Alex Novik

08/13/2025

Foundations of Programming: Python

Assignment 05

https://github.com/AlexBykov03/IntroToProg-Python-Mod05

Completing the 5th Assignment

Introduction

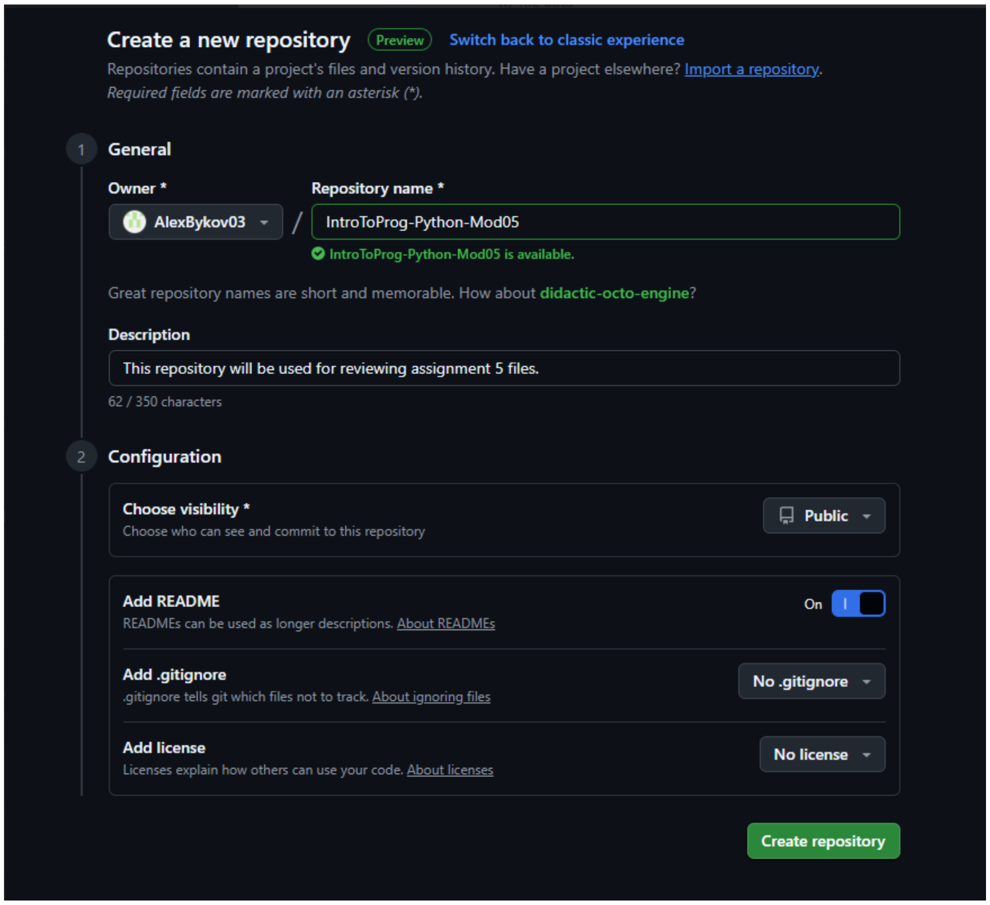
In this paper, I will explain the steps I took to complete the programming assignment from this module. The purpose of the assignment was to adjust the data processing of our menu-based Python program that we completed the previous week; in the same way we changed the data processing from the week before. This assignment had its own challenges and newer tasks. In this document, I will describe the steps I took in bringing everything to completion.

Reading and Preparing

I approached this assignment by first going to the Notes material. Reading through more than 30 pages of material isn't easy or quick – you could say that it isn't easy because it isn't quick. Anyway, I decided to start with the hardest part.

This week's module focused on 4 main concepts, these are: dictionaries, JSON files, Structured Error Handling, and GitHub. Since the goal of this document isn't to go into detail regarding these concepts, I will try to describe these simply in one sentence. Dictionaries are a way of parsing data by keeping a list or a set of data in key-value pairs. JavaScript Object Notation (or JSON) files are files comprised of dictionaries; they are useful in web application and configuration files. Structured Error Handling refers to setting error condition exceptions around parts of the code to avoid breaking the program and possibly informing the user of what could have gone wrong. Finally, GitHub, is an online "repository"; it is a more streamlined and useful alternative to network storage and tracking of code progress.

