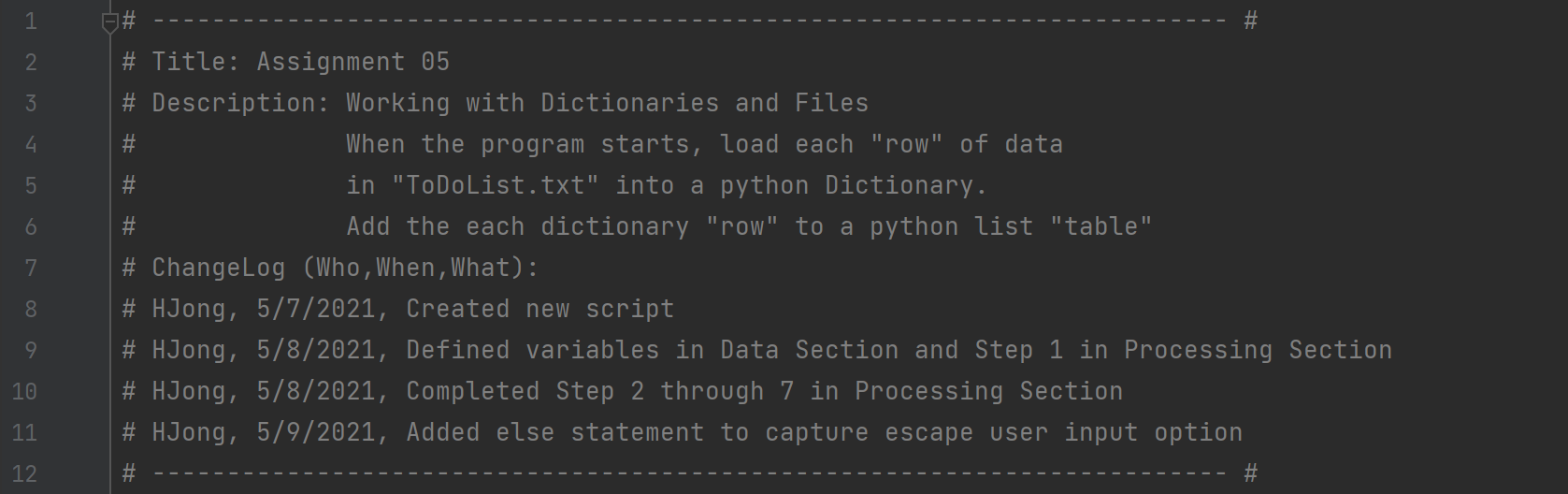


Figure Change Log of the Script

1. Data definition: This section includes the variables I used to perform tasks on the data, as shown in Figure 3.

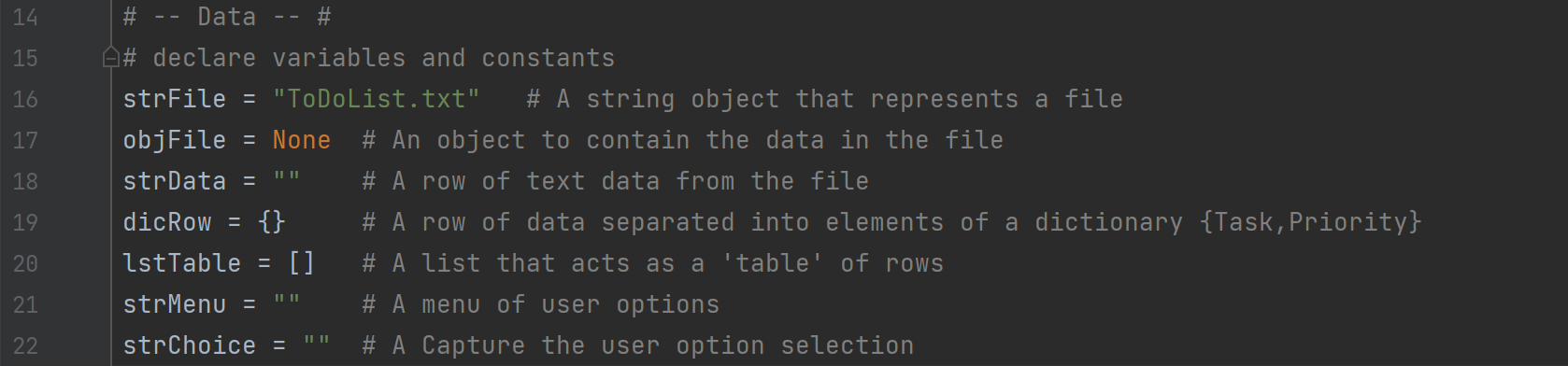


Figure Data Declaration Section of the Script

1. Data Processing: The data are loaded into a list table from the data file. See Figure 4.

