Surafel Tebeje

05/30/2025

Foundations Of Programming: Python

Assignment06

https://github.com/Surafel-01/IntroToProg-Python

Program organization using functions, classes, and SoC

Introduction:

Organizing a program using **functions**, **classes**, and the **Separation of Concerns (SoC)** principle in Python promotes clean, maintainable, and scalable code. Functions help break down tasks into smaller, reusable pieces, while classes enable encapsulation and the modeling of real-world entities. Applying SoC ensures each component or module of the program handles a distinct responsibility, such as input handling, data processing, or output generation, reducing complexity and making the code easier to test and modify. Together, these practices form the foundation of structured and modular Python programming (ChatGPT, May 2025). In this paper, I will try to demonstrate how to organize a program using Functions, Classes, and SoC.

Assignment instructions

Acceptance Criteria

Your program must include the following features and code to be accepted as complete:

File Name:

- The file is named Assignment06.py

Script Header:

1. The script header includes this text and has been updated with your name and the current date.

Constants:

- The constant **MENU: str** is set to the value:

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

- The constant **FILE_NAME: str** is set to the value "Enrollments.json"
- The constant values do not change throughout the program.

Variables:

- **menu_choice: str** is set to empty string.
- **students: list** : list is set to and empty list

Classes:

- The program includes a class named FileProcessor.
- The program includes a class named IO.
- All classes include descriptive document strings.

Functions:

- All functions include descriptive document strings.
- All functions with except blocks include calls to the function handling error messages.
- All functions use the @staticmethod decorator.
- The program includes functions with the following names and parameters:
    - output_error_messages(message: str, error: Exception = None)
    - output_menu(menu: str)
    - input_menu_choice()
    - output_student_courses(student_data: list)
    - input_student_data(student_data: list)
    - read_data_from_file(file_name: str, student_data: list):
    - write_data_to_file(file_name: str, student_data: list):

Input / Output:

1. On menu choice 1, the program prompts the user to enter the student's first name and last name, followed by the course name, using the input() function and stores the inputs in the respective variables.
2. Data collected for menu choice 1 is added to the **students** two-dimensional list of dictionaries rows.

- On menu choice 2, the program uses the print() function to show a string of comma-separated values for each row collected in the **students** variable.

Processing

- When the program starts, the contents of the "Enrollments.json" are automatically read into the **students** two-dimensional list of dictionary rows using the json.load() function. (**Tip:** Make sure to put some starting data into the file or you will get an error!)
- On menu choice 3, the program opens a file named "Enrollments.json" in write mode using the open() function. It writes the contents of the **students** variable to the file using the json.dump() function. Next the file is closed using the close() method. Finally, the program displays what was written to the file using the **students** variable.
- On menu choice 4, the program ends.

Error Handling

- The program provides structured error handling when the file is read into the list of dictionary rows.
- The program provides structured error handling when the user enters a first name.
- The program provides structured error handling when the user enters a last name.
- The program provides structured error handling when the dictionary rows are written to the file.

Test:

- The program takes the user's input for a student's first, last name, and course name.
- The program displays the user's input for a student's first, last name, and course name.
- The program saves the user's input for a student's first, last name, and course name to a JSON file. (check this in PyCharm or with a simple text editor like Notepad or TextEdit.)
- The program allows users to enter multiple registrations (first name, last name, course name).
- The program allows users to display multiple registrations (first name, last name, course name).
- The program allows users to save multiple registrations to a file (first name, last name, course name).
- The program runs correctly in both **PyCharm and** from the **console or terminal**.

Source Control:

- The script file and the knowledge document are hosted on a GitHub repository.
- A link to the repository is included in the knowledge document.
- A link to the repository is included in the GitHub links forum.

NOTE: The process and code needed to complete this assignment task is very similar to Modul06-Lab03!

