Irina Zubova

Feb 20, 2021

Foundations of Programming: Python

Assignment 06

https://github.com/i-zuzu/IntroToProg-Python

Adding Functions to a Script that Manages "ToDo list"

Introduction

In this document I'll describe in details how I worked on Assignment 06. The goal was to modify the starter script by adding more functions to organize the code. The starter code loaded data from a file into a Python List of Dictionary objects. The final code should manage the text file and be able to:

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

To perform this assignment I'll use the new knowledge on:

- Reading and understanding the code written by other person
- Creating the functions
- Using functions to organize the code
- Working with simple function class
- Creating dictionaries in Python

Drafting the Script

Checking the starter script

I saved the Assignment06_Starter file and the text file that was also provided in Assignment06 folder. Then I opened the script in PyCharm and checked the content and if the script worked. The script had the header and 4 sections:

1. *Data*

   There were variables and constants declared to use in the code

2. *Processor (the performs processing tasks)*

This section had functions class called Processor and 4 placeholders for functions that would perform reading data from the file, adding data to the list, removing data from the list and writing data to the file. Actually the first function to perform reading data from the file was already created and I used it as example

3. *Presentation (Input/Output)*

This section performs Input and Output tasks and has another function class IO. Functions to interact with the user (display the information and get user's inputs were there). 5 functions were completed and I needed to work on 2 placeholders.

4. *Main Body of the script*

This section places together pieces of the code from the previous sections. It had 4 steps:

Step 1 - When the program starts, Load data from ToDoFile.txt.

Step 2 - Display a menu of choices to the user

Step 3 Show current data

Step 4 - Process user's menu choice

Here I had to work on Step 4 to process user's choice like remove the existing task or add the new one.

Figure 1 shows the outcome of the starter script after running in PyCharm.

```
C:\Users\nh825c\Anaconda3\python.exe C:/Users/nh825c/Documents/_PythonClass/Assignment06/Assigment06.py
******* The current Tasks ToDo are: *******
update starter code (high)
write about the how I did it (high)
upload to new GitHub repository (high)
create repository web page (high)
submit to Canvas (high)
do a peer review (low)
*******************************************


        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


Press the [Enter] key to continue.
******* The current Tasks ToDo are: *******
update starter code (high)
write about the how I did it (high)
upload to new GitHub repository (high)
create repository web page (high)
submit to Canvas (high)
do a peer review (low)
*******************************************


        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
```

Figure 1. Screenshot of the Starter Script running in PyCharm

There was while loop under step 2 in Main Body section that continued displaying data from the ToDo file, the menu of options and offered to choose the option (from 1 to 5).

I revised the revision control in the script header and moved to Processing section to add functions to the code.

Adding code to Processing section

Function to add data to the list.
Below is the code from the started script:

```
 @staticmethod
def add_data_to_list(task, priority, list_of_rows):
    # TODO: Add Code Here!
    return list_of_rows, 'Success'
```

From these lines I understood the name of the function was `add_data_to_list` and this name gave idea on what the function had to perform. The function had 3 parameters (`task`, `priority` and `list_of_rows`) and it had to return `list_of_rows`.
I started with the docstring or function document headers to document what this function is for, the type and the definition of each parameter and what this function needed to return.
Since I had the code that performed the same task but without functions in the previous assignment05 I just had to convert this code to function:

Code from Assignment 5 without the function	Code converted to function

`strTask = input("Enter the task: ")`
`strPrior = input("Enter Priority(Low, Medium, High): " ).title()`
`tplPRIOR = ('Low', 'Medium', 'High')`
`while strPrior not in tplPRIOR:`
`  print("Please choose Low, Medium or High")`
`  strPrior = input("Enter Priority(Low, Medium, High): " ).title()`
 	
`dicRow = {"Task":strTask.title(), "Priority":strPrior.title()}`
 	
`if dicRow in lstTable:`
`  print('Already exist.')`
`else:`
`  lstTable.append(dicRow)`
`  print("\n Item added.")`
	`@staticmethod`
	`def add_data_to_list(task, priority, list_of_rows):`
	`  """Adds data entered by the user to a list of dictionary rows`
	`  :param task: (string) with the name of the task to do:`
	`  :param priority: (string) with the priority (Low, Medium, High):`
	`  :param list_of_rows: (list) you want to add data task and priority) to:`
	`  :return: (list) of dictionary rows`
	`  """`
	`  row = {"Task": task, "Priority": priority.lower()}`
	`  list_of_rows.append(row)`
	`  return list_of_rows, 'Success'`