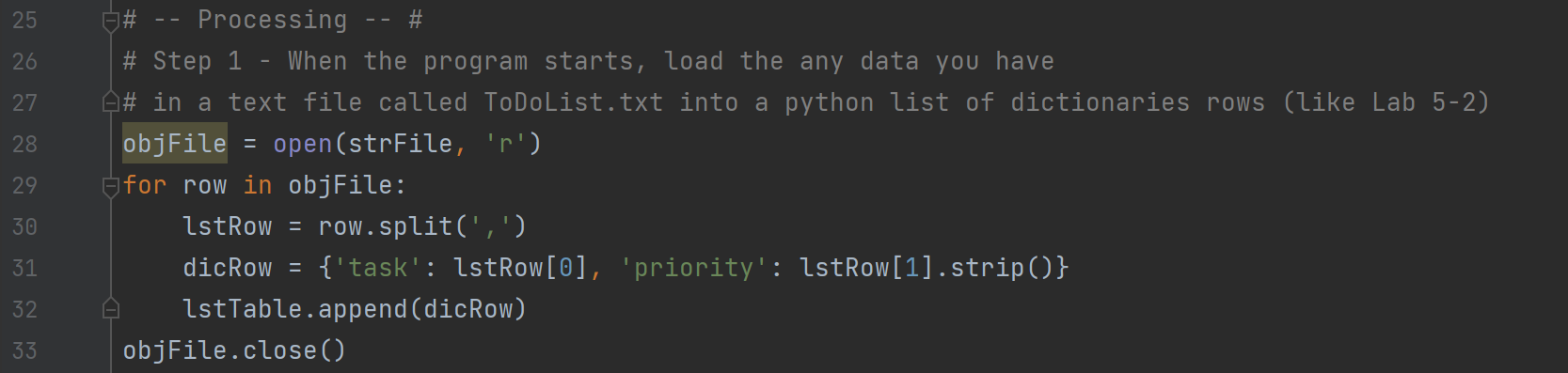


Figure Loading Data from File to Memory (a List Table)

1. Data input/output: The data input, output, data addition, and data removing are contained in a while loop, allowing the user to perform relevant tasks until the user exits the program. The beginning of the while loop is to print the option menu for the user to select from (Figure 5).

