Tim Randall
May 24th, 2023
Intro to Programming (Python) – IT FDN 110A Spring 2023
Assignment 06
https://github.com/trandallUW/IntroToProg-Python-Mod06

To Do List “Plus”

Introduction

The purpose of this assignment was to develop an understanding of how to take another person’s code and organize it using the “separation of concerns” – which is the framework of separating code by processing data vs. inputs / outputs to the user. The assignment also taught me some of the basics of functions and classes – which are ways to organize statements within the script. The assignment also had me learn how to create a basic website on Github. In general, this assignment was very difficult.

Overall Process

The first step to tackling this assignment was to download the starter code that was provided by Randall Root. My next step was to modify my header to input my notes into the change log. After understanding how the basics of functions and classes worked, I started to read the overall code.

At first, I read the code starting at the top and working down. The declared variables were helpful. But as I read further, I quickly became confused as I didn’t understand the overall goal of what the code was doing (starting with the functions first was not working for me). Eventually, I read the “main body” of the script and started to realize how the code was broken up into several mini-steps (functions). The functions were “bite-size” statements – which was different to the previous assignments. This gave me a starting point to understand how to break up my code and populate the missing functions & logic.

For the script, I decided to use the “Assignment05_Answer.py” file that was written by someone else. My reasoning was that my code for Assignment 05 didn’t work as well as I wanted it to, thus starting from a code written by someone else would at least guarantee me that I was starting from something that was working as intended.

Starting to Code – Input / Output

I decided to start with the input / output functions first as this was mainly just learning how to display and get information from the user. For this assignment, I tried to focus less on formatting the menu options and formatting the outputs to make them “pretty” – my goal was not to make the perfect script. I really just wanted to get it to work and cared less about line breaks / indentations of text.

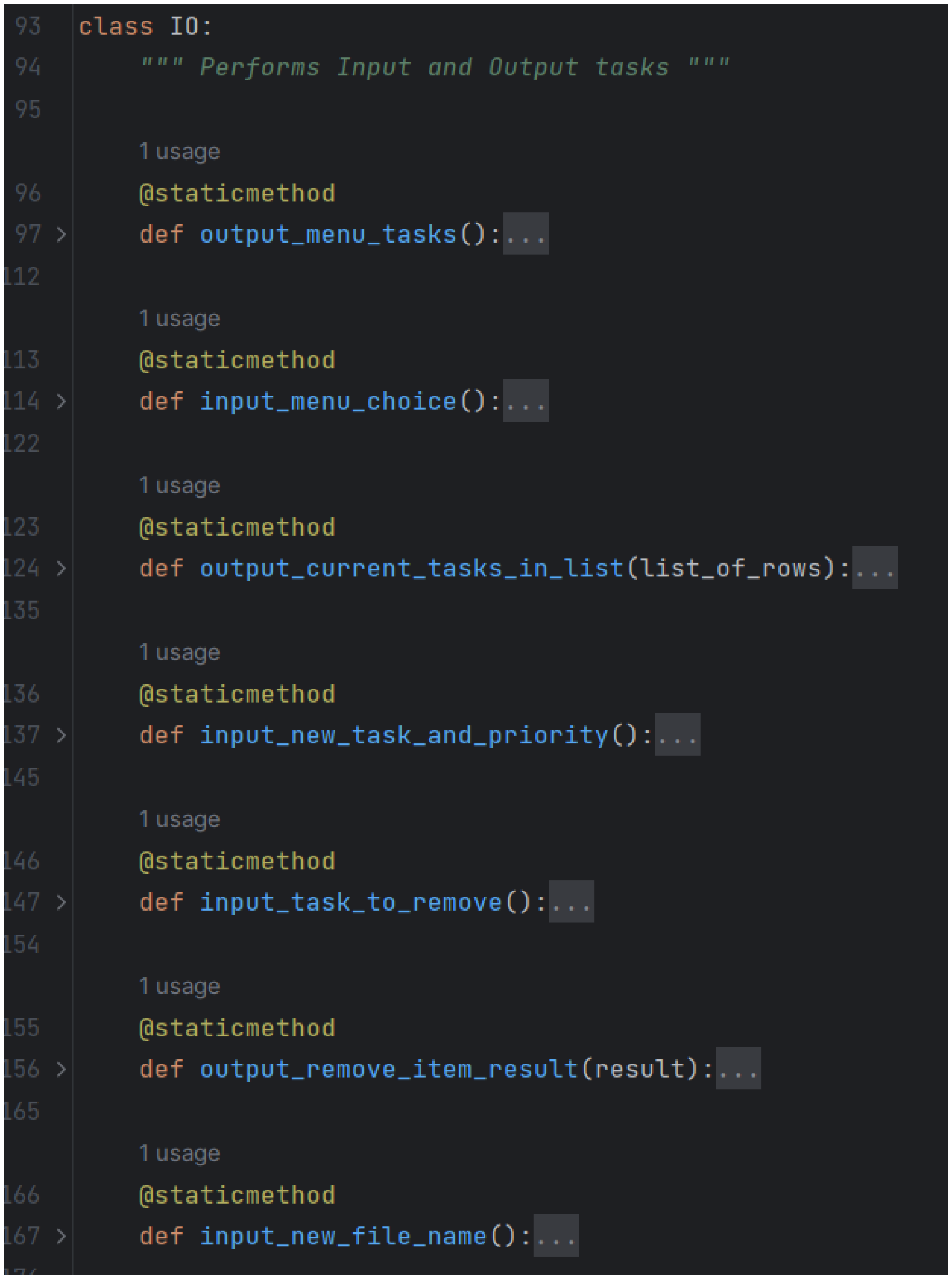
However, by using functions, I realized that the formatting of user outputs and inputs could quickly be upgraded (if I wanted to do this at a later point). The code could just be changed in one area with minimal effort. Great lesson learned for future use!

