Brooke Eisenhauer

November 23, 2021

Foundations of Programming: Python

Assignment 06

https://github.com/eisenhauerbrooke/IntroToProg-Python-Mod06.git

Functions

Introduction

This module focused on creating functions and parameters within a dictionary or list. Functions are powerful expressions that can be used to process and present data or groups of data. The script presents a menu to a user from a function and saves the data inputs in a dictionary.

Functions

Functions are expressions used to process and present a group of one or more statements. Functions must be defined and stored in memory before they are called by their name and parenthesis. Parameters are used data points or variables are passed into a function for processing. Not all functions require processing, so parameters are optional. There are two types of parameters: named and positional parameters. When using a named parameter, the name of the parameter matters more than the position of the parameter in reference to other parameters. On the other hand, positional parameters require appropriate positioning of the parameter in reference to other parameters and doesn't pay attention to the name of the parameter. After a function processes a named or positional parameters, it can return one or more return values.

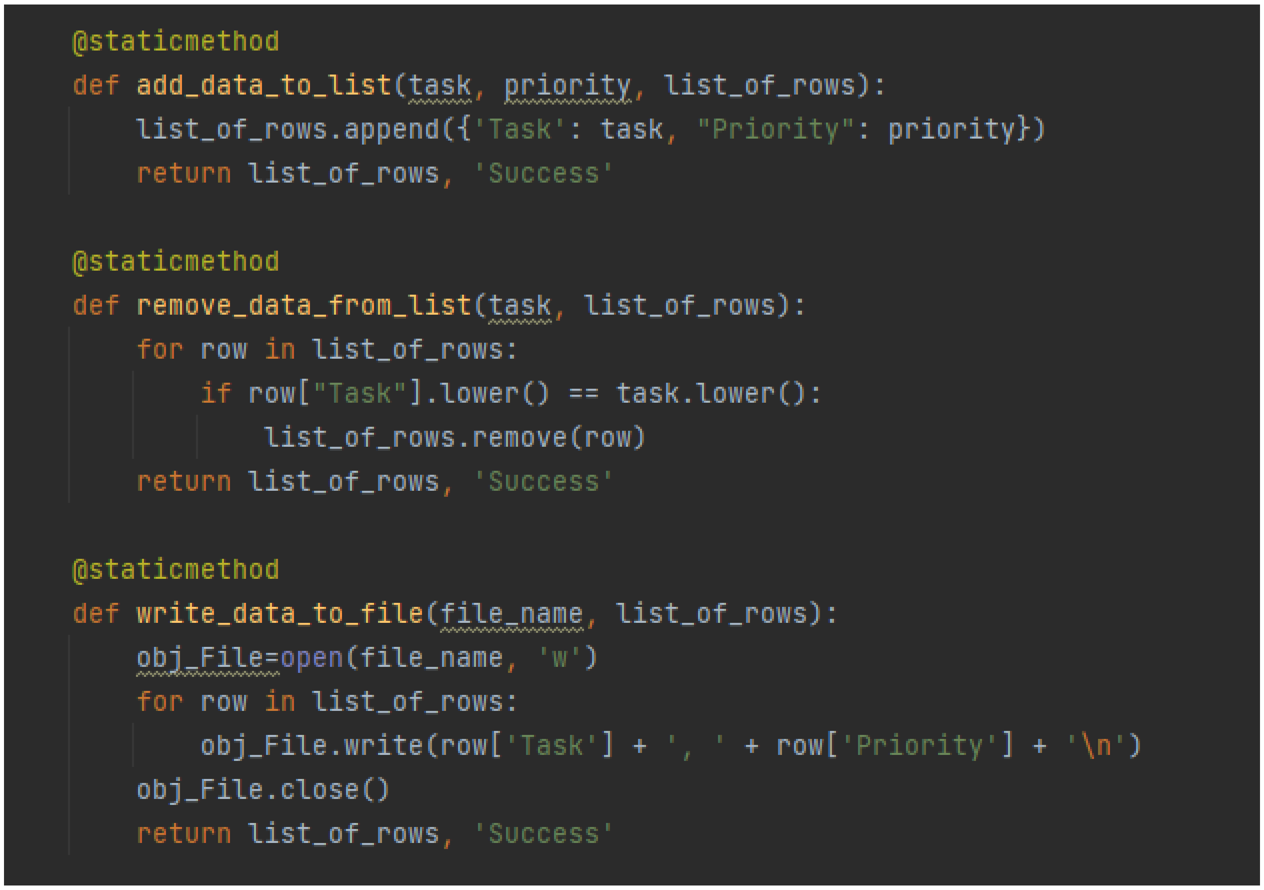
Using Functions in a Script

The script listed in Link 1, linked to GitHub (external), uses functions to the present a dictionary of data to the user and ask various questions.

Link 1: https://github.com/eisenhauerbrooke/IntroToProg-Python-Mod06.git

The script begins by stepping into the processing section, which will be accomplished in a class. Variables that require processing, such as reading data from file, adding data to a list, removing data from a list, and writing data to a file are defined in the Processor class.