The following syntax code was used in PyCharm and Console:

```
# ------------------------------------------------ #
# Title: A06 - Working with Functions, Classes and SoC
# Description: Demonstrates how to use Functions, classes and the SoC pattern
# ChangeLog: (Who, When, What)
# Surafel Tebeje,05.30.2025,Created Script
# ------------------------------------------------ #

import json

# Define the Data Constants
MENU:str='''\n---- 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
-----------------------------------------
```

```
\n'''
FILE_NAME:str="Enrollments.json"

#Define the data variables
menu_choice:str=''
students:list=[]

# Processing --------------------------------------- #
class FileProcessor:
    """ A collection of processing layer functions that work with Json files
        ChangeLog: (Who, When, What)
        Surafel Tebeje,05.30.2025,Created Class
       """

    @staticmethod
    def read_data_from_file(file_name:str, student_data:list):
        """This function reads data from a json file and loads into a list of dictionary rows
        ChangeLog: (Who, When, What)
        Surafel Tebeje,05.30.2025, created function
        parameter: file_name :string data with name of file to read from
        parameter: student_data: list of dictionary rows with multiple student registrations
        return: returns the list of dictionary rows
        """
        file=None  #a local variable
        try:
            file= open(file_name, "r")
            student_data = json.load(file)
            file.close()
        except FileNotFoundError as e:
            IO.output_error_messages("Text file must exist before running this script!", e)
        except Exception as e:
            IO.output_error_messages("There was a non-specific error!", e)
        finally:
            if file.closed == False:
                file.close()
        return student_data

    @staticmethod
    def write_data_to_file (file_name:str, student_data:list):
        """This function writes data to a json file
        ChangeLog: (Who, When, What)
        Surafel Tebeje,05.30.2025, created function"""
        file=None
        try:
             file = open(file_name, "w")
             json.dump(student_data, file)
             file.close()
        except TypeError as e:
            IO.output_error_messages("Please check that the data is a valid JSON format",e)
        except Exception as e:
            IO.output_error_messages("There was a non-specific error!", e)
        finally:
            if file.closed == False:
                file.close()
        print("The following data is saved in the file:\n")
        print(students)
class IO:
    """
        A collection of presentation layer functions that manage user input and output
        ChangeLog: (Who, When, What)
        Surafel Tebeje,05.30.2025,Created Class
        Surafel Tebeje,05.30.2025,Added menu output and input functions
        Surafel Tebeje,05.30.2025,Added a function to display the data
        Surafel Tebeje,05.30.2025,Added a function to display custom error messages
        """

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

                ChangeLog: (Who, When, What)
                Surafel Tebeje,05.30.2025,Created function
                :return: None
                """
        message="There is a problem with your data. Please try to find out!"
        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)
                Surafel Tebeje,05.30.2025,Created function"""
        print()
        print(menu)
        print()  # Adding extra space to make it look nicer.

    @staticmethod
    def input_menu_choice():
        """ This function gets a menu choice from the user
                :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("Please, choose only 1, 2, 3, or 4")
        except Exception as e:
            IO.output_error_messages(e.__str__())  # Not passing the exception object to avoid the technical
message
        return choice


    @staticmethod
    def input_student_data(student_data:list):
        """This function gets a student data from a user, and returns data.
        The function stores data into a list (students) which will be returned at the end.
            ChangeLog: (Who, When, What)
            Surafel Tebeje,05.30.2025,Created function"""

        try:
            student_first_name = input("What is the student's first name? ")
            if not student_first_name.isalpha():
                raise ValueError("The first name should not contain numbers")
            student_last_name = input("What is the student's last name? ")
            if not student_last_name.isalpha():
                raise ValueError("The last name should not contain numbers")
            course_name = input("What is the course's name? ")
            student_data = {"Student Name": student_first_name, "Student Last Name": student_last_name,
                            "Course Name": course_name}
            students.append(student_data)
            print(f"\nstudent {student_first_name} {student_last_name} is registered!\n")
        except ValueError as e:
            IO.output_error_messages ("That value is not the correct type of data!", e)
        except Exception as e:
            IO.output_error_messages("There was a non-specific error!", e)
        return student_data


    @staticmethod
    def output_student_courses(student_data: list):
        """This function displays the registered student data
        ChangeLog: (Who, When, What)
            Surafel Tebeje,05.30.2025,Created function"""

        for each in student_data:
            print(f"{each['Student Name']},{each['Student Last Name']},{each['Course Name']}\n")
```