As I completed reading the notes, I turned to the assignment document. My suspicions were confirmed; these elements were the focus this week. Before I started to write my code – or rather editing the starter file, I set up my GitHub account. It turns out I already had an account and all I had to do was to create a new public repository for this module as was described in the assignment.


Create a new repository Preview Switch back to classic experience

Repositories contain a project's files and version history. Have a project elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

1 General

Owner * AlexBykov03 / Repository name * IntroToProg-Python-Mod05

IntroToProg-Python-Mod05 is available.

Great repository names are short and memorable. How about didactic-octo-engine?

Description: This repository will be used for reviewing assignment 5 files.

62 / 350 characters

2 Configuration

Choose visibility * — Choose who can see and commit to this repository: Public

Add README — READMEs can be used as longer descriptions. About READMEs: On

Add .gitignore — .gitignore tells git which files not to track. About ignoring files: No .gitignore

Add license — Licenses explain how others can use your code. About licenses: No license

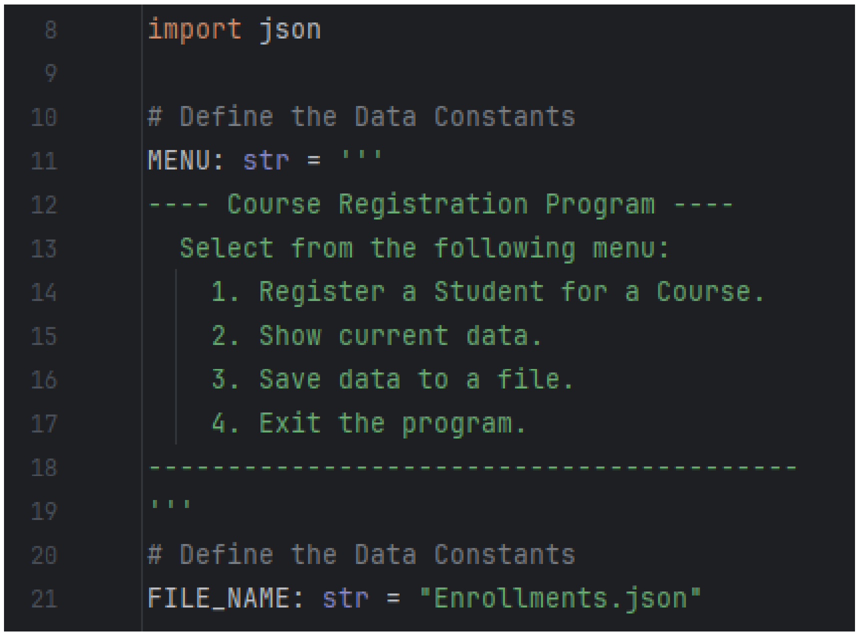
Figure 1. Creating a new GitHub repository.

After I complete writing this document, I will still have the task of compressing my work and uploading it onto Canvas and GitHub.

Writing the Code

Like I mentioned before, we were provided with a starter file that was a complete version of the previous assignment. To tailor it to this module, we had to adjust the methods of execution – the way we worked with the data. Here is how I went about making these changes.