The only additional functions that I added to the code were:

1. “output_remove_item_result” – a function that provided a result if the removal was successful

2. "input_new_file_name" – a function that asked the user for a new file name to load, if desired.

Figure 1 shows my Input / Output functions:

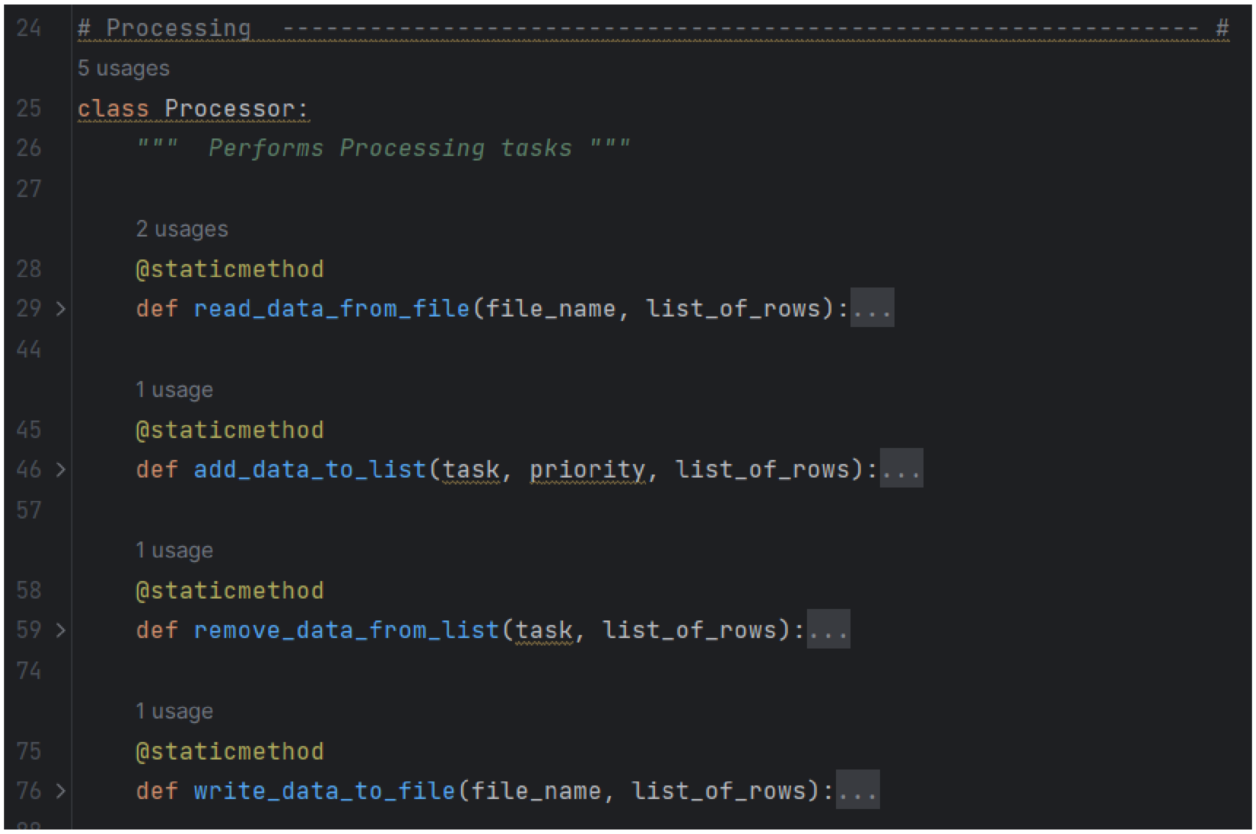


```
class IO:
    """ Performs Input and Output tasks """

    1 usage
    @staticmethod
    def output_menu_tasks():...

    1 usage
    @staticmethod
    def input_menu_choice():...

    1 usage
    @staticmethod
    def output_current_tasks_in_list(list_of_rows):...

    1 usage
    @staticmethod
    def input_new_task_and_priority():...

    1 usage
    @staticmethod
    def input_task_to_remove():...

    1 usage
    @staticmethod
    def output_remove_item_result(result):...

    1 usage
    @staticmethod
    def input_new_file_name():...
```

