Jason Johanneck
2022-May-23
IT FDN Programming 110
Module05 Assignment06

FDN Programming 110 Assignment 06

Github Repository Link: https://github.com/jjohanne514/IntroToProg_PythonAssignment06.git

Introduction

Assignment06 asks us to include the same Menu choices as Assignment 05 but wrap the IO and data processing code into functions that will invoked when handling user selected menu choices.

Building the script.

The script uses a pre-built python starter file where we are to complete the TODO sections. Here is my code listing from NotePad++.

Figure 1 - Assignment 06 Code Listing

```
# ------------------------------------------------------------------------ #
# Title: Assignment 06
# Description: Working with functions in a class,
#              When the program starts, load each "row" of data
#              in "ToDoList.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
# Jason Johanneck, 5/23/2022, Modified code to complete assignment 06
# ------------------------------------------------------------------------ #

# Data -------------------------------------------------------------------- #
# Declare variables and constants
strFileName = "ToDoList.txt"  # The name of the data file
file_obj = None  # An object that represents a file
row_dic = {}  # A row of data separated into elements of a dictionary {Task,Rank}
lstTable = []  # A list that acts as a 'table' of rows
choice_str = ""  # Captures the user option selection

# Processing  ------------------------------------------------------------- #
class Processor:
    """ Processes the data in a list of dictionaries to and from a text file """

    @staticmethod
    def read_file_to_list_of_dictionaries(file_name, list_of_dictionary_rows):
        list_of_dictionary_rows.clear()  # clear any old data before loading
        file = open(file_name, "r")  # Causes ERROR if file does not exist!
        for line in file:
            item, value = line.split(",")
            row = {"Task": item.strip(), "Rank": value.strip()}
            list_of_dictionary_rows.append(row)
        file.close()
        return list_of_dictionary_rows, 'success'

    @staticmethod
    def remove_data_from_list_of_dictionaries(list_of_dictionary_rows, task_to_remove):
        for row in list_of_dictionary_rows:
            if row["Task"].lower() == task_to_remove.lower():
                lstTable.remove(row)
                # print("row removed")
        return list_of_dictionary_rows, 'success'

    @staticmethod
    def add_data_to_list_of_dictionaries(list_of_dictionary_rows, item, value):
        row = {"Task": str(item).strip(), "Rank": str(value).strip()}
        list_of_dictionary_rows.append(row)
        return list_of_dictionary_rows, 'success'

    @staticmethod
    def write_file_from_list_of_dictionaries(file_name, list_of_dictionary_rows):
        file = open(file_name, "w")
        for row in list_of_dictionary_rows:
            file.write(row["Task"] + "," + row["Rank"] + "\n")
        file.close()
        return list_of_dictionary_rows, 'success'

# Presentation (Input/Output)  -------------------------------------------- #
class IO:
    @staticmethod
    def print_menu():
        print('''
        Menu of Options
        1) Load Data from File
        2) Add a new Task
        3) Remove an existing Task
        4) Save Data to File
        5) Exit Program
        ''')

    @staticmethod
    def input_menu_choice():
        choice = str(input("Which option would you like to perform? [1 to 4] - ")).strip()
        print()  # Add an extra line for looks
        return choice

    @staticmethod
    def print_current_list_items(list_of_rows):
        print("******* The Current Items Are: *******")
        for row in list_of_rows:
            print(row["Task"] + " (" + row["Rank"] + ")")
        print("*************************************************")
        print()  # Add an extra line for looks

    @staticmethod
    def input_task_and_rank():
        task = str(input("What is the Task? - ")).strip()
        rank = str(input("What is the Rank? - ")).strip()
        print()  # Add an extra line for looks
        return task, rank

    @staticmethod
    def input_task_to_remove():
        task = str(input("Remove which item? - ")).strip()
        print()  # Add an extra line for looks
        return task


# Main Body of the Script  ------------------------------------------------ #
while(True):
    IO.print_current_list_items(lstTable)
    IO.print_menu()  # Test menu
    strChoice = IO.input_menu_choice()    # Test user choice

    if (strChoice == "1"):  # 1) Load Data from File
        lstTable, status = Processor.read_file_to_list_of_dictionaries(strFileName, lstTable)
        if status == 'success':
            print('Done!')
    elif (strChoice == "2"):  # 2) Add a new item
        strTask, strRank = IO.input_task_and_rank()
        lstTable, status = Processor.add_data_to_list_of_dictionaries(lstTable, strTask,strRank)
        if status == 'success':
            print('Done!')
    elif (strChoice == "3"):  # 3) Remove an existing item
        strTask = IO.input_task_to_remove()
        lstTable, status = Processor.remove_data_from_list_of_dictionaries(lstTable, strTask)
        if status == 'success':
            print('Done!')
    elif (strChoice == "4"):  # 4) Save Data to File And Exit Program
        lstTable, status = Processor.write_file_from_list_of_dictionaries(strFileName, lstTable)
        if status == 'success':
            print('Done!')
    elif (strChoice == "5"):
        print("Goodbye!")
        break
```