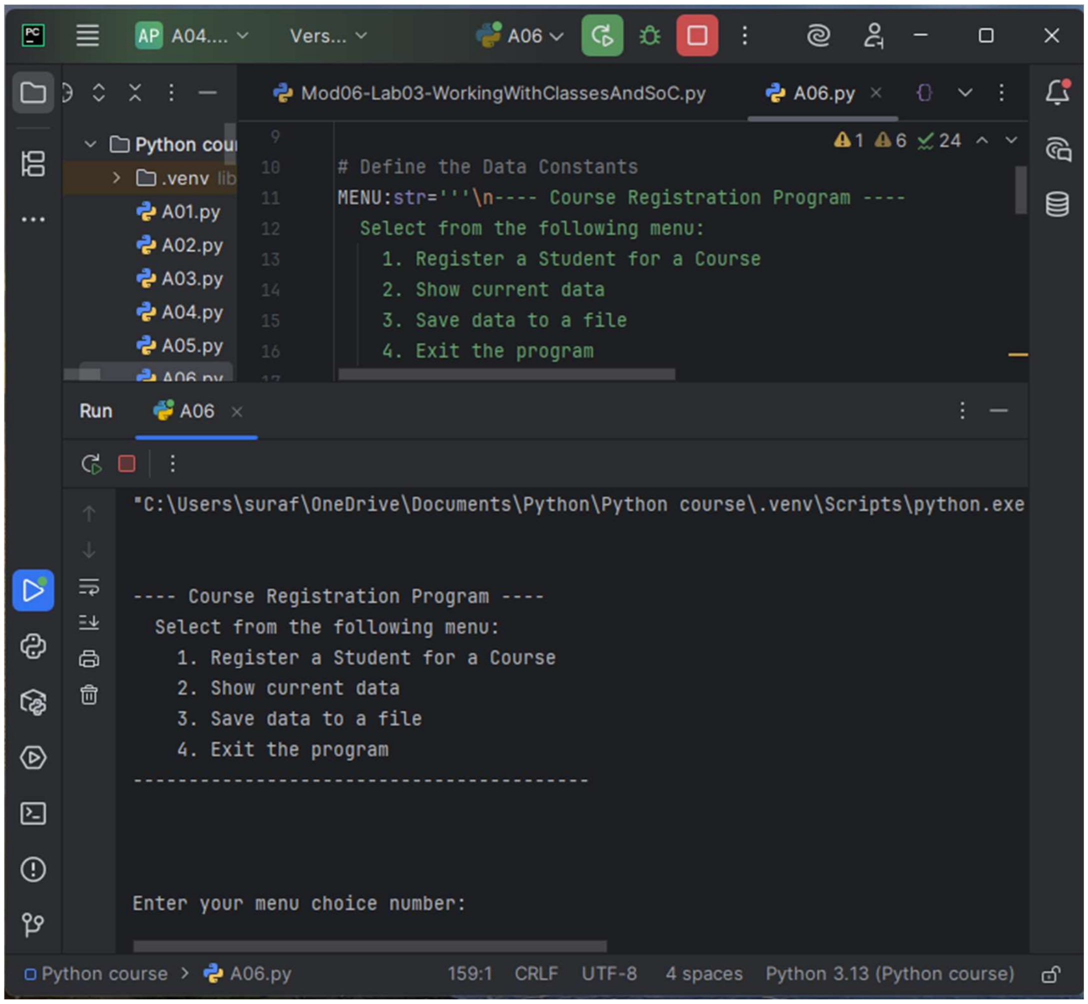
```
#  End of function definitions

# Beginning of the main body of this script

#Read contents of a file
students=FileProcessor.read_data_from_file(file_name=FILE_NAME, student_data=students)

# Repeat the follow tasks

while True:

    IO.output_menu(menu=MENU)

    menu_choice = IO.input_menu_choice()

    if menu_choice == "1":  # Register the student for a course
        IO.input_student_data(student_data=students)
        continue

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

    elif menu_choice == "3":  # Save data in a file
       FileProcessor.write_data_to_file(file_name= FILE_NAME, student_data=students)
       continue

    elif menu_choice == "4":  # End the program
        print("program Ended!")
        break  # out of the while loop
    else:
        print("Invalid menu choice")
```

Results from running the syntax codes: PyCharm

1. Reading the contents of a file:

   When the program reads the existing file and returns at the end.