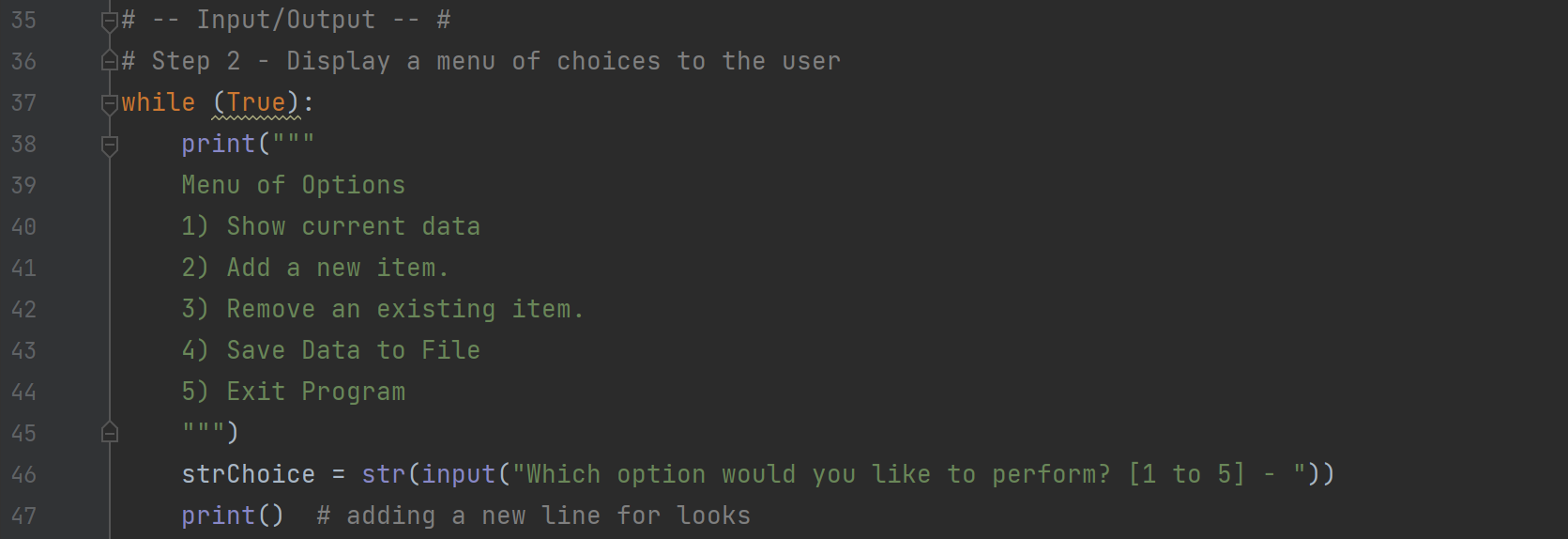


Figure Displaying the Menu of Options and Asking for User Selection

Based on the user selection, various tasks (showing data, adding data, removing data, recording data, and exiting the program) can be performed using the *if* statements within the while loop. Figure 6 shows the implementation of options 1-3 (showing, adding, and removing data) within the while loop.