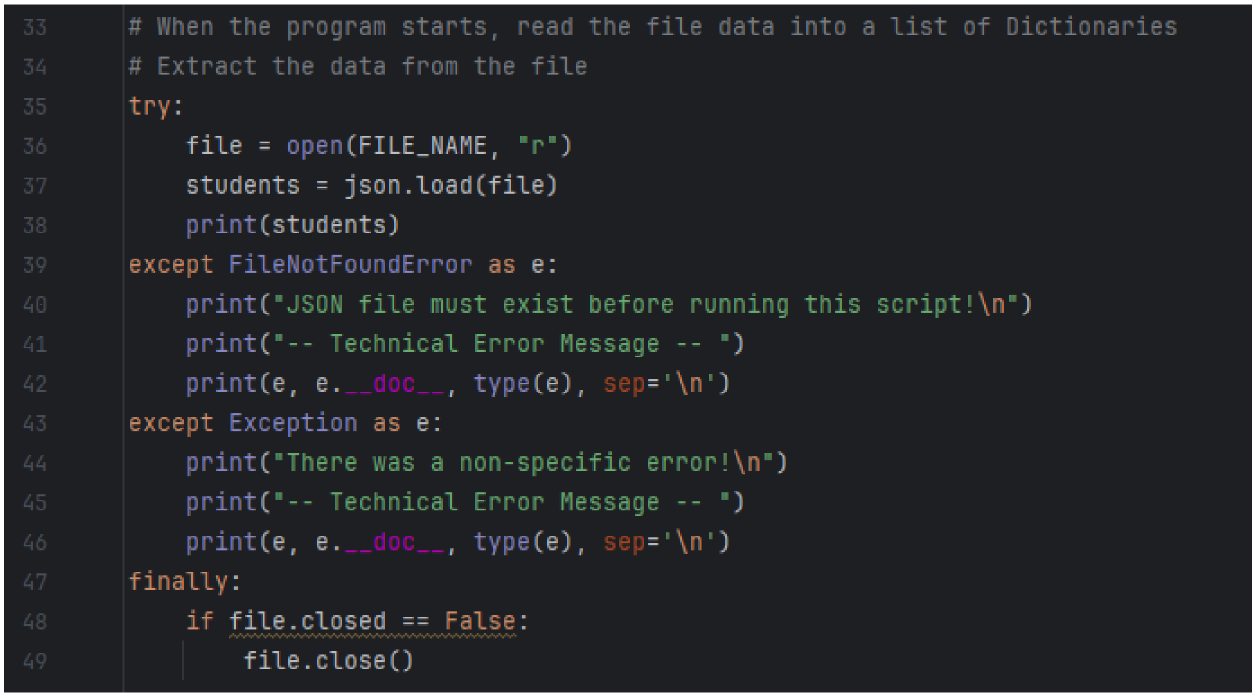
First, the file that I am working with is a JSON file and not a CSV. Thus, the “FILE_NAME” constant had to be changed to “Enrollments.json”. To work with this type of file more easily, I could use the json library of functions such as load() and dump(); but to do that I had to first import the json library.



```
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"
```

Figure 2. First adjustments.

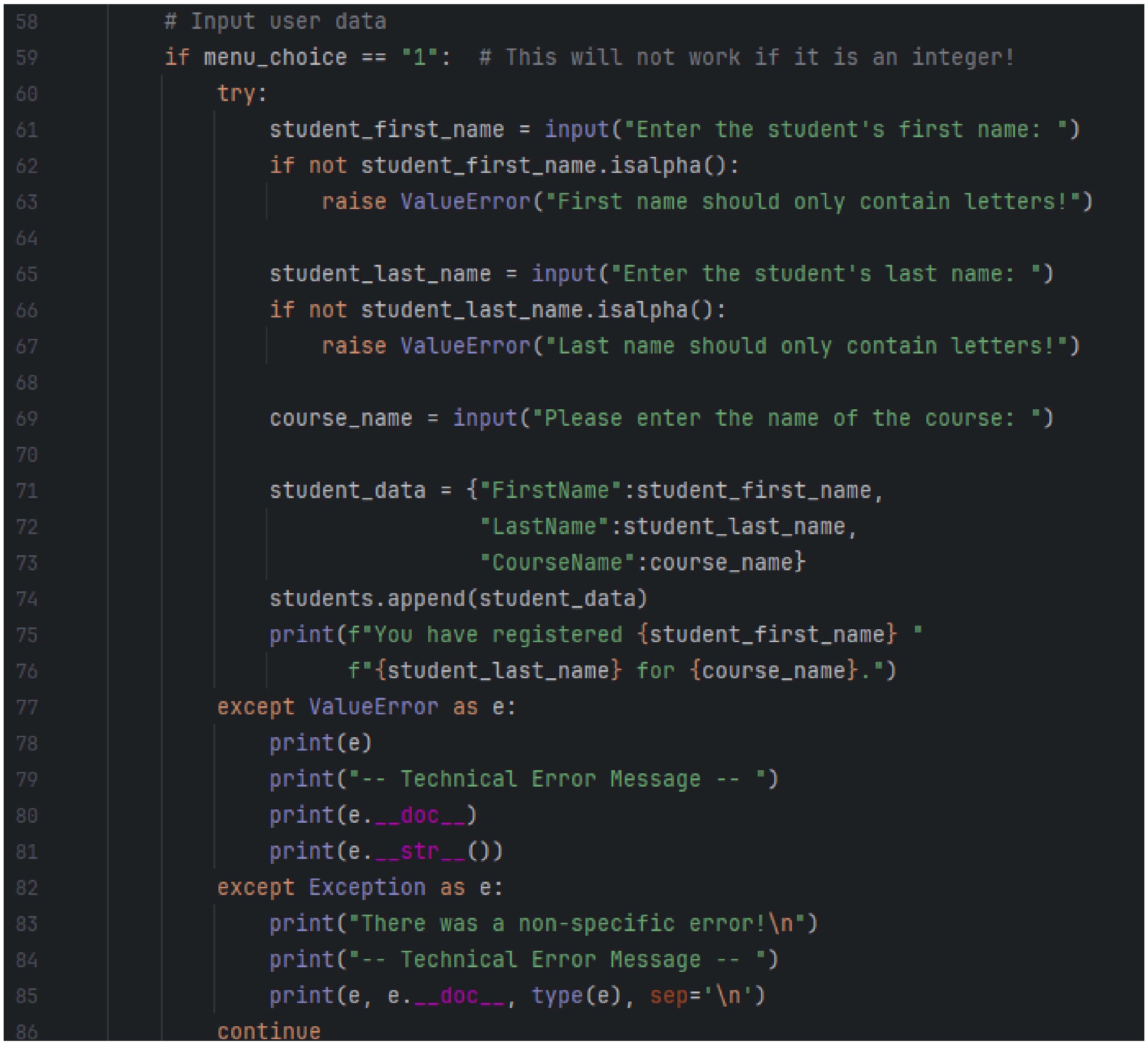
Next, I had to open the JSON file into the file variable using the argument “file = open(FILE_NAME, “r”)”. Then load the data from the file into “students” using the “json.load(file)” function. At first, I tried doing this part the way it was shown in one of the demos – by reading the data through a loop line by line. However, I realized that this function loads the data over into list of dictionaries. You can refer to the figure below to see the way I implemented the code.



```
# When the program starts, read the file data into a list of Dictionaries
# Extract the data from the file
try:
    file = open(FILE_NAME, "r")
    students = json.load(file)
    print(students)
except FileNotFoundError as e:
    print("JSON 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 non-specific error!\n")
    print("-- Technical Error Message -- ")
    print(e, e.__doc__, type(e), sep='\n')
finally:
    if file.closed == False:
        file.close()
```

