Name: Cari Williams

Date: 12 November 2025

Course: Foundations Of Programming with Python

Github: https://github.com/carilynette/IntroToProg-Python-Mod05

A05: Working with Dictionaries and Exception Handling

Introduction

This project builds on previous assignments and expands skills in working with dictionaries, JSON files, and exception handling. In this assignment, I modified the starter script to store course registration data in dictionaries instead of lists, load and save data from a JSON file, and handle possible runtime errors using try/except/finally blocks. These changes made the program more robust, easier to maintain, and more realistic for real-world data-processing tasks.

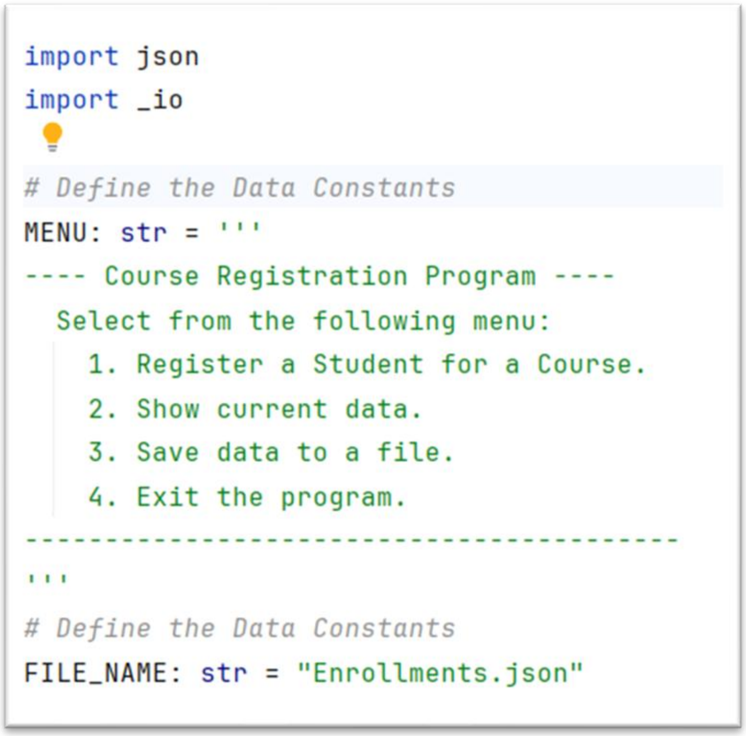
Starter file

I began by using the same process used in previous assignments - I opened the starter file in PyCharm, saved it as *Assignment05-CWilliams.py* in my PythonCourse folder and moved *Enrollments.json* into the same folder, along with updating the header.

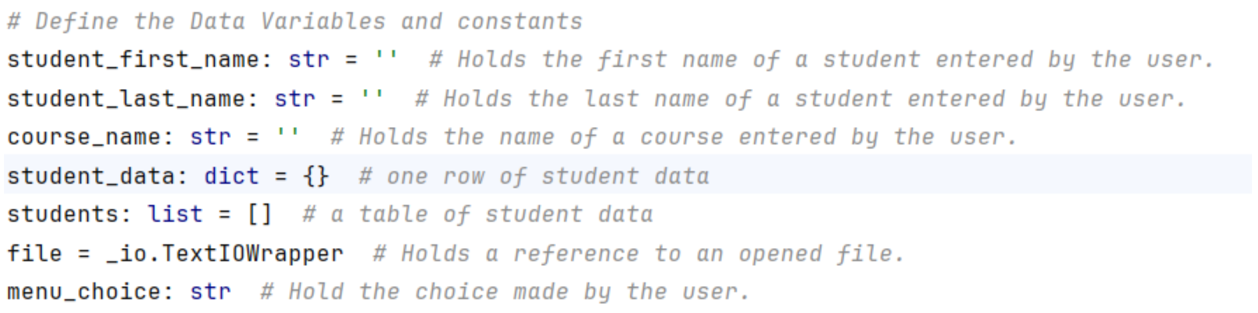
Updating the Data Constants & Variables

I updated the file format from a CSV to a JSON file, *Enrollments.json*. JSON offers a more flexible structure for storing data pairs, which aligns more with dictionaries.

I updated the variables to use dictionaries instead of lists, and I added type hints using _io.TextIOWrapper instead of initializing file = None. The starter script also required importing both json and _io to support JSON file operations and to prepare for more complex file handling. Each student is now represented as a dictionary containing first name, last name, course name. This structure made the data more descriptive and the code easier to read.



```python
import json
import _io

# 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"
```



```python
# Define the Data Variables and constants
student_first_name: str = ''  # Holds the first name of a student entered by the user.
student_last_name: str = ''  # Holds the last name of a student entered by the user.
course_name: str = ''  # Holds the name of a course entered by the user.
student_data: dict = {}  # one row of student data
students: list = []  # a table of student data
file = _io.TextIOWrapper  # Holds a reference to an opened file.
menu_choice: str  # Hold the choice made by the user.
```

