Brian Christopherson

05-JUN-2025

Python 100

Assignment07

https://github.com/cb658/IntroToProg-Python-Mod07

A student registration program using data classes, objects and functions, with Separation of Concerns patterning

Introduction

This paper will discuss Assignment07, which involves writing a Python program that registers new students and adds them to the student enrollment. To do this, the program reads the enrollment file and writes the user-input data back to the enrollments file in the JavaScript Object Notation (JSON) format.

The program presents a menu to the user and uses a *while* loop and conditional logic to control the flow of the program. The program uses constants, variables, dictionaries and a list to store the data. Data classes and functions are used to organize the program and display the data, and the program is further organized in patterns based on the Separation of Concerns (SoC) design principle.

For error checking, the program uses data validation via the setter property, and also uses the try-except structured error handling method.

This program was written in the PyCharm IDE. Documentation of the program's testing can be found in Appendix A; the full code is in the GitHub repo https://github.com/cb658/IntroToProg-Python-Mod07.

Writing the Python program

The program creation process began with a review of the acceptance criteria for the program. The acceptance criteria included using a specific name for the program file, a script header and the use of specific constants and variables. The constants were strings presenting a multi-line menu and the name of the file to be used for the read and write operations. For the multi-line menu, a triple-quoted string was used. The menu presented four choices to the user.

Input and Output options

The acceptance criteria defined input and output options based on the user choices from the menu. These are described in the Main body section of this document.

Classes and Class Properties

The acceptance criteria required utilizing specific classes and functions, with descriptive document strings. Two data class instances are used, along with a processing class and a presentation class. The classes are:

- Person *(data class)*
  - *Properties:*
    - *student_first_name*
    - *student_last_name*
- Student *(data class, a child of the Person class)*
  - *Properties:*
    - *student_first_name (inherited)*
    - *student_last_name (inherited*
    - *course_name*

- FileProcessor *(processing class)*
- IO *(presentation class)*

Functions

The acceptance criteria also required specific functions, also with descriptive document strings, calls to function handling error message, and the use of the @staticmethod decorator. The @staticmethod is called because the functions are not trying to change the class or modify objects. The function names and parameters are listed below:

- 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):

The code contained a while statement to loop through the various input choices. The looping allows for another of the acceptance criteria, which was to allow the user to enter multiple student registrations, if desired, and store all those registrations.

The acceptance criteria defined some structured error handling at various points within the program execution:

- when the file is read in
- the user's input of the first name
- the user's input of the last name
- lastly, when the dictionary rows are written to the file.

The last requirements of the acceptance criteria required specific tests of the program input and output functions, as well as confirming the program runs correctly in both PyCharm and the console.

Program Organization

Following the Separation of Concerns (SoC) design principle, the program is divided into three sections: Data, Processing and Presentation.

Data section

The ***Data*** section includes the declaration of variables and defining the data classes. The data classes use the getter functions to apply title case formatting, while the setter function to set and validate the data.

Processing section

The ***Processing*** section contains the ***FileProcessor*** class, which contains two functions:

- ***read_data_from_file****:* for reading the json file and writing the student data to memory.
- ***write_data_to_file****:* for converting the student objects to a dictionary for writing to the json file. This function also calls the **IO. *output_student_and_course_names*** to display the enrollment data to the user and the ***IO.output_error_messages*** for structured error handling..

Presentation section

The ***Presentation*** section contains the ***IO*** class, with five functions:

- ***output_error_message****s*: for displaying error messages to the user
- ***output_menu***: for displaying the MENU string constant to the user and present the program choices
- ***input_menu_choice***: for accepting the user's input choices. This function includes error handling and calls the *output_error_messages* function
- ***output_student_and_course_names***: for displaying the student name(s) and course enrollment(s) to the user, using the **student** object
- ***input_student_data***: for accepting the student first name, last name and the course name and adding to the list of enrollments. Print statements advise the user that the student has been enrolled, and to enter option 3 to save the data. This function includes error handling and calls the ***output_error_messages*** function.
-

Main body

The main body of the program begins with a call to the ***FileProcessor.read_data_from_file*** function to read the data into the students list of lists. A ***while*** loop begins for presenting the menu choices to the user, using the **IO.output_menu** function. Next, the user input is collected via the ***IO.input_menu_choice*** function and set to the ***menu_choice*** string variable**.**

- If the user enters choice ***1***, the ***IO.input_student_data*** function is called to add the entered data to the **students** list.
- If the user enters choice ***2***, the ***IO.output_student_and_course_names*** function is called to display the current data to the user.
- If the user enters choice ***3***, the ***FileProcessor.write_data_to_file*** function is called to write the student data via the *json.dump()* function to the Enrollments.json file.
- If the user enters choice ***4***, the program exits via a ***break*** statement.

