Nathan Thomas

November 11, 2025

IT FND 110 A

Assignment 5

https://github.com/nthomas28/IntroToProg-Python-Mod05

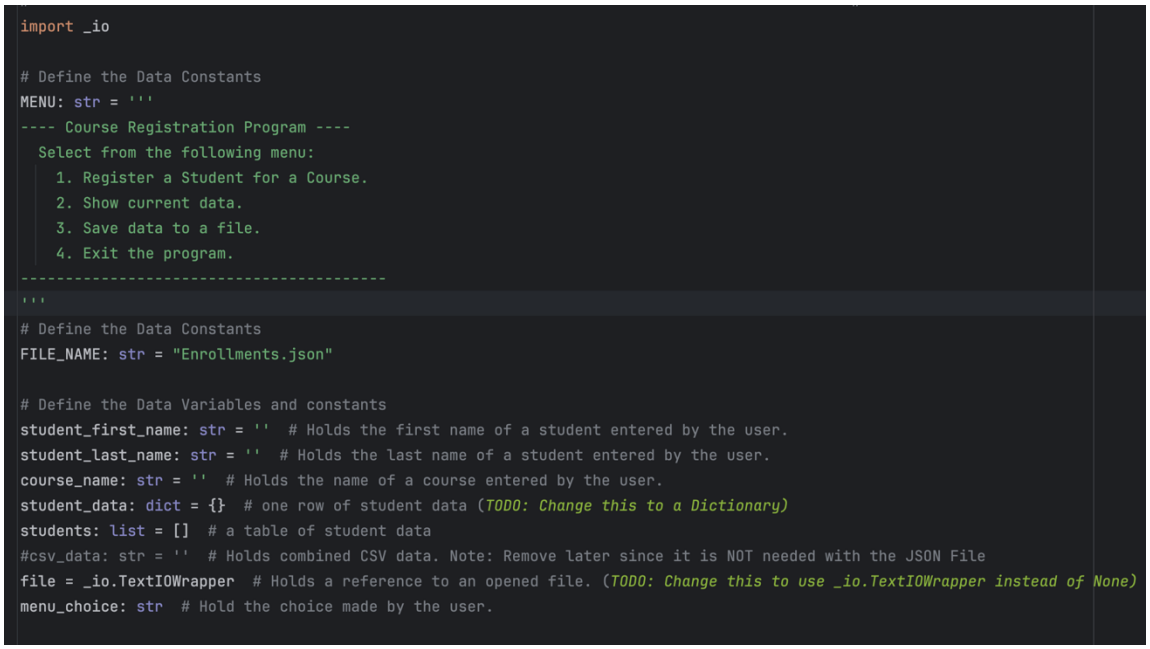
Programming with Advanced Collections of Data

Introduction

For this assignment, we were instructed to create a program that uses dictionaries and exception handling to present and save data to a **.json** file. Unlike last week's assignment, where we used **.csv** files, **.json** files enable us to accomplish the same data processing but with less code required. This program will also demonstrate exception handling by presenting exception error messages wherever a user's input doesn't match the class type prompted. For example, an exception error will show if a user inputs a numerical value instead of an alphabetical name.

Defining Constants & Variables

For this program, we needed to change a constant and a few variables to reflect our **.json** file instead of our previously used **.csv** file. Our **FILE_NAME** constant has been altered to pull up a **.json** file, which we will use to present our student data in this program. The variable **student_data** has been set to an empty dictionary instead of a list like we've used in previous assignments. Our variable, **file** has been set to **_io.TextIOWrapper,** which would enable us to utilize the **file.close()** command to close our **.json** file.



```
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"

# 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 (TODO: Change this to a Dictionary)
students: list = []  # a table of student data
#csv_data: str = ''  # Holds combined CSV data. Note: Remove later since it is NOT needed with the JSON File
file = _io.TextIOWrapper  # Holds a reference to an opened file. (TODO: Change this to use _io.TextIOWrapper instead of None)
menu_choice: str  # Hold the choice made by the user.
```

Figure: These are the constants and variables used in this program.

