Katie London
11/14/2021
Foundations of Programming: Python
Assignment05
https://github.com/KatieLondon/IntroToProg-Python

Creating a To-Do List

Introduction

The purpose of this assignment is to create a to-do list to manage a user's tasks and priorities. The user is able to view and modify their list by reading from and writing to a text file. Instead of reading the data into a tuple like we did in the previous assignment, this week, we are reading the data into a dictionary object. One of the big differences between tuples and dictionaries is that elements stored within them are accessed with a key instead of with numerical indices. Finally, following the separation of concerns principle, in this script, we attempt to separate the code into three distinct sections: data, processing and presentation. However, the processing and presentation steps turn out to be too intertwined and cannot effectively be separated.

To-Do List Script

Data

The data section is the beginning of the script where we declare variables and constants. For the to-do list script, we are declaring multiple string variables, a dictionary object to hold a row of data and a list object to hold multiple dictionary elements. Declaring variables in Python is purely optional but helps with the readability of the code.

Processing and Presentation

The processing section comprises the bulk of the code and is where we perform tasks on the data to accomplish our objective. The first step is to read data from a text file. We store each row of data in a dictionary and each dictionary object is stored within a list to create a table of data. When reading text data from a file we need to use the strip command to remove any hidden carriage returns.

Once we have the data loaded the next step is to give the user a menu of options to choose from. This involves printing the menu to the screen and capturing the user's input. The user can choose from viewing the current data, adding new items, removing items, saving the data or exiting the program. Using a while loop with the conditional expression set to true, we continue to give the user a menu of options to choose from until they choose to exit the program. While these steps fall into the presentation category, they cannot be separated from the processing steps.

Once the user has chosen a menu option, we use conditional statements with the if-elif construct to compare the string input from the user to each menu option. Within each if-elif case, there are additional steps. To show the current data, we use a for loop construct to print each row of the table to the screen. Since each row is stored as a dictionary object, we need to use the keys "Task" and "Priorities" to access the corresponding elements. Here, again, we use the strip function to remove hidden carriage returns. To add a new to-do list item we use input from the user to create a new row stored as a dictionary object and append it to the list. To remove an item from the to-do list, we use a for loop construct to search the table for the item matching the one specified by the user. Once we find the matching item, we use the remove function to delete the row. To save the data, we open a text document and loop through each row of data stored in the list and write the data to the file. Finally, the last option is to exit the program which is accomplished with a break command.

Running Python Script

I successfully ran the ToDoList script in PyCharm and tested each of the 5 menu choices, as shown in figure 1, 2 and 3 below. In figure 1, I selected option 1 to display the current data. Next, I choose option 2 to add the item "water plants" to the table with "priority 3".

