Tellrell White

05/28/2025

Foundations of Programming: Python

Assignment 6

https://github.com/twhite320/IntroToProg-Python-Mod6

Functions

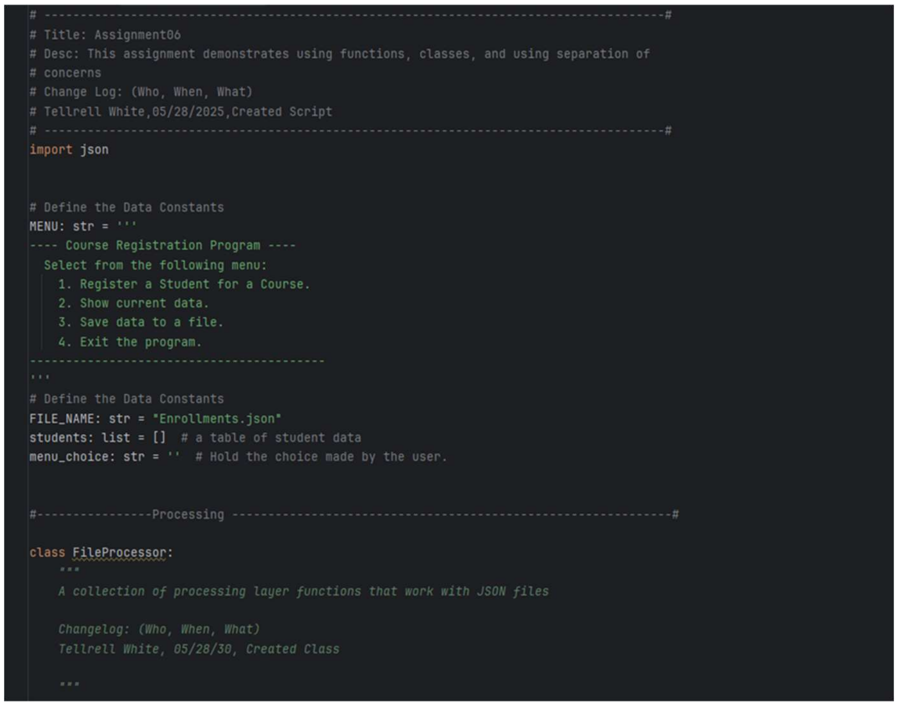
Introduction

In this document I will discuss the steps I used to create and test a Python script that displays a message about a student's registration in a Python course. This assignment is very similar to Assignment 5; however, the additional concepts of functions, classes, and separation of concerns (SoC) were added to complete this assignment using the Pycharm Integrated Development Environment (IDE).

Development of Python Script

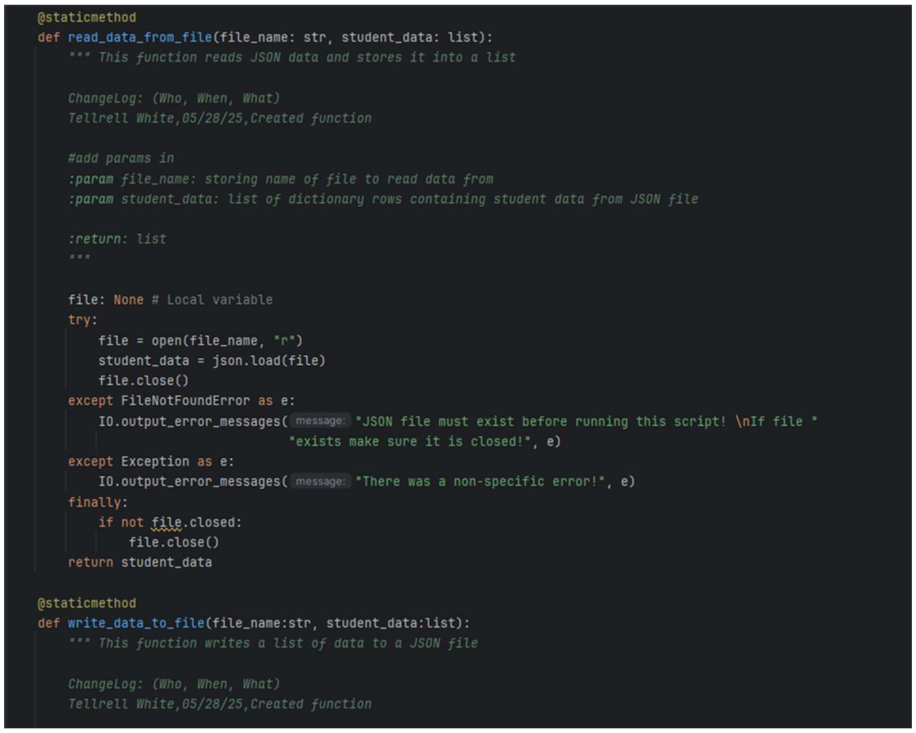
The Python script created for this assignment displayed in Figures 1-7 was developed using the PyCharm IDE, which is an environment that allows programmers to write, debug, and manage Python projects for a host of different applications. The header section displayed at the top of the script shows my name, a short description, and a change log. After the header section, all the "setup code" including importing the "JSON" module used for file operations, defining the constants, data variables, and the two classes, "FileProcessor" and "IO", and their respective functions to be used in the script are shown. Next follows the body of the script which leverages the class functions defined in the "setup" section of the script to accomplish a particular task based on the user's input like what we've seen in past assignments. Functions were developed to accomplish tasks such as reading and writing to files, taking user input, and displaying data to the console for the user. Code from previous assignments was leveraged in the development of these class functions.