Figure 3. Reading the data from the JSON file.

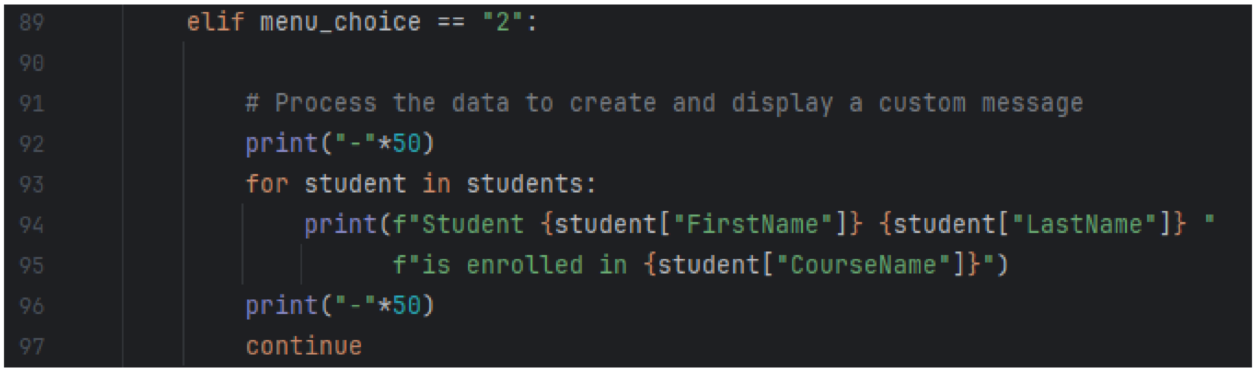
For the first menu choice code, all that needed to be done was assign the variables taken from user input and assigning them to keys in the dictionary (in the same key format of the JSON file). Refer to the figure below for my code implementation.



```python
    # Input user data
    if menu_choice == "1":  # This will not work if it is an integer!
        try:
            student_first_name = input("Enter the student's first name: ")
            if not student_first_name.isalpha():
                raise ValueError("First name should only contain letters!")

            student_last_name = input("Enter the student's last name: ")
            if not student_last_name.isalpha():
                raise ValueError("Last name should only contain letters!")

            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_first_name} "
                  f"{student_last_name} for {course_name}.")
        except ValueError as e:
            print(e)
            print("-- Technical Error Message -- ")
            print(e.__doc__)
            print(e.__str__())
        except Exception as e:
            print("There was a non-specific error!\n")
            print("-- Technical Error Message -- ")
            print(e, e.__doc__, type(e), sep='\n')
        continue
```

Figure 4. Code for menu choice 1.