Figure 1 - Input/Output

Moving on to Processing Data

After getting menu items and input/outputs to work, I then started to focus on how to process the data. Most of the functions were easy. The "remove data from list" function was really difficult for me to grasp how to get it to work. It took me several sessions and much research / looking at examples to finally get something that worked. The challenge that I was having was understanding the local variables and naming conventions. Then translating some of the starter code and piecing together previous code from previous assignments. Eventually – I was able to get it to work. Figure 2 shows my code.



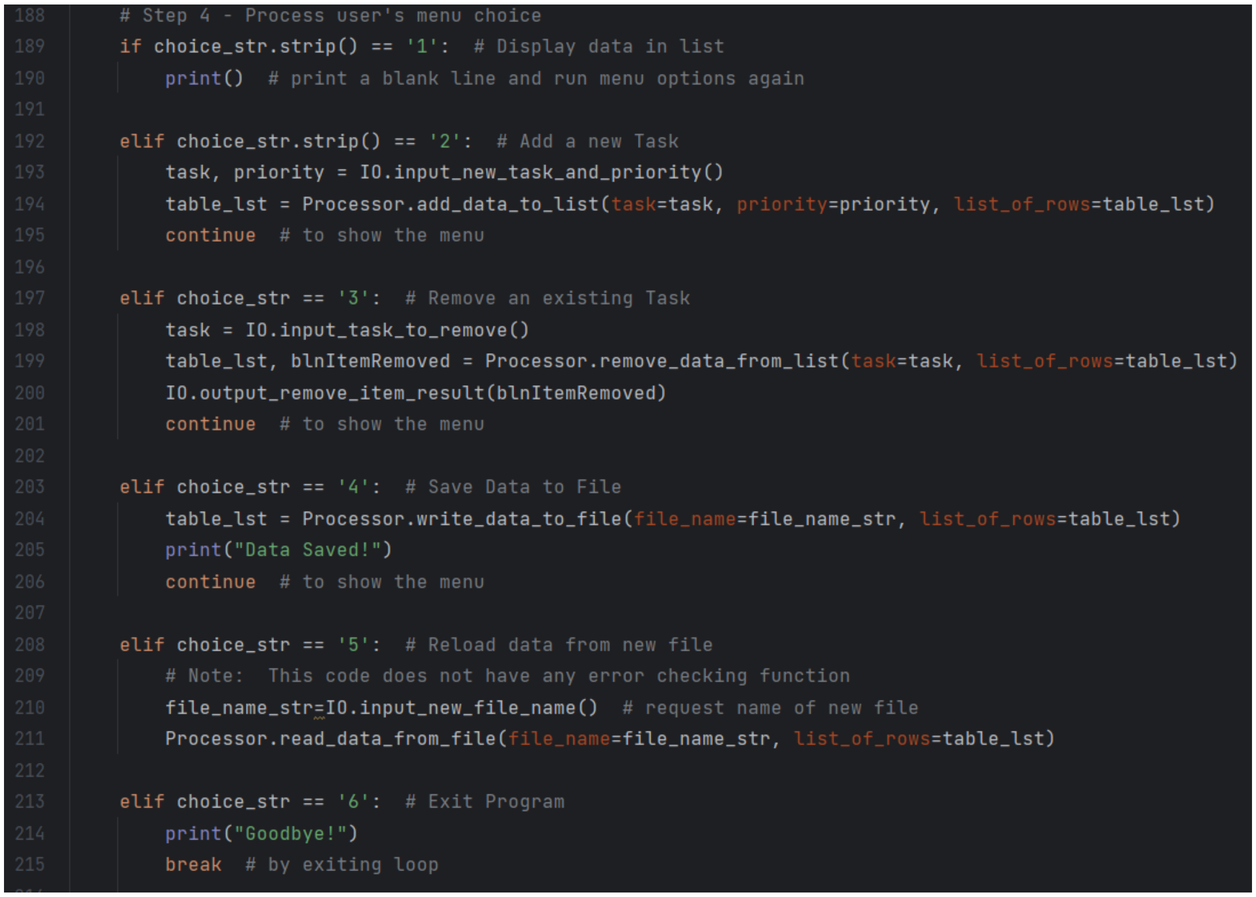
```
# Processing -------------------------------------------------------------- #
5 usages
class Processor:
    """  Performs Processing tasks """

    2 usages
    @staticmethod
    def read_data_from_file(file_name, list_of_rows):...

    1 usage
    @staticmethod
    def add_data_to_list(task, priority, list_of_rows):...

    1 usage
    @staticmethod
    def remove_data_from_list(task, list_of_rows):...

    1 usage
    @staticmethod
    def write_data_to_file(file_name, list_of_rows):...
```