2. When the program runs, it displays the menu and prompts for the menu choice number:

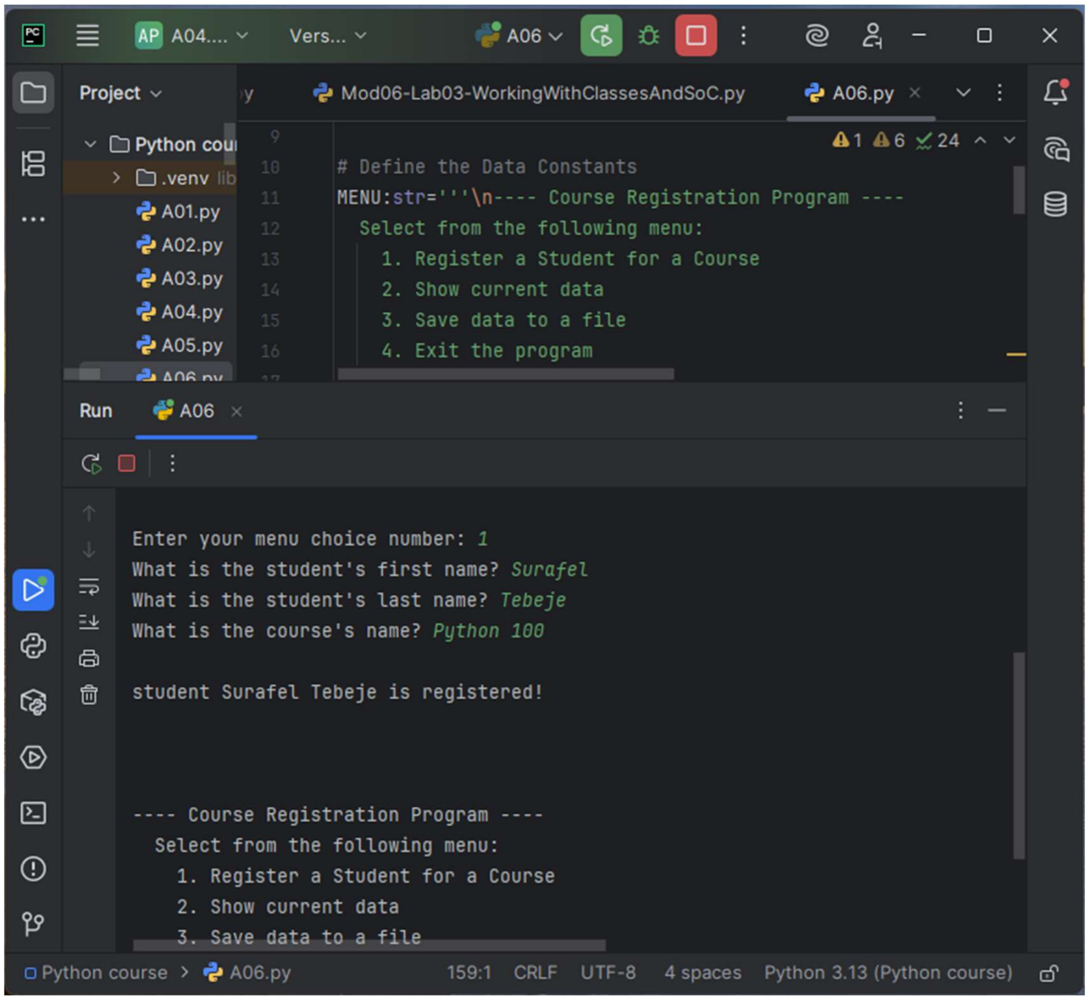


```
# Define the Data Constants
MENU:str='''\n---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course
    2. Show current data
    3. Save data to a file
    4. Exit the program
```

```
"C:\Users\suraf\OneDrive\Documents\Python\Python course\.venv\Scripts\python.exe


---- 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:
```

Figure 1: Displaying menu choices

3. Results of Choice 1: The program prompts the menu choice, and upon user input, it stores the entered data into the respective variables/functions. It also confirms the registration by displaying a message.

