Paulo Roberto Barreto Alves

08/20/2025

Foundations of Python Programming

Assignment 06

https://github.com/pagh59258/IntroToProg-Python-Mod06

Functions

Introduction

On module 06, I acquired knowledge on some Python programming tools and techniques. With this new knowledge, I created a Python script very similar to the one I have developed in Module 05, but adding some extra techniques, such as: **the use of classes, functions and SoC (Separation of Concerns Pattern).**

Reading Python 100 Module 06 notes.doc file

- Organizing your code
  - Functions
- Global vs local variables
- Parameters
  - Using parameters
  - Default parameter values
  - Overloaded functions
- Return values
- Classes and functions
  - Static classes
  - Document strings (Docstrings)
- The Separation of Concerns Pattern
  - Organizing your code based on concerns
  - Using the Separations of Concerns Pattern

Performing Module 06 labs

Throughout the module, I was asked to apply the learned knowledge into some practice labs, such as:

- Mod06-Lab01-Working with Functions
- Mod06-Lab02-Working with Parameters
- Mod06-Lab03-Working with Classes and SoC

Creating a Python script

I created a new file named Assignment06.py, which as the extension indicates, is a Python script. This Python script demonstrates using constants, variables, and print statements to display a message about a student's registration for a course. This program is very similar to Assignment05, but it adds the use of Classes, Functions and Separation of Concerns (SoC).

Here are some notes about this Python script:

- To avoid errors, we need to have some data already populated on the "Enrollments.json" file before starting script execution.
- SoC (Separation of Concerns) technique was implementing these 2 layers:
    - Data Layer
    - Presentation Layer
- On the Data Layer, the following Classes and functions were defined:
    - Data Layer
        - FileProcessor class
            - read_data_from_file function
            - write_data_to_file function
    - Presentation Layer
        - IO class
            - output_error_messages function
            - output_menu function
            - input_menu_choice function
            - input_student_data function
            - output_current_student_data function
            - output_check_unsaved_student_data function
- When the program starts, the function “FileProcessor.read_data_from_file” is invoked to extract data from the "Enrollments.json" file and return it on the students list of dictionaries.
- A “While” loop is started
    - Function “IO.input_menu_choice” is invoked to display a menu with 4 options
        - Only accept options are 1, 2, 3 or 4. If any other option is entered, an error is shown and menu is displayed again;
        - Regarding Menu Option “1”:
            - “IO.input_student_data” function is invoked
                - Function does a critique to not allow invalid fields:
                    - Student’s first name cannot be numeric or blank;
                    - Students’ last name cannot be numeric or blank;
                    - Course name cannot be blank.
            - Users are allowed to make several enrollments;
- Regarding Menu Option “2”:
    - Function “IO.output_current_student_data” is invoked

        - It displays current student registrations (including the ones not yet saved to the json file)
- Regarding Menu Option “3”:
    - Function “FileProcessor.write_data_to_file“ is invoked
        - Writes current student registrations to json file
        - Display list of student registrations saved
- Regarding Menu Option “4”:
    - Function “IO.output_check_unsaved_student_data“ is invoked
        - It checks if there is unsaved data and offer the user the option to save it or not;
        - Program is terminated

On Figure 1 below, we can see the initial Enrollment.json file (provided inside assignment zip file), must contain some data:

Figure 1: Enrollment.json initial file

On Figure 2 below, we can see the basic coding used for Script header, Import and constants & variables definition:

Assignment06.py

```python
# ------------------------------------------------------------------------ #
# Title: Assignment06
# Desc: This assignment demonstrates using Classes and Functions
# Change Log: (Who, When, What)
#   PAlves,8/20/2025,Created Script
# ------------------------------------------------------------------------ #

# ------------------------------------------------------------------------ #
# ------------------------- Imports ------------------------------------- #
# ------------------------------------------------------------------------ #
import json
from io import TextIOWrapper


# ------------------------------------------------------------------------ #
# ------------------ 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
----------------------------------------
'''

FILE_NAME: str = "Enrollments.json"


# ------------------------------------------------------------------------ #
# ---------------------- Define the Global Data Variables ---------------- #
# ------------------------------------------------------------------------ #
menu_choice: str  # Hold the choice made by the user.
students: list = []  # a table of student data


```

Figure 2: Assignment06.py Python script header, imports, constants and variables

On Figure 3 below, we can see the Data Layer – FileProcessor class – read_data_from_file function:

Assignment06.py

```python

# ------------------------------------------------------------------------ #
# ------------------------- Data Layer ----------------------------------- #
# ------------------------------------------------------------------------ #

# --------------------- FileProcessor class  ------------------------------ #
class FileProcessor:  4 usages
    """
    A collection of processing layer functions that work with Json files

    ChangeLog: (Who, When, What)
    PAlves, 8/20/2025,Created Class

    """

    # ---------------------- read_data_from_file function ------------------- #
    @staticmethod  2 usages
    def read_data_from_file(file_name: str, student_data: list):
        """ This function read data from the JSON file into student_data list

        ChangeLog: (Who, When, What)
        PAlves, 8/20/2025,Created function

        :return: None
        """

        try:
            file: TextIOWrapper = open(file_name, "r")
            student_data = json.load(file)
            file.close()

        except FileNotFoundError as e:
            IO.output_error_messages( message: "Text file must exist before\
             running this script!", 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

```

Figure 3: PyCharm – Assignment06.py Python read_data_from_file function

On Figure 4 below, we can see the Data Layer – FileProcessor class – write_data_to_file function:

```
class FileProcessor:  4 usages
        return student_data


    # ---------------------- write_data_to_file function ---------------------- #
    @staticmethod  2 usages
    def write_data_to_file(file_name: str, student_data: list):
        """ This function writes data onto the JSON file

        ChangeLog: (Who, When, What)
        PAlves, 8/20/2025,Created function

        :return: None
        """

        try:
            file: TextIOWrapper = open(file_name, "w")
            json.dump(student_data, file, indent=2)
            file.close()
            print("-" * 65)
            print("The following data was saved to file:")
            print("-" * 65)
            for student in student_data:
                print(f"{student['FirstName']} {student['LastName']} ,"
                      f"is enrolled in {student['CourseName']}.")
            print("-" * 65)

        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()
```

Figure 4: PyCharm – Assignment06.py Python write_data_to_file function

On Figure 5 below, we can see the Presentation Layer – IO class – output_error_messages function:

Assignment06.py

```
118
119     # ------------------------------------------------------------------------ #
120     # --------------------- Presentation Layer --------------------------------- #
121     # ------------------------------------------------------------------------ #
122
123     # ------------------------- IO class  ------------------------------------- #
124     class IO:  16 usages
125         """
126         A collection of presentation layer functions that manage user
127         input and output
128
129         ChangeLog: (Who, When, What)
130         PAlves, 8/20/2025,Created Class
131         """
132
133         # ------------------ output_error_messages function --------------------- #
134         @staticmethod  11 usages
135         def output_error_messages(message: str, error: Exception = None):
136             """ This function displays a custom error messages to the user
137
138             ChangeLog: (Who, When, What)
139             PAlves, 8/20/2025,Created function
140
141             :return: None
142             """
143             print("-" * 65)
144             print(message, end="\n\n")
145             print("-" * 65)
146             if error is not None:
147                 print("-" * 65)
148                 print("-- Technical Error Message -- ")
149                 print(error, error.__doc__, type(error), sep='\n')
150                 print("-" * 65)
151
```

Figure 5: PyCharm – Assignment06.py Python output_error_messages function

On Figure 6 below, we can see the output_menu function:

```python

    # ---------------------- output_menu function ---------------------- #
    @staticmethod  1 usage
    def output_menu(menu: str):
        """ This function displays the menu of choices to the user

        ChangeLog: (Who, When, What)
        PAlves, 8/20/2025,Created function

        :return: None
        """
        print()
        print(menu)
        print()  # Adding extra space to make it look nicer.

```

Figure 6: PyCharm – Assignment06.py Python output_menu function

On Figure 7 below, we can see the input_menu_choice function:

```python

    # ------------------- input_menu_choice function ---------------------- #
    @staticmethod  1 usage
    def input_menu_choice():
        """ This function gets a menu choice from the user

        ChangeLog: (Who, When, What)
        PAlves, 8/20/2025,Created function

        :return: string with the users choice
        """
        choice = "0"
        try:
            choice = input("Enter your menu choice number: ")
            if choice not in ("1","2","3","4"):  # Note these are strings
                raise Exception("Error: Please, choose only 1, 2, 3, or 4")

        except Exception as e:
            IO.output_error_messages(e.__str__())

        return choice