Adding data to a list is accomplished by defining the function and parameters, appending the existing data with new data, and returning the results. To remove data from the list, the script searches for the task in the dictionary and removes it if found. To write data to the file, the script writes the list of data into dictionary rows and transfers it to the file, as shown in Figure 1.

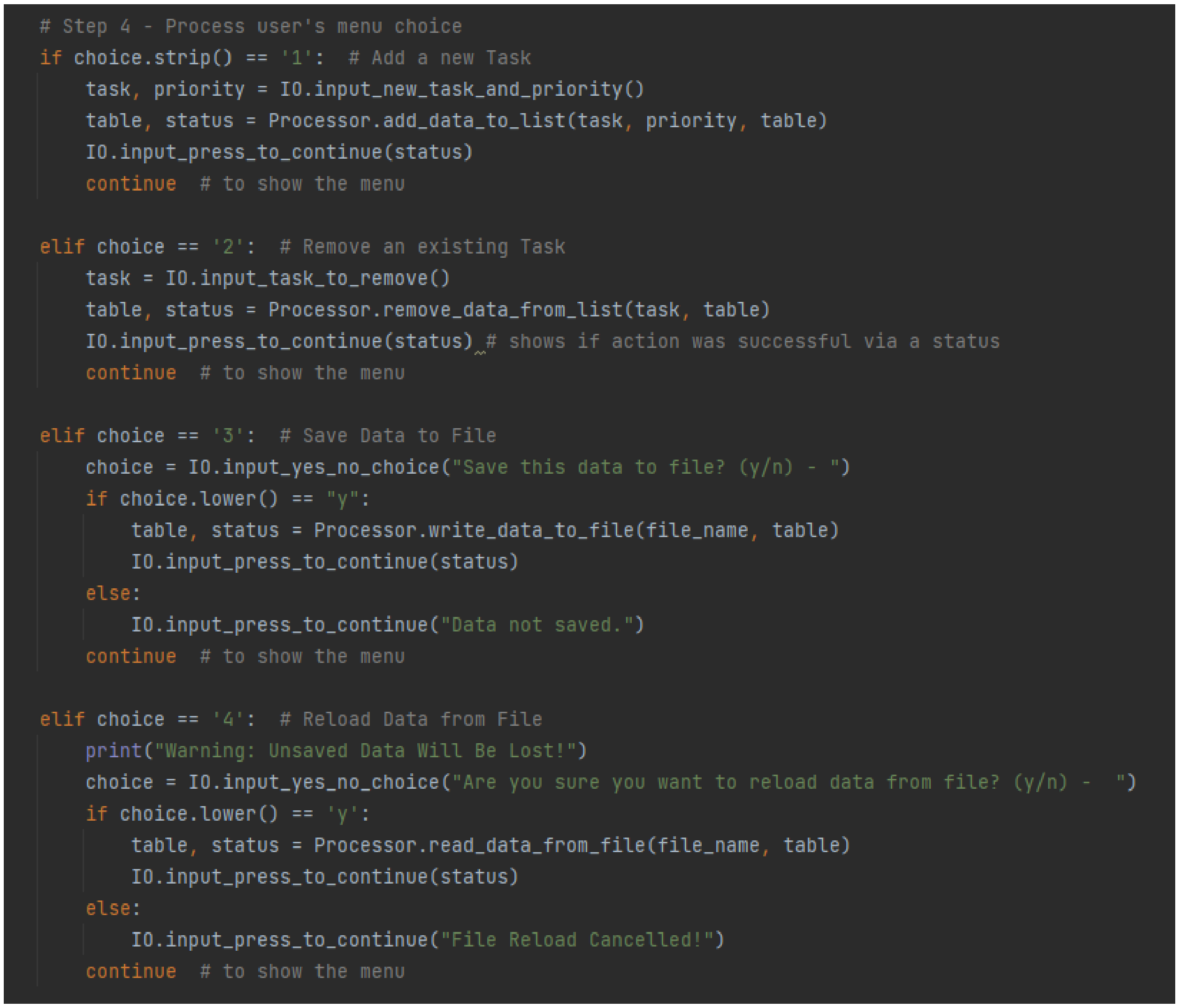


```
    @staticmethod
    def add_data_to_list(task, priority, list_of_rows):
        list_of_rows.append({'Task': task, "Priority": priority})
        return list_of_rows, 'Success'

    @staticmethod
    def remove_data_from_list(task, list_of_rows):
        for row in list_of_rows:
            if row["Task"].lower() == task.lower():
                list_of_rows.remove(row)
        return list_of_rows, 'Success'

    @staticmethod
    def write_data_to_file(file_name, list_of_rows):
        obj_File=open(file_name, 'w')
        for row in list_of_rows:
            obj_File.write(row['Task'] + ', ' + row['Priority'] + '\n')
        obj_File.close()
        return list_of_rows, 'Success'
```