Figure 2 - Processing Data

Main Body

The main body of the script was mainly untouched from the starter code that was provided by Randall Root. My only revisions were to re-order the menu numbers just based on the additional functions that I added. The "read data from file" code is not perfect – I realize that if the user enters a file name that doesn't exist, the code will give errors. I didn't try to tackle any additional logic checks – this will be for future assignments. Also – I was not a fan of how the script prints the tasks after every user input. If I had more time, I would probably try to change the logic on that (display the data only if the user asks). This would be relatively straightforward but not something that I needed to do this week 😊

Figure 3 shows the main body of my script.

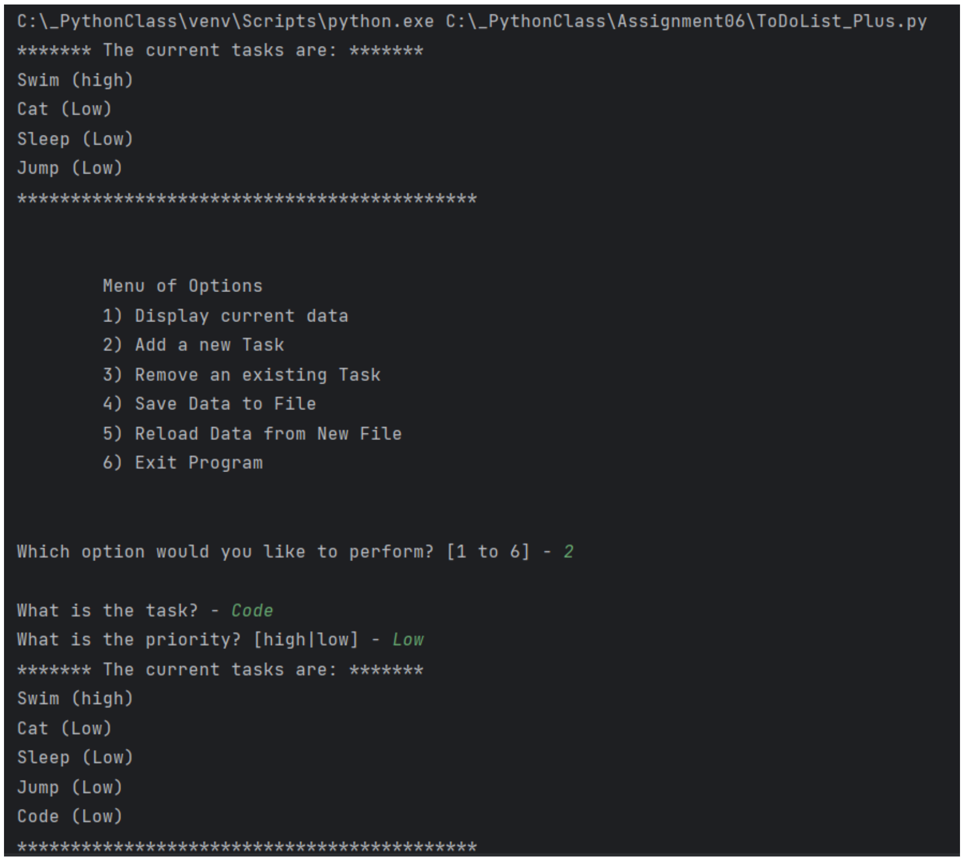


```python
    # Step 4 - Process user's menu choice
    if choice_str.strip() == '1':  # Display data in list
        print()  # print a blank line and run menu options again

    elif choice_str.strip() == '2':  # Add a new Task
        task, priority = IO.input_new_task_and_priority()
        table_lst = Processor.add_data_to_list(task=task, priority=priority, list_of_rows=table_lst)