The concept of SoC was used in this assignment to separate the program into distinct layers providing scalability, maintainability, and reusability. These layers include the "Processing Layer" responsible for implementing the core functionality of the program including things such as data processing, calculations, and decisions. The other layer used is the "Presentation Layer" for presenting information to the user and capturing user input. The figures show comments within the script that display the different layers used for this program and the classes and their respective functions. To illustrate, the "IO Class", which is used for Presentation layer implementation, contains functions that output data to the console for things such as error handling, and displaying student registration data, and has functions for taking user input for things such as menu selection and inputting student registration data. This is highlighted below in the figures.



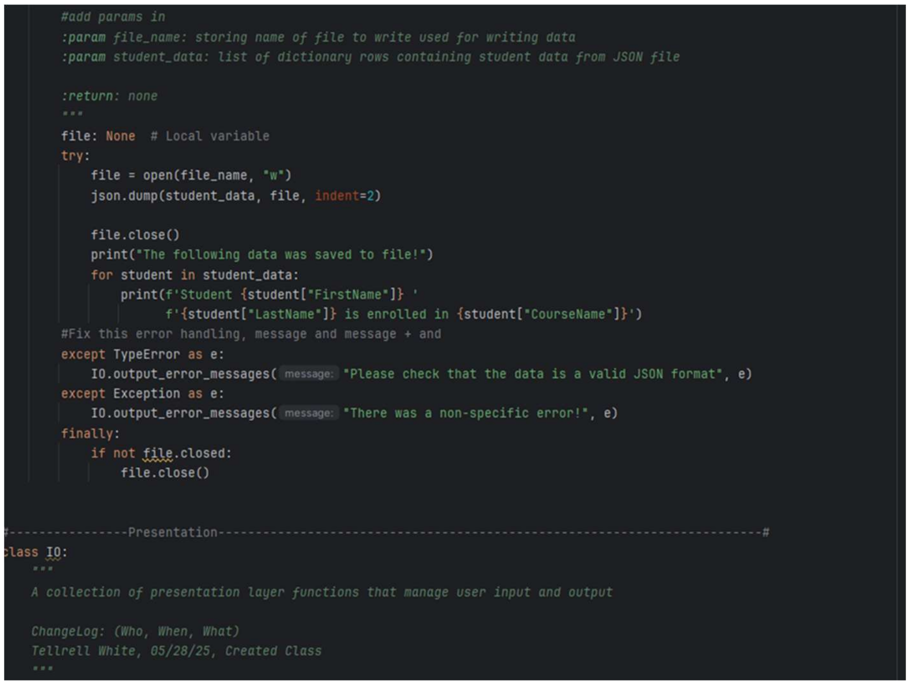
```python
# -----------------------------------------------------------------------------------------#
# Title: Assignment06
# Desc: This assignment demonstrates using functions, classes, and using separation of
# concerns
# Change Log: (Who, When, What)
# Tellrell White,05/28/2025,Created Script
# -----------------------------------------------------------------------------------------#
import json


# Define the Data Constants
MENU: str = '''
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------
'''
# Define the Data Constants
FILE_NAME: str = "Enrollments.json"
students: list = []  # a table of student data
menu_choice: str = ''  # Hold the choice made by the user.


#---------------Processing -------------------------------------------------------------#

class FileProcessor:
    """
    A collection of processing layer functions that work with JSON files

    Changelog: (Who, When, What)
    Tellrell White, 05/28/30, Created Class

    """
```