Testing the program

To test the program within PyCharm, the Run Module (F5) command was invoked. The program presented the menu, accepted the user input, then presented it and the read-in data when the 'current data' option was chosen. The program also saved the data to the Enrollments.json file and exited if the 'exit program' choice was called, thus meeting the acceptance criteria. The student error handling blocks were tested by changing the name of the Enrollments.json file and entering numerals when entering the student's first and last names. The program was then tested by opening the Windows console and calling the Assignment07.py file. The program also ran successfully via the console. Screenshots showing the PyCharm and console tests can be found in Appendix A.

Summary

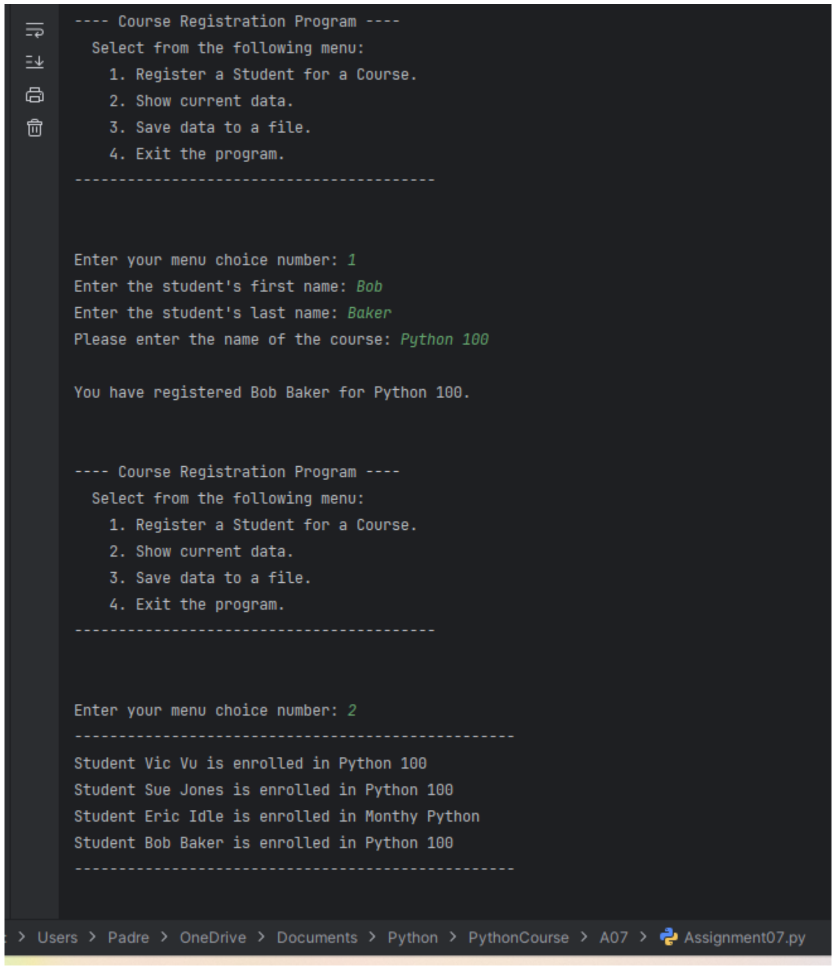
This assignment covered several concepts, such as declaring constants and variables and defining classes, objects and functions. The program used *input()* and *print()* functions; processed data using dictionaries, lists and files in the JSON format. The program also included structured error handling. The program utilized Separation of Concerns (SoC) patterns to organize the classes and functions.

The program successfully accepted input from the program's user, of a student's first name and last name and course for enrollment. That input was then output to the user, then added to the existing "**students**" enrollments list, which was read and assigned memory when the enrollments file was opened. The updated "**students**" enrollments list was then successfully written to the enrollments file. Errors with the file, and errors with the input of the student’s first name and last name were addressed within the program. The program was successfully tested in the PyCharm IDE and the Windows Command Prompt console.