Reading Current Data

For this program, we're attempting to pull data from a **JSON** file and add it to our **students** list to present all current data to a user. After initially opening our **JSON** file under "**read"**, we can use the **json.load(file)** code to extract any data currently in our **Enrollments.json** file into our program. Added into this section is our first exception handling, where an error will be presented if the **.json** file does not exist or an unspecified error occurs. I built this using a **try/except** function, where this program will "try" to open our **JSON** file unless an exception appears, in which this program will present an error message. For our "file does not exist" error, we needed to include the exception specification **FileNotFoundError** since this is an error we're specifically trying to prevent. For all other exception errors, we use the **Exception** specification instead.

```
try:
    file = open(FILE_NAME, "r")
    students = json.load(file)
    file.close()
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 non-specific error!\n")
    print("-- Technical Error Message -- ")
    print(e, e.__doc__, type(e), sep='\n')
finally:
    if file.closed == False:
        file.close()
```

Figure: This code opens and reads our JSON file and saves the data to our students list.

Saving New Data to Our List

Menu option 1 is the first condition of the **if:** loop used in this program. This information will be formatted as a comma-separated value and added to the variable **student_data** along with the already saved data from **Enrollments.json**. I performed this by adding the code **students.append(student)**, which will add our user's data to our current **students** dictionary. Included in this section of the program is our second and third exceptions, which would prevent a user from inputting a numerical value instead of an alphabetical value. To accomplish this, the code **if not student_first_name.isalpha(),** which asks if the user input was an alphabetical value. If this is **true** our program will present a **ValueError** stating that a first name shouldn't contain a number. I repeated this code for our **student_last_name** variable as well.

```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("The first name should not contain numbers.")

        student_last_name = input("Enter the student's last name: ")
        if not student_last_name.isalpha():
            raise ValueError("The last name should not contain numbers.")

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

Figure: This code prompts a user input, which is saved to our dictionary.

Presenting the Current Data

If a user were to select menu option 2, this program would print all current registration data in the **Enrollments.json** file. Now that there is more than one condition for the **if:** loop in this program, I used **elif**. I also included a nested **for** condition, which identifies each student or row of my **students** list to be printed on a separate row. If menu option 2 were to be selected first, this program would present all current data saved to **Enrollments.json**.

```python
# 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(student)
    continue
```

Figure: This code presents the current data back to the user unless unavailable.

Saving to a .CSV File

When a user selected menu option 3, I wanted to save the information stored in my list **students** from our **Enrollments.json**. For this, I used my constant **FILE_NAME,** which I had previously assigned to the **.json** document. Unlike when I extracted that data from the start of my program, I used the **write** mode so that the save .**json** document would be overwritten. This wouldn't affect any student data that has already been saved to our **Enrollments.json**, because we already extracted the saved information to our **students** list at the beginning of this program. Our current **students** variable will include that data previously included into my **JSON** file. I included our final exception errors in this section, where our program will verify that our **students** list is formatted for a **JSON** file before being added to our **Enrollments.json**.

```
# Save the data to a file
elif menu_choice == "3":
    try:
        file = open(FILE_NAME, "w")
        json.dump(students, file)
        file.close()
        continue
    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.closed == False:
            file.close()
```

Figure: The code used to save our data to a separate file called "Enrollments.csv".

Closing the Menu Loop and Ending the Program

The final menu option, 4, would be to end the program. **Break** is used to end our menu selection loop, which was built using **if, elif** and **else** commands. Included is a final **else()** command, presenting a request for a user to select a proper option from our menu if they haven't.

```
    # 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")
```

Figure: This code closes our program for menu option 4.

Summary

This program uses dictionaries, lists, **JSON** files and exception handling to create a student registration program that would allow a user to access current data and add more student registrations to our **Enrollments.json** file. This is accomplished by first reading and extracting the current data from our **JSON** file to be saved to our **students** list. A user can add new student registration information to **students** variable, which would build upon our existing list. Using exception handling, this program can manage potential user errors by presenting a custom error message to the user after improper inputs.