Figure 1: Python Script in Pycharm



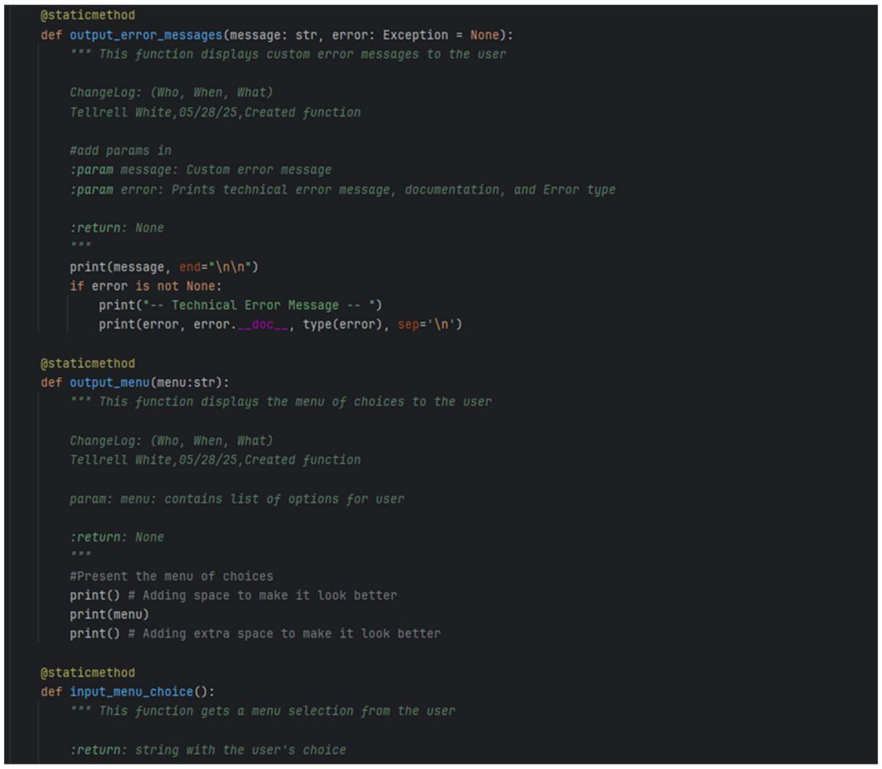
```python
    @staticmethod
    def read_data_from_file(file_name: str, student_data: list):
        """ This function reads JSON data and stores it into a list

        ChangeLog: (Who, When, What)
        Tellrell White,05/28/25,Created function

        #add params in
        :param file_name: storing name of file to read data from
        :param student_data: list of dictionary rows containing student data from JSON file

        :return: list
        """

        file: None # Local variable
        try:
            file = open(file_name, "r")
            student_data = json.load(file)
            file.close()
        except FileNotFoundError as e:
            IO.output_error_messages( message: "JSON file must exist before running this script! \nIf file "
                                      "exists make sure it is closed!", e)
        except Exception as e:
            IO.output_error_messages( message: "There was a non-specific error!", e)
        finally:
            if not file.closed:
                file.close()
        return student_data

    @staticmethod
    def write_data_to_file(file_name:str, student_data:list):
        """ This function writes a list of data to a JSON file

        ChangeLog: (Who, When, What)
        Tellrell White,05/28/25,Created function
```