I gave the variables and parameters inside the function simple names as they cannot be access outside the function due to encapsulation. I kept the function simple and didn't add the while loop to restrict user in the choice of priorities.
Now I had to test new function works. I copied the function to a separate test file. I had to call the function by just using the name of the function followed by parentheses. Also I provided values to the function through the parameters. `('play', 'low', lstTable)` in the code below are required arguments or variables passed from the caller to the function. Then I displayed the result with the `print()` function.
Code in the test file:

```
lstTable = []  # A list that acts as a 'table' of rows
def add_data_to_list(task, priority, list_of_rows):
    """Adds data entered by the user to a list of dictionary rows

    :param task: (string) with the name of the task to do:
    :param priority: (string) with the priority (Low, Medium, High):
    :param list_of_rows: (list) you want to add data task and priority) to:
    :return: (list) of dictionary rows
    """
    # strTask = input("Enter the task: ")
    # strPriority = input("Enter Priority(Low, Medium, High): ").title()
    row = {"Task": task, "Priority": priority.lower()}
    list_of_rows.append(row)
    return list_of_rows, 'Success'

add_data_to_list('play', 'low', lstTable)
print(lstTable)
```

The output after running test for `add_data_to_list` function is shown on Figure 2. As planned the data provided to the function was converted to dictionary and put to list:

```
[{'Task': 'play', 'Priority': 'low'}]

Process finished with exit code 0
```

Figure 2. Output of the `add_data_to_list` function.

Functions to remove data from the list

Below is the code from the started script:

```
@staticmethod
def remove_data_from_list(task, list_of_rows):
    # TODO: Add Code Here!
    return list_of_rows, 'Success'
```

The name of the function is `add_data_to_list`, the function has 2 parameters (`task` and `list_of_rows`) and it returns `list_of_rows`.
I started with the documentation string and then convert the piece of code form Assignemnt05 to the function:

Code from Assignment 5 without the function	Code converted to function

`strTask = input("Please enter the task you want to delete: ").title()`
`for dicRow in lstTable:`
`  if strTask == dicRow['Task']:`
`  lstTable.remove(dicRow)`
`  print("\n Item removed.")`
`  break`
`else:`
`  print(strTask, "is not in the Task list.")`
	`@staticmethod`
	`def remove_data_from_list(task, list_of_rows):`
	`  """Deletes data entered by the user from a list of dictionary rows`
	
	`  :param task: (string) with the name of the task to delete:`
	`  :return: (list) of dictionary rows`
	`  """`
	`  for row in list_of_rows:`
	`  if task == row["Task"]:`
	`  list_of_rows.remove(row)`
	`  break`
	`  return list_of_rows, 'Success'`

I used for loop inside the function which tells the program to go through each line of the table and remove the task if it matches the task provided by the user.

I run the test of this new function in the separate test file. I created simple `lstTable` list that had only 2 lines, provided 2 required variables to the function and printed the results.
Code in the test file:

```
lstTable = [{"Task": 'play' , "Priority": 'low'}, {"Task": 'eat' , "Priority": 'hign'}]

def remove_data_from_list(task, list_of_rows):
    # TODO: DONE
    """Deletes data entered by the user from a list of dictionary rows
```

```
    :param task: (string) with the name of the task to delete:
    :return: (list) of dictionary rows
    """
    for row in list_of_rows:
        if task == row["Task"]:
            list_of_rows.remove(row)
            break
    return list_of_rows, 'Success'

remove_data_from_list('eat', lstTable)
print(lstTable)
```

The output after running test for `remove_data_from_list` function is shown on Figure 3. As planned the function returned the list with only one line as the other line was deleted.

```
[{'Task': 'play', 'Priority': 'low'}]

Process finished with exit code 0
```

Figure 3. Output of the `remove_data_from_list` function.