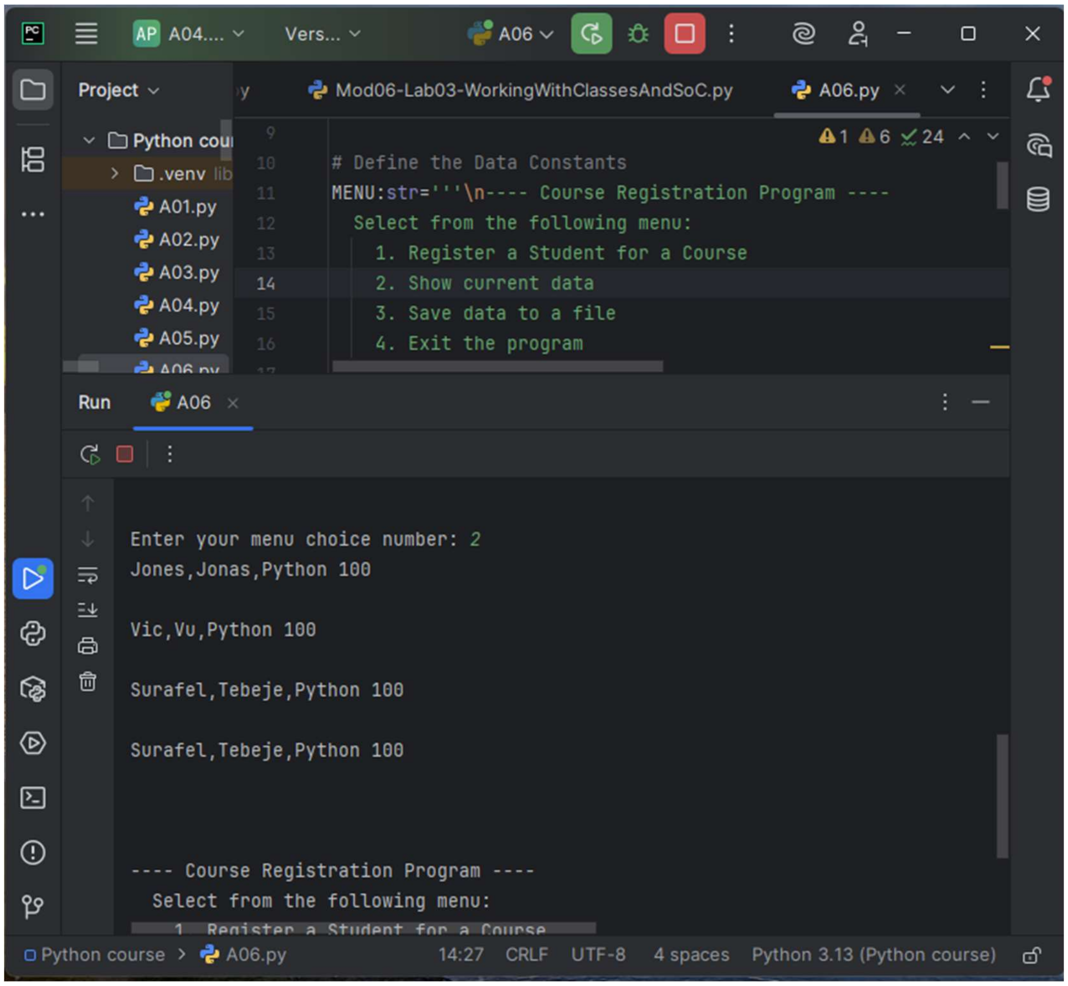


```
# Define the Data Constants
MENU:str='''\n---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course
    2. Show current data
    3. Save data to a file
    4. Exit the program
```

```
Enter your menu choice number: 1
What is the student's first name? Surafel
What is the student's last name? Tebeje
What is the course's name? Python 100

student Surafel Tebeje is registered!



---- 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
```

