Rachael Leahy
18 August 2025
IT FDN 110A
Assignment 05
https://github.com/outlawthruhiker/IntroToProg-Python-Mod05

Advanced Collections and Error Handling

Introduction

This week's course material discussed dictionaries, working with JSON files, structured error handling, and managing code files in general and via GitHub specifically.

The assignment tasked us with creating a script that demonstrated our ability to open and use a JSON file, save information in a dictionary format, and use structured error handling. The codes continue to build on each other, so I will only cover in this document what I added and changed to the code I submitted in Assignment 4 (see **bold** items).

Creating the Script

Assignment Directives

Our assignment this week included multiple directives. Those that differ from or add onto Assignment03.py are called out in bold and described later in the document. Steps repeated from Assignment03.py (mostly included in the starter script) are not presented in bold and not detailed further in this document.
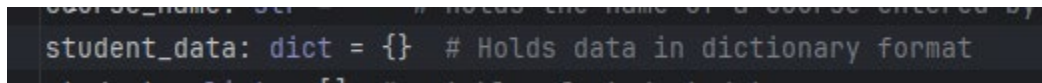
1. Name the .py file "Assignment05."
2. Populate an appropriate script header.
3. Create constants:
   a. "MENU" set to a multi-line string value.
   b. "FILE_NAME" set to the string value, "Enrollments.json." Last week this file was a .csv instead of a .json, but I'm going to trust that you know I know how to create a basic file outside of Python at this point.
4. Create variables:
   a. A string variable, "student_first_name," populated with user input.
   b. A string variable, "student_last_name," populated with user input.
   c. A string variable, "course_name" populated with user input.
   d. An object, "file_obj" set to None.
   e. A string variable, "menu_choice" populated with user input which directs the loop.
   f. **A dictionary variable, "student_data" set to an empty dictionary variable.**
   g. A list variable, "students," set to an empty list.
5. **Upon opening the script, it was to read the contents of "Enrollments.json" automatically into the list variable 'students' as a two-dimensional list.**

6. Allow the user to pick a menu choice (1, 2, 3, 4) and create a script loop which completes various actions based on the menu choice.
    a. Menu choice 1.) The user is prompted to enter the student's first and last names, as well as the course name. These user entries are then stored in the respective variables, **as well as added to the dictionary named student_data. Student_data is then appended to students.**
    b. *Menu choice 2.)* The data entered are printed to the screen as a string of comma-separated values for each row collected in the 'students' variable.
    c. **Menu choice 3.) The entered data are saved to Enrollments.json and the data saved in the file are displayed to the user.**
    d. Menu choice 4.) The program prints, "Program Ended" and the loop is ended.
    e. **The program provides structured error handling when:**
        i. **The file is read into the list of dictionary rows**
        ii. **The user enters their first and last names**
        iii. **When the dictionary rows are written to the file.**

Completing the Assignment
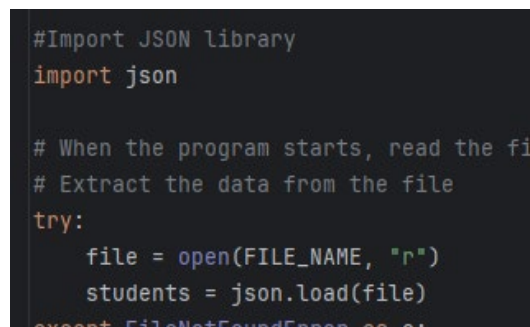
To accomplish these goals, I did the following.

1. I created a dictionary variable, "student_data" set to an empty dictionary variable (Figure 1).


```
student_data: dict = {}  # Holds data in dictionary format
```

Figure 1. Screenshot showing "student_data" dictionary variable

2. On opening, the script reads the contents of "Enrollments.json" into the list, "students" (Figure 2).


```
#Import JSON library
import json

# When the program starts, read the fi
# Extract the data from the file
try:
    file = open(FILE_NAME, "r")
    students = json.load(file)
```

Figure 2: "Enrollments.json" is read into the list variable "students"

3. Upon choosing Menu Choice 1, user-entered first name, last name, and course name are added to the dictionary "student_data" and appended to "students" (Figure 3).

```python
student_data = {
    "Name": student_first_name + " " + student_last_name,
    "Course": course_name
}
students.append(student_data)
print(f"You have registered {student_first_name} {student_last_name} for {course_name}.")
continue
```

Figure 3: Data are added to the dictionary variable and subsequently appended to the list variable.