Figure 2: Python Script in Pycharm



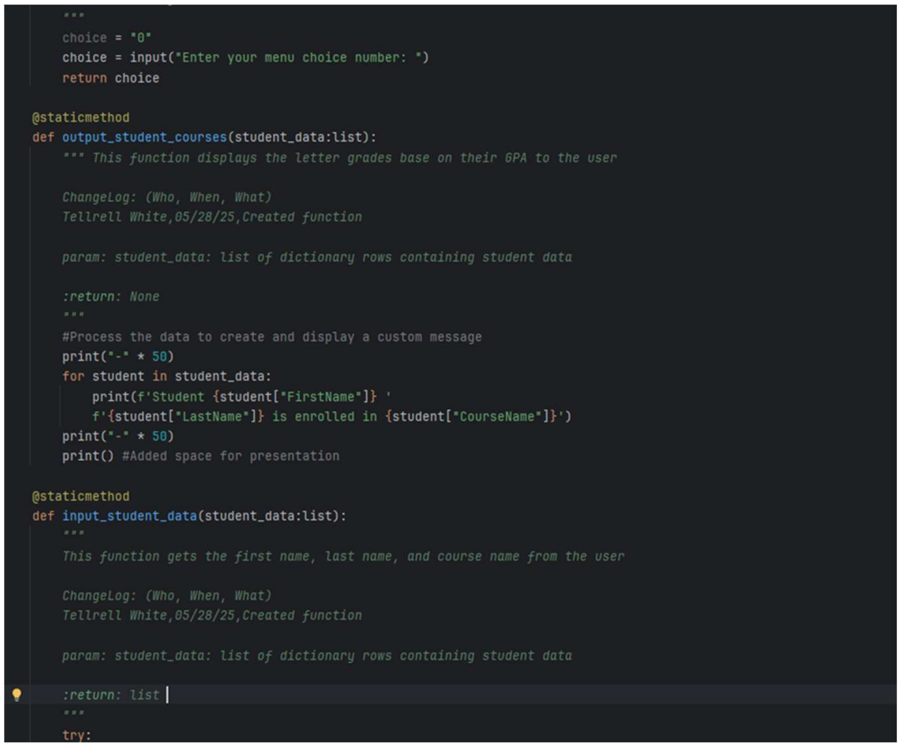
```
        #add params in
        :param file_name: storing name of file to write used for writing data
        :param student_data: list of dictionary rows containing student data from JSON file

        :return: none
        """
        file: None  # Local variable
        try:
            file = open(file_name, "w")
            json.dump(student_data, file, indent=2)

            file.close()
            print("The following data was saved to file!")
            for student in student_data:
                print(f'Student {student["FirstName"]} '
                      f'{student["LastName"]} is enrolled in {student["CourseName"]}')
        #Fix this error handling, message and message + and
        except TypeError as e:
            IO.output_error_messages( message: "Please check that the data is a valid JSON format", e)
        except Exception as e:
            IO.output_error_messages( message: "There was a non-specific error!", e)
        finally:
            if not file.closed:
                file.close()


#---------------Presentation---------------------------------------------------------------------#
class IO:
    """
    A collection of presentation layer functions that manage user input and output

    ChangeLog: (Who, When, What)
    Tellrell White, 05/28/25, Created Class
    """
```

Figure 3: Python Script in Pycharm



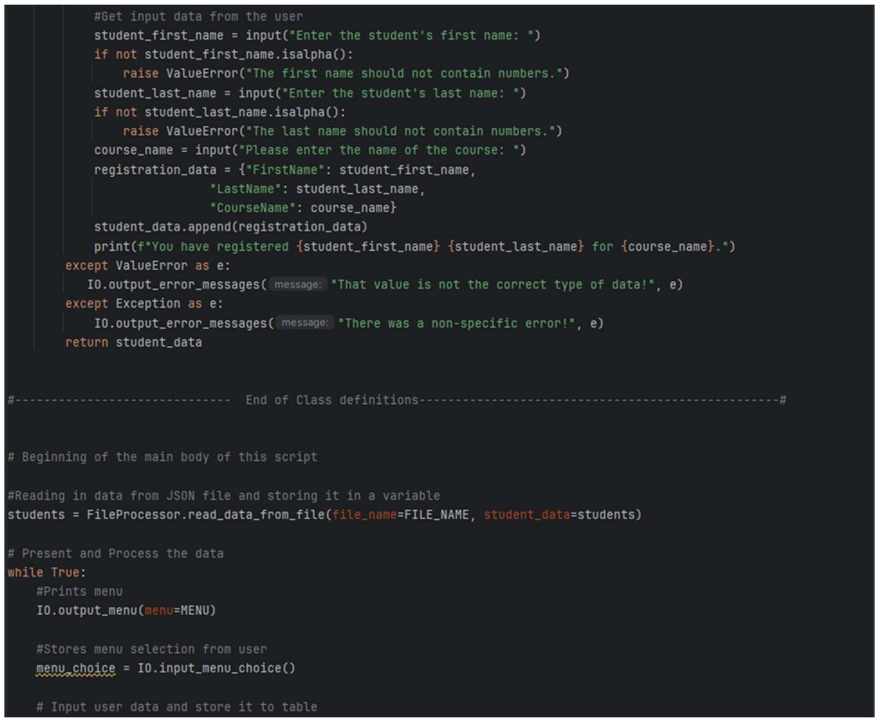
```
    @staticmethod
    def output_error_messages(message: str, error: Exception = None):
        """ This function displays custom error messages to the user

        ChangeLog: (Who, When, What)
        Tellrell White,05/28/25,Created function

        #add params in
        :param message: Custom error message
        :param error: Prints technical error message, documentation, and Error type

        :return: None
        """
        print(message, end="\n\n")
        if error is not None:
            print("-- Technical Error Message -- ")
            print(error, error.__doc__, type(error), sep='\n')

    @staticmethod
    def output_menu(menu:str):
        """ This function displays the menu of choices to the user

        ChangeLog: (Who, When, What)
        Tellrell White,05/28/25,Created function

        param: menu: contains list of options for user

        :return: None
        """
        #Present the menu of choices
        print() # Adding space to make it look better
        print(menu)
        print() # Adding extra space to make it look better

    @staticmethod
    def input_menu_choice():
        """ This function gets a menu selection from the user

        :return: string with the user's choice
```