The Python code was broken out into three sections:

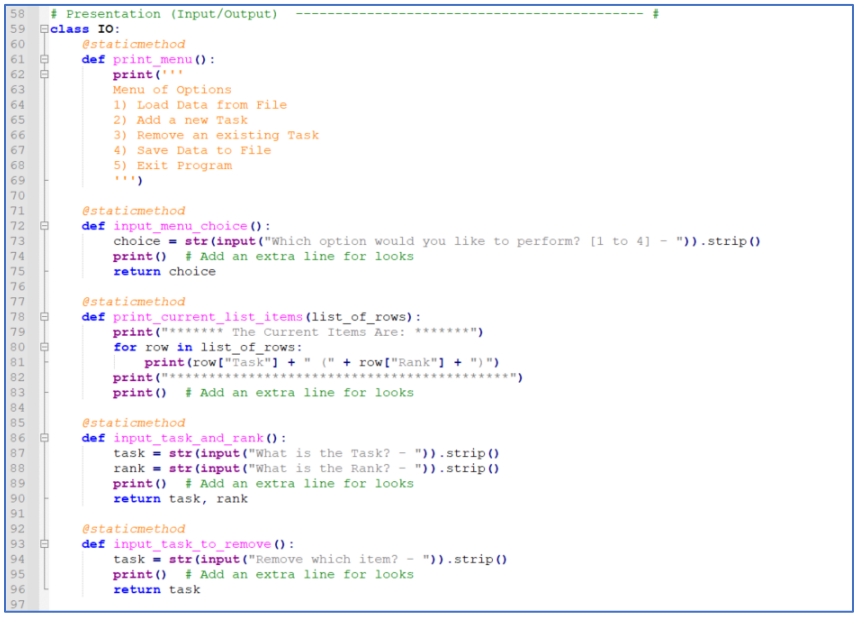
Section one: A function class for Data Processing was defined that included four functions:

Figure 2- Processor Class with four functions

```
# Processing  ------------------------------------------------------------- #
class Processor:
    """ Processes the data in a list of dictionaries to and from a text file """

    @staticmethod
    def read_file_to_list_of_dictionaries(file_name, list_of_dictionary_rows):
        list_of_dictionary_rows.clear()  # clear any old data before loading
        file = open(file_name, "r")  # Causes ERROR if file does not exist!
        for line in file:
            item, value = line.split(",")
            row = {"Task": item.strip(), "Rank": value.strip()}
            list_of_dictionary_rows.append(row)
        file.close()
        return list_of_dictionary_rows, 'success'

    @staticmethod
    def remove_data_from_list_of_dictionaries(list_of_dictionary_rows, task_to_remove):
        for row in list_of_dictionary_rows:
            if row["Task"].lower() == task_to_remove.lower():
                lstTable.remove(row)
                # print("row removed")
        return list_of_dictionary_rows, 'success'

    @staticmethod
    def add_data_to_list_of_dictionaries(list_of_dictionary_rows, item, value):
        row = {"Task": str(item).strip(), "Rank": str(value).strip()}
        list_of_dictionary_rows.append(row)
        return list_of_dictionary_rows, 'success'

    @staticmethod
    def write_file_from_list_of_dictionaries(file_name, list_of_dictionary_rows):
        file = open(file_name, "w")
        for row in list_of_dictionary_rows:
            file.write(row["Task"] + "," + row["Rank"] + "\n")
        file.close()
        return list_of_dictionary_rows, 'success'

# Presentation (Input/Output)  ------------------------------------------- #
```