Functions to write data to file

Below is the code from the started script:

```
@staticmethod
def write_data_to_file(file_name, list_of_rows):
    # TODO: Add Code Here!
    return list_of_rows, 'Success'
```

The name of the function is `write_data_to_file`, the function has 2 parameters (`file_name` and `list_of_rows`) and it returns `list_of_rows`.

I started with the documentation string and then convert the piece of code form Assignemnt05 to the function:

Code from Assignment 5 without the function	Code converted to function

`objFile = open("ToDoList.txt", "w")`
`for dicRow in lstTable:`
`  objFile.write(dicRow["Task"] + ", " + dicRow["Priority"] + '\n')`
`print("Data saved to file")`
`objFile.close()`
	`@staticmethod`
	`def write_data_to_file(file_name, list_of_rows):`
	`  """ Writes data to a file from list of dictionary rows`
	`  :param file_name: (string) with name of file:`
	`  :param list_of_rows: (list) you want write to file:`
	`  :return: (list) of dictionary rows`
	`  """`
	`  file = open(file_name, "w")`
	`  for row in list_of_rows:`
	`  file.write(row["Task"] + ", " + row["Priority"] + '\n')`
	`  file.close()`
	`  return list_of_rows, 'Success'`

Again, I used simple names for variables inside the function as they will only be used within this function.

The code to test the new function on separate file is below:

```
lstTable = [{"Task": 'play' , "Priority": 'low'}, {"Task": 'eat' , "Priority":
'high'}]

def write_data_to_file(file_name, list_of_rows):
    """ Writes data to a file from list of dictionary rows

    :param file_name: (string) with name of file:
    :param list_of_rows: (list) you want write to file:
    :return: (list) of dictionary rows
    """
    file = open(file_name, "w")
    for row in list_of_rows:
        file.write(row["Task"] + ", " + row["Priority"] + '\n')
    file.close()
    return list_of_rows, 'Success'

write_data_to_file("TesFile.txt", lstTable)
```

The output after running test for `write_data_to_file` function is shown on Figure 4. As desired data from the list (or table) was written to the file.

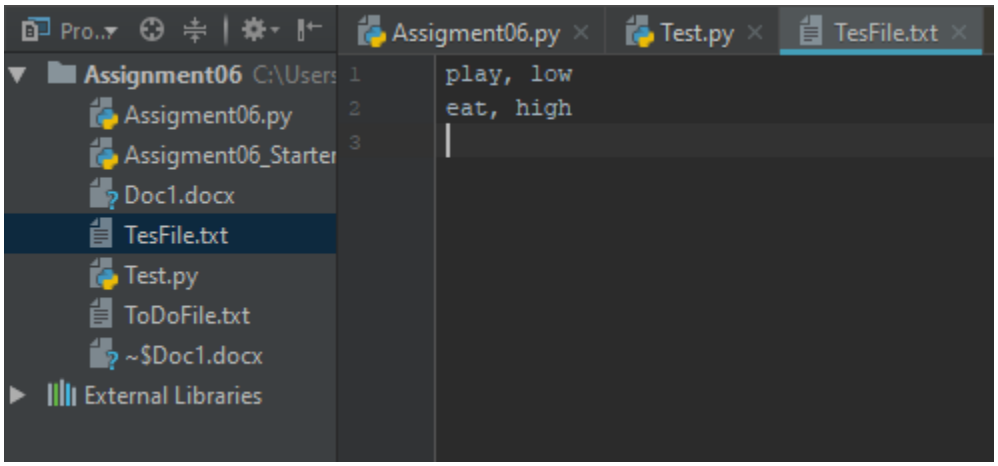

Figure 4. Output of the `write_data_to_file` function.