Figure 4:Python Script in Pycharm



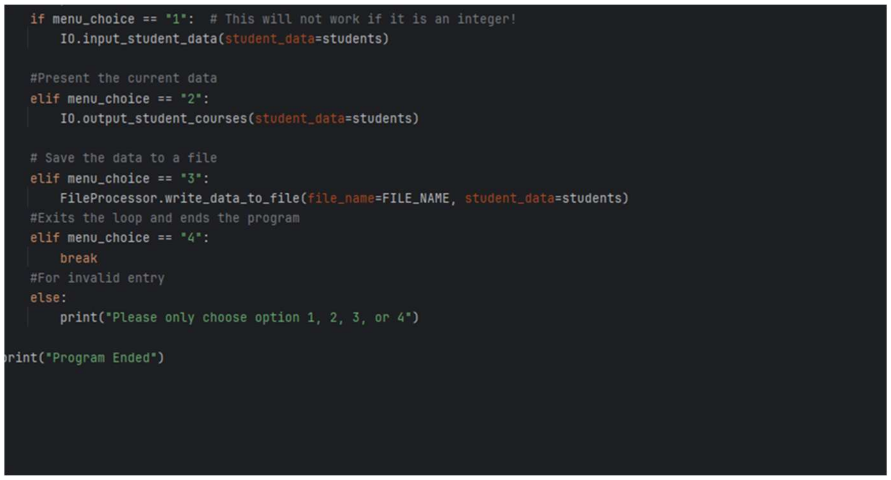
```
        """
        choice = "0"
        choice = input("Enter your menu choice number: ")
        return choice

    @staticmethod
    def output_student_courses(student_data:list):
        """ This function displays the letter grades base on their GPA to the user

        ChangeLog: (Who, When, What)
        Tellrell White,05/28/25,Created function

        param: student_data: list of dictionary rows containing student data

        :return: None
        """
        #Process the data to create and display a custom message
        print("-" * 50)
        for student in student_data:
            print(f'Student {student["FirstName"]} '
            f'{student["LastName"]} is enrolled in {student["CourseName"]}')
        print("-" * 50)
        print() #Added space for presentation

    @staticmethod
    def input_student_data(student_data:list):
        """
        This function gets the first name, last name, and course name from the user

        ChangeLog: (Who, When, What)
        Tellrell White,05/28/25,Created function

        param: student_data: list of dictionary rows containing student data

        :return: list
        """
        try:
```