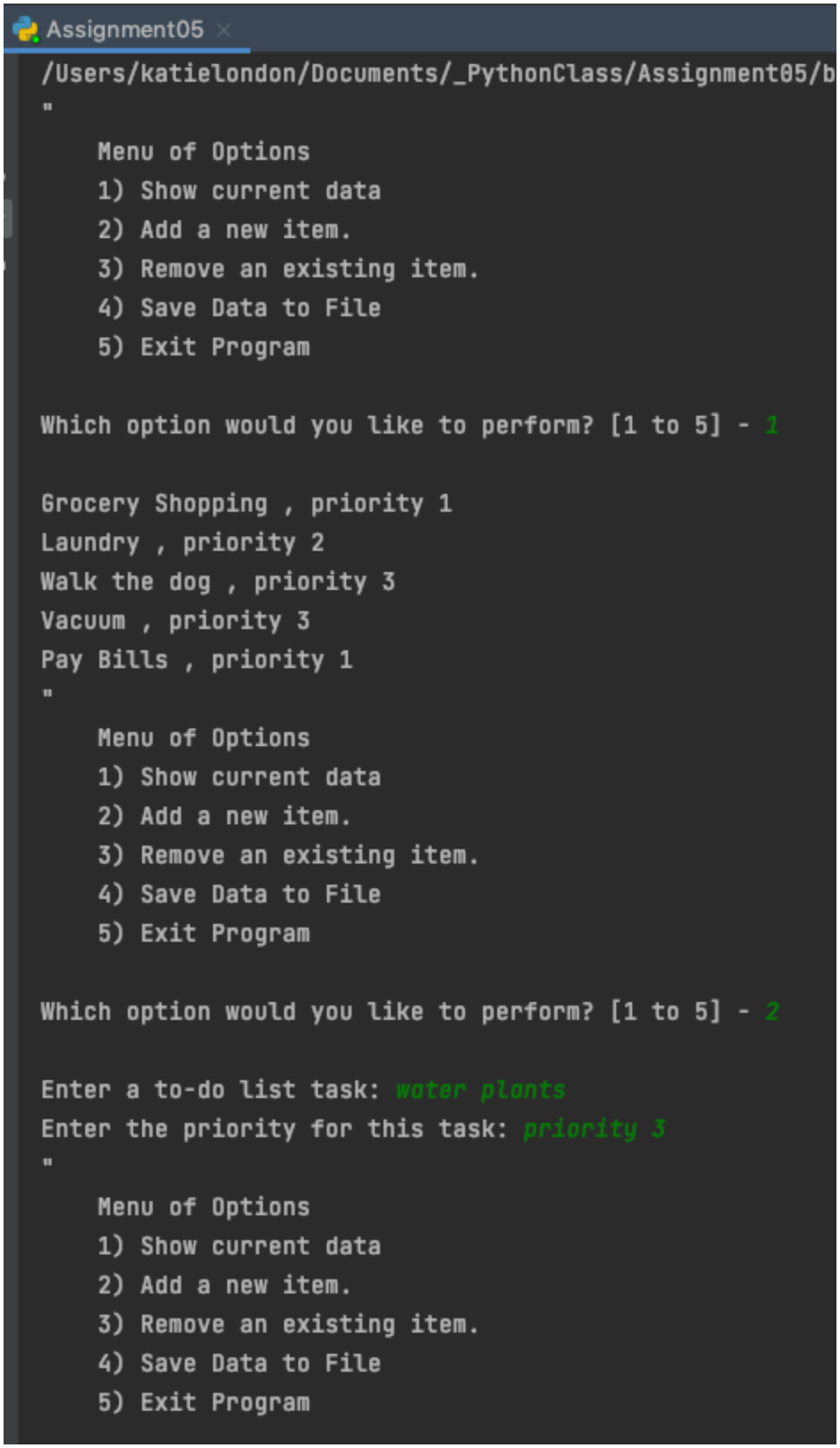

Figure 1: Screenshot of Menu option choice 1 and 2 in PyCharm

In figure 2 below, I confirmed that “water plants” with “priority 3” was added to the table by choosing option 1 again to display the data. Next, to test menu option 3, remove an item, I entered “laundry” and showed the data before and after to confirm that the item

specified was removed.