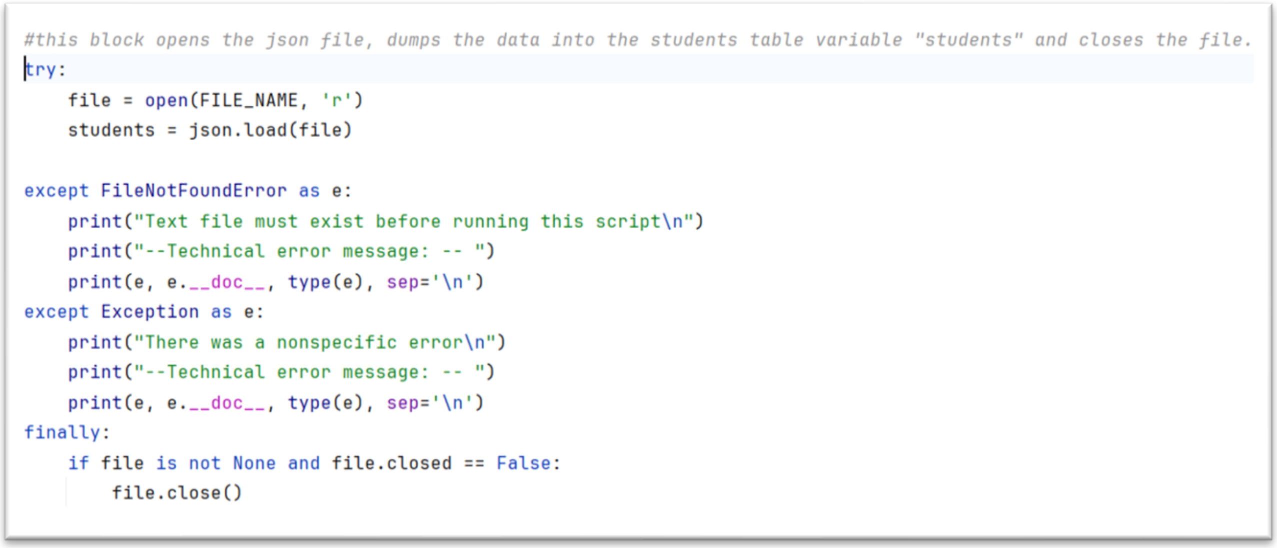
Figure 1: Updated variables

Reading data from a JSON file

The program was updated to process the data into the JSON file. Previously the data in *Enrollments.csv* was read, split then transformed into a user-friendly format. With a JSON file, opening and reading the data in *Enrollments.json* is a much simpler process, requiring less code to implement. Json.load() loads existing data from the file into a Python list and json.dump() saves the updated data to the file in the same format. Because JSON uses curly braces { } to represent objects, the string formatting in the program also needed to be updated for dictionary-based output.

Error Handling

This module introduces the important concept of error handling. I added the Try//xcept, Finally block to handle any errors with loading and reading the file. FileNotFoundError is caught if the JSON file does not yet exist. The Finally block ensures the file is closed safely even if an error occurs.



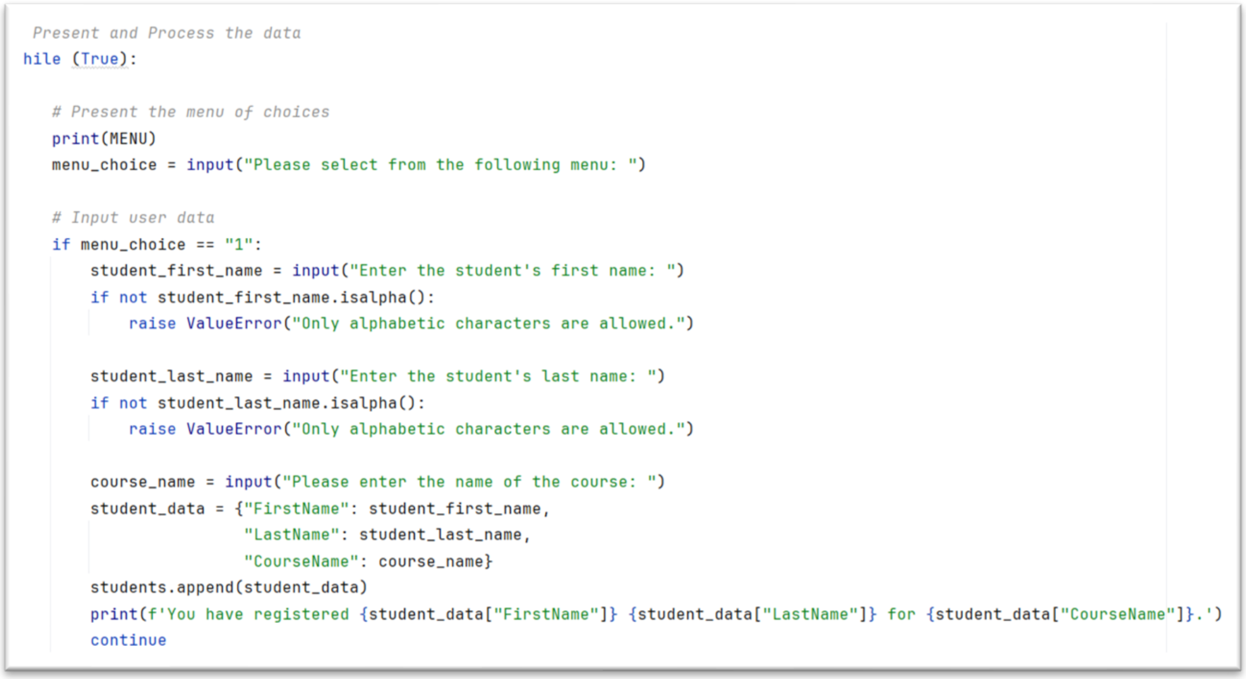
```
#this block opens the json file, dumps the data into the students table variable "students" and closes the file.
try:
    file = open(FILE_NAME, 'r')
    students = json.load(file)

except FileNotFoundError as e:
    print("Text file must exist before running this script\n")
    print("--Technical error message: -- ")
    print(e, e.__doc__, type(e), sep='\n')
except Exception as e:
    print("There was a nonspecific error\n")
    print("--Technical error message: -- ")
    print(e, e.__doc__, type(e), sep='\n')
finally:
    if file is not None and file.closed == False:
        file.close()
```

Figure 2: Reading JSON file with error handling

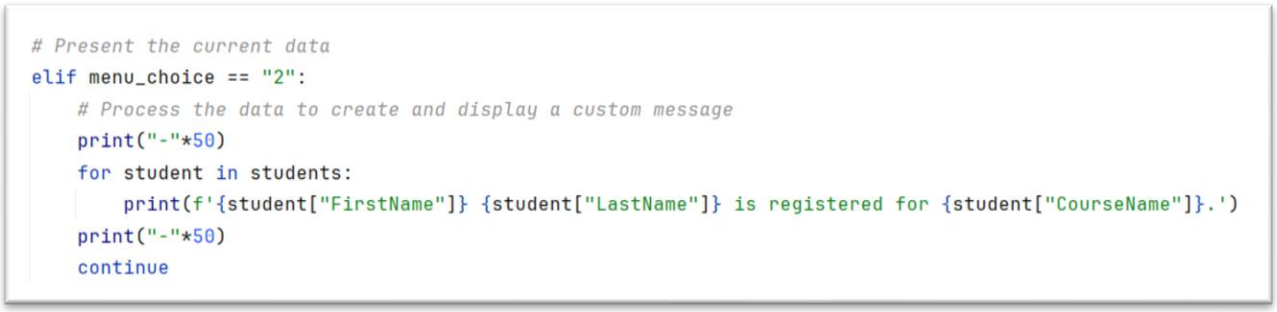
Displaying the menu and processing input