Using a loop, I made a print statement for menu choice 2 and called out the required values using keys.

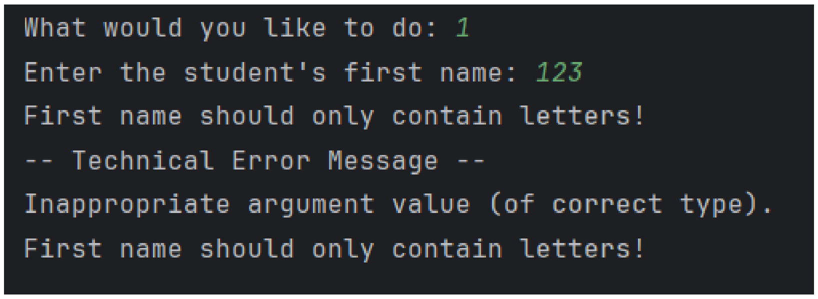


```
        elif menu_choice == "2":

            # Process the data to create and display a custom message
            print("-"*50)
            for student in students:
                print(f"Student {student["FirstName"]} {student["LastName"]} "
                      f"is enrolled in {student["CourseName"]}")
            print("-"*50)
            continue
```

Figure 5. Code for menu choice 2.

For menu choice 3, I now had to use the “json.dump()” function to save the data to the JSON file.

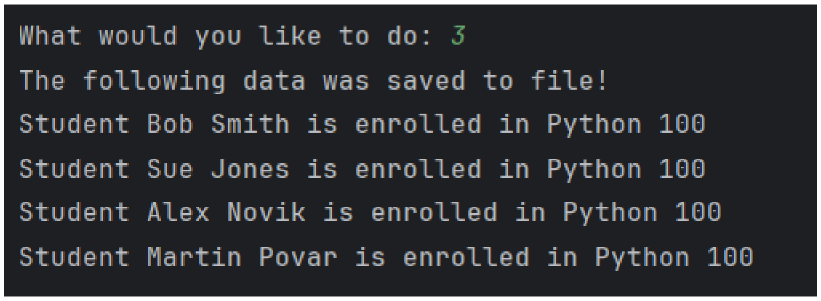
The assignment also required the implementation of exceptions for structured error handling. As you can see in most of the previous figures, I wrapped the exceptions around the code.



```
What would you like to do: 1
Enter the student's first name: 123
First name should only contain letters!
-- Technical Error Message --
Inappropriate argument value (of correct type).
First name should only contain letters!
```

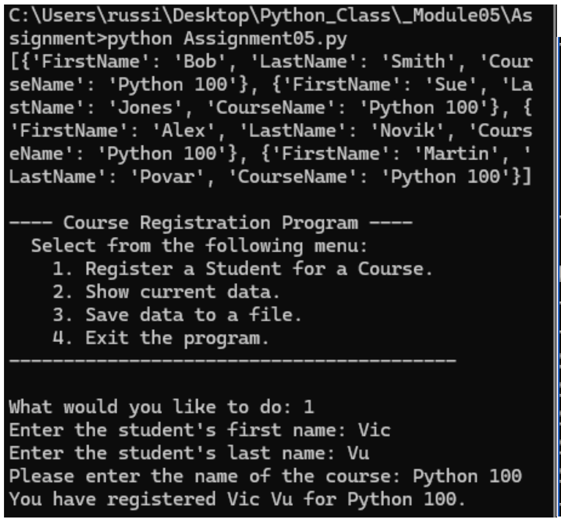
Figure 6. Testing value error exception.

