Rabiya Wasiq

11/12/2023

Introduction to Programming with Python

Assignment05

ADVANCED COLLECTIONS AND ERROR HANDLING

Introduction

This week we went further into Advanced collections of data, namely dictionaries, and lists of dictionaries. We learned to add and remove data from dictionaries and then subsequently adding data from dictionaries to files and reading data from files into dictionaries. We also learned about JSON files and the code to write and read data from JSON files.
We also learned error handling by way of defining exceptions and raising specific exceptions that we anticipate from a specific code.

Creating Python Script

For the purpose of this assignment I will be adding to the code from Assignment 04. I will be using the same Python script and demonstrating reading and writing data from a JSON file. I will demonstrate the use of dictionaries to store data. We will be storing our student data in the form of a list of dictionaries instead of a list of lists.
We will also see the various possible methods of handling that may arise when writing or reading data from files and when receiving input from the user.

Dictionaries

Dictionaries are used to store data values in key:value pairs.

A dictionary is a collection which is ordered*, changeable and does not allow duplicates. Dictionaries are written using curly brackets, dictionary items are presented in key:value pairs, and can be referred to by using the key name.

Example : {‘First_Name”:”Vic”, “Last_Name”:”Vu”}

JSON Files

JSON stands for JavaScript Object Notation and is a lightweight format for storing and transporting data.

Python has the built-in module json , which allow us to work with JSON data.

Starter Code from Assignment04

We start by using Open Tab in Pycharm and navigating to the location of Python script of Assignment 04. We will copy the script and make changes as we proceed.

We will use the same format of adding a script header and using pseudocode for problem solving.

```
# ------------------------------------------------------------------------
-------------- #
# Title: Assignment05
# Desc: This Assignment Demonstrates Advanced Collections and Error Handling
# Change Log: (Who, When, What)
#   Rabiya Wasiq,11/11/2030,Created Script
# ------------------------------------------------------------------------
-------------- #
```

Figure 1 – Script Header

Declaring and assigning Variables / Constants

Declaring and assigning variables are the first steps to writing a code. It is best practice to declare the variables and constants that you intend to use throughout the script at the beginning.

```
# Define the Data Constants
MENU: str = '''
---------------------------------------
---- Course Registration Program ----
  Select from the following menu:
    1. View all students registered to date
    2. Register a New Student for a Course
    3. Show New student registration details
    4. Save New student data to a file
    5. Exit the program

---------------------------------------
'''
import json
from json import JSONDecodeError
# Define the Data Variables

student_first_name: str = ''
student_last_name: str = ''
```

```
course_name: str = ''
csv_data: str = ''
FILENAME: str = 'Enrollments.json'
file = None
menu_choice: str = ''
second_student: str = ''
student_data: list = []
students: list[dict[str,str]] = []
```

Figure 2: Declaring and assigning constants and variables.

In (**Figure 2)** I have defined the constants and variables that I intend to use in this program.

You can see that I have used '''triple quotation marks''' for my CONSTANT MENU , this preserves the formatting in the string and we do not need to use escape characters to add additional formatting. I have assigned my variables with empty string values except for the variable file which has None value assigned. My variable student_data is a list whereas the variable students is a two dimensional list of dictionaries.

Read data from JSON file and store in a variable for the program

As mentioned earlier I want to read data recorded in a JSON file and store it to variable in my program. My file contains data in the form of list of dictionaries. To read data from a JSON file I will start by importing the built in JSON Module in Pycharm that allows us to work with JSON data.

```
import json
from json import JSONDecodeError
```