The testing of the program is shown in Appendix A.

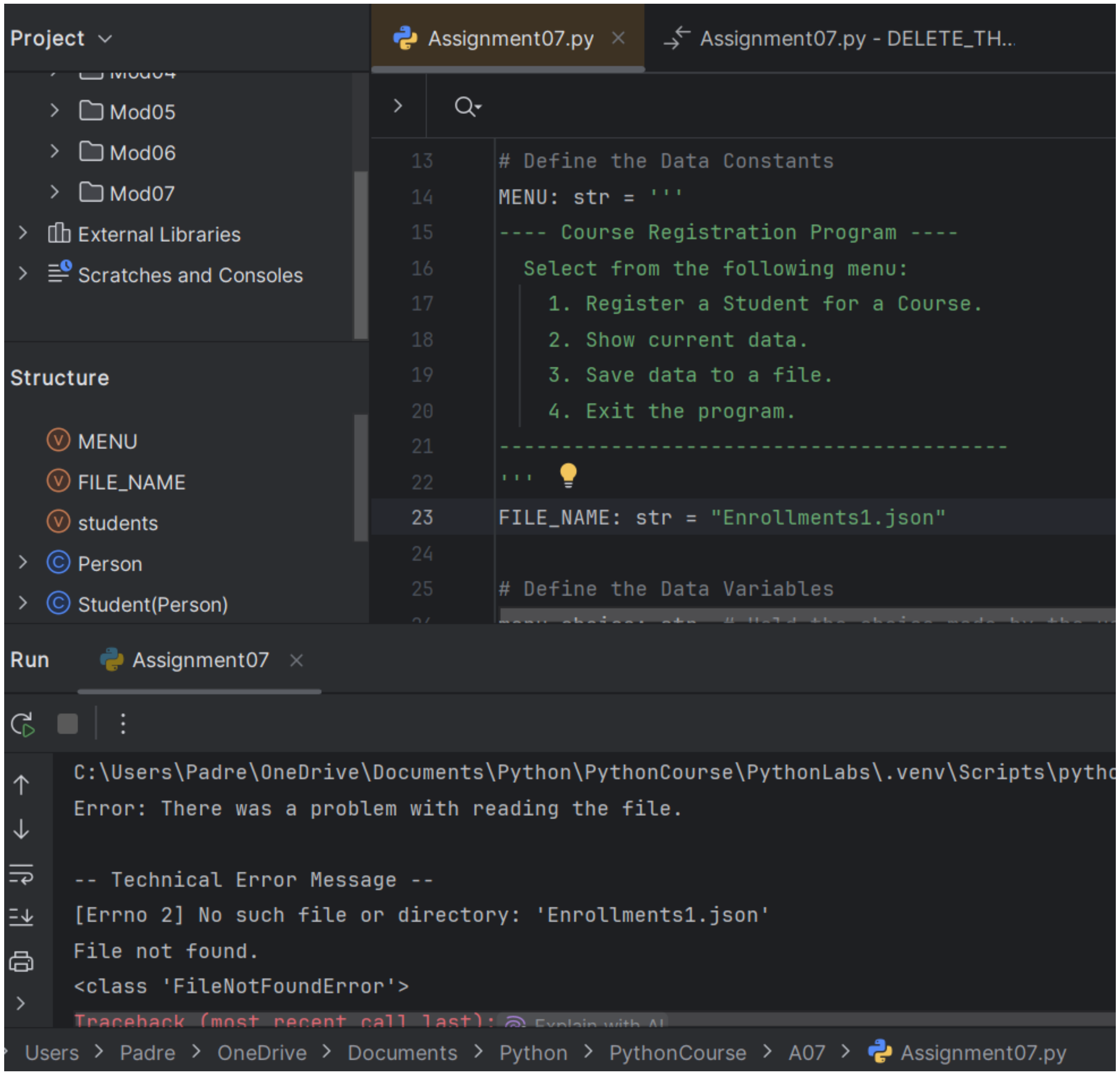
Appendix A – Testing the program

PyCharm testing



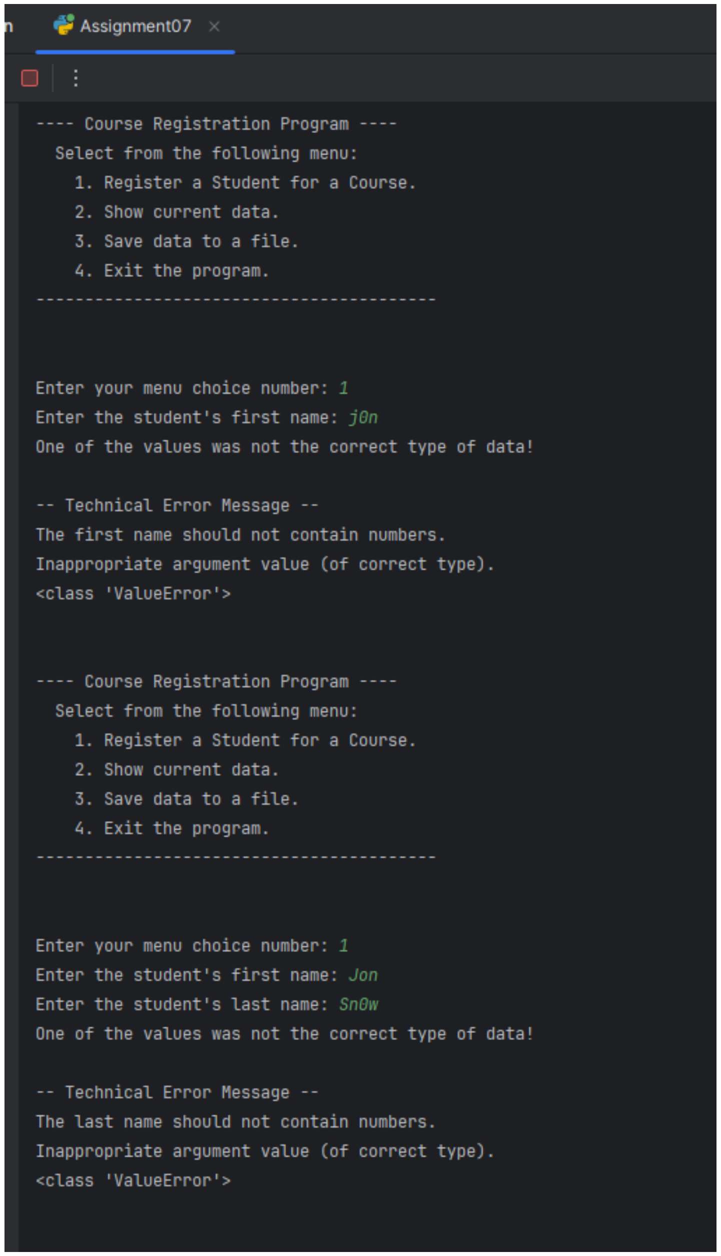
```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------



Enter your menu choice number: 1
Enter the student's first name: Bob
Enter the student's last name: Baker
Please enter the name of the course: Python 100

You have registered Bob Baker for Python 100.


---- 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 Vic Vu is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Eric Idle is enrolled in Monthy Python
Student Bob Baker is enrolled in Python 100
--------------------------------------------------
```

Users › Padre › OneDrive › Documents › Python › PythonCourse › A07 › Assignment07.py