        continue  # to show the menu

    elif choice_str == '3':  # Remove an existing Task
        task = IO.input_task_to_remove()
        table_lst, blnItemRemoved = Processor.remove_data_from_list(task=task, list_of_rows=table_lst)
        IO.output_remove_item_result(blnItemRemoved)
        continue  # to show the menu

    elif choice_str == '4':  # Save Data to File
        table_lst = Processor.write_data_to_file(file_name=file_name_str, list_of_rows=table_lst)
        print("Data Saved!")
        continue  # to show the menu

    elif choice_str == '5':  # Reload data from new file
        # Note:  This code does not have any error checking function
        file_name_str=IO.input_new_file_name()  # request name of new file
        Processor.read_data_from_file(file_name=file_name_str, list_of_rows=table_lst)

    elif choice_str == '6':  # Exit Program
        print("Goodbye!")
        break  # by exiting loop
```

Figure 3 - Main Body

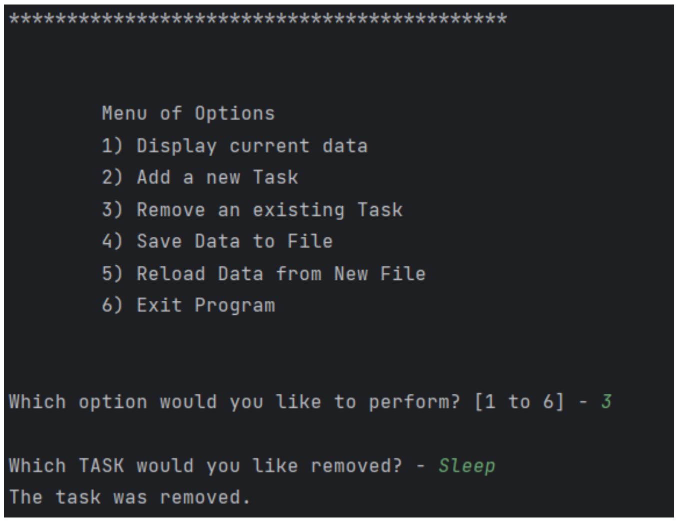
Final Code

The code works. Figure 4, 5, 6, and 7 show Pycharm outputs. Figure 8 shows command prompt outputs for a few functions (all other functions the same).



