Anthony Giles

August 16, 2022

IT FDN 100 B

Assignment 06

To Do List Program

Introduction

This week's assignment involves creating a simple program that will demonstrate how to create *functions* with the use of *classes*. We also will be uploading program data to GitHub.

Creating the Program

File was created in PyCharm using starter file per instruction. (Figure 1)

ile.txt | Assigment06.py

```
# ------------------------------------------------------------------------ #
# 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
#AGiles,8.16.2022,Modified code to complete assignment 06, completed required functions
# ------------------------------------------------------------------------ #

# Data ------------------------------------------------------------------- #
# Declare variables and constants
file_name_str = "ToDoFile.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,Priority}
```

Figure 1: Shows starter file with modified header

Create Required Process Functions

Created the following functions by using the *def* statement and specifying required *parameters* for passing variables. "**add_data_to_list**" function (Figure 2), "**remove_data_from_list**" function (Figure 2.1), "**write_data_to_file**" function (Figure 2.2)

```python
@staticmethod
def add_data_to_list(task, priority, list_of_rows):
    """ Adds data to a list of dictionary rows

    :param task: (string) with name of task:
    :param priority: (string) with name of priority:
    :param list_of_rows: (list) you want filled with file data:
    :return: (list) of dictionary rows
    """
    row = {"Task": str(task).strip(), "Priority": str(priority).strip()}
    list_of_rows.append(row)
    return list_of_rows
```

Figure 2: Shows " add_data_to_list" function

```python
@staticmethod
def remove_data_from_list(task, list_of_rows):
    """ Removes data from a list of dictionary rows

    :param task: (string) with name of task:
    :param list_of_rows: (list) you want filled with file data:
    :return: (list) of dictionary rows
    """
    for row in list_of_rows:
        if row["Task"].lower().strip() == task:
            bolFound = True
            list_of_rows.remove(row)
            print("Task has been removed")
            break
        else:
            bolFound = False
    if bolFound == False:
        print("Task was not found")
    return list_of_rows
```

Figure 2.1: Shows " remove_data_from_list" function

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

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

Figure 2.2: Shows " write_data_to_file" function

Create Required Input Functions

Created the following required input functions and returned specified values.
"**input_new_task_and_priority**" function (Figure 3), "**input_task_to_remove**" (Figure 3.1)

```python
@staticmethod
def input_new_task_and_priority():
    """  Gets task and priority values to be added to the list

    :return: (string, string) with task and priority
    """
    task = input("Task: ")
    priority = input("Priority: ")
    return task, priority
```

Figure 3: Shows "input_new_task_and_priority" function

```python
@staticmethod
def input_task_to_remove():
    """  Gets the task name to be removed from the list

    :return: (string) with task
    """
    task = input("Task to be removed: ").strip().lower()
    return task
```

Figure 3.1: Shows "input_task_to_remove" function

Executing Program

Per instructions program was executed from PyCharm. (Figure 4)

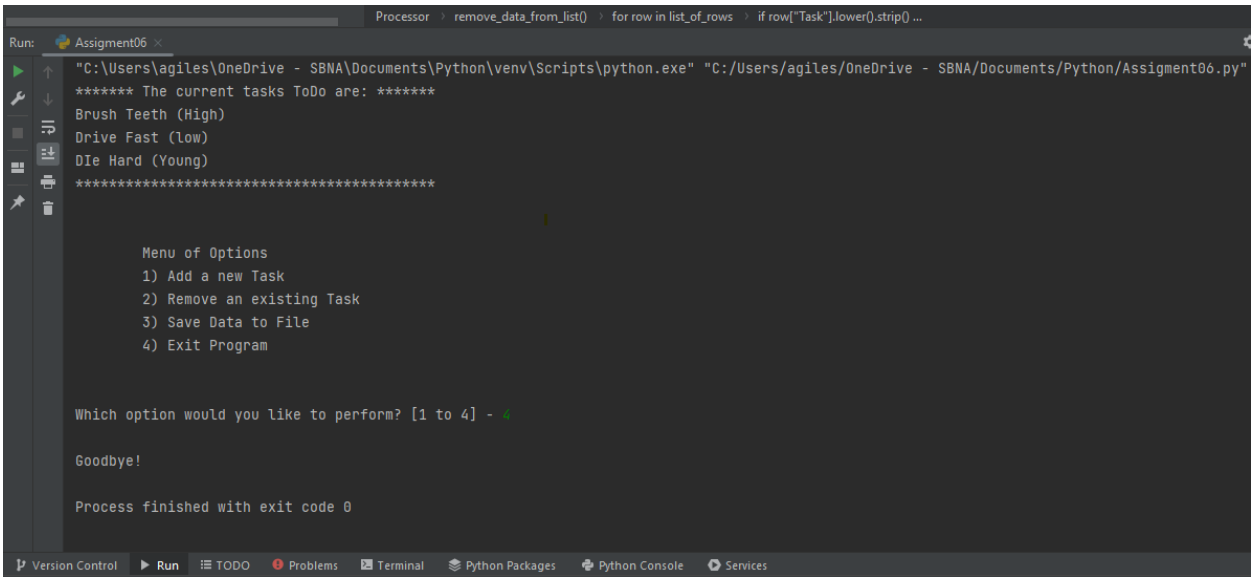

Figure 4: Shows program execution

Display data file from within PyCharm. (Figure 4.1)