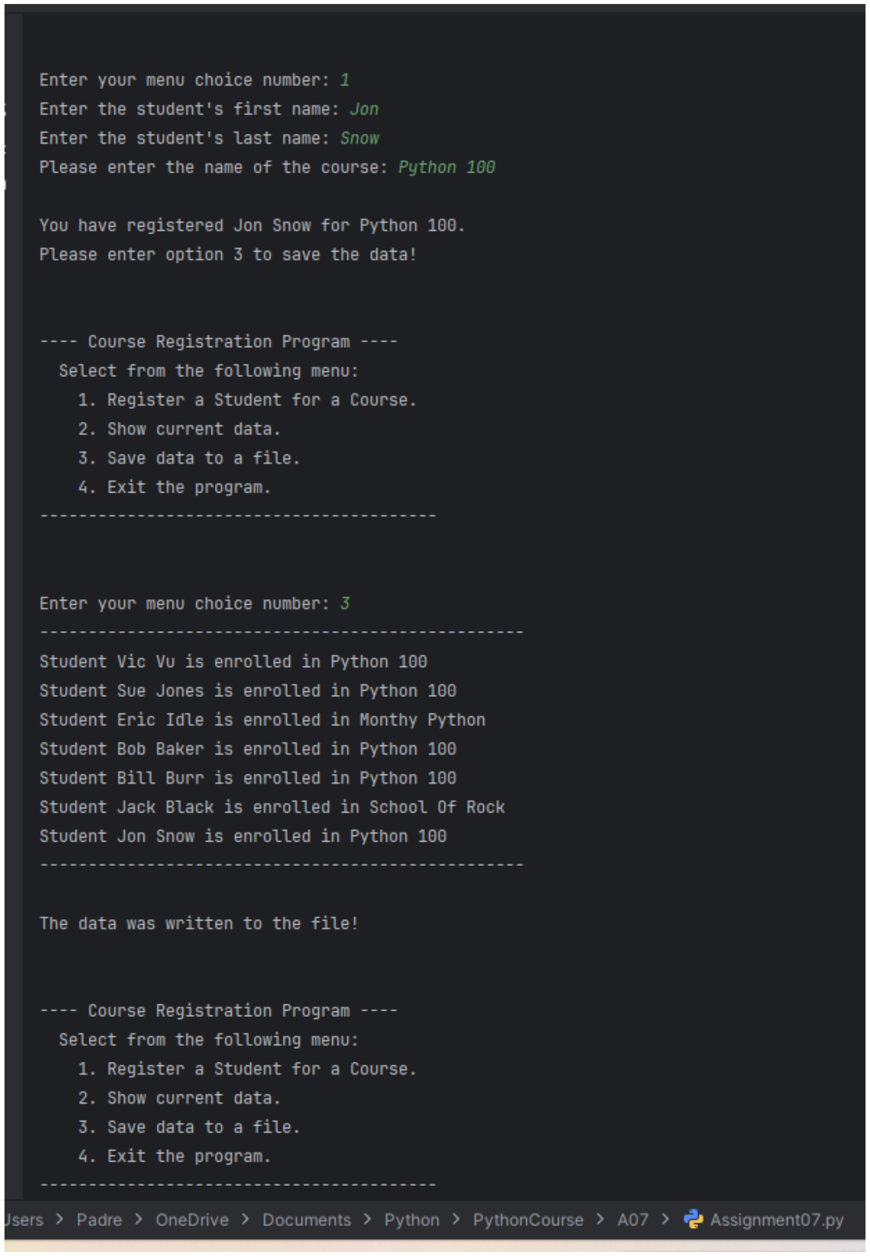


```
# Define the Data Constants
MENU: str = '''
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------
'''
FILE_NAME: str = "Enrollments1.json"

# Define the Data Variables
```

```
C:\Users\Padre\OneDrive\Documents\Python\PythonCourse\PythonLabs\.venv\Scripts\pytho
Error: There was a problem with reading the file.

-- Technical Error Message --
[Errno 2] No such file or directory: 'Enrollments1.json'
File not found.
<class 'FileNotFoundError'>
Traceback (most recent call last):
```