Figure 5: Python Script in Pycharm



```
            #Get input data from the user
            student_first_name = input("Enter the student's first name: ")
            if not student_first_name.isalpha():
                raise ValueError("The first name should not contain numbers.")
            student_last_name = input("Enter the student's last name: ")
            if not student_last_name.isalpha():
                raise ValueError("The last name should not contain numbers.")
            course_name = input("Please enter the name of the course: ")
            registration_data = {"FirstName": student_first_name,
                                "LastName": student_last_name,
                                "CourseName": course_name}
            student_data.append(registration_data)
            print(f"You have registered {student_first_name} {student_last_name} for {course_name}.")
        except ValueError as e:
           IO.output_error_messages( message: "That value is not the correct type of data!", e)
        except Exception as e:
            IO.output_error_messages( message: "There was a non-specific error!", e)
        return student_data


#-----------------------------  End of Class definitions--------------------------------------------------#


# Beginning of the main body of this script

#Reading in data from JSON file and storing it in a variable
students = FileProcessor.read_data_from_file(file_name=FILE_NAME, student_data=students)

# Present and Process the data
while True:
    #Prints menu
    IO.output_menu(menu=MENU)

    #Stores menu selection from user
    menu_choice = IO.input_menu_choice()

    # Input user data and store it to table
```

Figure 6: Python Script in Pycharm



```
    if menu_choice == "1":  # This will not work if it is an integer!
        IO.input_student_data(student_data=students)

    #Present the current data
    elif menu_choice == "2":
        IO.output_student_courses(student_data=students)

    # Save the data to a file
    elif menu_choice == "3":
        FileProcessor.write_data_to_file(file_name=FILE_NAME, student_data=students)
    #Exits the loop and ends the program
    elif menu_choice == "4":
        break
    #For invalid entry
    else:
        print("Please only choose option 1, 2, 3, or 4")

rint("Program Ended")
```

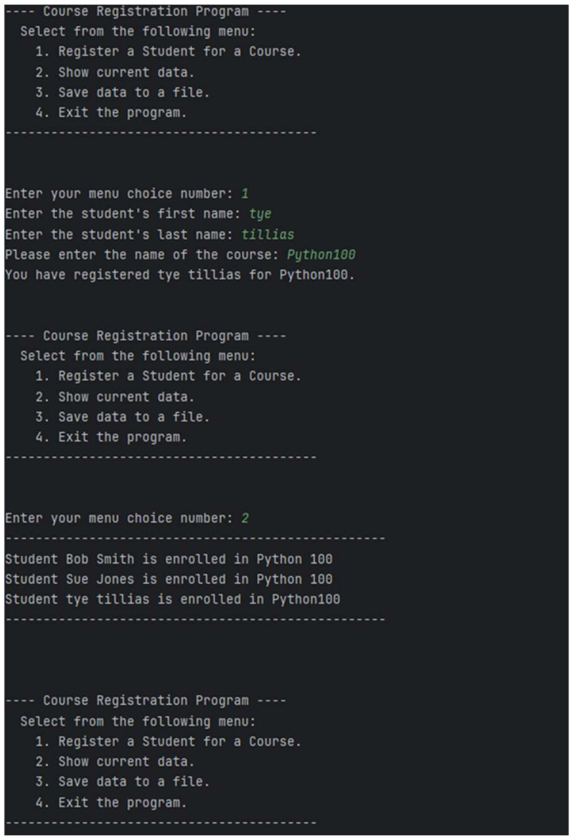
Figure 7: Python Script in Pycharm

Testing of Python Script

To validate that the script developed and shown in the figures above worked as intended, two methods were used to test the behavior of the script. Within the Pycharm IDE, I ran the "Assignment06.py script to not only ensure that there were no errors and that the script would run, but also to verify behavior of the script was correct. Figures 8, 9, and 10 show the output from running the "Assignment06.py" script in Pycharm. As shown in the figures, the user inputs a value matching a menu choice, and the conditional statement matching the user's input will be executed. As shown in the figure, the user first selects "menu choice 1", which prompts the user for a first name, last name and course information. During the second loop or iteration, the user is shown the menu again, and user selects "option 2", which displays the registration information contained within the list "students".