```

Figure 7: PyCharm – Assignment06.py Python script input_menu_choice function

On Figure 8 below, we can see the input_student_data function:

```python
    # ---------------- input_student_data function ------------------------ #
    @staticmethod  1 usage
    def input_student_data(student_data: list):
        """ This function gets the first name, last name, and Course Name

        ChangeLog: (Who, When, What)
        PAlves, 8/20/2025,Created function

        :return: None
        """

        try:
            # Input the data
            student_first_name = input("What is the student's first name? ")
            if not student_first_name.isalpha():
                raise ValueError("First name cannot contain numbers or be blank.")

            student_last_name = input("What is the student's last name? ")
            if not student_last_name.isalpha():
                raise ValueError("Last name cannot contain numbers or be blank.")

            course_name = input("What is the course name? ")
            if course_name =="":
                raise ValueError("Course name should not be blank.")

            student = {"FirstName": student_first_name,
                       "LastName": student_last_name,
                       "CourseName": course_name}
            student_data.append(student)

            print("-"* 65)
            print("List of students currently registered for courses:")
            print("-"* 65)
            for student in students:
                print(f"{student['FirstName']} {student['LastName']} "\
                      f"is enrolled in {student['CourseName']}.")
            print("-"* 65)
            print("IMPORTANT")
            print("-Some of these registrations might not be yet saved to file")
            print("-Make sure you use save registrations to file before exit")
            print("-"* 65)

        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
```

Figure 8: PyCharm – Assignment06.py Python script input_student_data function

On Figure 9 below, we can see the output_current_student_data function:

```
238
239     # ---------- output_current_student_data function -------------------- #
240     @staticmethod  1 usage
241     def output_current_student_data(student_data: list):
242         """ This function Displays the current student data
243
244         ChangeLog: (Who, When, What)
245         PAlves, 8/20/2025,Created function
246
247         :return: None
248         """
249
250         try:
251             print("-" * 65)
252             print("List of students currently registered for courses:")
253             print("-" * 65)
254             for student in student_data:
255                 print(f"{student['FirstName']} {student['LastName']} "\
256                       f"is enrolled in {student['CourseName']}.")
257             print("-" * 65)
258             print("IMPORTANT")
259             print("- Some of these registrations might not be yet saved")
260             print("- Make sure you use save registrations before exit")
261             print("-" * 65)
262
263
264         except ValueError as e:
265             IO.output_error_messages(e)
266
267         except Exception as e:
268             IO.output_error_messages( message: "There was a\
269              non-specific error!", e)
270
```

Figure 9: PyCharm – Assignment06.py Python script output_current_student_data function

On Figure 10 below, we can see the output_check_unsaved_student_data function:

```python
    # ------------- output_check_unsaved_student_data function -------------- #
    @staticmethod  1 usage
    def output_check_unsaved_student_data(student_data: list):
        """ This function Checks if there are any unsaved student data

        ChangeLog: (Who, When, What)
        PAlves, 8/20/2025,Created function

        :return: None
        """

        rec_on_file: list = []  # a table of student data already saved on JSON file

        try:
            rec_on_file = FileProcessor.read_data_from_file(file_name=FILE_NAME\
                                                            ,student_data=rec_on_file)

            if (len(student_data) != len(rec_on_file)):
                print("-" * 65)
                print("Warning: There are registrations not yet saved.")
                print("-" * 65)
                pend_save = input("Do you want to save the data? (y/n): ")
                if (pend_save == "y"):
                    # Invoke FileProcessor.write_data_to_file function to save data
                    FileProcessor.write_data_to_file(file_name=FILE_NAME,\
                                                     student_data=students)
                    print("-" * 65)
                    print("Unsaved data was written to JSON file!")
                    print("-" * 65)

                else:
                    print("-" * 65)
                    print("Pending data were not saved to file!")
                    print("-" * 65)

        except ValueError as e:
            IO.output_error_messages(e)

        except Exception as e:
            IO.output_error_messages( message: "There was a non-specific error!", e)
```

Figure 10: PyCharm – Assignment06.py Python script output_check_unsaved_student_data function

On Figure 11 below, we can see the Python script main body:

```python

# ------------------------------------------------------------------------ #
# ----------------- Script Main body ------------------------------------- #
# ------------------------------------------------------------------------ #

# When the program starts:
#      Read from the Json file to extract data
#      Load extracted/read data into student_data list of lists (table)

students = FileProcessor.read_data_from_file(file_name=FILE_NAME, \
                                             student_data=students)

# ------------ Infinite loop until menu_option 4 is chosen ----------------- #
while True:
    IO.output_menu(menu=MENU)
    menu_choice = IO.input_menu_choice()

    # ---------- menu_option 1 display current student data --------------- #
    if menu_choice == "1":
        students = IO.input_student_data(student_data=students)
        continue

    # ---------- menu_option 2 get new student data ---------------------- #
    elif menu_choice == "2":
        IO.output_current_student_data(student_data=students)
        continue

    # ---------- menu_option 3 save student data to JSON file ------------ #
    elif menu_choice == "3":
        FileProcessor.write_data_to_file(file_name=FILE_NAME, \
                                         student_data=students)
        continue

    # - menu_option 4 check unsaved data and break loop to finish script -- #
    elif menu_choice == "4":
        IO.output_check_unsaved_student_data(student_data=students)
        break

# ------------------------------------------------------------------------ #
# -------------------- End of script  ------------------------------------ #
# ------------------------------------------------------------------------ #
```

Figure 11: PyCharm – Assignment06.py Python script main body

Then I executed the script via PyCharm and via Windows command shell.