Figure 2: Results of choice 1

4. Results of Choice 2: The program displays the stored data.

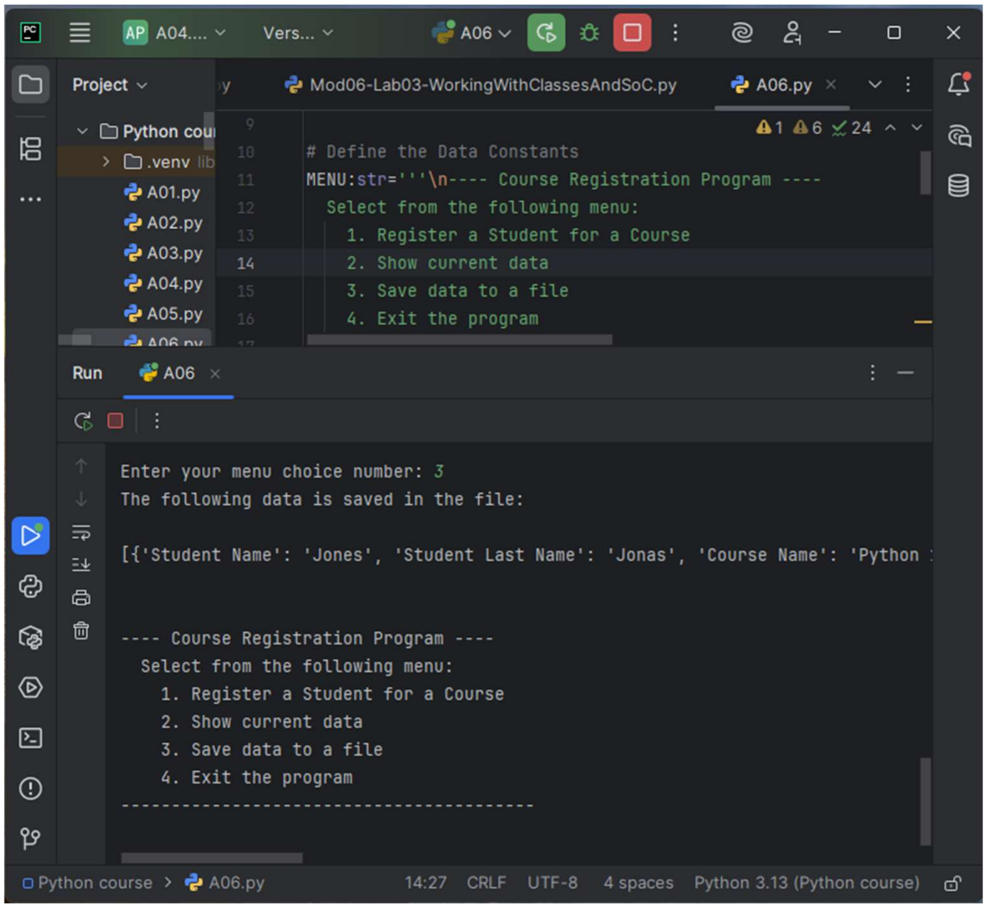


```
# Define the Data Constants
MENU:str='''\n---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course
    2. Show current data
    3. Save data to a file
    4. Exit the program
```

```
Enter your menu choice number: 2
Jones,Jonas,Python 100

Vic,Vu,Python 100

Surafel,Tebeje,Python 100

Surafel,Tebeje,Python 100



---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course
```