Adding code to Presentation section

Now moving to 2 functions defined in IO class. Below is the code from the starter script for these 2 functions.

```
@staticmethod
def input_new_task_and_priority():
    pass  # TODO: Add Code Here!
    # return task, priority

@staticmethod
def input_task_to_remove():
    pass  # TODO: Add Code Here!
    # return task
```

The names of these functions give the idea on what the functions need to do. These functions dn't have any parameters, they just get input from the user using the input function. The code with docstring for these functions is below:

```
@staticmethod
def input_new_task_and_priority():
    """ Gets the name of a new task and priority to add to the list from a user
    :return: string, string
    """
    task = input("Enter the task: ")
    priority = input("Enter Priority(low, medium or high): ").lower()
    return task, priority
    # return task, priority

@staticmethod
def input_task_to_remove():
    """ Gets the name of a task to remove from the list from a user
    :return: string
    """
    task = input("Please enter the task you want to remove: ")
    return task
```

Adding code to main Body of the Script

I needed to call the functions defined in the previous sections to perform all actions rom the menu of options.

```
# Step 4 - Process user's menu choice
if strChoice.strip() == '1':  # Add a new Task
    # TODO: Add Code Here
    IO.input_press_to_continue(strStatus)
    continue  # to show the menu
```

If the user chooses option 1, then 2 functions are used (or called) to get from the user data on the name of a new task and priority and add these data to the list: `input_new_task_and_priority()` and `add_data_to_list`.
Also to notify user that the action was implemented, I used `input_press_to_continue` function from IO class that has optional message as a parameter. I had to specify the name of the class where the functions were located. The code to perform these actions is below:

```
 Step 4 - Process user's menu choice
if strChoice.strip() == '1':  # Add a new Task
    strTask, strPriority = IO.input_new_task_and_priority()
    Processor.add_data_to_list(strTask, strPriority, lstTable)
    strStatus = "\n Task added.\n"
    IO.input_press_to_continue(strStatus)
    continue  # to show the menu
```

Next option from the menu is 2 where the existing task has to be deleted and here's the code from the starter file:

```
elif strChoice == '2':  # Remove an existing Task
    # TODO: Add Code Here
    IO.input_press_to_continue(strStatus)
    continue  # to show the menu
```

I assigned the `input_task_to_remove()` function to a variable `strTask` and then I called `remove_data_from_list ()` function and passed 2 arguments – the `strTask` and the `lstTable` list. I used `input_press_to_continue()` function from IO class that has optional message as a parameter. This time I used the message "Item removed" as the argument to notify the user the task was implemented.

```
elif strChoice == '2':  # Remove an existing Task
    strTask = IO.input_task_to_remove()
    Processor.remove_data_from_list(strTask, lstTable)
    strStatus = "\n Item removed.\n"
    IO.input_press_to_continue(strStatus)
    continue  # to show the menu
```

Now moving to the next option to save data to the file, below is the starter code:

```
elif strChoice == '3':   # Save Data to File
    strChoice = IO.input_yes_no_choice("Save this data to file? (y/n) - ")
    if strChoice.lower() == "y":
        # TODO: Add Code Here!
        IO.input_press_to_continue(strStatus)
```

I used here write_data_to_file function from the processor class:

```
elif strChoice == '3':   # Save Data to File
    strChoice = IO.input_yes_no_choice("Save this data to file? (y/n) - ")
    if strChoice.lower() == "y":
        Processor.write_data_to_file(strFileName, lstTable)
        strStatus = "\n Data saved to file.\n"
        IO.input_press_to_continue(strStatus)
```

The last one piece of code I had to work on from the starter script:

```
elif strChoice == '4':  # Reload Data from File
    print("Warning: Unsaved Data Will Be Lost!")
    strChoice = IO.input_yes_no_choice("Are you sure you want to reload data from
file? (y/n) -  ")
    if strChoice.lower() == 'y':
        # TODO: Add Code Here!
        IO.input_press_to_continue(strStatus)
```

I just called here the functions to read data from file, display current task and notify the user:

```
elif strChoice == '4':  # Reload Data from File
    print("Warning: Unsaved Data Will Be Lost!")
    strChoice = IO.input_yes_no_choice("Are you sure you want to reload data from
file? (y/n) -  ")
    if strChoice.lower() == 'y':
        Processor.read_data_from_file(strFileName, lstTable)
        IO.print_current_Tasks_in_list(lstTable)
        strStatus = "\n Data reloaded from file.\n"
        IO.input_press_to_continue(strStatus)
```

