Jessica Wray
22 November 2021
Foundations of Programming, Python
Assignment 06
GitHub URL: https://github.com/jkwray/IntroToProg-Python-Mod06

Working with Classes and Functions

Introduction

This assignment introduced several new concepts as well as gave us some in depth experience of what it is like to jump into someone else's code and build on top of it. While the tasks the script performs are almost identical to the script from last week's assignment, the actual program itself is very different. It's another example that highlights how programming the same task can be done in a variety of different ways.

Creating the Script

To begin, I simply changed the script header by adding to the change log and then read through the code to understand what had already been done and what was left to do. It was well documented and easy to follow so I started at the beginning of the menu options and once I got one of them to work, I moved onto the next. Having the classes and functions pre-defined made the task much more manageable than having to start from scratch.

For the most part I was able to re-purpose code I wrote for last week's assignment and simply change the names of variables, arguments, and parameters to match what was in the starter file for this week. I started with Menu Option 1 "Add a new task" and identified where the code was incomplete. I copied a lot of my code from last week and added it to the related functions in the Processor class and the IO class. Once I got that working, I did the same thing with Menu Options 2 and 3 as Option 4 was already completed in the starter file. The script is quite long so I only included a screen shot below of the functions being called. To see the full code please refer to the included .py file in the assignment.

```python
# Step 1 - When the program starts, Load data from ToDoFile.txt.
Processor.read_data_from_file(file_name_str, table_lst)  # read file data

# Step 2 - Display a menu of choices to the user
while (True):
    # Step 3 Show current data
    IO.output_current_tasks_in_list(table_lst)  # Show current data in the list/table
    IO.output_menu_tasks()  # Shows menu
    choice_str = IO.input_menu_choice()  # Get menu option

    # Step 4 - Process user's menu choice
    if choice_str.strip() == '1':  # Add a new Task
        # call function to get user input
        task_str, priority_str = IO.input_new_task_and_priority()
        # call function to save the new task to the current list
        Processor.add_data_to_list(task_str.strip(), priority_str.strip(), table_lst)
        continue  # to show the menu

    elif choice_str == '2':  # Remove an existing Task
        # get task to be removed
        task_str = IO.input_task_to_remove()
        print(task_str)
        # call function to remove the task from the list
        Processor.remove_data_from_list(task_str, table_lst)
        continue  # to show the menu

    elif choice_str == '3':  # Save Data to File
        # call function to save the current list to a file
        Processor.write_data_to_file(file_name_str, table_lst)
        continue  # to show the menu

    elif choice_str == '4':  # Exit Program
        print("Goodbye!")
        break  # and Exit
```

Figure 1: Assignment06.py

Running the Script

To run the script in PyCharm I right clicked and chose “Run” from the dropdown menu. I then ran through each menu item in order. I first verified that the list of tasks displayed in the output where the same as the ones in the source file.