4. I created structured error handling for when the file is read into the list of dictionary rows (Figure 4), the user enters their first and last names (Figure 5), and the dictionary rows are written to the file (Figure 6).

```python
try:
    file = open(FILE_NAME, "r")
    students = json.load(file)
except FileNotFoundError as e:
    if file.closed==False:
        file.close()
    print("JSON file must exist before running this script.\n")
    print("Built in Python error info:")
    print(e, e.__doc__, type(e), sep="\n")
except json.JSONDecodeError as e:
    if file.closed==False:
        file.close()
    print("JSON file is invalid. \n")
    print("Built in Python error info:")
    print(e, e.__doc__, type(e), sep="\n")
except Exception as e:
    if file.closed==False:
        file.close()
    print("There was a non-specific error.\n")
    print("Built in Python error info:")
    print(e, e.__doc__, type(e), sep="\n")
```

Figure 4: Structured error handling for when the file is read into the list of dictionary rows.

```python
if menu_choice == "1":  # This will not work if it is an integer!
    while True:
        try:
            student_first_name = input("Enter the student's first name: ")
            if not student_first_name.isalpha():
                raise ValueError("The first name must contain only letters.")
            break
        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("Built in Python error info:")
            print(e, e.__doc__, type(e), sep="\n")
    while True:
        try:
            student_last_name = input("Enter the student's last name: ")
            if not student_last_name.isalpha():
                raise ValueError("The last name must contain only letters.")
            break
        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("Built in Python error info:")
            print(e, e.__doc__, type(e), sep="\n")
```

Figure 5: Structured error handling for users entering their fist and last names.

```python
elif menu_choice == "3":
    try:
        file = open(FILE_NAME, "w")
        json.dump(students, file)
        file.close()
        print("The following data are saved to the file:")
        for student in students:
            print(f"{student['Name']}, {student['Course']}")

    except PermissionError as e:
        if file.closed == False:
            file.close()
        print("Permission denied. Cannot write to file. \n")
        print("Built in Python error info:")
        print(e, e.__doc__, type(e), sep="\n")
    except Exception as e:
        if file.closed == False:
            file.close()
        print("There was a non-specific error.\n")
        print("Built in Python error info:")
        print(e, e.__doc__, type(e), sep="\n")
    continue
```

Figure 6: Structured error handling for when the dictionary rows are written to the file.

Testing the Script

1. To test the script in PyCharm, I first saved the final draft and then ran it.
   a. After choosing menu option 1.), I was prompted to enter a first and last name, and a course name (Figure 7) and I was prompted to make another selection.
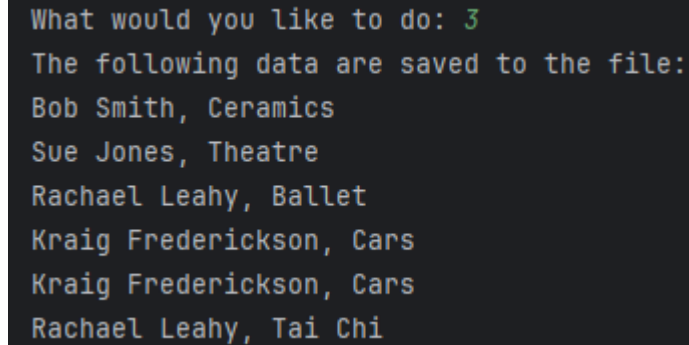

Figure 7: Screenshot showing the input prompts and print command output in PyCharm.