Figure 3 : Results of choice 2

5. Results of Choice 3: The program stores the data into a file and displays the stored data.

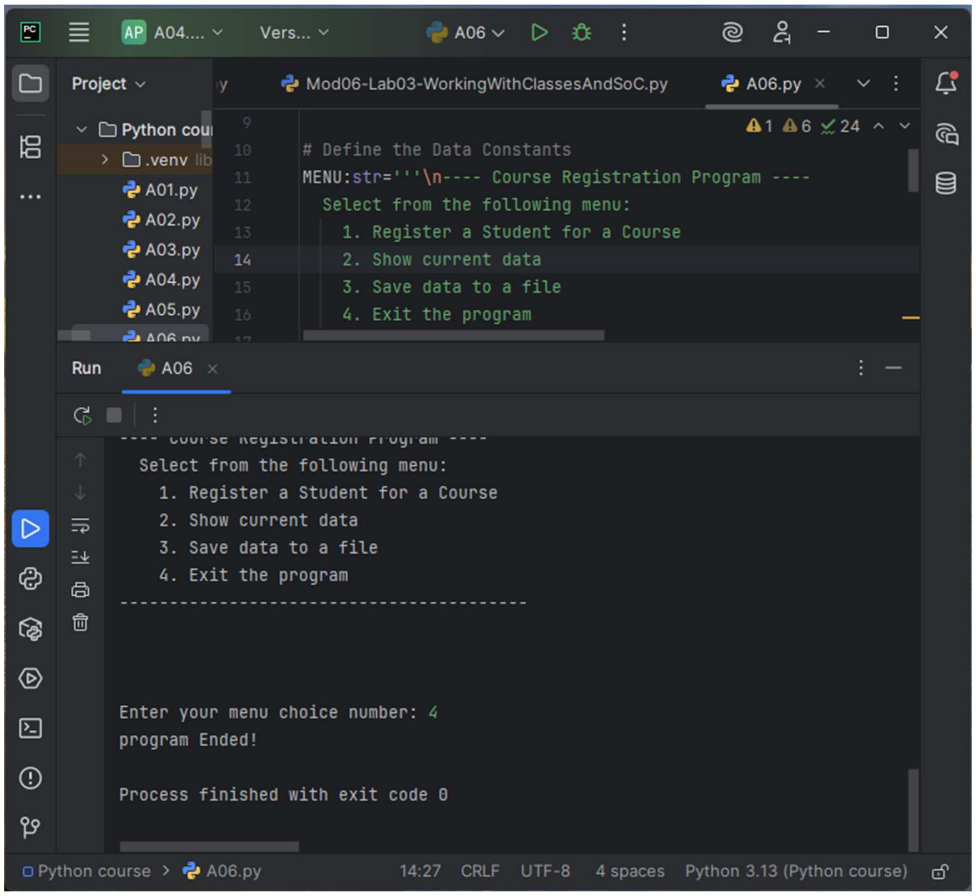


```
# Define the Data Constants
MENU:str='''\n---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course
    2. Show current data
    3. Save data to a file
    4. Exit the program
```

```
Enter your menu choice number: 3
The following data is saved in the file:

[{'Student Name': 'Jones', 'Student Last Name': 'Jonas', 'Course Name': 'Python


---- 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 4: Results of choice 3

6. Results of Choice 4: The program ends, and displays a message.

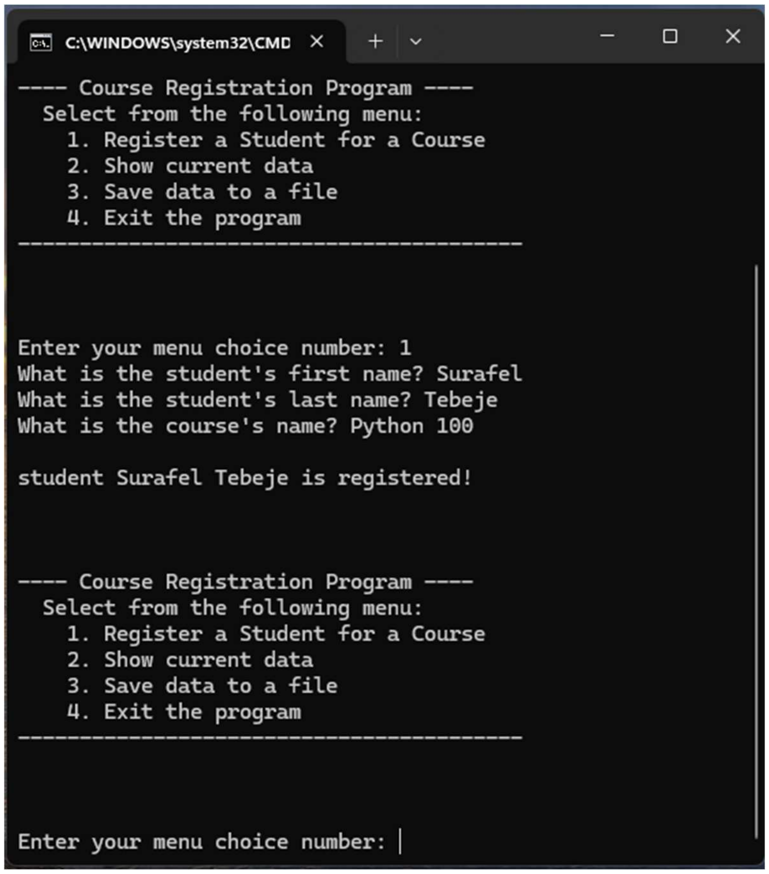


```
# Define the Data Constants
MENU:str='''\n---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course
    2. Show current data
    3. Save data to a file
    4. Exit the program
```

```
---- 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!

Process finished with exit code 0
```

Figure 5: Results of choice 4

Running code on the console:



```
---- 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? Surafel
What is the student's last name? Tebeje
What is the course's name? Python 100

student Surafel Tebeje is registered!



---- 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: 
```

Figure 6: Results of the Console

Summary:

In summary, I demonstrated that Python programs can be well organized using Functions, classes, and SoC. This improves code modularity, scalability, and debugging.