Final Script

```
# ------------------------------------------------------------------------------ #
# Title: Assignment 06
# Description: Working with functions in a class,
#              When the program starts, load each "row" of data
#              in "ToDoToDoList.txt" into a python Dictionary.
#              Add the each dictionary "row" to a python list "table"
# ChangeLog (Who,When,What):
# RRoot,1.1.2030,Created started script
# RRoot,1.1.2030,Added code to complete assignment 5
# IZubova,2.19.2021,Modified code to complete assignment 6
# ------------------------------------------------------------------------------ #

# Data ---------------------------------------------------------------------- #
# Declare variables and constants
strFileName = "ToDoFile.txt"  # The name of the data file
objFile = None   # An object that represents a file
dicRow = {}  # A row of data separated into elements of a dictionary {Task,Priority}
lstTable = []  # A list that acts as a 'table' of rows
strChoice = ""  # Captures the user option selection
strTask = ""  # Captures the user task data
strPriority = ""  # Captures the user priority data
strStatus = ""  # Captures the status of an processing functions

# Processing  ------------------------------------------------------------- #
class Processor:
    """  Performs Processing tasks """

    @staticmethod
    def read_data_from_file(file_name, list_of_rows):
        """ Reads data from a file into a list of dictionary rows

        :param file_name: (string) with name of file:
        :param list_of_rows: (list) you want filled with file data:
        :return: (list) of dictionary rows
        """
        list_of_rows.clear()  # clear current data
        file = open(file_name, "r")
        for line in file:
            task, priority = line.split(",")
            row = {"Task": task.strip(), "Priority": priority.strip()}
            list_of_rows.append(row)
        file.close()
        return list_of_rows, 'Success'

    @staticmethod
    def add_data_to_list(task, priority, list_of_rows):
        """Adds data entered by the user to a list of dictionary rows

        :param task: (string) with the name of the task to do:
        :param priority: (string) with the priority (Low, Medium, High):
        :param list_of_rows: (list) you want to add data task and priority) to:
        :return: (list) of dictionary rows
        """
        row = {"Task": task, "Priority": priority.lower()}
        list_of_rows.append(row)
        return list_of_rows, 'Success'

    @staticmethod
    def remove_data_from_list(task, list_of_rows):
        """Deletes data entered by the user from a list of dictionary rows
```

```
        :param task: (string) with the name of the task to delete:
        :return: (list) of dictionary rows
        """
        for row in list_of_rows:
            if task == row["Task"]:
                list_of_rows.remove(row)
                break
        return list_of_rows, 'Success'

    @staticmethod
    def write_data_to_file(file_name, list_of_rows):
        """ Writes data to a file from list of dictionary rows

        :param file_name: (string) with name of file:
        :param list_of_rows: (list) you want write to file:
        :return: (list) of dictionary rows
        """
        file = open(file_name, "w")
        for row in list_of_rows:
            file.write(row["Task"] + ", " + row["Priority"] + '\n')
        file.close()
        return list_of_rows, 'Success'

# Presentation (Input/Output)  -------------------------------------------- #
class IO:
    """ Performs Input and Output tasks """

    @staticmethod
    def print_menu_Tasks():
        """  Display a menu of choices to the user

        :return: nothing
        """
        print('''
        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
        ''')
        print()  # Add an extra line for looks

    @staticmethod
    def input_menu_choice():
        """ Gets the menu choice from a user

        :return: string
        """
        choice = str(input("Which option would you like to perform? [1 to 5] -
")).strip()
        print()  # Add an extra line for looks
        return choice

    @staticmethod
    def print_current_Tasks_in_list(list_of_rows):
        """ Shows the current Tasks in the list of dictionaries rows

        :param list_of_rows: (list) of rows you want to display
        :return: nothing
        """
        print("******* The current Tasks ToDo are: *******")
```

```
        for row in list_of_rows:
            print(row["Task"] + " (" + row["Priority"] + ")")
        print("******************************************")
        print()  # Add an extra line for looks

    @staticmethod
    def input_yes_no_choice(message):
        """ Gets a yes or no choice from the user

        :return: string
        """
        return str(input(message)).strip().lower()

    @staticmethod
    def input_press_to_continue(optional_message=''):
        """ Pause program and show a message before continuing

        :param optional_message:  An optional message you want to display
        :return: nothing
        """
        print(optional_message)
        input('Press the [Enter] key to continue.')

    @staticmethod
    def input_new_task_and_priority():
        """ Gets the name of a new task and priority to add to the list from a user
        :return: string, string
        """
        task = input("Enter the task: ")
        priority = input("Enter Priority(low, medium or high): ").lower()
        return task, priority
        # return task, priority

    @staticmethod
    def input_task_to_remove():
        """ Gets the name of a task to remove from the list from a user
        :return: string
        """
        task = input("Please enter the task you want to remove: ")
        return task

# Main Body of Script  ------------------------------------------------------ #

# Step 1 - When the program starts, Load data from ToDoFile.txt.
Processor.read_data_from_file(strFileName, lstTable)  # read file data

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

    # Step 4 - Process user's menu choice
    if strChoice.strip() == '1':  # Add a new Task
        strTask, strPriority = IO.input_new_task_and_priority()
        Processor.add_data_to_list(strTask, strPriority, lstTable)
        strStatus = "\n Task added.\n"
        IO.input_press_to_continue(strStatus)
        continue  # to show the menu

    elif strChoice == '2':  # Remove an existing Task
        strTask = IO.input_task_to_remove()
```

```
        Processor.remove_data_from_list(strTask, lstTable)
        strStatus = "\n Item removed.\n"
        IO.input_press_to_continue(strStatus)
        continue  # to show the menu

    elif strChoice == '3':   # Save Data to File
        strChoice = IO.input_yes_no_choice("Save this data to file? (y/n) - ")
        if strChoice.lower() == "y":
            Processor.write_data_to_file(strFileName, lstTable)
            strStatus = "\n Data saved to file.\n"
            IO.input_press_to_continue(strStatus)
        else:
            IO.input_press_to_continue("Save Cancelled!")
        continue  # to show the menu

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

    elif strChoice == '5':  #  Exit Program
        print("Goodbye!")
        break   # and Exit
```