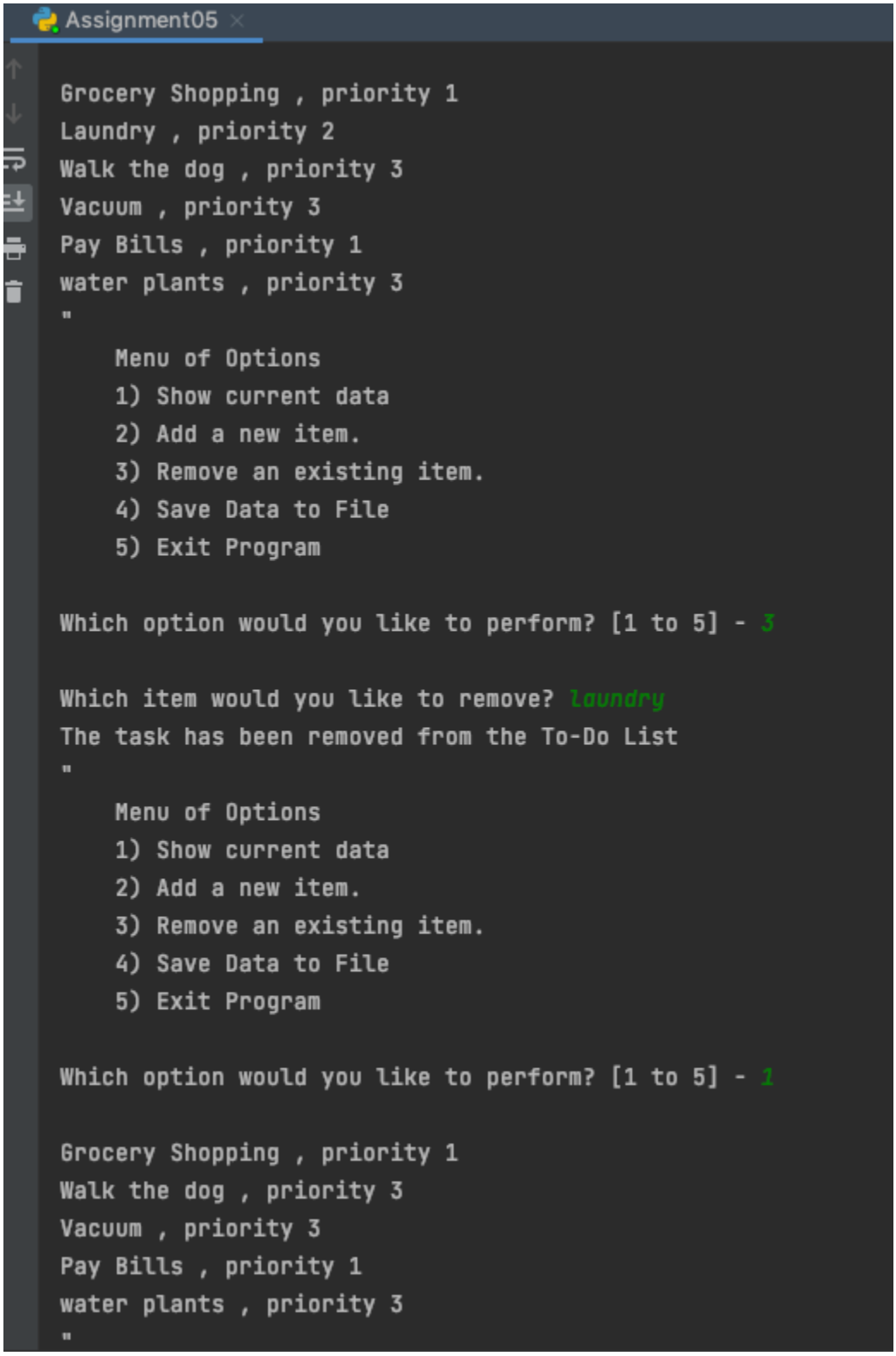

Figure 2: Screenshot of Menu option choice 3 working in PyCharm

In figure 3, I choose the option to save the data and then exit the program

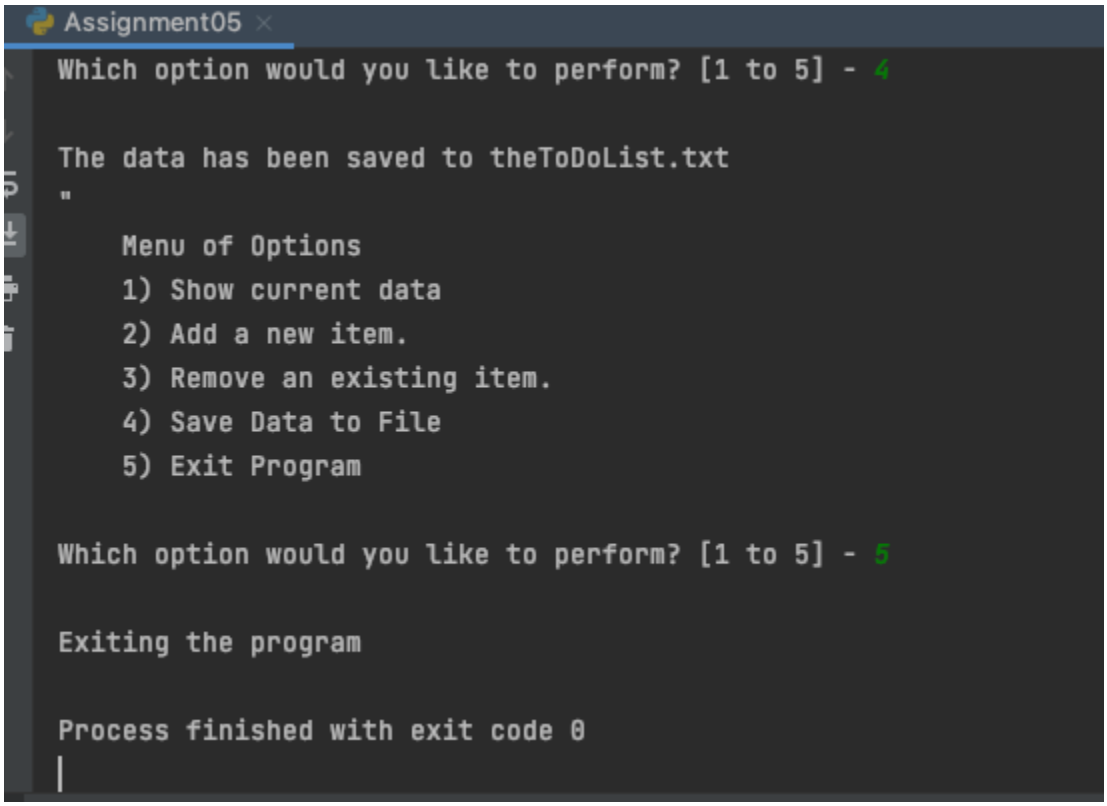

Figure 3: Screenshot of Menu option choice 4 and 5 working in PyCharm

After confirming that the script and each menu option is working correctly, I tested the script by running it on the terminal. This time, I only ran each menu option once.

```
Last login: Fri Nov  5 13:47:42 on ttys000
(base) Katies-MacBook:~ katielondon$ cd /Documents/_PythonClass/Assignment05
-bash: cd: /Documents/_PythonClass/Assignment05: No such file or directory
(base) Katies-MacBook:~ katielondon$ cd Documents/_PythonClass/Assignment05
(base) Katies-MacBook:Assignment05 katielondon$ ls
Assignment05.py ToDoList.py     ToDoList.txt    bin             lib
(base) Katies-MacBook:Assignment05 katielondon$ python ToDoList.py
"
    Menu of Options
    1) Show current data
    2) Add a new item.
    3) Remove an existing item.
    4) Save Data to File
    5) Exit Program

Which option would you like to perform? [1 to 5] - 1

Grocery Shopping , priority 1
Laundry , priority 2
Walk the dog , priority 3
Vacuum , priority 3
Pay Bills , priority 1
"
    Menu of Options
    1) Show current data
    2) Add a new item.
    3) Remove an existing item.
    4) Save Data to File
    5) Exit Program

Which option would you like to perform? [1 to 5] - 2

Enter a to-do list task: water plants