   b. After choosing menu option 2.), the entered data were printed to the screen and I was prompted to make another selection.
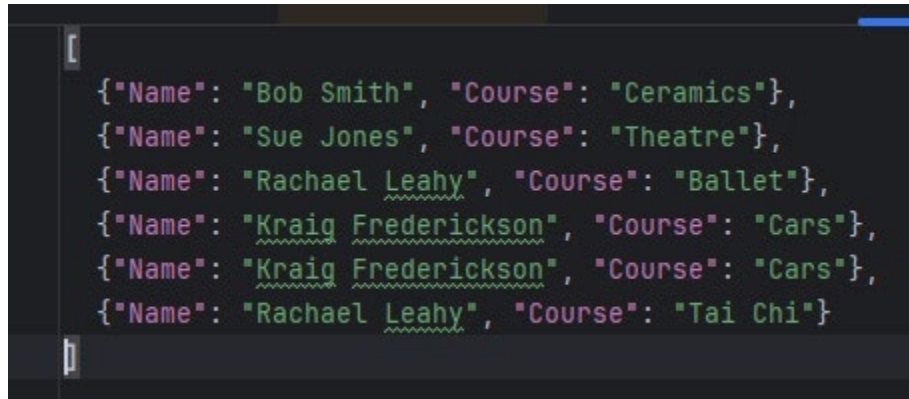

Figure 8: Screenshot showing the input data printed to the screen after selecting menu choice 2.

c. After choosing menu option 3.), the entered data were also printed to the screen (Figure 9) and the data were written into "Enrollments.json" (Figure 10).

Figure 9: Screenshot showing the entered data printed to the screen after selecting menu choice 3.

Figure 10. Screenshot showing the input data entered into a .json file called "Enrollments.json."

2. To test the script in the Command Line, I first navigated to the appropriate directory and then entered the command 'python "Assignemnt05.py"' both to open the appropriate file to be read and to ensure the computer interpreted the file as a python script. Results were identical to those in PyCharm and are presented below (Figures 11 & 12).

```
---- 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: Sarah
Enter the student's last name: Leahy
Please enter the name of the course: Drivers Ed
You have registered Sarah Leahy for Drivers Ed.

---- 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
----------------------------------------------------
Bob Smith, Ceramics
Sue Jones, Theatre
Rachael Leahy, Ballet
Kraig Frederickson, Cars
Kraig Frederickson, Cars
Rachael Leahy, Tai Chi
Sarah Leahy, Drivers Ed
----------------------------------------------------

---- 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 are saved to the file:
Bob Smith, Ceramics
Sue Jones, Theatre
Rachael Leahy, Ballet
Kraig Frederickson, Cars
Kraig Frederickson, Cars
Rachael Leahy, Tai Chi
Sarah Leahy, Drivers Ed

---- 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.
----------------------------------------
```
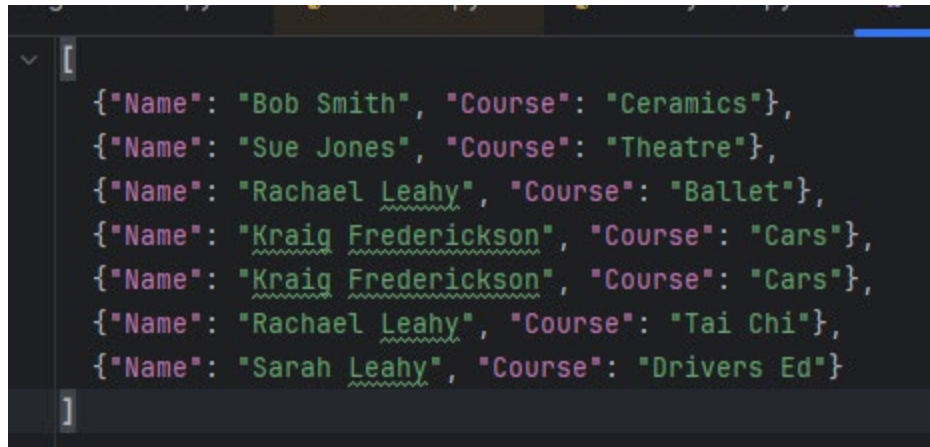
Figure 11. Screenshot showing output from Command Line for this assignment

Figure 12. .json file after running the script through the Command Line

Summary

This was a good assignment, but creating these documents takes *a lot* of time and there's very little educational value add.