Section two: A function class for screen IO was created with five Input/Output functions:

Figure 3- IO Class with five functions defined



```
# Presentation (Input/Output)  ------------------------------------------- #
class IO:
    @staticmethod
    def print_menu():
        print('''
        Menu of Options
        1) Load Data from File
        2) Add a new Task
        3) Remove an existing Task
        4) Save Data to File
        5) Exit Program
        ''')

    @staticmethod
    def input_menu_choice():
        choice = str(input("Which option would you like to perform? [1 to 4] - ")).strip()
        print()  # Add an extra line for looks
        return choice

    @staticmethod
    def print_current_list_items(list_of_rows):
        print("******* The Current Items Are: *******")
        for row in list_of_rows:
            print(row["Task"] + " (" + row["Rank"] + ")")
        print("*****************************************")
        print()  # Add an extra line for looks

    @staticmethod
    def input_task_and_rank():
        task = str(input("What is the Task? - ")).strip()
        rank = str(input("What is the Rank? - ")).strip()
        print()  # Add an extra line for looks
        return task, rank

    @staticmethod
    def input_task_to_remove():
        task = str(input("Remove which item? - ")).strip()
        print()  # Add an extra line for looks
        return task

```

Section three: The main body of the script contains logic for processing menu selections:

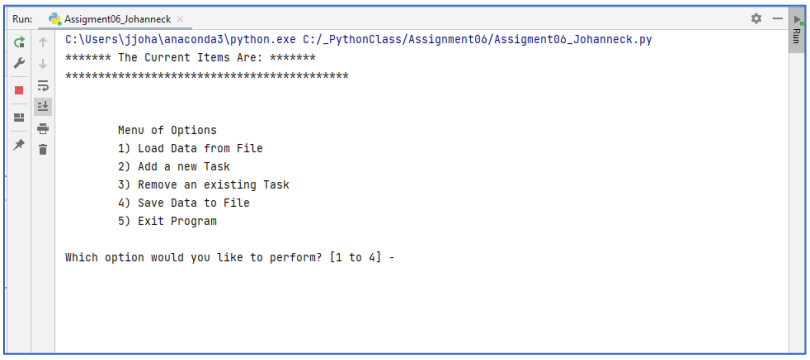
Figure 4 - Main Script section for Menu Processing Logic

```
# Main Body of the Script  ------------------------------------------------------ #
while(True):
    IO.print_current_list_items(lstTable)
    IO.print_menu()   # Test menu
    strChoice = IO.input_menu_choice()    # Test user choice

    if (strChoice == "1"):  # 1) Load Data from File
        lstTable, status = Processor.read_file_to_list_of_dictionaries(strFileName, lstTable)
        if status == 'success':
            print('Done!')
    elif (strChoice == "2"):  # 2) Add a new item
        strTask, strRank = IO.input_task_and_rank()
        lstTable, status = Processor.add_data_to_list_of_dictionaries(lstTable, strTask,strRank)
        if status == 'success':
            print('Done!')
    elif (strChoice == "3"):  # 3) Remove an existing item
        strTask = IO.input_task_to_remove()
        lstTable, status = Processor.remove_data_from_list_of_dictionaries(lstTable, strTask)
        if status == 'success':
            print('Done!')
    elif (strChoice == "4"):  # 4) Save Data to File And Exit Program
        lstTable, status = Processor.write_file_from_list_of_dictionaries(strFileName, lstTable)
        if status == 'success':
            print('Done!')
    elif (strChoice == "5"):
        print("Goodbye!")
        break
```

Running the script.