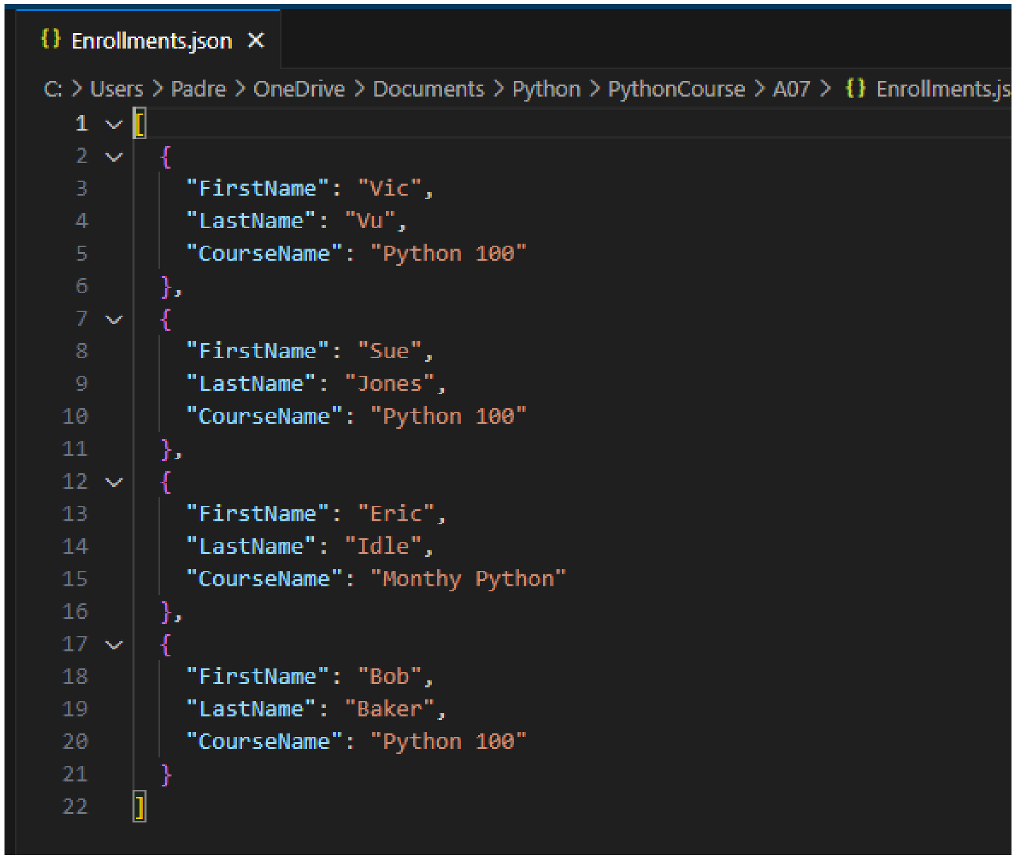
```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------



Enter your menu choice number: 1
Enter the student's first name: j0n
One of the values was not the correct type of data!

-- Technical Error Message --
The first name should not contain numbers.
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
Enter the student's first name: Jon
Enter the student's last name: Sn0w
One of the values was not the correct type of data!

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



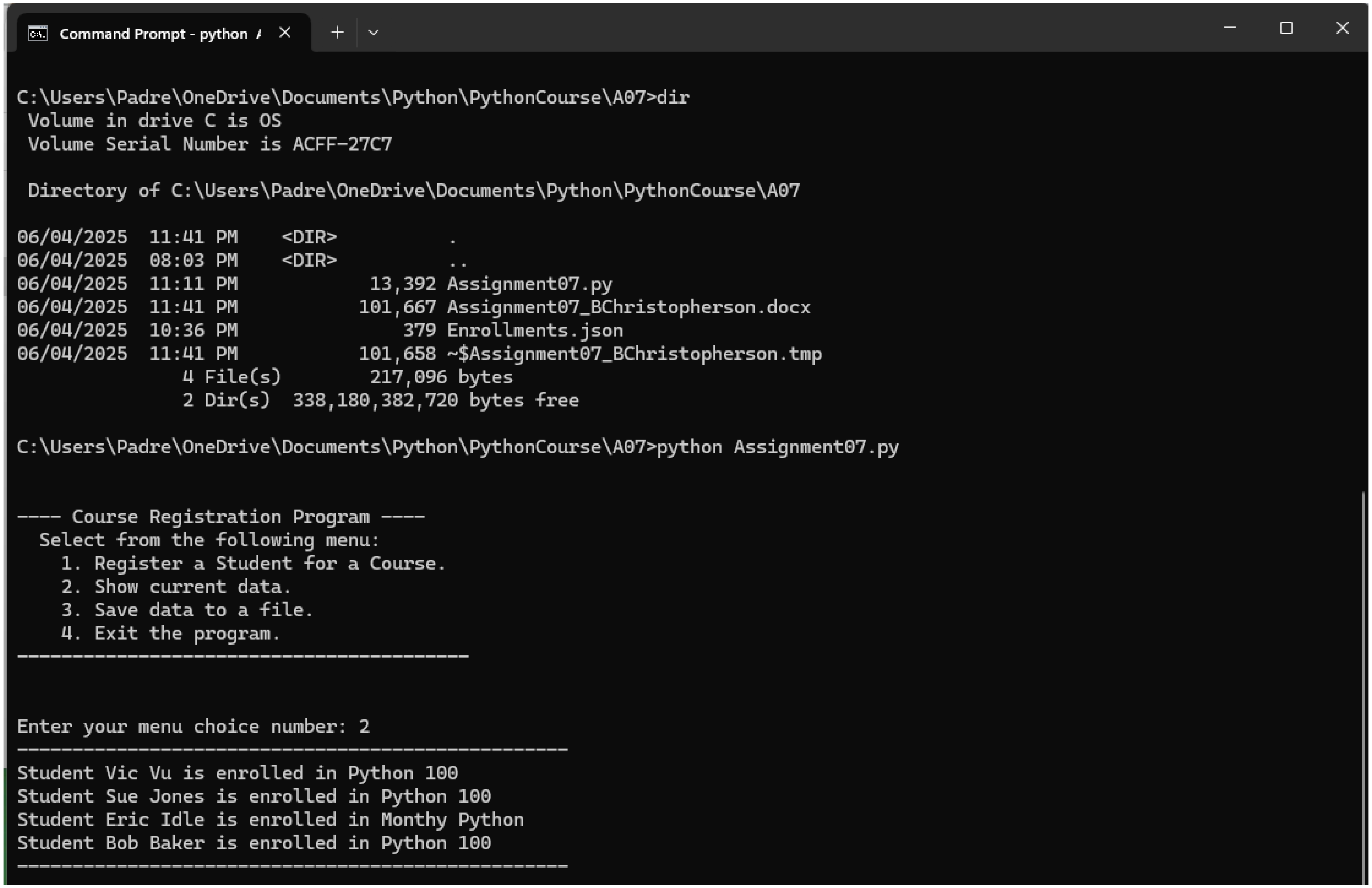
```
Enter your menu choice number: 1
Enter the student's first name: Jon
Enter the student's last name: Snow
Please enter the name of the course: Python 100