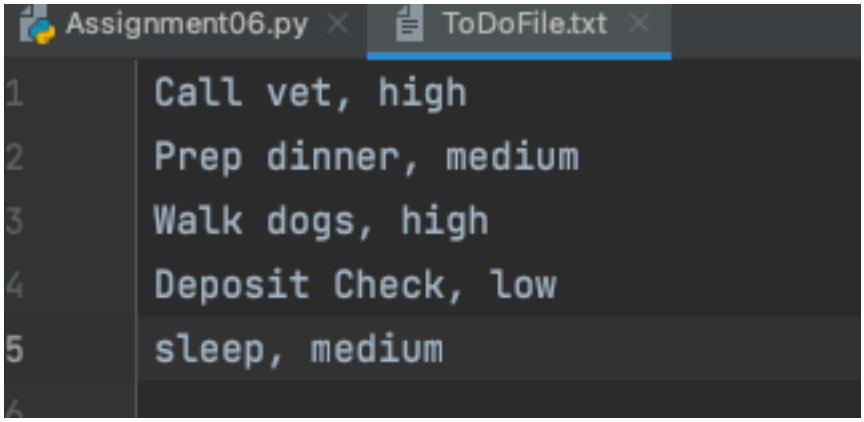

Figure 2: Original task list in .txt file

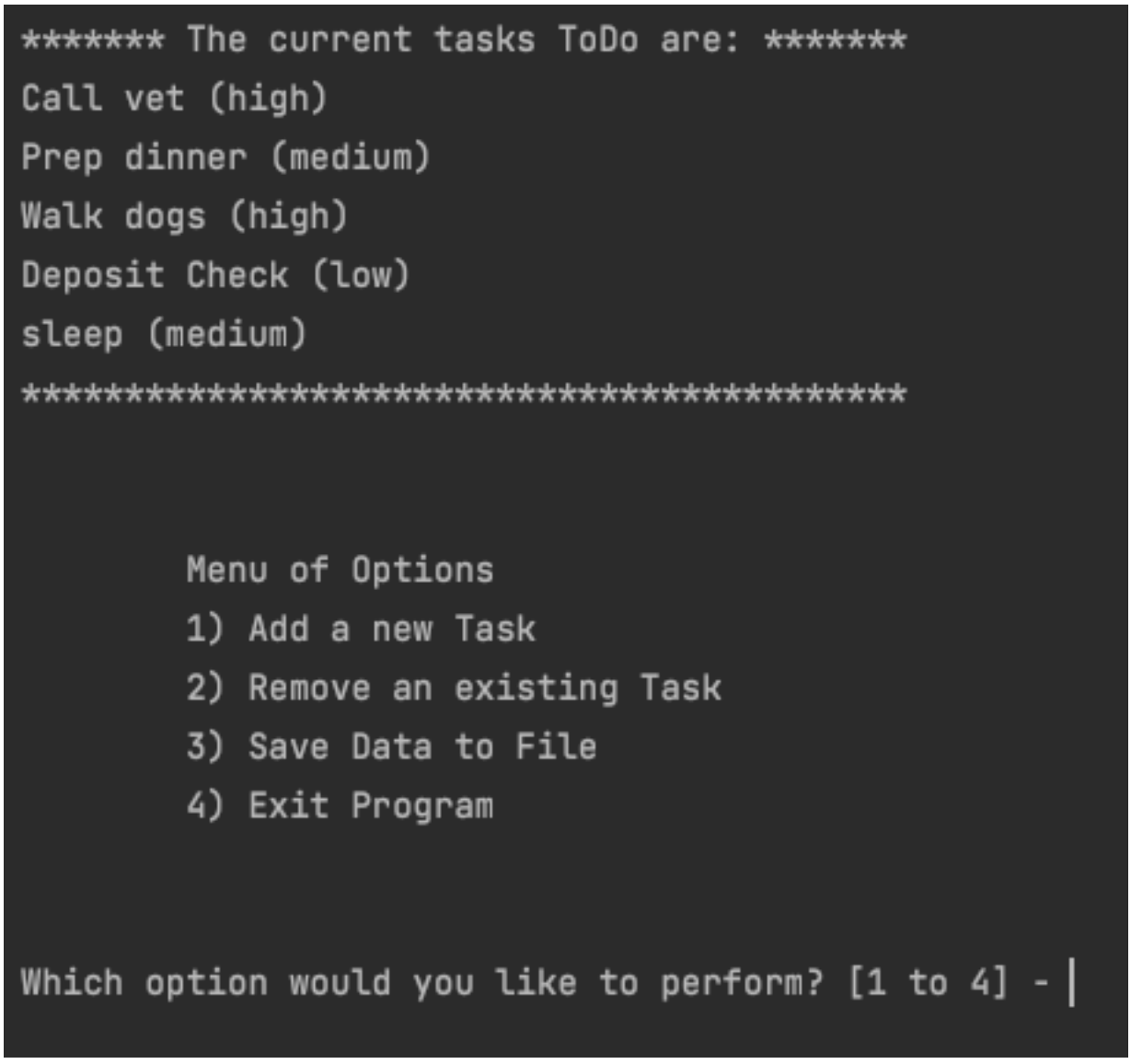

Figure 3: Beginning task list in PyCharm

As you can see from the Figure 2 and Figure 3, the lists are identical which shows the script is pulling the data from the right source.