My Json file currently contains the following data **(Figure 3)**

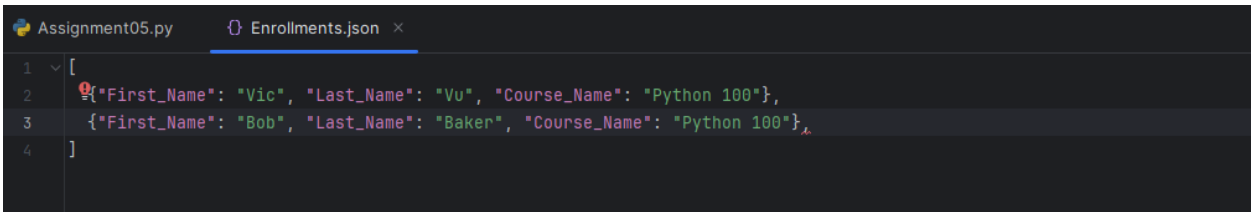

Figure 3 – JSON file with existing data

To read data from JSON file I open the file in read mode and use the code students = json.load(file), **(Figure 4)**this function reads the data from the data from the file and loads it in the variable students, while maintaining the same format it is stored in the file. As JSON supports objects, strings, integers, floats and Booleans we don't need to parse data while importing or use for loops as in the case of csv or txt files,

```
# Read Data from Json file
try:
    file = open(FILENAME, 'r')
    students = json.load(file)
    file.close()
```

```
except FileNotFoundError as e:
    print(e, e.__doc__, e.__str__(),type(e), sep='\n')
    print('Json file not found, creating it...')
    file = open(FILENAME, 'w')
except JSONDecodeError as e:
    print(e, e.__doc__, e.__str__(), type(e), sep='\n')
    print('Json file does not contain any data, resetting it..')
    file = open(FILENAME, 'w')
    json.dump(students, file)
except Exception as e:
    print('Unexpected Technical error')
    print('-' * 50)
    print(e, e.__doc__, e.__str__(), type(e), sep='\n')
finally:
    if not file.closed:
        file.close()
```

Figure 4 – Reading data from Json file

In (**Figure 4**) I have wrapped my code in a try and except function. I know that we may encounter a few known errors when opening and reading data from files, hence I have created exceptions for those errors as I don't want my program to crash. The first one is the File not found error, in the case my file does not exist Python would create a file by that name. The second one is the JSONDecodeError, in the case my file exists but does not have any data to read, Python would open the file in write mode and add data from the variable students that I had declared and assigned value of empty string at the beginning of the code.

The third one is the catch all exception which would show the description of any other error that may arise.

The Finally block is the catch all which would be executed in all the scenarios, it ensures that we close the file for all the options.

While loop

As we move on with the code we add the While Loop which prompts the user with a menus of choices, menu option 1 would show the data read from the Json file formatted using string formatting. As our data in stored in the form of list of dictionaries, we use a for loop to iterate over each dictionary and can access values using Key Names instead of index **(Figure 5)**

```
while True:

    # Present the menu of choices

    print(MENU)

    # Input user data
    menu_choice = input("Please choose an option from the above menu: ")
    print()

    # Menu choice 1 shows the data extracted from the JSon and saved in the
two-dimensional list
    if menu_choice == "1":
```

```
        for student_data in students:
            print(f'Student Full Name :{student_data["First_Name"]}
{student_data["Last_Name"]} | Course Name : {student_data["Course_Name"]}')

    # Getting student details from the user
```

Figure 5 - While Loop prompting the user to enter menu choice and formatting data from the JSON file.

I now want the user to choose the menu option, the program will then execute further based on the input received here from the user.

Raising exceptions while getting input from user

Since our program relies heavily on input from the user, we want to ensure that the user adds input in the correct format. Here we want to make sure that the user enters alphabetic characters only for the Student First and Last Name. For the said purpose I have raised a value error, which tells Python to raise a Valuerror when the user enters a numeric value and prompt on the screen for the user to add the correct value. Wrapping the code in a While Loop makes sure that user is prompted to enter the student name till they enter it in the correct format which breaks the loop (**Figure 6)**

```
# Getting student details from the user

elif menu_choice == "2":
    while True:
        try:
            student_first_name = input("Enter the student's first name: ").capitalize()
            if not student_first_name.isalpha():
                raise ValueError('Student First Name can only contain alphabetic characters')
            break
        except ValueError as e:
            print(e)


    while True:
        try:
            student_last_name = input("Enter the student's last name: ").capitalize()
            if not student_first_name.isalpha():
                raise ValueError('Student First Name can only contain alphabetic characters')
            break
        except ValueError as e:
            print(e)

    course_name = input("Enter the course name: "
```

Figure 6 Raising Value error for user input in the wrong format

The rest of the menu options work the same as the previous assignment making use of If, elif and else statements. **(Figure 7)**

```
# Present the current data using f string for string formatting
elif menu_choice == "3":
    # prompting the user to enter student details in case they skipped a step
    if student_first_name == '' or student_last_name == '' or course_name ==
'':
        print('Please enter student details again')

    else:
```

```
        message = f'{student_first_name} {student_last_name} has registered
for {course_name}'
        print(message)


# Save the data to a file
elif menu_choice == "4":
    # prompting the user to enter student details in case they skipped a step
    if student_first_name == '' or student_last_name == '' or course_name ==
'':
        print('Please enter student details again')

    # storing input data as dictionary
    else:
        student_data = {"First_Name": student_first_name, "Last_Name":
student_last_name, "Course_Name": course_name}
        students.append(
            student_data)  # adding new student data to the existing data
extracted from json file and stored in 2D list of dictionaries
```