Running the Final code

Assignment06 is asking to run the script in both OS command/shell window and PyCharm. Figure 5 captures the image of the script running in PyCharm:

```
C:\Users\nh825c\Anaconda3\python.exe C:/Users/nh825c/Documents
******* The current Tasks ToDo are: *******
update starter code (high)
write about the how I did it (high)
upload to new GitHub repository (high)
create repository web page (high)
submit to Canvas (high)
do a peer review (low)
sleep (low)
play (low)
*****************************************


        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 the task: walk
Enter Priority(low, medium or high): medium

 Task added.

Press the [Enter] key to continue.
******* The current Tasks ToDo are: *******
update starter code (high)
write about the how I did it (high)
upload to new GitHub repository (high)
create repository web page (high)
submit to Canvas (high)
do a peer review (low)
sleep (low)
play (low)
walk (medium)
*****************************************


        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] - 2

Please enter the task you want to remove: sleep

 Item removed.

Press the [Enter] key to continue.3
******* The current Tasks ToDo are: *******
update starter code (high)
write about the how I did it (high)
upload to new GitHub repository (high)
create repository web page (high)
submit to Canvas (high)
do a peer review (low)
play (low)
walk (medium)
*****************************************


        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] - 3

Save this data to file? (y/n) - y

 Data saved to file.

Press the [Enter] key to continue.
```

```
******* The current Tasks ToDo are: *******
update starter code (high)
write about the how I did it (high)
upload to new GitHub repository (high)
create repository web page (high)
submit to Canvas (high)
do a peer review (low)
play (low)
walk (medium)
*****************************************


        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] - 4

Warning: Unsaved Data Will Be Lost!
Are you sure you want to reload data from file? (y/n) - y
******* The current Tasks ToDo are: *******
update starter code (high)
write about the how I did it (high)
upload to new GitHub repository (high)
create repository web page (high)
submit to Canvas (high)
do a peer review (low)
play (low)
walk (medium)
*****************************************


 Data reloaded from file.

Press the [Enter] key to continue.
******* The current Tasks ToDo are: *******
update starter code (high)
write about the how I did it (high)
upload to new GitHub repository (high)
create repository web page (high)
submit to Canvas (high)
do a peer review (low)
play (low)
walk (medium)
*****************************************


        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] - 5

Goodbye!

Process finished with exit code 0
```