From the PyCharm application:

Here is the Initial Display after pressing the run button:



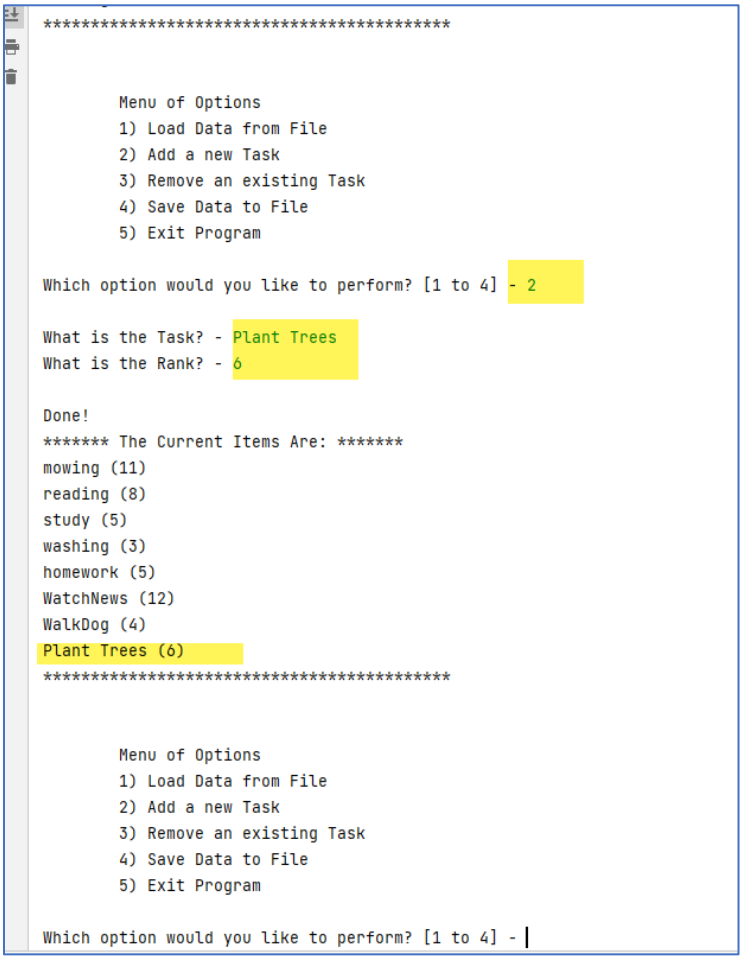
Here is output after selecting the option to Load the ToDoList.txt data:

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

Done!
******* The Current Items Are: *******
mowing (11)
reading (8)
study (5)
shave (7)
washing (3)
homework (5)
WatchNews (12)
WalkDog (4)
**********


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

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

Here is the output after selecting the menu choice to remove a task:

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


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

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

Remove which item? - shave

Done!
******* The Current Items Are: *******
mowing (11)
reading (8)
study (5)
washing (3)
homework (5)
WatchNews (12)
WalkDog (4)
*****************************************


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

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

Here I am selecting the menu option to add a new Task for Planting Trees:



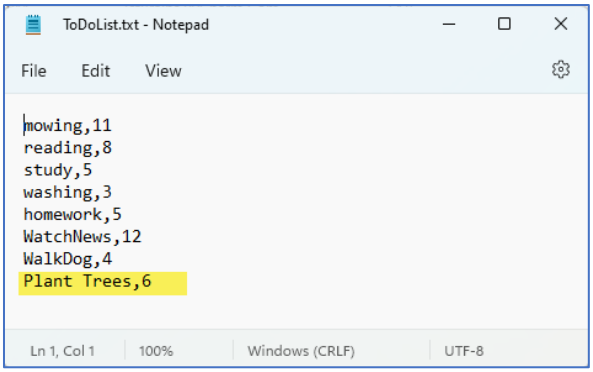
```
*****************************************


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

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

What is the Task? - Plant Trees
What is the Rank? - 6

Done!
******* The Current Items Are: *******
mowing (11)
reading (8)
study (5)
washing (3)
homework (5)
WatchNews (12)
WalkDog (4)
Plant Trees (6)
*****************************************


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

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

This is the output after selecting menu option #4 to Save data to disk:

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


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

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

Done!
******* The Current Items Are: *******
mowing (11)
reading (8)
study (5)
washing (3)
homework (5)
WatchNews (12)
WalkDog (4)
Plant Trees (6)
*****************************************


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

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

Here I am showing the contents of ToDoList.txt after saving to disk:

Figure 5- Contents of ToDoList.txt after Menu Option 4 selected

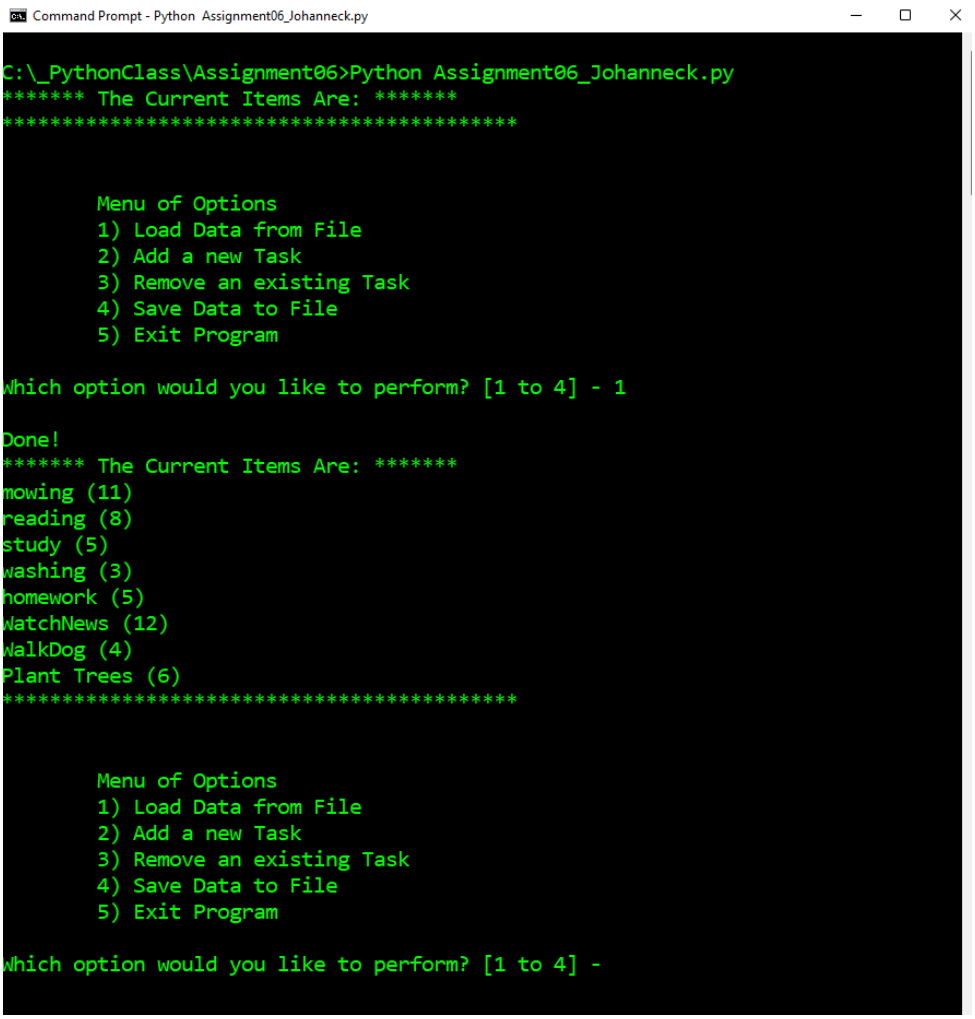


Finally, here is PyCharm output after selecting menu option #5 to Exit:

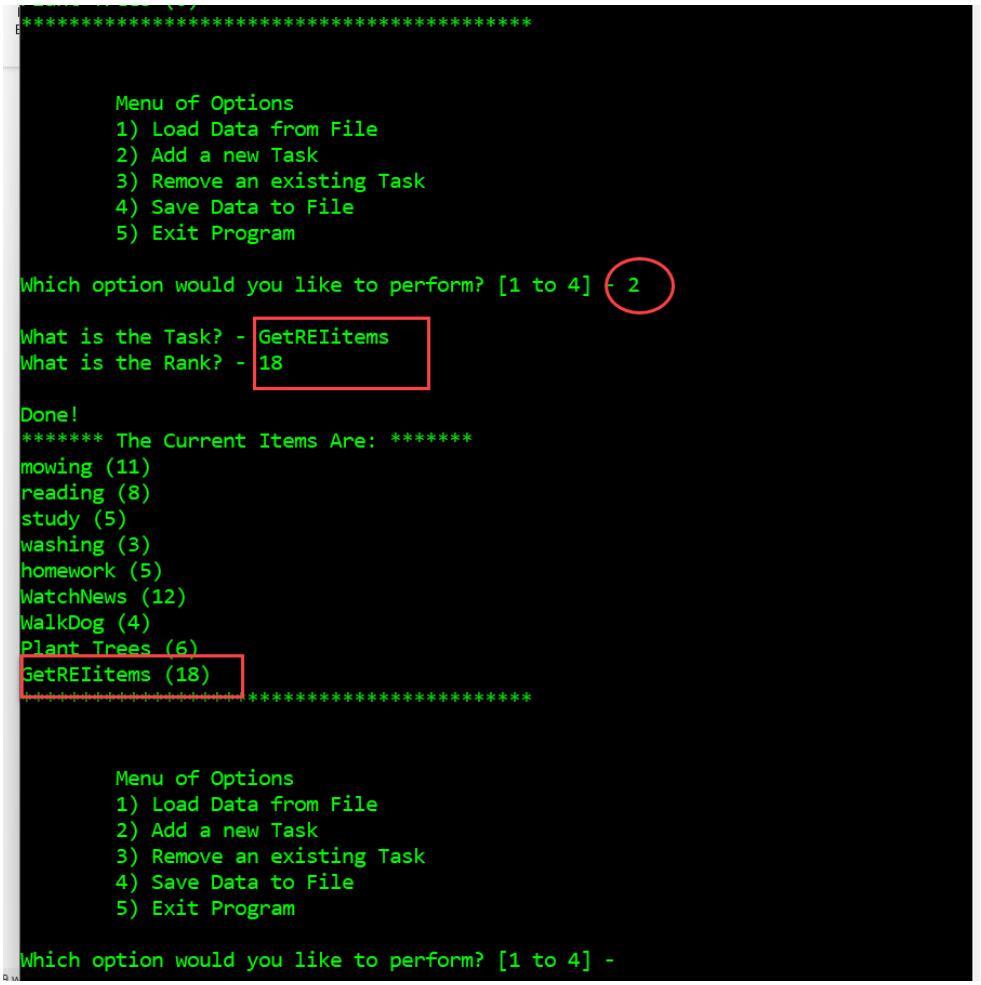
```
*****************************************