I chose option one that allows me to add a new task and priority to the existing list to begin. I input “workout” with a priority level of “high”. As you can see in Figure 4 below the script took those inputs and appended it the existing list.

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

Please enter task: Workout
Please assign priority level ["high", "medium", or "low"]: high
Your task has been added!

******* The current tasks ToDo are: *******
Call vet (high)
Prep dinner (medium)
Walk dogs (high)
Deposit Check (low)
sleep (medium)
Workout (high)
*******************************************
```

Figure 4: Adding a task to the list in PyCharm

Next, I chose option 2 which allows the user to remove a task from the list. I chose to remove “sleep” and then verified that the script executed as expected. You can see the output below.

```
Which option would you like to perform? [1 to 4] - 2


What task would you like to remove: sleep
sleep

"sleep" has been removed from your task list.

******* The current tasks ToDo are: *******
Call vet (high)
Prep dinner (medium)
Walk dogs (high)
Deposit Check (low)
Workout (high)
****************************************
```

Figure 5: Successfully deleting a task to the list in PyCharm

I also wanted to test my code that handled input that wasn't on the current list, so I chose option 2 again and typed "run". You can see the results in the output image below. Once the "was NOT found" message is displayed, the script asks the user to select an option from the menu (not shown in Figure 6)

```
Which option would you like to perform? [1 to 4] - 2


What task would you like to remove: run

"run" was NOT found on your task list

******* The current tasks ToDo are: *******
Call vet (high)
Prep dinner (medium)
Walk dogs (high)
Deposit Check (low)
sleep (medium)
****************************************
```

Figure 6: Unsuccessfully deleting a task to the list in PyCharm

The final test was making sure the script saved the update list to the original text file. I chose option 3 and then verified that the file was indeed updated with the changes I had made in the previous steps.

```
Which option would you like to perform? [1 to 4] - 3

Data has been saved!