Figure 7 – Menu options 3 and 4

For menu choice 4 you can see that I have I am storing input data as a dictionary with key names and values. I have then added the dictionary in the 2D list using the append function.**(Figure 7)**

Writing data to Json file

While writing data to JSON file I have again added provision for possible errors using try and exception functions **(Figure 8)**

Since I now know that my file has been created and it has valid data I don't need to add exceptions for FileNotFound error and JsonDecodeError.

```
#Writing Data to JSON file and adding exceptions
    try:
        file = open(FILENAME, 'w')  # using the write function, to truncate the file
        json.dump(students, file)
        file.close()
        print('Student registration details recorded\n')
    except Exception as e:
        print('Unexpected Technical error')
        print('-' * 50)
        print(e, e.__doc__, e.__str__(), type(e), sep='\n')
    finally:
        if not file.closed:
            file.close()
```

Figure 8 : Writing data to JSON file

Exit the While loop

Menu choice 5 allows the user the user to exit the program **(Figure 9)**

```
# Stop the loop
elif menu_choice == "5":
    user_input = input("Do you wish to exit the program?? (Y/N)
").capitalize()
    if user_input == "Y":
        print()
        print('Pausing the program till you press Enter...')
        break

else:
    print('Incorrect menu choice, please try again')  # incase the user
enters the wrong menu-choise
    continue
```

Figure 9 – Exit the While loop

Check data in JSON file

If I open my Json file it should have the previous data and the new data added after running the program

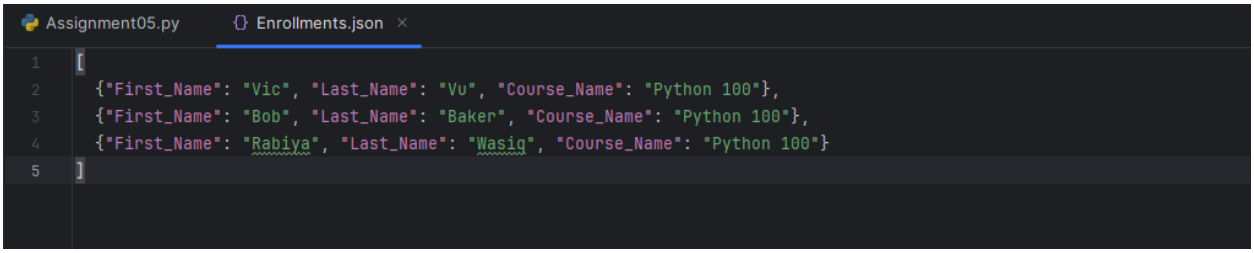

Figure 10 – JSON file with old and new data

Running the code in PyCharm and Command Prompt and link to GitHub

I was successfully able to run the program both in PyCharm and also in Command prompt by navigating to the program directory where the program is saved.

I have also uploaded my program at my GitHub account at the following link:

Assignment05 – GitHub link

Below you can see my program running in both Python and Command Prompt, the try and except function for user input ensure that the user enters the correct format for first name and last name

Pycharm

```
----------------------------------------

Please choose an option from the above menu: 2

Enter the student's first name: 123
Student First Name can only contain alphabetic characters
Enter the student's first name: Rabiya
Enter the student's last name: 456
Student First Name can only contain alphabetic characters
Enter the student's last name: Wasiq
Enter the course name: Python 100

----------------------------------------
---- Course Registration Program ----
  Select from the following menu:
    1. View all students registered to date
    2. Register a New Student for a Course
    3. Show New student registration details
    4. Save New student data to a file
    5. Exit the program

----------------------------------------

Please choose an option from the above menu: 3
```

Command Prompt

```
----------------------------------------

Please choose an option from the above menu: 2

Enter the student's first name: 1232
Student First Name can only contain alphabetic characters
Enter the student's first name: Rabiya
Enter the student's last name: 343
Student First Name can only contain alphabetic characters
Enter the student's last name: Wasiq
Enter the course name: Python 100

----------------------------------------
---- Course Registration Program ----
  Select from the following menu:
    1. View all students registered to date
    2. Register a New Student for a Course
    3. Show New student registration details
    4. Save New student data to a file
    5. Exit the program

----------------------------------------

Please choose an option from the above menu: 3
```

Summary

I was successfully able to read data from a Json file and load it in my program for further use. Using dictionaries instead of lists , made accessing specific values much easier by way of key names. I was able to make provisions for possible errors while opening and reading data from files. I was also able to raise errors to ensure that I receive input from the user in the correct format and that my data remains clean.

Most challenging part for me was to add a while loop around the try and except function for user input. I wanted to make sure the user is prompted for the student's name again till we receive the data in the correct format and that we don't end up adding empty variables to my file.