Figure 5. Screenshot of the Script running in PyCharm

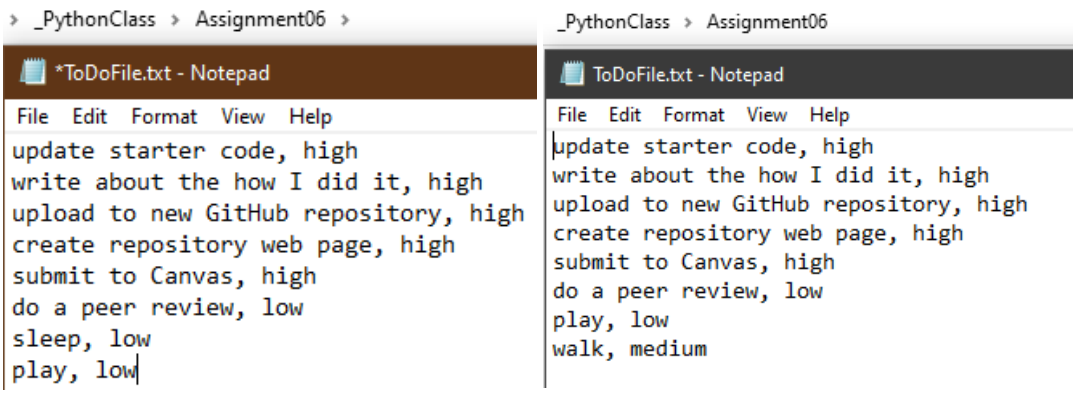

Figure 6. Screenshot of the ToDo list file before (on the left) and after (on the right) the script runs

Then I ran the script in Terminal. Figure 7 shows the results:

Figure 7. Screenshot of the Script running in Windows PowerShell

And the last picture (Figure 8) verifies the data in the file was changed according to the user inputs. The original text file is in the upper left corner, the final version after user's changes in in the lower right corner.

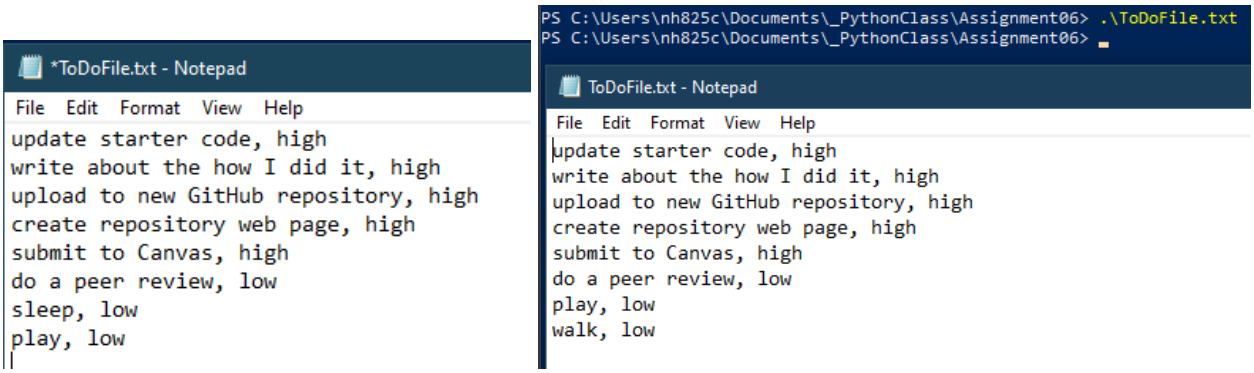

Figure 8. Print screens of the text file before (on the left) and after (on the right) running the script in PowerShell

Summary

In this assignment I learned how to make the code more clear with the separation of control programming pattern. I practiced to create functions and work with its components like parameters and docstring. I also got exposed to the classes – the way of grouping functions. Using knowledge on functions, list and dictionaries I was able to make changes in the text file like adding and deleting rows and then I finally saved the updated data back to the text file.