The overall menu structure remains unchanged. It uses a **while True** loop to keep showing the menu until the user chooses 4 to exit. The menu allows the user to:

1. Register a student for a course
2. Show current data
3. Save data to a file
4. Exit the program



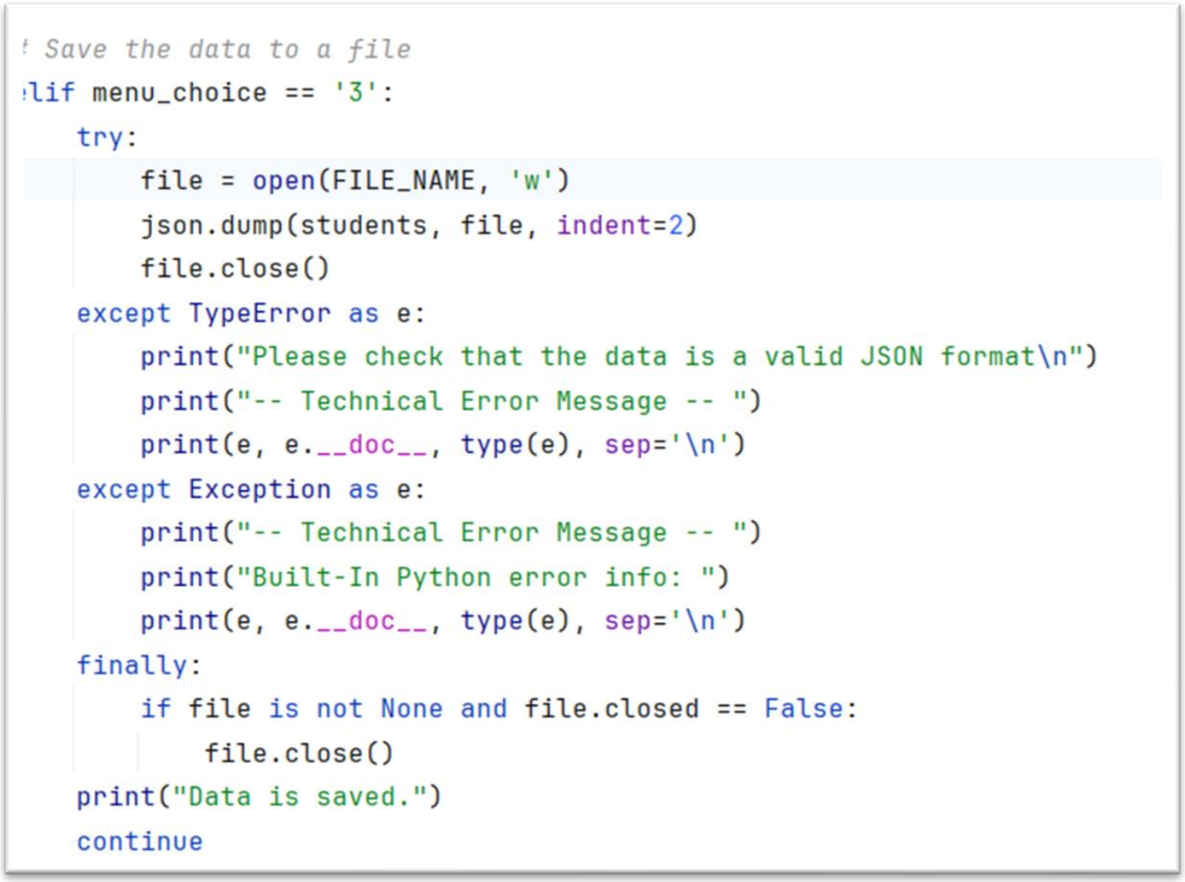
```
Present and Process the data
hile (True):

    # Present the menu of choices
    print(MENU)
    menu_choice = input("Please select from the following menu: ")

    # Input user data
    if menu_choice == "1":
        student_first_name = input("Enter the student's first name: ")
        if not student_first_name.isalpha():
            raise ValueError("Only alphabetic characters are allowed.")

        student_last_name = input("Enter the student's last name: ")
        if not student_last_name.isalpha():
            raise ValueError("Only alphabetic characters are allowed.")

        course_name = input("Please enter the name of the course: ")
        student_data = {"FirstName": student_first_name,
                        "LastName": student_last_name,
                        "CourseName": course_name}
        students.append(student_data)
        print(f'You have registered {student_data["FirstName"]} {student_data["LastName"]} for {student_data["CourseName"]}.')
        continue
```



```
# Present the current data
elif menu_choice == "2":
    # Process the data to create and display a custom message
    print("-"*50)
    for student in students:
        print(f'{student["FirstName"]} {student["LastName"]} is registered for {student["CourseName"]}.')
    print("-"*50)
    continue
```

Figure 3: Show a menu and take input

If the user selects option 1, the program prompts for the student's first and last name and the course name. The data is added to **student_data** dictionary with the .append function. The format was updated for using a dictionary versus a list. When option 2 is chosen, the program loops through students and prints each student’s information. Input validation error handling has been added to prevent numeric data from being entered into an alphabetical field (first and last name). A custom message displays if the error is thrown.