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

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

Goodbye!

Process finished with exit code 0
```

Running the script from the Windows Command prompt:

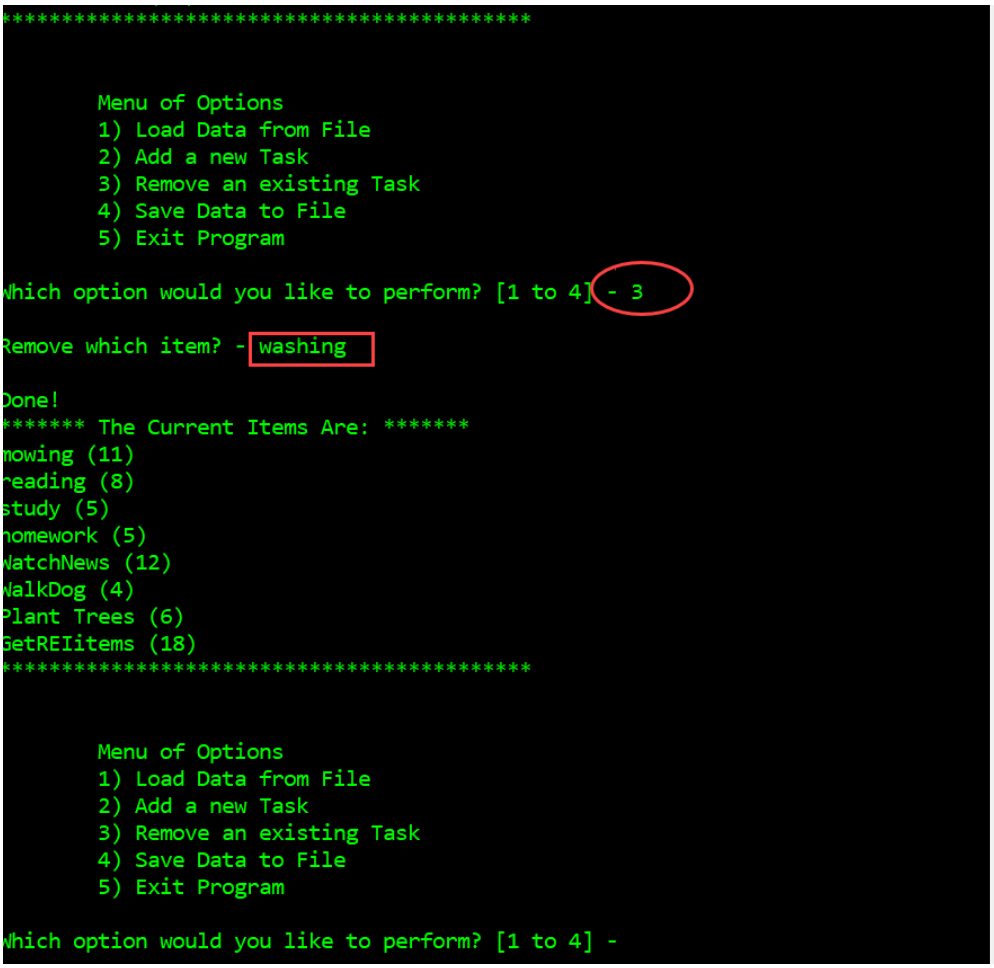
Here I run the Assignment06_Johanneck.py and then selecting the menu option to load data from file:



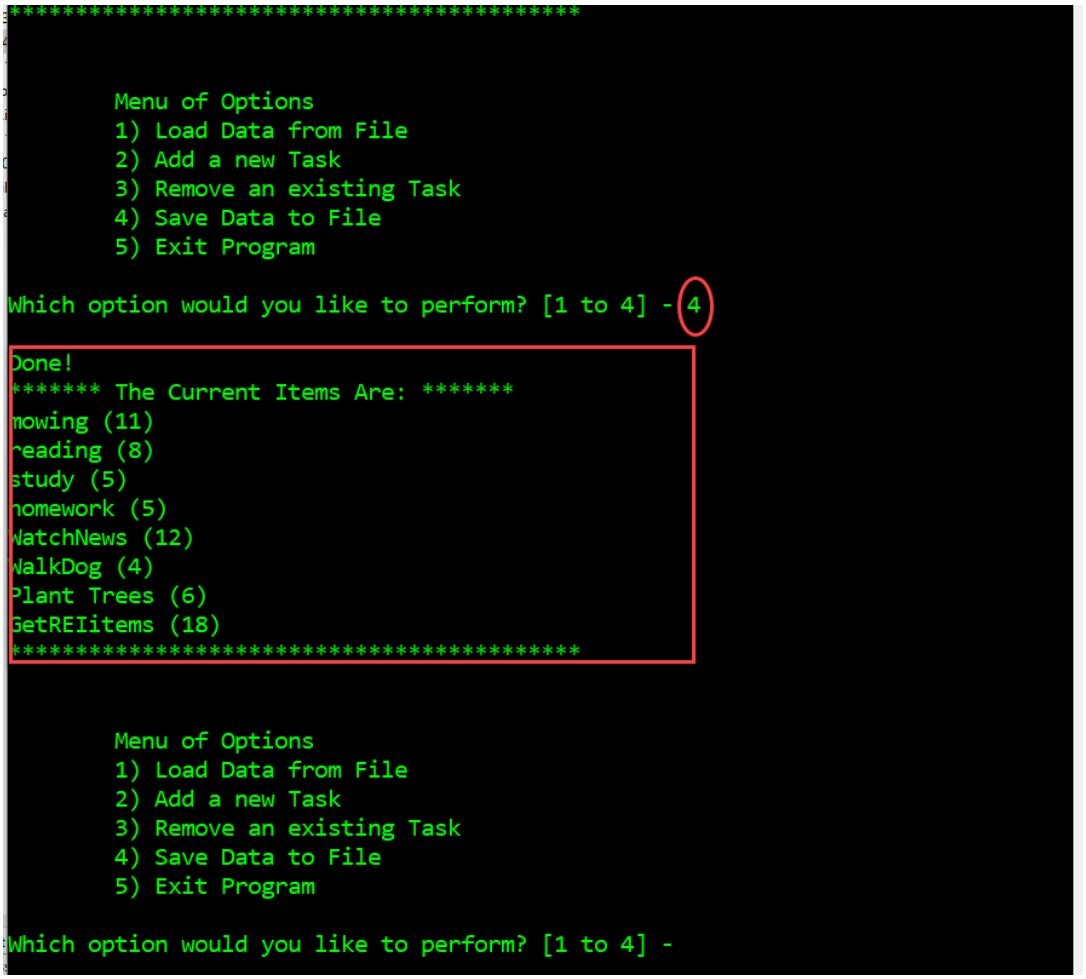
The following screen shows the addition of a new task:



The below screen print is from my choice to remove an item from the list:

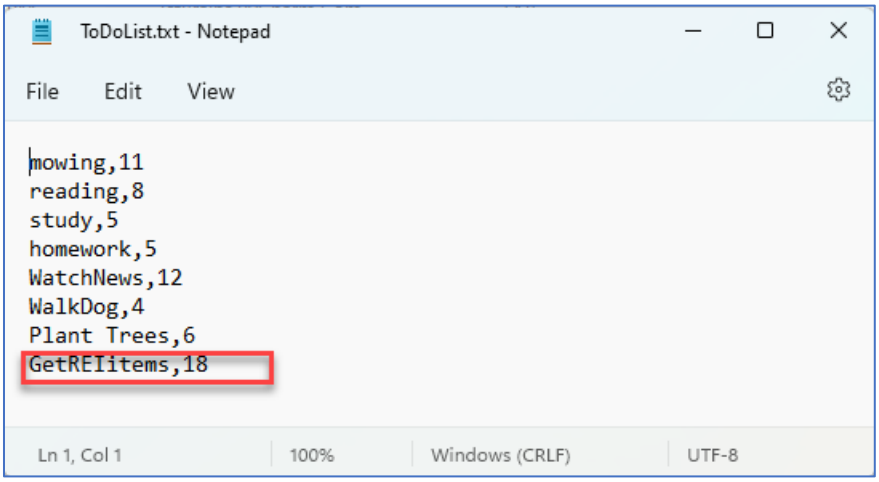