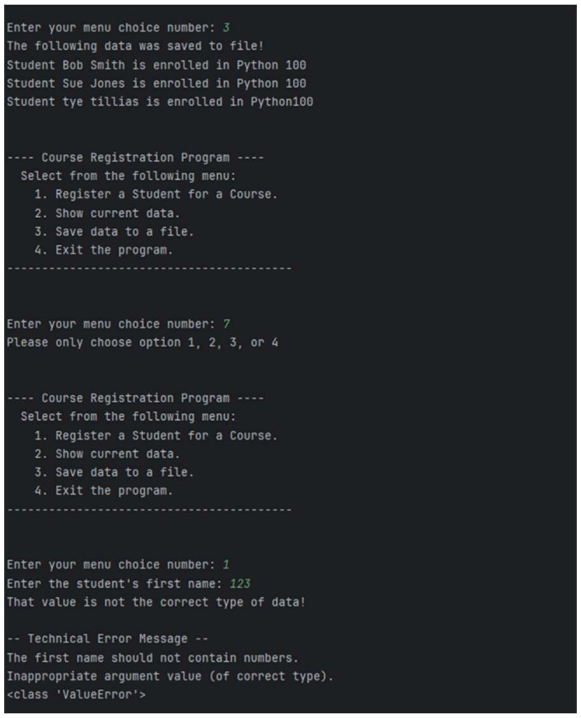
Next, the user selects "option 3", which saves the user's registration information to a JSON file and prints the student registration information written to the screen. The contents of the original JSON file, "Enrollments.json" is shown in Figure 11 and the modified JSON file containing all the data written to the file from the "students" variable in shown in Figure 12. During the next loop the menu is presented, and the user enters a value of "7", which isn't a valid choice, and the user is shown the message "Enter a valid value (1-4)" and the program restarts the loop. Next, the user selects menu choice 1 and inputs "123" when prompted for a first name. Eror handling is demonstrated by printing a message telling the user that the first name cannot contain numbers along with additional information. Next, when prompted for a menu selection, the user enters the value of “4” and this ends the program.

Another method of testing the script was through using the command terminal in windows. After launching the command terminal and navigating to the correct directory where the script exists, I launched the python interpreter using the “Python” command and provided the name of the script as an argument to run it. The output of running the script is shown in Figures 13 and 14. Like the results obtained from Pycharm, when selecting menu choices 1 through 4, the program displayed correct results.



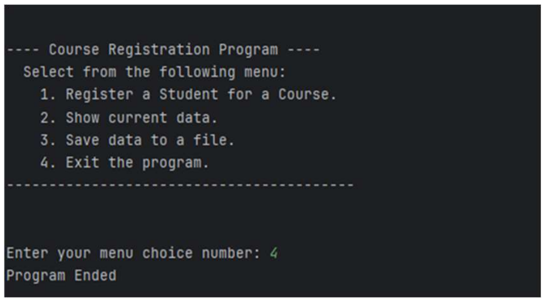
```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------



Enter your menu choice number: 1
Enter the student's first name: tye
Enter the student's last name: tillias
Please enter the name of the course: Python100
You have registered tye tillias for Python100.


---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------



Enter your menu choice number: 2
--------------------------------------------------
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student tye tillias is enrolled in Python100
--------------------------------------------------




---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------
```

Figure 8: Output from Pycharm



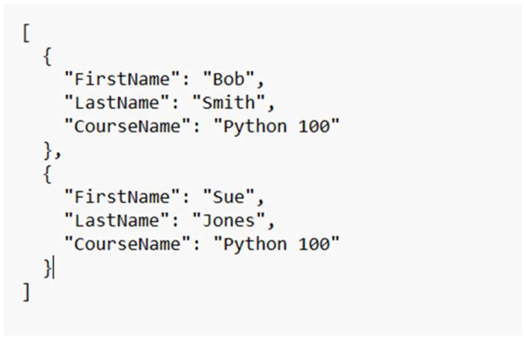
```
Enter your menu choice number: 3
The following data was saved to file!
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student tye tillias is enrolled in Python100


---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------


Enter your menu choice number: 7
Please only choose option 1, 2, 3, or 4


---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------


Enter your menu choice number: 1
Enter the student's first name: 123
That value is not the correct type of data!

-- Technical Error Message --
The first name should not contain numbers.
Inappropriate argument value (of correct type).
<class 'ValueError'>
```

Figure 9: Output from Pycharm



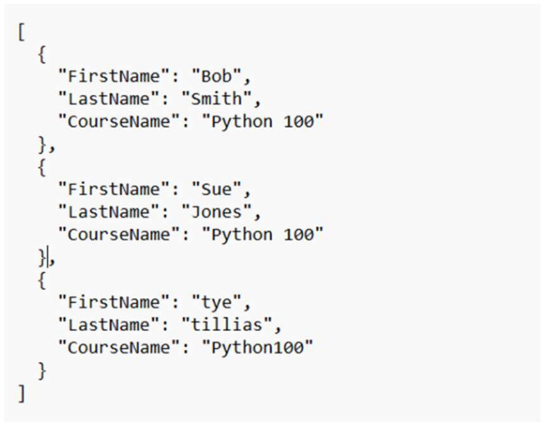
```

---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------


Enter your menu choice number: 4
Program Ended
```