Enter the priority for this task: 3
"
    Menu of Options
    1) Show current data
    2) Add a new item.
    3) Remove an existing item.
    4) Save Data to File
    5) Exit Program

Which option would you like to perform? [1 to 5] - 3

Which item would you like to remove? vacuum
The task has been removed from the To-Do List
"
    Menu of Options
    1) Show current data
    2) Add a new item.
    3) Remove an existing item.
    4) Save Data to File
    5) Exit Program

Which option would you like to perform? [1 to 5] - 4

The data has been saved to theToDoList.txt
"
    Menu of Options
    1) Show current data
    2) Add a new item.
    3) Remove an existing item.
    4) Save Data to File
    5) Exit Program

Which option would you like to perform? [1 to 5] - 5

Exiting the program
```

Figure 4: Screenshot of the ToDoList script successfully running in Terminal

Summary

In this assignment, we wrote a script to manage a to-do list. The script reads data from a text file and loads it into a list object where each list element contains a dictionary object used to represent one row of the table. The user is able to choose from the following menu options: show data, add new item, remove an existing item, save data and exit. To execute the menu option chosen, we used a conditional if-elif construct. Within option 1, option 3 and option 4, we used a for loop to accomplish the task. Finally, I ran the script in both PyCharm and the Terminal to verify that the script worked.