******* The current tasks ToDo are: *******
Call vet (high)
Prep dinner (medium)
Walk dogs (high)
Deposit Check (low)
Workout (high)
*******************************************
```

Figure 7: Saving data to file in PyCharm

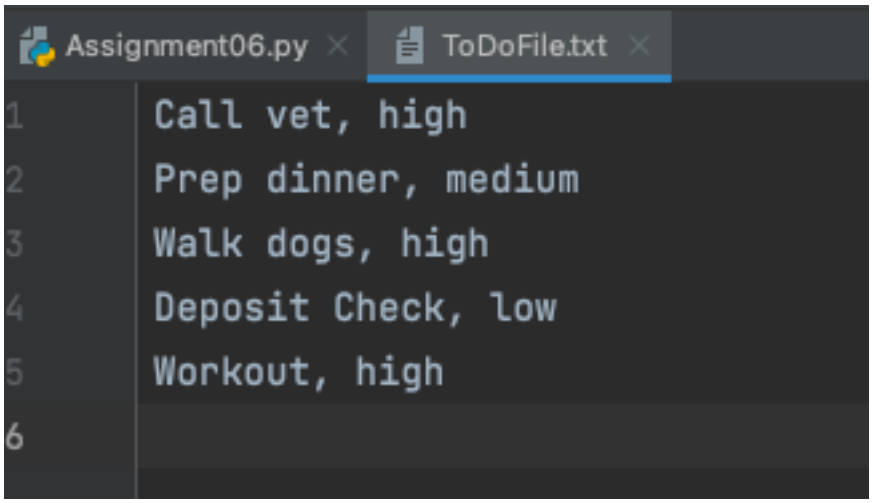

Figure 8: Update text file

Finally, I ran through all the steps again in a terminal window to verify that the script executed as expected and it did as you can see by the figure below.

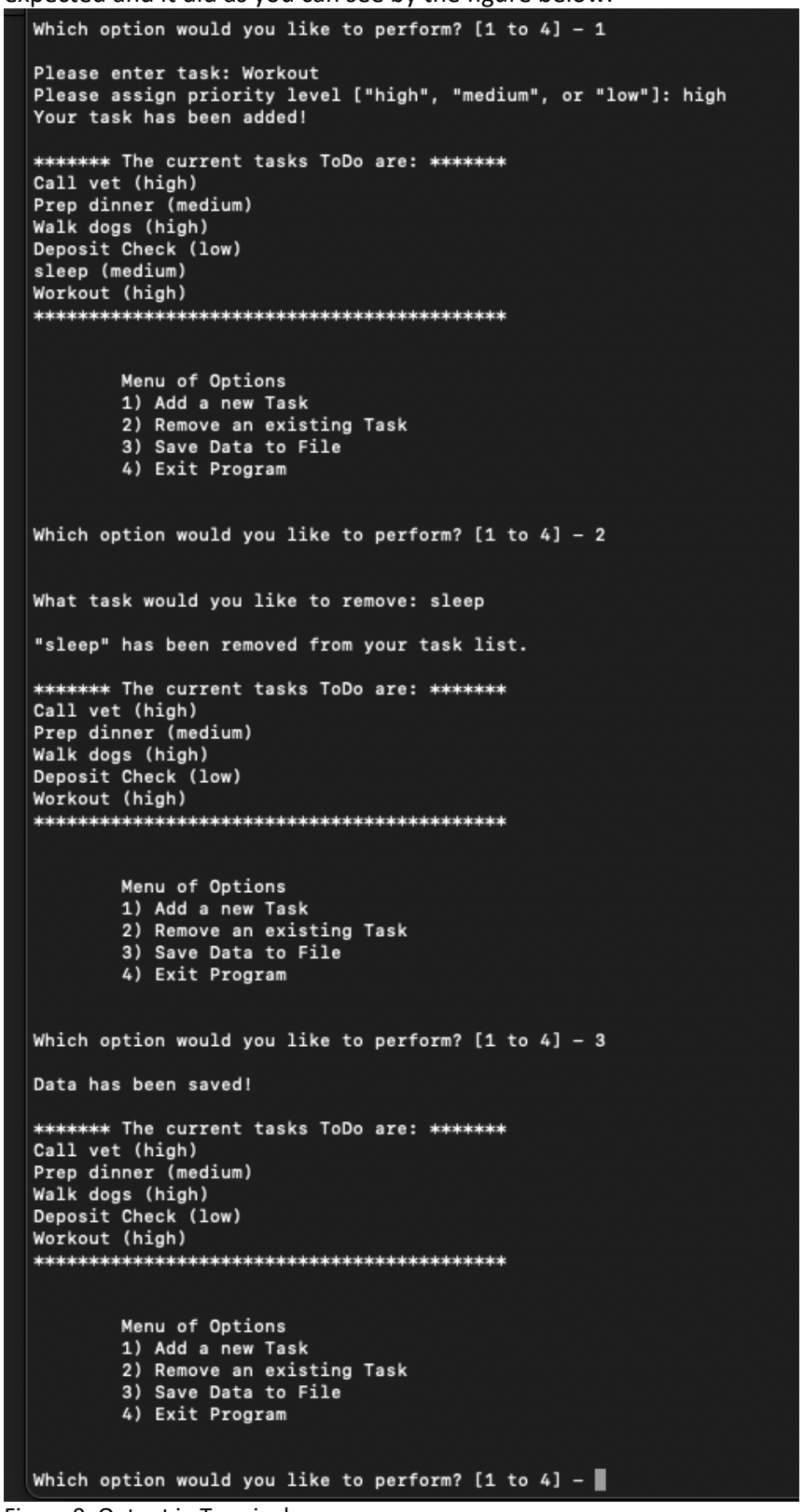

Figure 9: Output in Terminal

Conclusion

My favorite part of this assignment is how clean the code is now that we are using classes and functions. Because I was a software engineer in my early career, the previous weeks have been challenging because the code we were asked to write seemed very messy and unorganized. It's refreshing to be able to now move past that and write clean, well-documented, and organized code. I look forward to building on these concepts next week.