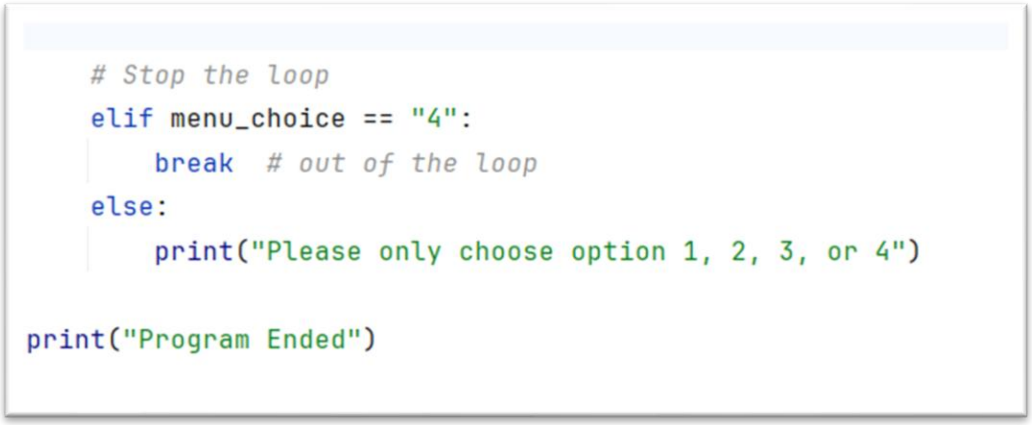


```
 Save the data to a file
lif menu_choice == '3':
    try:
        file = open(FILE_NAME, 'w')
        json.dump(students, file, indent=2)
        file.close()
    except TypeError as e:
        print("Please check that the data is a valid JSON format\n")
        print("-- Technical Error Message -- ")
        print(e, e.__doc__, type(e), sep='\n')
    except Exception as e:
        print("-- Technical Error Message -- ")
        print("Built-In Python error info: ")
        print(e, e.__doc__, type(e), sep='\n')
    finally:
        if file is not None and file.closed == False:
            file.close()
    print("Data is saved.")
    continue
```

When option 3 is chosen, the program opens *Enrollments.json* in write mode. Each student’s information in the dictionary is added to the file with json.dump. The TypeError handles incorrect data types during saving. Then the file is closed safely. The saved data and a confirmation message are displayed confirming it worked.

Exiting the program loop

When the user selects option 4, the program prints an exit message and uses break to stop the loop and end execution. If a selection other than 1-4 is made, a message is displayed to the user.



```
    # Stop the loop
    elif menu_choice == "4":
        break  # out of the loop
    else:
        print("Please only choose option 1, 2, 3, or 4")

print("Program Ended")
```

Testing the program

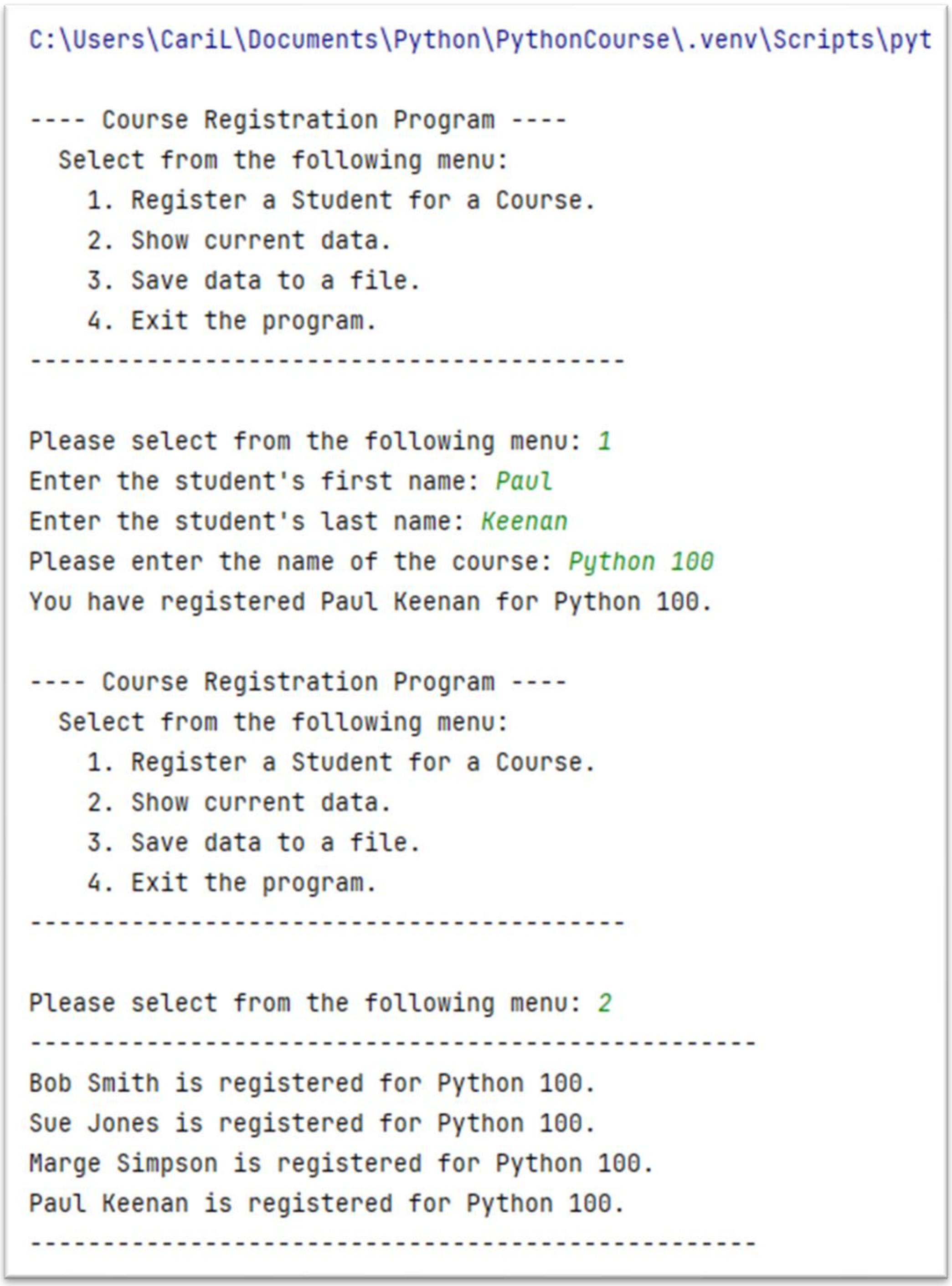
I tested the program several times in both PyCharm and the command line to ensure each menu option worked correctly.