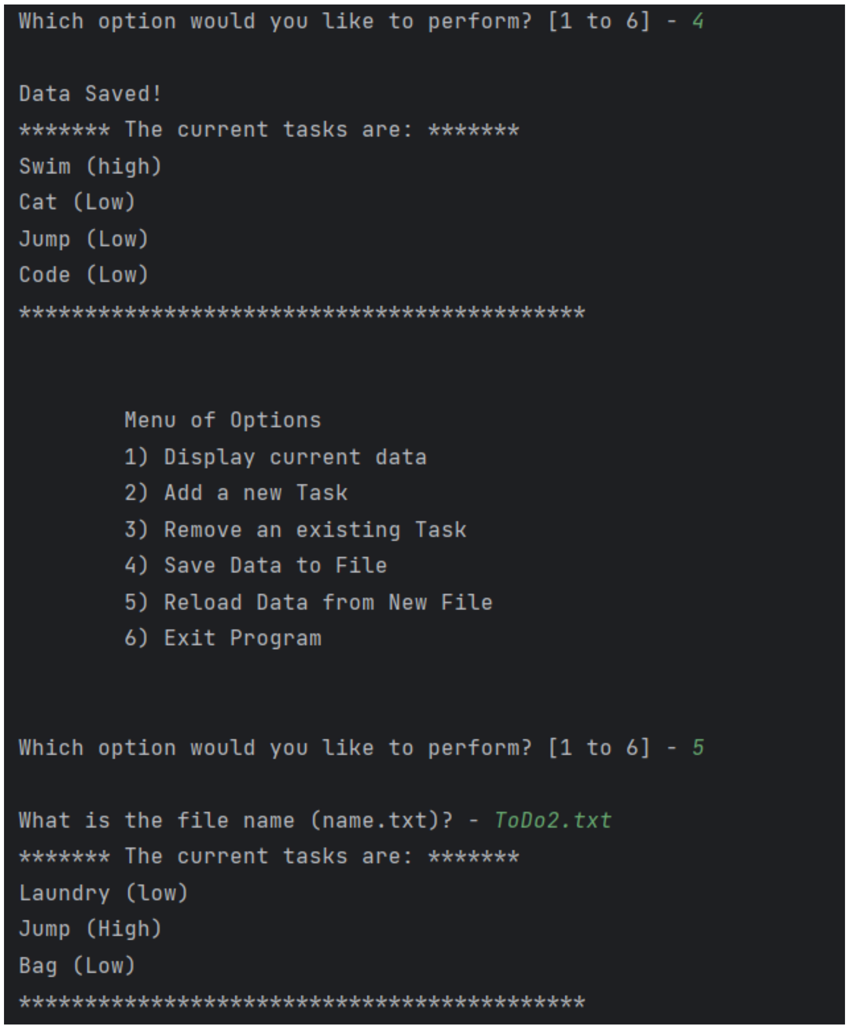
```
C:\_PythonClass\venv\Scripts\python.exe C:\_PythonClass\Assignment06\ToDoList_Plus.py
******* The current tasks are: *******
Swim (high)
Cat (Low)
Sleep (Low)
Jump (Low)
*******************************************


        Menu of Options
        1) Display current data
        2) Add a new Task
        3) Remove an existing Task
        4) Save Data to File
        5) Reload Data from New File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 2

What is the task? - Code
What is the priority? [high|low] - Low
******* The current tasks are: *******
Swim (high)
Cat (Low)
Sleep (Low)
Jump (Low)
Code (Low)
*******************************************
```

Figure 4 - Pycharm Add Data



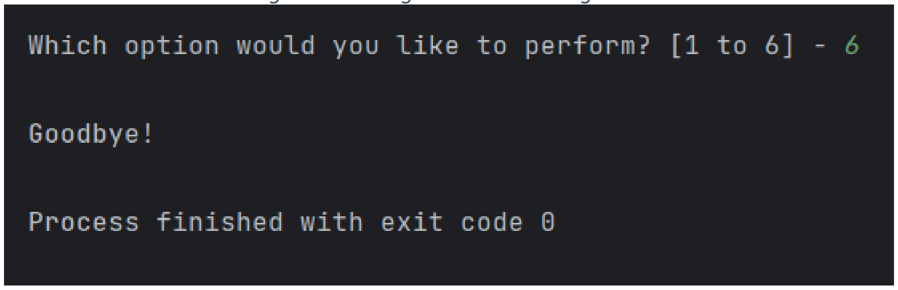
```
*******************************************


        Menu of Options
        1) Display current data
        2) Add a new Task
        3) Remove an existing Task
        4) Save Data to File
        5) Reload Data from New File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 3

Which TASK would you like removed? - Sleep
The task was removed.
```