Figure 1: Adding, Removing, and Writing Data to File

After writing data to the specified file, the script transitions into the input/output section, which is accomplished in a new class. The script displays a menu of options to the user and receives the user selection from the menu. If the user chooses to input a new task and priority, the script prompts the user to input a new task and priority individually and adds the data to a dictionary. The user can select to display the current data in the dictionary, save data to the dictionary, remove data from the dictionary, or exit without saving.

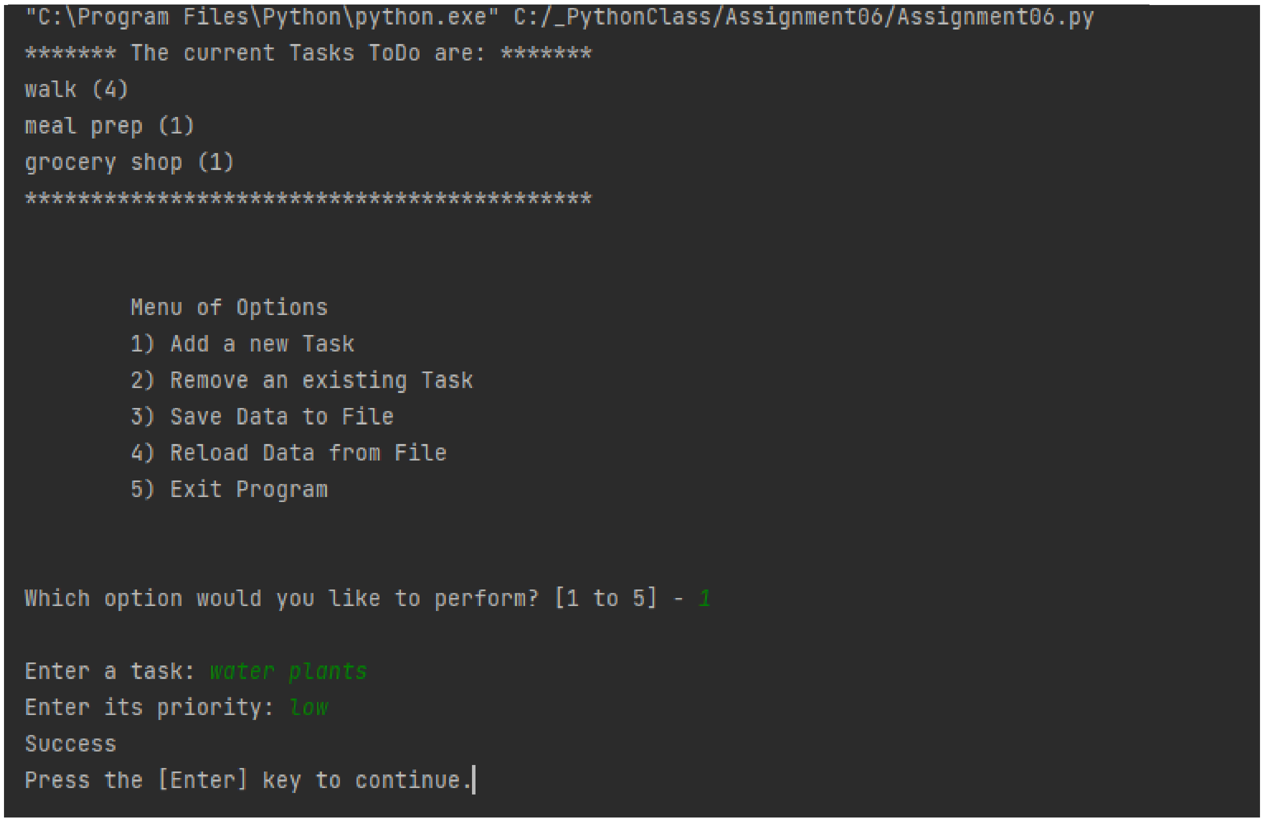
In the main body of the script, the script loads any existing data from the dictionary. The script then processes the users' inputs as previously identified. Figure 2 shows IO functions used to process the data based on the users selection.



```
# Step 4 - Process user's menu choice
if choice.strip() == '1':  # Add a new Task
    task, priority = IO.input_new_task_and_priority()
    table, status = Processor.add_data_to_list(task, priority, table)
    IO.input_press_to_continue(status)
    continue  # to show the menu

elif choice == '2':  # Remove an existing Task
    task = IO.input_task_to_remove()
    table, status = Processor.remove_data_from_list(task, table)
    IO.input_press_to_continue(status) # shows if action was successful via a status
    continue  # to show the menu

elif choice == '3':  # Save Data to File
    choice = IO.input_yes_no_choice("Save this data to file? (y/n) - ")
    if choice.lower() == "y":
        table, status = Processor.write_data_to_file(file_name, table)
        IO.input_press_to_continue(status)
    else:
        IO.input_press_to_continue("Data not saved.")
    continue  # to show the menu

elif choice == '4':  # Reload Data from File
    print("Warning: Unsaved Data Will Be Lost!")
    choice = IO.input_yes_no_choice("Are you sure you want to reload data from file? (y/n) -  ")
    if choice.lower() == 'y':
        table, status = Processor.read_data_from_file(file_name, table)
        IO.input_press_to_continue(status)
    else:
        IO.input_press_to_continue("File Reload Cancelled!")
    continue  # to show the menu
```