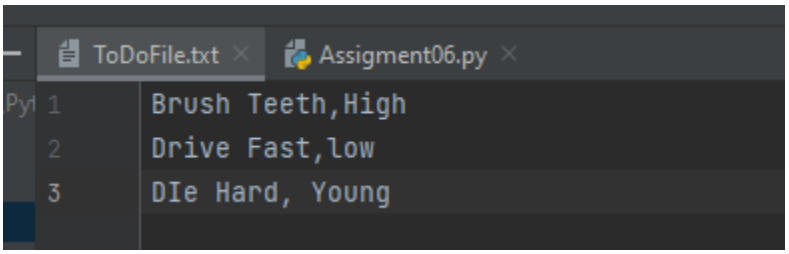

Figure 4.1: Shows item entry in data file

Per instructions program was executed from terminal. (Figure 4.2)

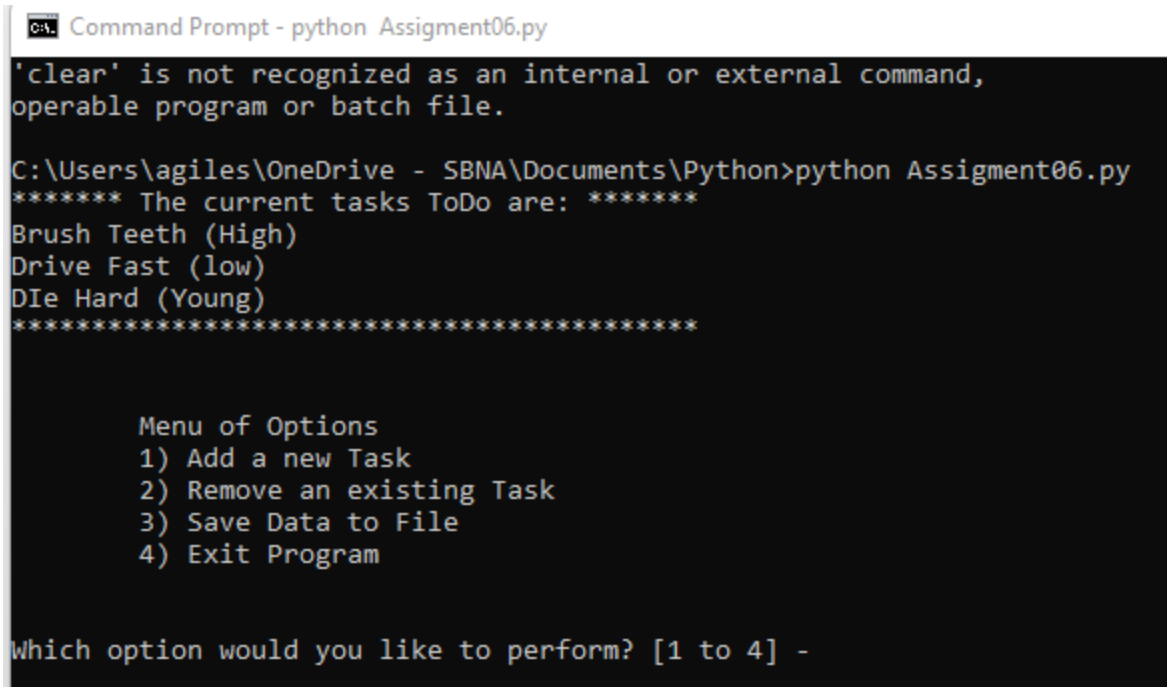

Figure 4.2: Shows program execution

Opened data file to confirm data entry. (Figure 4.3)

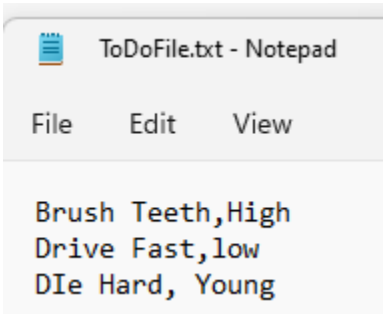

Figure 4.3: Shows item entry in data file

Summary

This week's assignment involved creating a simple program that demonstrated how to create *functions* with the use of *classes*. We also uploaded program data to GitHub with the addition of a GitHub Web Page.