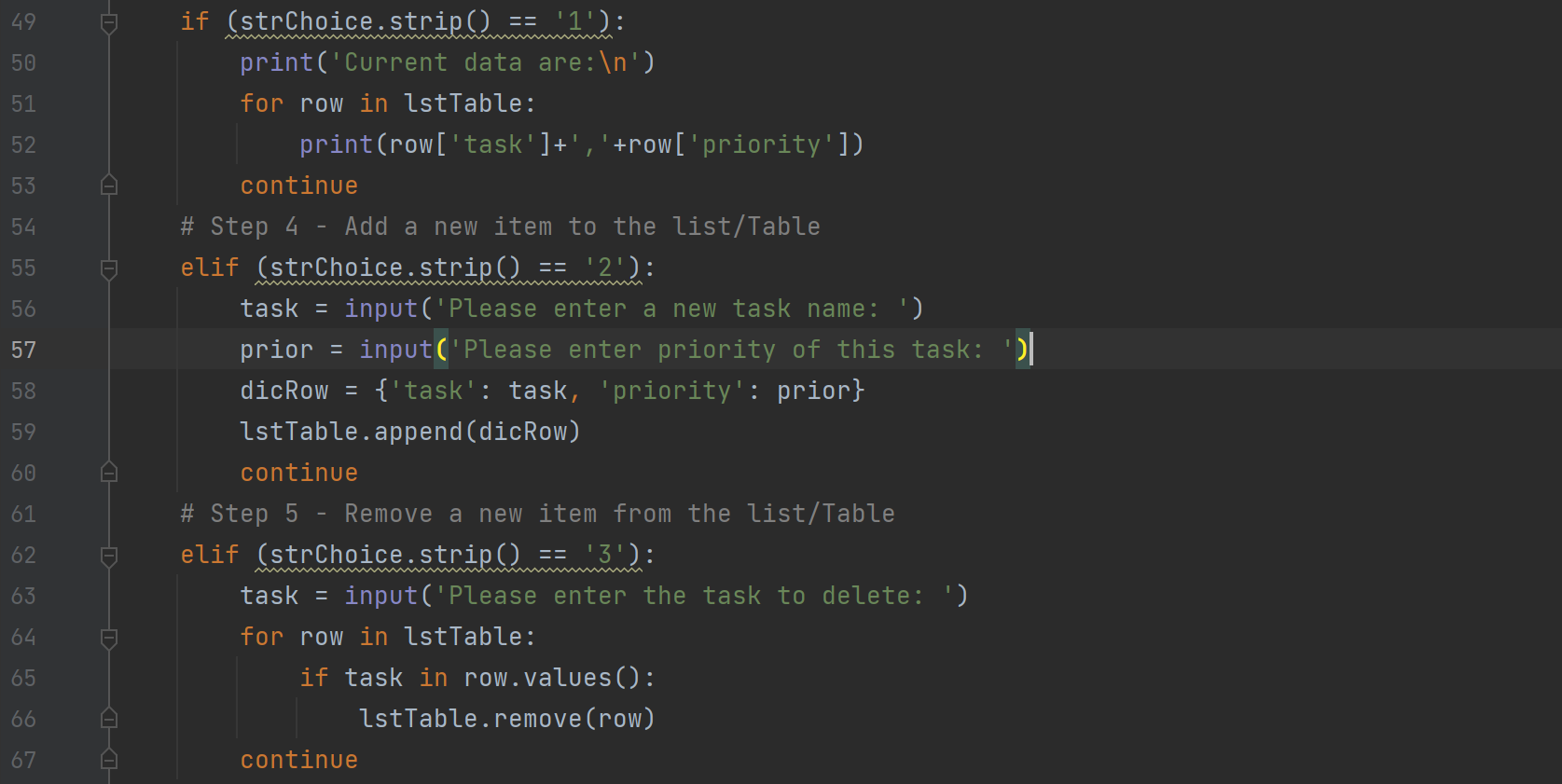


Figure Showing, Adding, and Removing Data in Memory (the List Table)

Figure 7 shows the implementation of options 4-5 (writing data to file and exiting the program). I also added an option with an *else* statement to capture any user input outside of 1-5.