- Option 1 - the program successfully registered a student and displayed a confirmation message.
- Option 2 – the program displayed all current registration data in a clean, readable list.
- Option 3 – saved the new entries to *Enrollments.json* file.
- An invalid menu option displayed the intended message.
- Option 4 – exited the program and ended the loop as expected.

All functions performed as intended, and the output in the .json file matched the user input from the testing session.

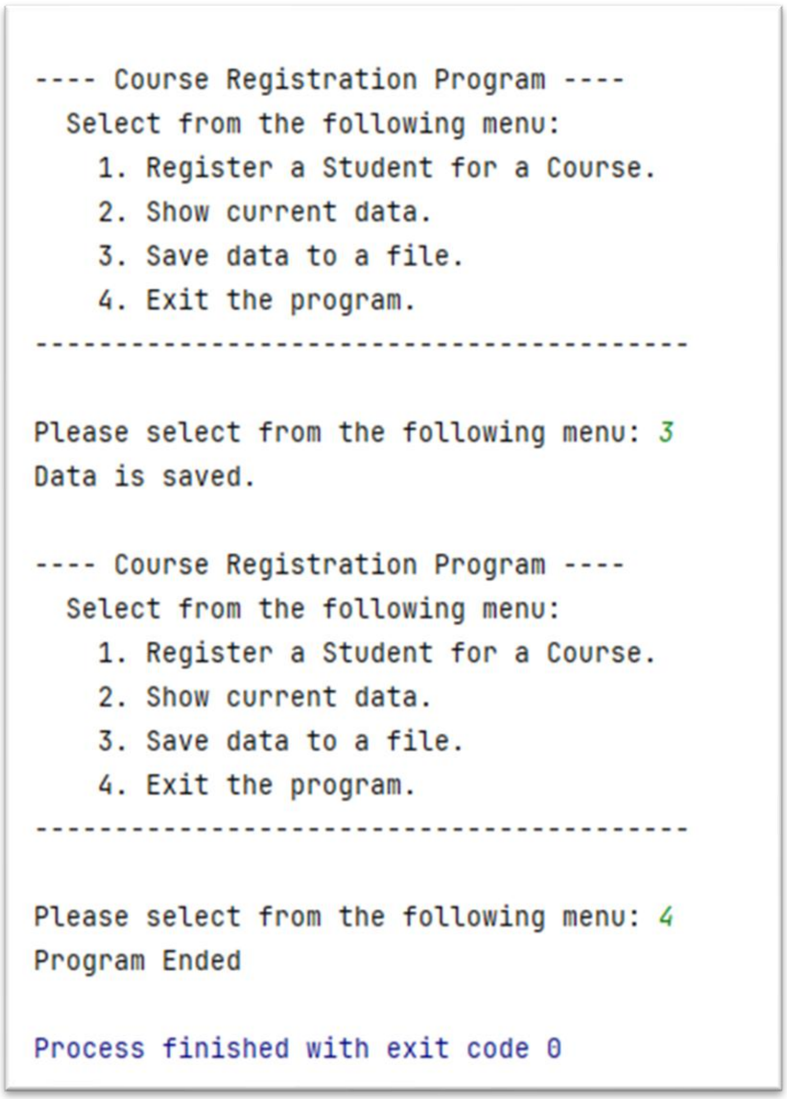
Uploading to GitHub

To maintain version control and keep a record of my coursework, I uploaded my Assignment 05 script the *Enrollments.json* file and this document to my GitHub repository. For this assignment, I used GitHub's web interface: I simply dragged the files directly into the upload window on GitHub. After adding a short commit message describing the update, I saved the changes to my main branch. This process ensures my work is backed up, easy to access, and included in my growing portfolio of Python projects.

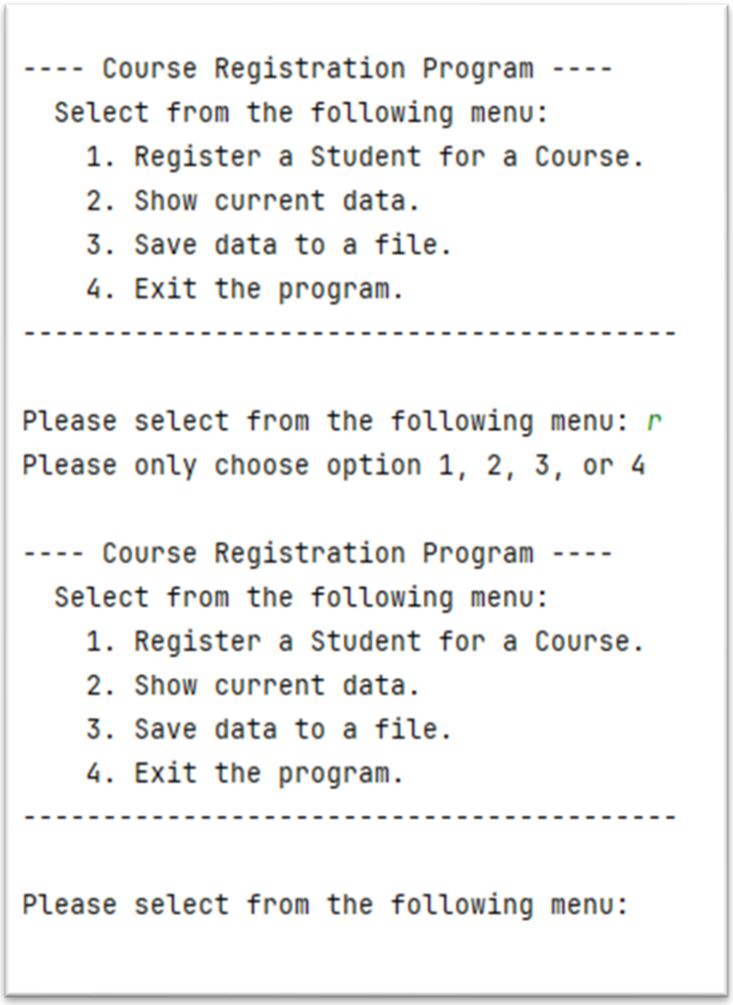


```
C:\Users\CariL\Documents\Python\PythonCourse\.venv\Scripts\pyt

---- 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.
-----------------------------------------

Please select from the following menu: 1
Enter the student's first name: Paul
Enter the student's last name: Keenan
Please enter the name of the course: Python 100
You have registered Paul Keenan 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.
-----------------------------------------

Please select from the following menu: 2
--------------------------------------------------
Bob Smith is registered for Python 100.
Sue Jones is registered for Python 100.
Marge Simpson is registered for Python 100.
Paul Keenan is registered for Python 100.
--------------------------------------------------
```