Executing script on PyCharm

Figure 12 shown below displays the Assignment06.py Python script menu using PyCharm.

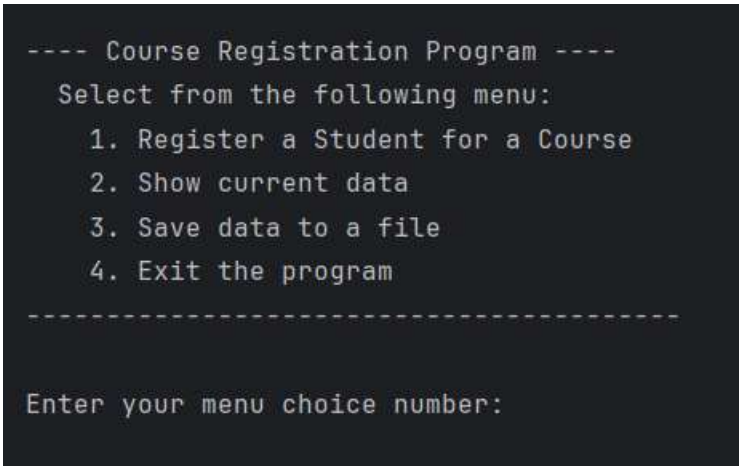

Figure 12: PyCharm – Assignment06.py script menu

Figure 13 shown below displays the critique if an invalid menu option is chosen (either alphabetic options or integers different from 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:
------------------------------------------------------------------
Error: Please, choose only 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: a
------------------------------------------------------------------
Error: Please, choose only 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: 5
------------------------------------------------------------------
Error: Please, choose only 1, 2, 3, or 4

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

Figure 13: PyCharm – Assignment06.py script menu invalid option critique

Figure 14 below shows error validation on Student's First and Last name and Course name invalid entries:

```
---- 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
What is the student's first name?
-------------------------------------------------------------------
That value is not the correct type of data!

-------------------------------------------------------------------
-------------------------------------------------------------------
-- Technical Error Message --
First name cannot contain numbers or be blank.
Inappropriate argument value (of correct type).
<class 'ValueError'>
-------------------------------------------------------------------


---- 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
What is the student's first name? 123
-------------------------------------------------------------------
That value is not the correct type of data!

-------------------------------------------------------------------
-------------------------------------------------------------------
-- Technical Error Message --
First name cannot contain numbers or be blank.
Inappropriate argument value (of correct type).
<class 'ValueError'>
-------------------------------------------------------------------
```

```
Enter your menu choice number: 1
What is the student's first name? Vic
What is the student's last name?
-------------------------------------------------------------------
That value is not the correct type of data!

-------------------------------------------------------------------
-------------------------------------------------------------------
-- Technical Error Message --
Last name cannot contain numbers or be blank.
Inappropriate argument value (of correct type).
<class 'ValueError'>
-------------------------------------------------------------------


---- 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
What is the student's first name? Vic
What is the student's last name? 123
-------------------------------------------------------------------
That value is not the correct type of data!

-------------------------------------------------------------------
-------------------------------------------------------------------
-- Technical Error Message --
Last name cannot contain numbers or be blank.
Inappropriate argument value (of correct type).
<class 'ValueError'>
-------------------------------------------------------------------
```

```
---- 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
What is the student's first name? Vic
What is the student's last name? Vu
What is the course name?
------------------------------------------------------------------
That value is not the correct type of data!

------------------------------------------------------------------
------------------------------------------------------------------
-- Technical Error Message --
Course name should not be blank.
Inappropriate argument value (of correct type).
<class 'ValueError'>
------------------------------------------------------------------
```

Figure 14: PyCharm – Assignment06.py script validation on Student's First and Last name and Course name invalid entries