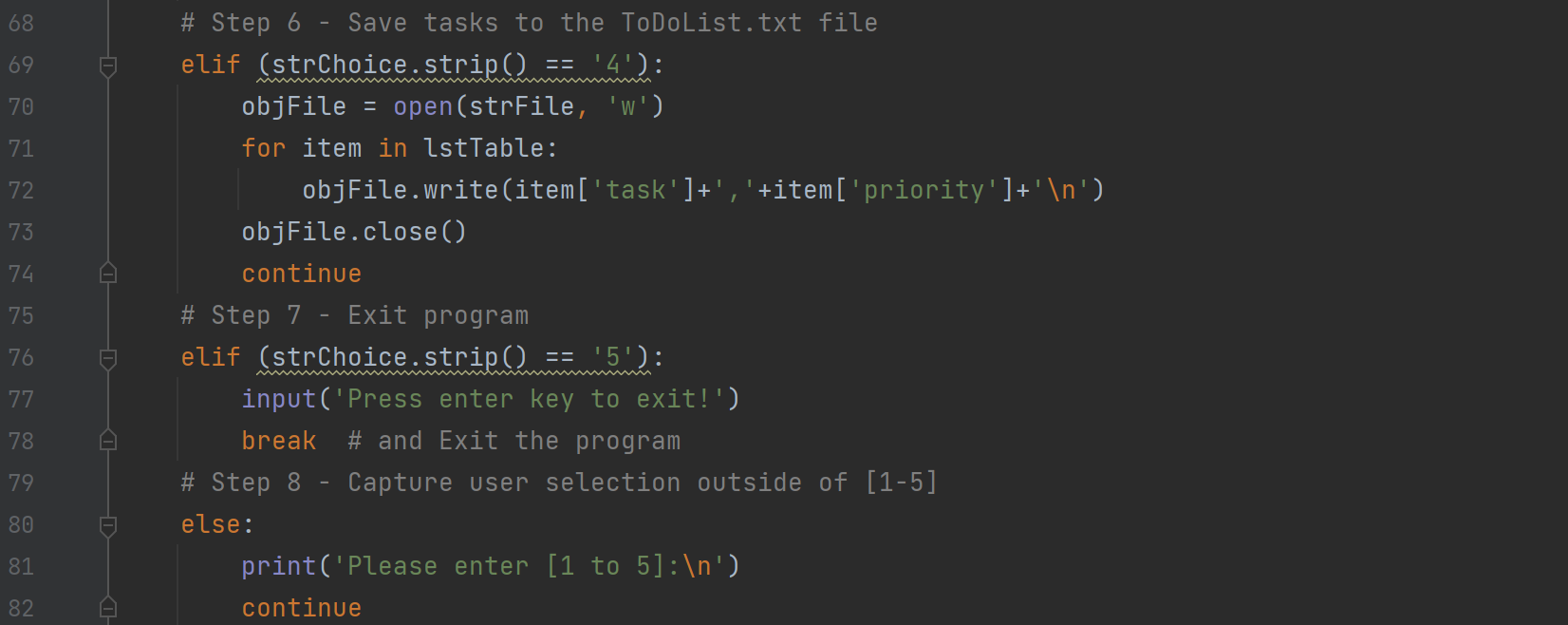


Figure Writing Data into File and Exiting the Program

## Executing the Script in PyCharm

I ran the script by right clicking then picking “run ‘Assignment05’” from within the PyCharm script window (hotkey: “Ctrl+Shift+F10”). The initial data content in the “ToDoList.txt” is as shown in Figure 8.

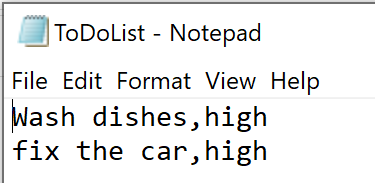


Figure Initial items in ToDoList.txt

In this test I wrote the code to display existing data, then add one item of task {‘task’: ’clean floor’, ‘priority’: ‘medium’) and remove {‘task’: ’fix the car’, ‘priority’: ‘high’}. Finally, I saved the data back to the file and then exited the program. As shown in Figure 9 is a screenshot of the run steps in PyCharm.