Figure 4: Option 1 testing

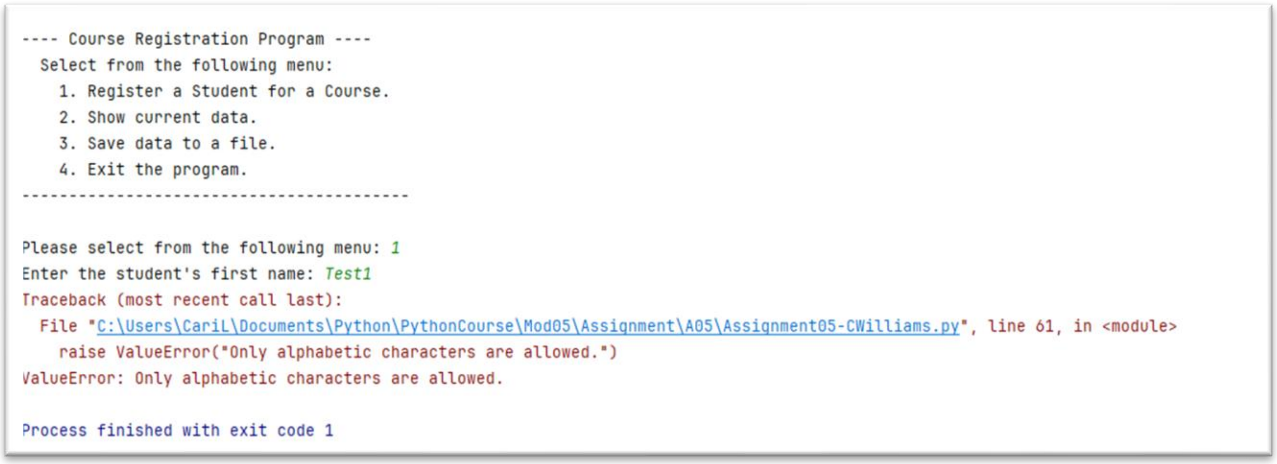


```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------

Please select from the following menu: 3
Data is saved.

---- 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.
-----------------------------------------

Please select from the following menu: 4
Program Ended

Process finished with exit code 0
```

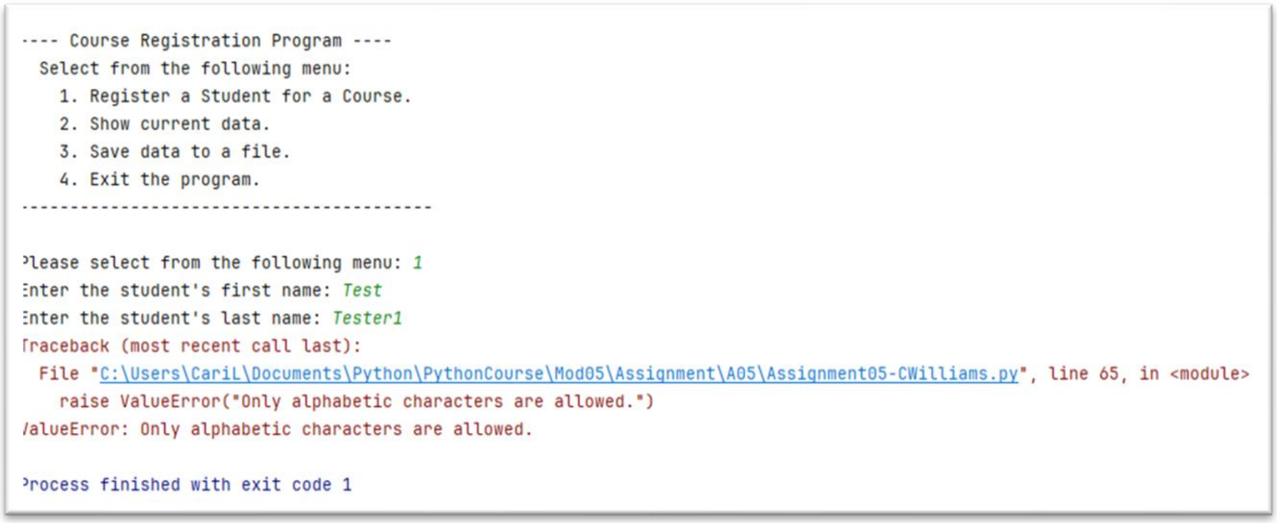


```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------

Please select from the following menu: r
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.
-----------------------------------------

Please select from the following menu:
```