You have registered Jon Snow for Python 100.
Please enter option 3 to save the data!


---- 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
--------------------------------------------------
Student Vic Vu is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Eric Idle is enrolled in Monthy Python
Student Bob Baker is enrolled in Python 100
Student Bill Burr is enrolled in Python 100
Student Jack Black is enrolled in School Of Rock
Student Jon Snow is enrolled in Python 100
--------------------------------------------------

The data was written to the 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.
-----------------------------------------
```

Jsers > Padre > OneDrive > Documents > Python > PythonCourse > A07 > Assignment07.py

]


{} Enrollments.json ✕
C: > Users > Padre > OneDrive > Documents > Python > PythonCourse > A07 > {} Enrollments.js

```
[
  {
    "FirstName": "Vic",
    "LastName": "Vu",
    "CourseName": "Python 100"
  },
  {
    "FirstName": "Sue",
    "LastName": "Jones",
    "CourseName": "Python 100"
  },
  {
    "FirstName": "Eric",
    "LastName": "Idle",
    "CourseName": "Monthy Python"
  },
  {
    "FirstName": "Bob",
    "LastName": "Baker",
    "CourseName": "Python 100"
  }
]
```



```
C:\Users\Padre\OneDrive\Documents\Python\PythonCourse\A07>dir
 Volume in drive C is OS
 Volume Serial Number is ACFF-27C7

 Directory of C:\Users\Padre\OneDrive\Documents\Python\PythonCourse\A07

06/04/2025  11:41 PM    <DIR>          .
06/04/2025  08:03 PM    <DIR>          ..
06/04/2025  11:11 PM            13,392 Assignment07.py
06/04/2025  11:41 PM           101,667 Assignment07_BChristopherson.docx
06/04/2025  10:36 PM               379 Enrollments.json
06/04/2025  11:41 PM           101,658 ~$Assignment07_BChristopherson.tmp
               4 File(s)        217,096 bytes
               2 Dir(s)  338,180,382,720 bytes free

C:\Users\Padre\OneDrive\Documents\Python\PythonCourse\A07>python Assignment07.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: 2
--------------------------------------------------
Student Vic Vu is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Eric Idle is enrolled in Monthy Python
Student Bob Baker is enrolled in Python 100
--------------------------------------------------
```