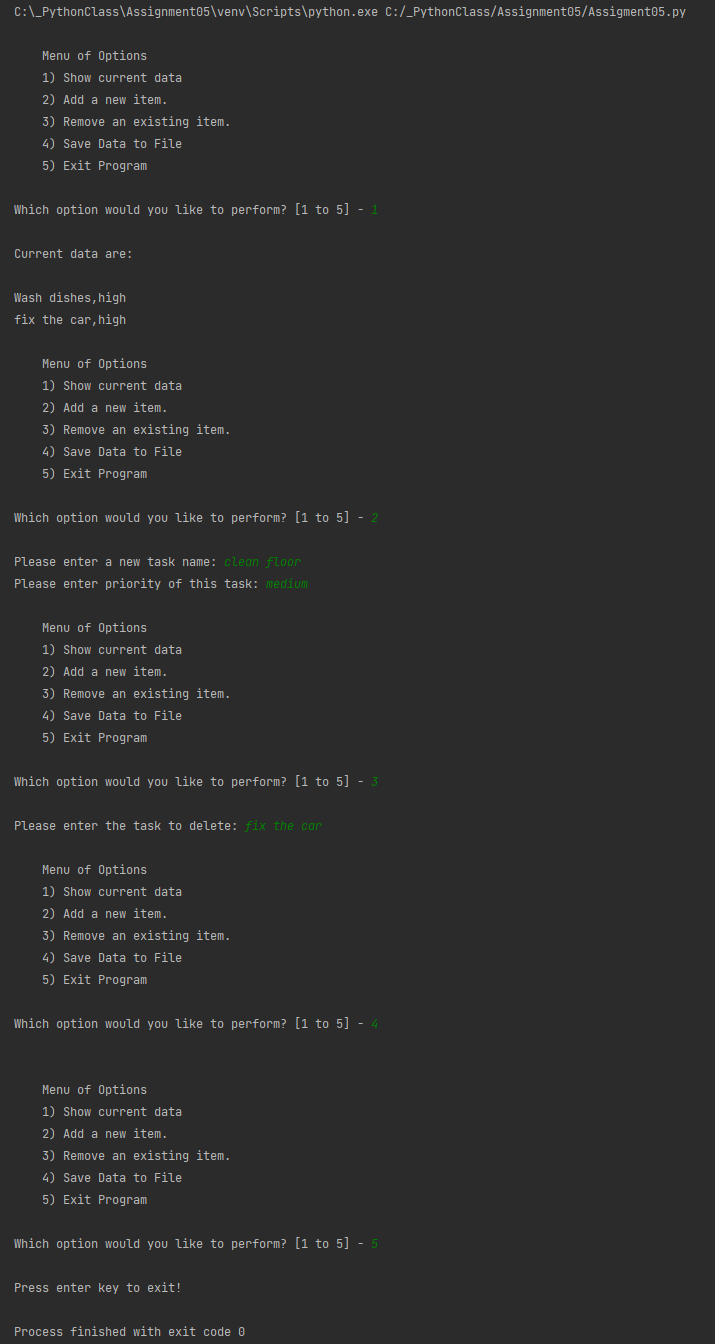


Figure Test Run Results in PyCharm

## Checking the Output File

To confirm that the script worked as intended, the final data in “ToDoList.txt” after the test run ended is shown in Figure 10.

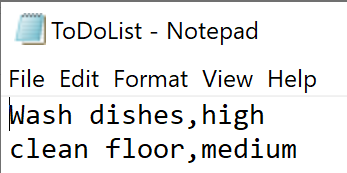


Figure Final items in ToDoList.txt after the Run in PyCharm

## Executing the Script in Windows Command Prompt

Before the run in the Windows command prompt, the initial content of the “ToDoList.txt” was set the same as in Figure 10. I followed the same steps in the run in PyCharm, except that I added one item of task and priority {‘task’: ‘grocery shopping’, ‘priority’: ‘low’} and remove {‘task’: ‘Wash dishes’, ‘priority’: ‘high’}. I went to “C:\\_PythonClass\Assignment05\” and run Assignment05.py. As shown in Figure 11, I added and removed an item as described above, the script ran exactly the same as in PyCharm in Figure 9.