Figure 5: Testing an invalid entry

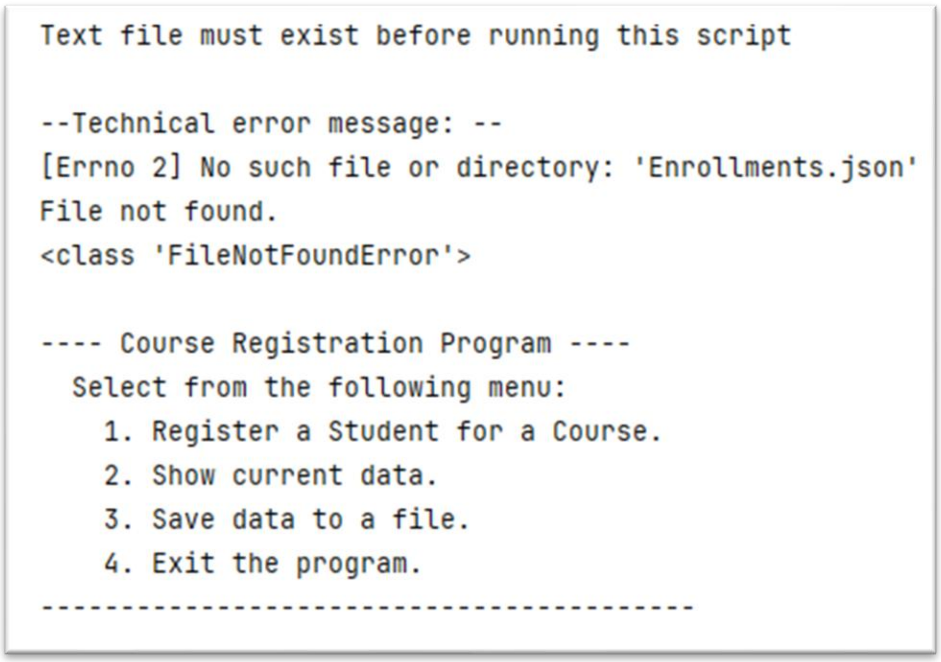


```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------

Please select from the following menu: 1
Enter the student's first name: Test1
Traceback (most recent call last):
  File "C:\Users\CariL\Documents\Python\PythonCourse\Mod05\Assignment\A05\Assignment05-CWilliams.py", line 61, in <module>
    raise ValueError("Only alphabetic characters are allowed.")
ValueError: Only alphabetic characters are allowed.

Process finished with exit code 1
```



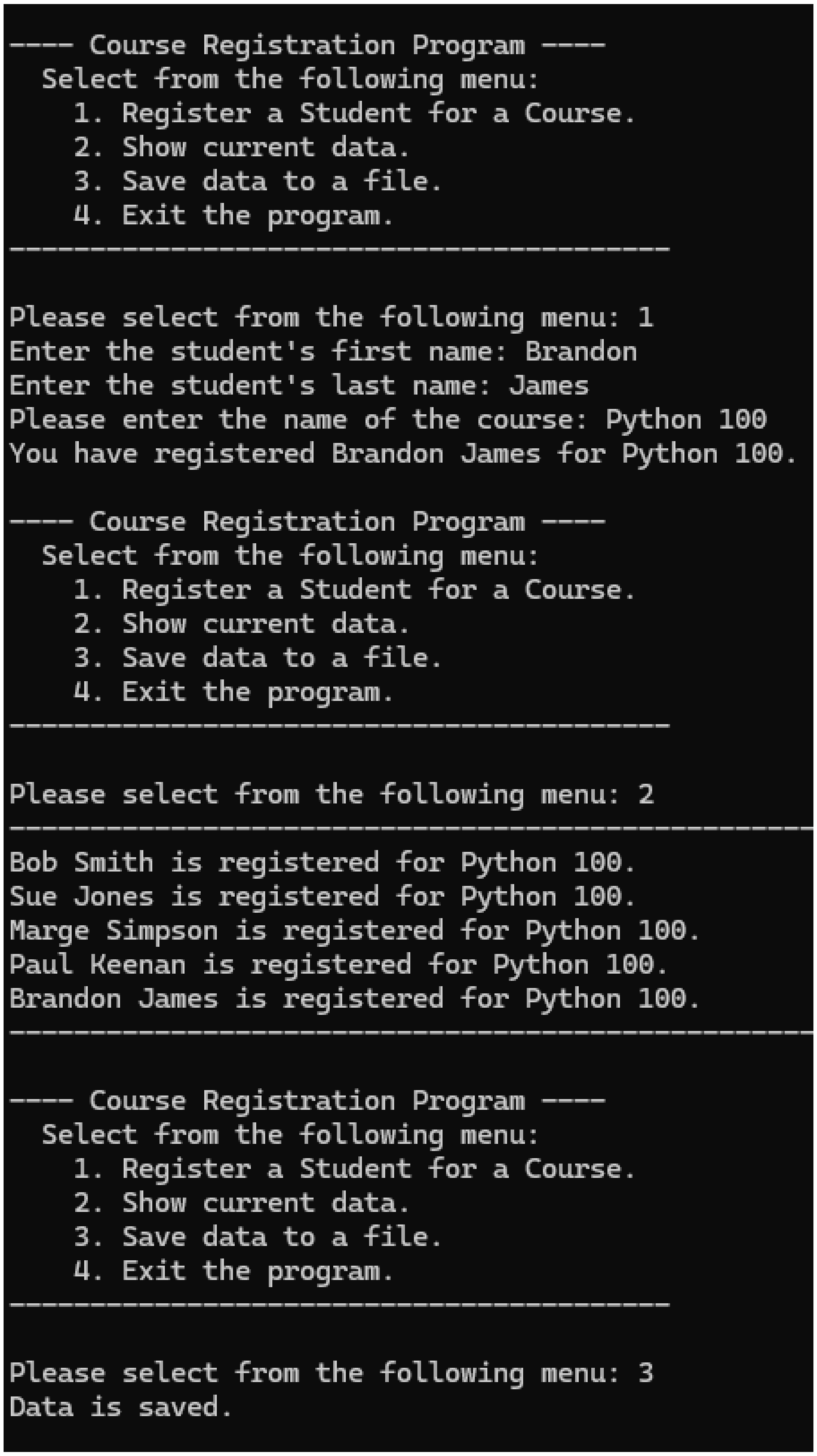
```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------

Please select from the following menu: 1
Enter the student's first name: Test
Enter the student's last name: Tester1
Traceback (most recent call last):
  File "C:\Users\CariL\Documents\Python\PythonCourse\Mod05\Assignment\A05\Assignment05-CWilliams.py", line 65, in <module>
    raise ValueError("Only alphabetic characters are allowed.")
ValueError: Only alphabetic characters are allowed.

Process finished with exit code 1
```

Figure 6: Testing input validation error handling in PyCharm.