In this next screen print, I chose to Save the current list table of dictionary rows to disk:

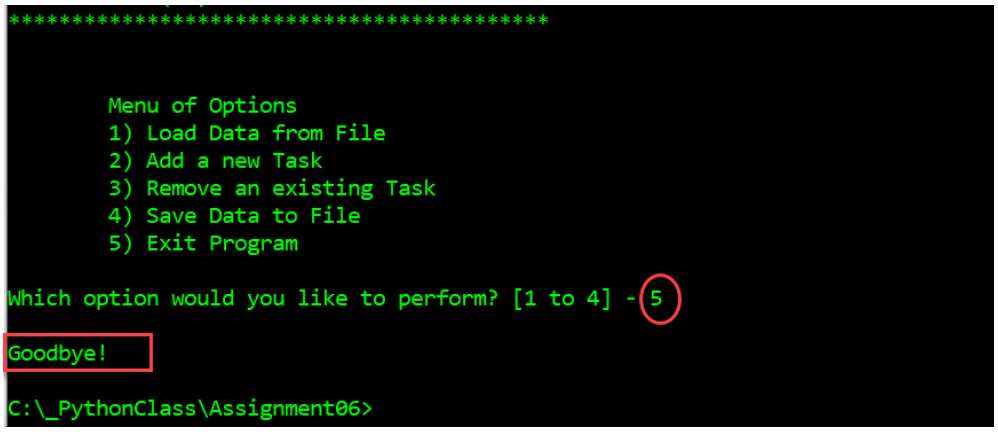


The contents of the ToDoList.txt file after removing the Washing Task and Saving the list of Tasks to Disk:

Figure 6 - contents of ToDoList.txt from Notepad



Finally, after selecting option #5 to Exit:



Summary

In this assignment we learned how to consolidate screen I/O and table processing into functions. We also learned how pass parameters into functions and evaluate function return arguments.