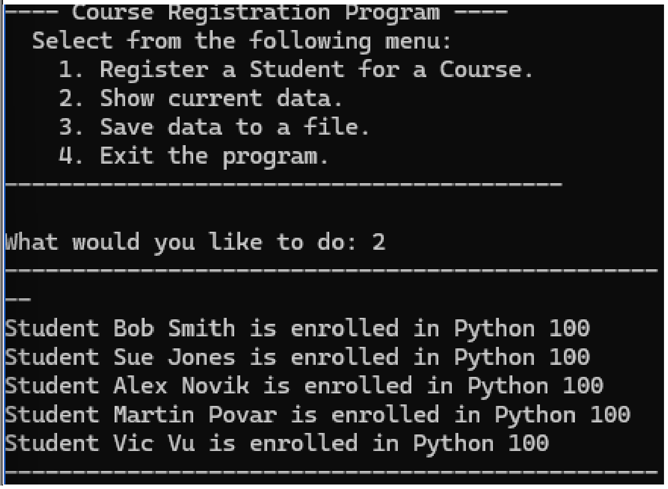
Finally, I tested the code to make sure that it and the exception worked as expected.

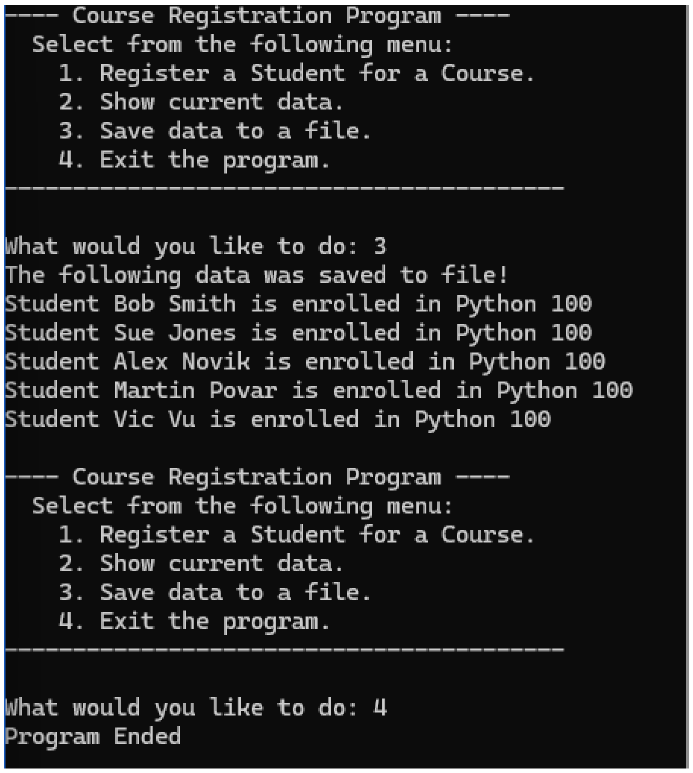


```
What would you like to do: 3
The following data was saved to file!
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Alex Novik is enrolled in Python 100
Student Martin Povar is enrolled in Python 100
```

Figure 7. Testing the code in PyCharm. Successfully saving the data.


```
C:\Users\russi\Desktop\Python_Class\_Module05\As
signment>python Assignment05.py
[{'FirstName': 'Bob', 'LastName': 'Smith', 'Cour
seName': 'Python 100'}, {'FirstName': 'Sue', 'La
stName': 'Jones', 'CourseName': 'Python 100'}, {
'FirstName': 'Alex', 'LastName': 'Novik', 'Cours
eName': 'Python 100'}, {'FirstName': 'Martin', '
LastName': 'Povar', 'CourseName': '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.
-----------------------------------------

What would you like to do: 1
Enter the student's first name: Vic
Enter the student's last name: Vu
Please enter the name of the course: Python 100
You have registered Vic Vu 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.
-----------------------------------------

What would you like to do: 2
------------------------------------------------
--
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Alex Novik is enrolled in Python 100
Student Martin Povar is enrolled in Python 100
Student Vic Vu is enrolled in 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.
-----------------------------------------

What would you like to do: 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 Alex Novik is enrolled in Python 100
Student Martin Povar is enrolled in Python 100
Student Vic Vu is enrolled in 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.
-----------------------------------------

What would you like to do: 4
Program Ended
```

Figures 8, 9, 10. Running the code in Command Prompt.

Now it is time to upload the code (and this document) onto GitHub.

Summary

In this assignment, I learned a lot about dictionaries, JSON files, exceptions, and GitHub. There seems to be a broad pattern of building on previous knowledge to learn and create something new. Completing the work takes time, but the growth of knowledge and experience is in its own way rewarding. I am looking forward to what we will accomplish next week.