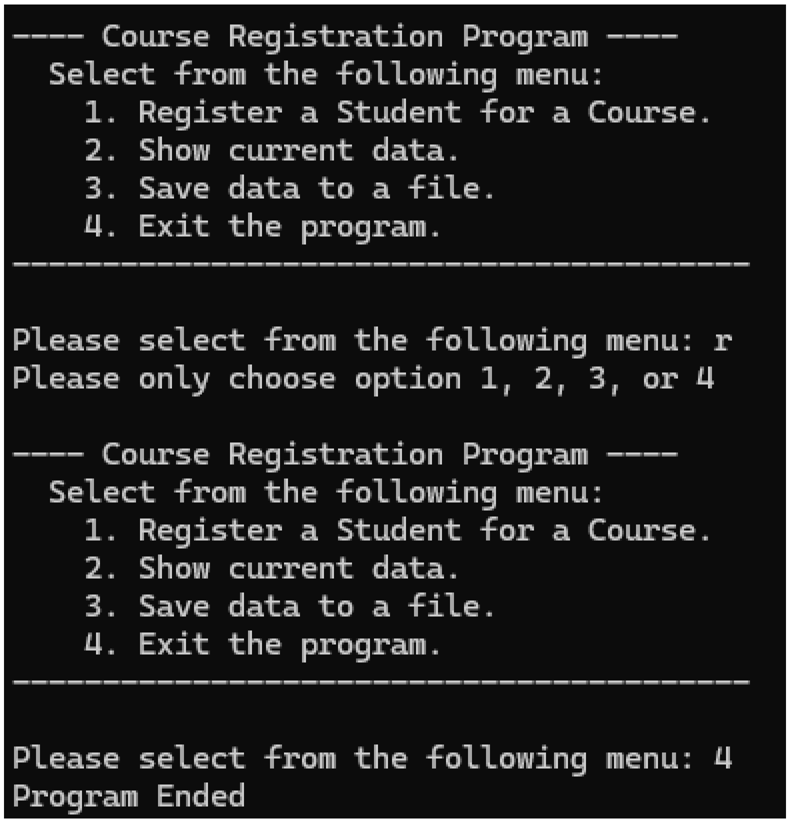


```
Text file must exist before running this script

--Technical error message: --
[Errno 2] No such file or directory: 'Enrollments.json'
File not found.
<class 'FileNotFoundError'>

---- 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 7: Testing FileNotFound Error

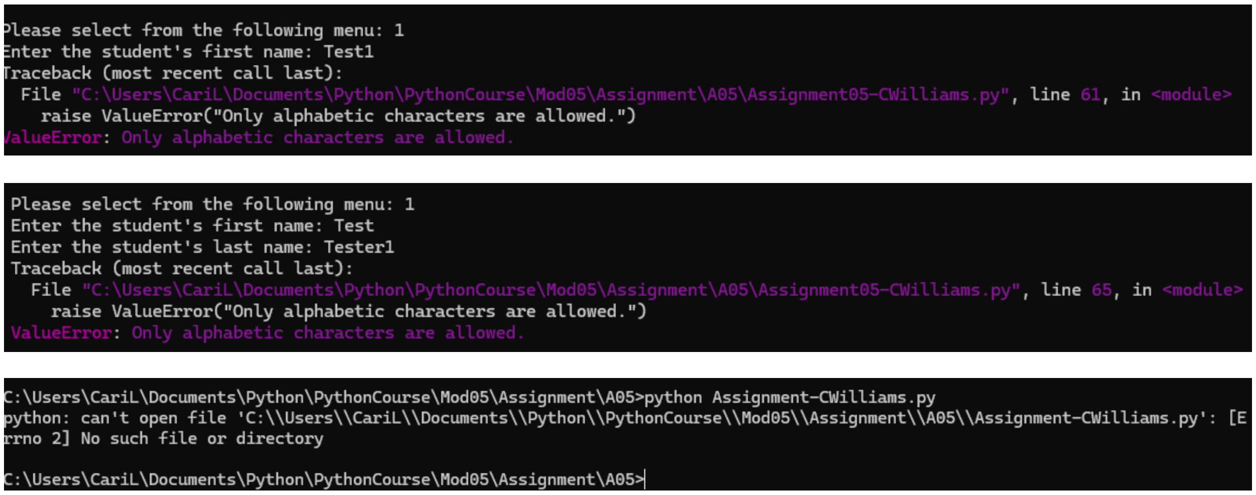


```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------

Please select from the following menu: 1
Enter the student's first name: Brandon
Enter the student's last name: James
Please enter the name of the course: Python 100
You have registered Brandon James 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.
-----------------------------------------

Please select from the following menu: 2
--------------------------------------------------
Bob Smith is registered for Python 100.
Sue Jones is registered for Python 100.
Marge Simpson is registered for Python 100.
Paul Keenan is registered for Python 100.
Brandon James is registered 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.
-----------------------------------------

Please select from the following menu: 3
Data is saved.
```



```
---- 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.
-----------------------------------------

Please select from the following menu: r
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.
-----------------------------------------

Please select from the following menu: 4
Program Ended
```

Figure 8: Testing in cmd



```
Please select from the following menu: 1
Enter the student's first name: Test1
Traceback (most recent call last):
  File "C:\Users\CariL\Documents\Python\PythonCourse\Mod05\Assignment\A05\Assignment05-CWilliams.py", line 61, in <module>
    raise ValueError("Only alphabetic characters are allowed.")
ValueError: Only alphabetic characters are allowed.
```

```
Please select from the following menu: 1
Enter the student's first name: Test
Enter the student's last name: Tester1
Traceback (most recent call last):
  File "C:\Users\CariL\Documents\Python\PythonCourse\Mod05\Assignment\A05\Assignment05-CWilliams.py", line 65, in <module>
    raise ValueError("Only alphabetic characters are allowed.")
ValueError: Only alphabetic characters are allowed.
```

```
C:\Users\CariL\Documents\Python\PythonCourse\Mod05\Assignment\A05>python Assignment-CWilliams.py
python: can't open file 'C:\\Users\\CariL\\Documents\\Python\\PythonCourse\\Mod05\\Assignment\\A05\\Assignment-CWilliams.py': [E
rrno 2] No such file or directory

C:\Users\CariL\Documents\Python\PythonCourse\Mod05\Assignment\A05>
```

Figure 9: Testing error handling

Summary

Assignment 05 strengthened my understanding of dictionaries, JSON file handling, and Python exception handling. I successfully converted the program from CSV to JSON, implemented robust error handling, ensured safe file operations, validated user input, and improved the overall structure of the program. The final script is more flexible, easier to maintain, and follows good programming practices that will be useful in larger Python projects.