Figure 2: IO Functions

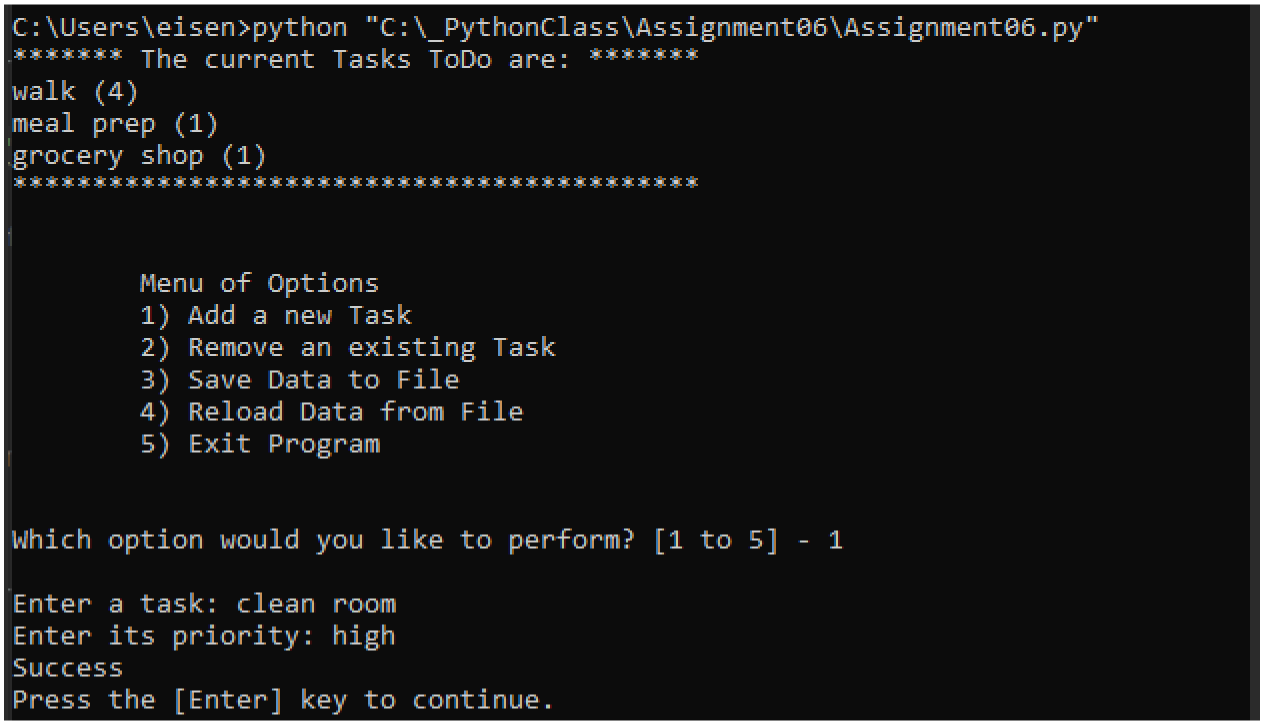
After completing the data processing, the file execution in PyCharm can be seen in Figure 3.



```
"C:\Program Files\Python\python.exe" C:/_PythonClass/Assignment06/Assignment06.py
******* The current Tasks ToDo are: *******
walk (4)
meal prep (1)
grocery shop (1)
*******************************************


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


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

Enter a task: water plants
Enter its priority: low
Success
Press the [Enter] key to continue.
```

Figure 3: PyCharm Execution

Similarly, the file executed in the Command Prompt is shown in Figure 4.



```
C:\Users\eisen>python "C:\_PythonClass\Assignment06\Assignment06.py"
******* The current Tasks ToDo are: *******
walk (4)
meal prep (1)
grocery shop (1)
*******************************************


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


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

Enter a task: clean room
Enter its priority: high
Success
Press the [Enter] key to continue.
```

Figure 4: Command Module Execution

Summary

Functions were used in the script to display a menu of options to the user and save the outputs within a dictionary. Functions can be used to process and present data as long as the function is defined prior to processing or presentation.