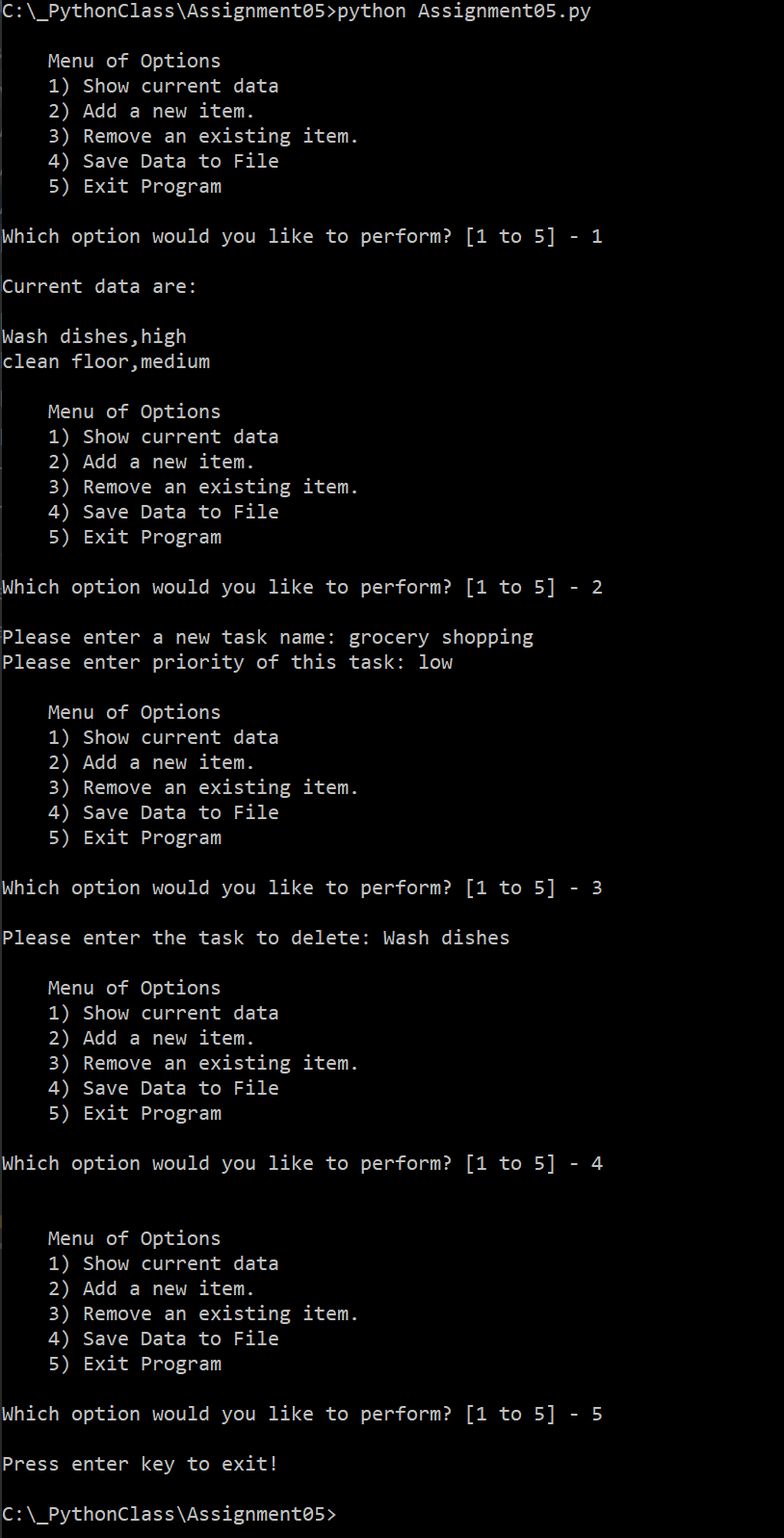


Figure Run Results in Windows Command Prompt

## Checking the Output File Again

I went ahead to check the data in “ToDoList.txt” in the working folder. Now the new data in the file are verified and shown in Figure 12.