Figure 15 below shows a couple of valid registrations made via Menu Option 1:

```
---- 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
What is the student's first name? Vic
What is the student's last name? Vu
What is the course name? Java 404
-----------------------------------------------------------------
List of students currently registered for courses:
-----------------------------------------------------------------
Bob Smith is enrolled in Python 100.
Sue Jones is enrolled in Python 100.
Vic Vu is enrolled in Java 404.
-----------------------------------------------------------------
IMPORTANT
-Some of these registrations might not be yet saved to file
-Make sure you use save registrations to file before exit
-----------------------------------------------------------------


---- 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
What is the student's first name? John
What is the student's last name? Doe
What is the course name? Law 785
-----------------------------------------------------------------
List of students currently registered for courses:
-----------------------------------------------------------------
Bob Smith is enrolled in Python 100.
Sue Jones is enrolled in Python 100.
Vic Vu is enrolled in Java 404.
John Doe is enrolled in Law 785.
-----------------------------------------------------------------
IMPORTANT
-Some of these registrations might not be yet saved to file
-Make sure you use save registrations to file before exit
-----------------------------------------------------------------
```

Figure 15: PyCharm – Assignment06.py script shows a couple of valid registrations via menu option 1

Figure 16 shown below displays the output of Menu option 2, when the current data is shown:

```
---- 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
------------------------------------------------------------------
List of students currently registered for courses:
------------------------------------------------------------------
Bob Smith is enrolled in Python 100.
Sue Jones is enrolled in Python 100.
Vic Vu is enrolled in Java 404.
John Doe is enrolled in Law 785.
------------------------------------------------------------------
IMPORTANT
- Some of these registrations might not be yet saved
- Make sure you use save registrations before exit
------------------------------------------------------------------
```

Figure 16: PyCharm – Assignment06.py script menu option 2 output showing current data

Figure 17 shown below displays the output of Menu option 3, when data is saved to JSON file and displayed:

```
---- 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:
------------------------------------------------------------------
Bob Smith ,is enrolled in Python 100.
Sue Jones ,is enrolled in Python 100.
Vic Vu ,is enrolled in Java 404.
John Doe ,is enrolled in Law 785.
------------------------------------------------------------------
```

Figure 17: PyCharm – Assignment06.py script menu options 3 output

Figure 18 shown below displays the menu options 4 (when there is pending data to be saved to file):

```
---- 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
What is the student's first name? John
What is the student's last name? Meyer
What is the course name? Singing 245
---------------------------------------------------------------------
List of students currently registered for courses:
---------------------------------------------------------------------
Bob Smith is enrolled in Python 100.
Sue Jones is enrolled in Python 100.
Vic Vu is enrolled in Java 404.
John Doe is enrolled in Law 785.
John Meyer is enrolled in Singing 245.
---------------------------------------------------------------------
IMPORTANT
-Some of these registrations might not be yet saved to file
-Make sure you use save registrations to file before exit
---------------------------------------------------------------------


---- 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
---------------------------------------------------------------------
Warning: There are registrations not yet saved.
---------------------------------------------------------------------
Do you want to save the data? (y/n): y
---------------------------------------------------------------------
The following data was saved to file:
---------------------------------------------------------------------
Bob Smith ,is enrolled in Python 100.
Sue Jones ,is enrolled in Python 100.
Vic Vu ,is enrolled in Java 404.
John Doe ,is enrolled in Law 785.
John Meyer ,is enrolled in Singing 245.
---------------------------------------------------------------------
---------------------------------------------------------------------
Unsaved data was written to JSON file!
```

```
Process finished with exit code 0
```

Figure 18: PyCharm – Assignment06.py script menu option 4 output (when there is pending data)

Figure 19 shown below displays the menu option 4, which exits the program (and no pending enrollment):

```
---- 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
-------------------------------------------------------------------
List of students currently registered for courses:
-------------------------------------------------------------------
Bob Smith is enrolled in Python 100.
Sue Jones is enrolled in Python 100.
Vic Vu is enrolled in Java 404.
John Doe is enrolled in Law 785.
John Meyer is enrolled in Singing 245.
-------------------------------------------------------------------
IMPORTANT
- Some of these registrations might not be yet saved
- Make sure you use save registrations before exit
-------------------------------------------------------------------


---- 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

Process finished with exit code 0
```

Figure 19: PyCharm – Assignment06.py script menu option 4 with no pending enrollment

Figure 20 shown below displays the final content of the Enrollments.json file:

Figure 20: PyCharm – Assignment06.py script – final content on Enrollment.json file

Executing script on Windows command shell (cmd)

Figure 21 shown below, displays the successful execution of Assignment06.py Python script using Windows command shell (cmd).

Figure 21: Windows command shell (CMD) - Execution of Assignment06.py script

Summary

The creation and execution of this third Python script was a great way to enhance my Python programming using some of the new knowledge I learned on Module 06, including **the use of Classes, Functions and SoC (Separation of Concerns Pattern).**