Figure 10: Output from Pycharm



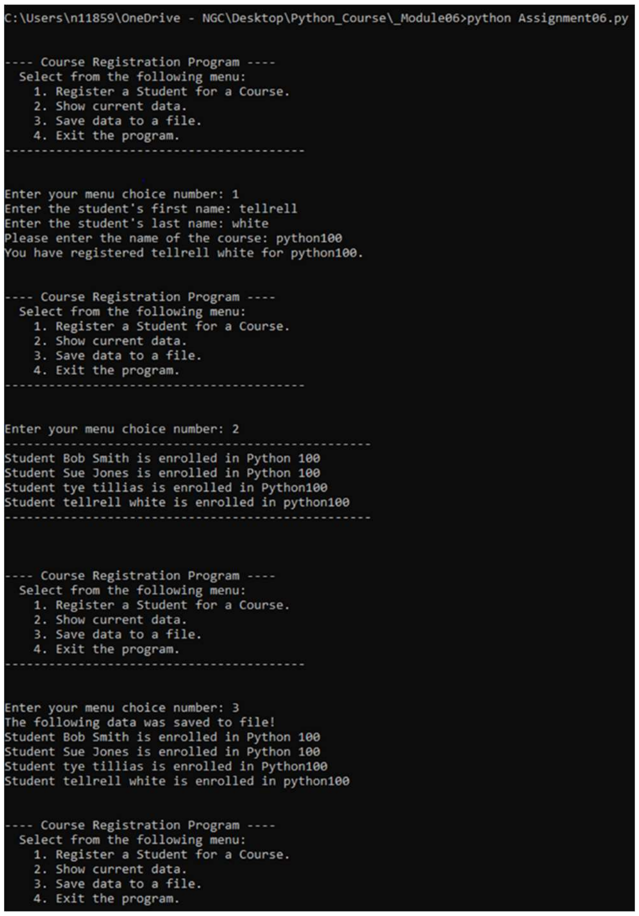
```
[
  {
    "FirstName": "Bob",
    "LastName": "Smith",
    "CourseName": "Python 100"
  },
  {
    "FirstName": "Sue",
    "LastName": "Jones",
    "CourseName": "Python 100"
  }
]
```

Figure 11: Original JSON File



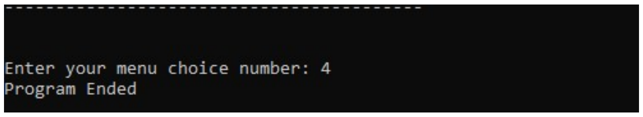
```
[
  {
    "FirstName": "Bob",
    "LastName": "Smith",
    "CourseName": "Python 100"
  },
  {
    "FirstName": "Sue",
    "LastName": "Jones",
    "CourseName": "Python 100"
  },
  {
    "FirstName": "tye",
    "LastName": "tillias",
    "CourseName": "Python100"
  }
]
```

Figure 12: Modified JSON File



```
C:\Users\n11859\OneDrive - NGC\Desktop\Python_Course\_Module06>python Assignment06.py

---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------


Enter your menu choice number: 1
Enter the student's first name: tellrell
Enter the student's last name: white
Please enter the name of the course: python100
You have registered tellrell white for python100.

---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------


Enter your menu choice number: 2
--------------------------------------------------
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student tye tillias is enrolled in Python100
Student tellrell white is enrolled in python100
--------------------------------------------------



---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------


Enter your menu choice number: 3
The following data was saved to file!
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student tye tillias is enrolled in Python100
Student tellrell white is enrolled in python100

---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
```

Figure 13: Windows Terminal Output



```
-----------------------------------------


Enter your menu choice number: 4
Program Ended
```

Figure 14: Windows Terminal Output

Summary

As discussed in this document, a Python script was developed utilizing the concepts of functions, classes and SoC learned in module 6 to display to a user information about registration into a Python course. This script was tested using Pycharm, the terminal in windows, and examination of a JSON file for correct behavior.