Figure 5 - Pycharm Remove Data



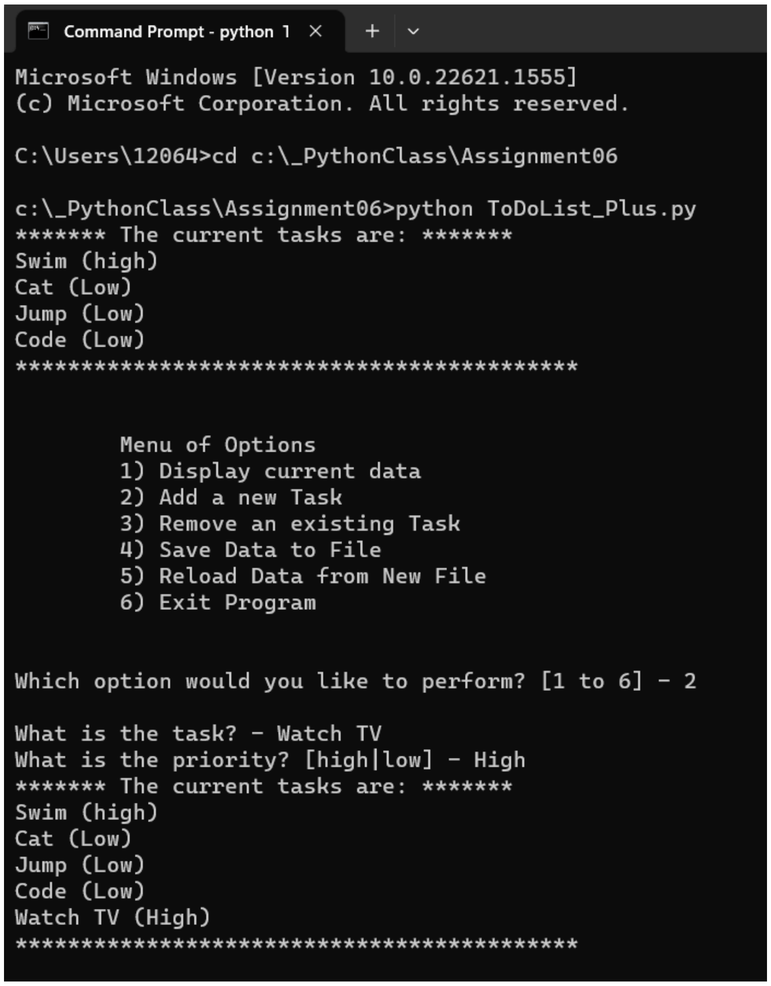
```
Which option would you like to perform? [1 to 6] - 4

Data Saved!
******* The current tasks are: *******
Swim (high)
Cat (Low)
Jump (Low)
Code (Low)
*******************************************


        Menu of Options
        1) Display current data
        2) Add a new Task
        3) Remove an existing Task
        4) Save Data to File
        5) Reload Data from New File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 5

What is the file name (name.txt)? - ToDo2.txt
******* The current tasks are: *******
Laundry (low)
Jump (High)
Bag (Low)
*******************************************
```

Figure 6 - Saving Data and Reading Data



```
Which option would you like to perform? [1 to 6] - 6

Goodbye!

Process finished with exit code 0
```

Figure 7 - Exiting the program Pycharm



Command Prompt - python 1

```
Microsoft Windows [Version 10.0.22621.1555]
(c) Microsoft Corporation. All rights reserved.

C:\Users\12064>cd c:\_PythonClass\Assignment06

c:\_PythonClass\Assignment06>python ToDoList_Plus.py
******* The current tasks are: *******
Swim (high)
Cat (Low)
Jump (Low)
Code (Low)
*******************************************


        Menu of Options
        1) Display current data
        2) Add a new Task
        3) Remove an existing Task
        4) Save Data to File
        5) Reload Data from New File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 2

What is the task? - Watch TV
What is the priority? [high|low] - High
******* The current tasks are: *******
Swim (high)
Cat (Low)
Jump (Low)
Code (Low)
Watch TV (High)
*******************************************
```

Figure 8 - Command Prompt

GitHub Webpage

I followed the instructions to create a webpage. I wasn't able to get a published URL and will need to troubleshoot that step.
https://github.com/trandallUW/IntroToProg-Python-Mod06/blob/main/docs/index.md

Summary:

This assignment was a lesson in taking someone else's code, organizing it into functions, and getting it to work as intended. The majority of the assignment taught me how to understand how functions can split up code into several steps. Most of my time was learning how someone else has structured the script and then walking through each step in an organized fashion to make sure it worked before moving onto the next section. The hardest part was learning how to remove data. Overall – I have learned how functions and "separation of concerns" can help make larger scripts more digestible and easier to adapt for future use in other applications.