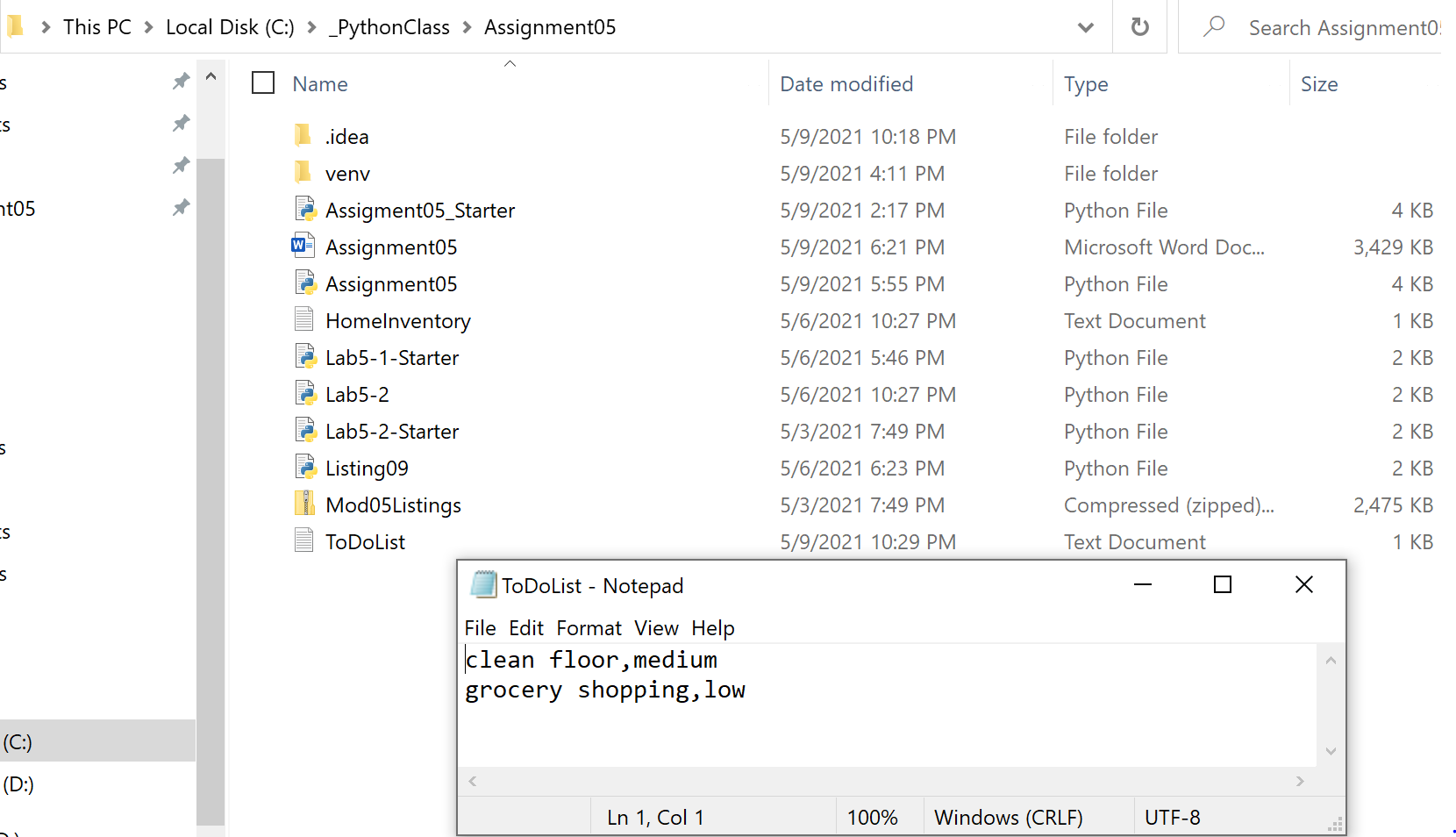


Figure Data in ToDoList.txt After the Script was Executed in Windows Command

# Summary

In module 5, I have learned about lists and dictionaries with various methods, and how they can be used to store and update a collection of data. I also learned how to load the data from a text file to a table, operate on these data, and then save them to the file. The learning is demonstrated by the script submitted.

The collaborative environment of programming makes it crucial to standardize the practice of coding (such as formatting) in order to make pieces of code fit well between multiple developers who work in the same project. The use of functions enables more organized and flexible programs. The *try-except* block allows developers to write user-friendly codes for error handling purposes. The source-control applications such as GitHub are powerful tools to allow developers to work in a collaborative way.