Name: Joseph Rideout **Date :** November 15, 2023 **Course:** Using json files to work with data
Mod05-Assignment 05

Python JSON: Read, Write

Introduction

This document introduces the fifth assignment, which showcases the utilization of read and write operations to manage data from a JSON file. Covering the fundamental methods, including **json.load()**, **json.dump()**, demonstrating the practical application of these methods.
Additionally, I found Mr. Randel Root live sessions to be helpful for understanding python for this week assignment.

Json in Python

JSON (**J**ava**S**cript **O**bject **N**otation) is a popular data format used for representing structured data. JSON is often used to transmit data between a server and a web application, as well as to store configuration data. Data is represented as key-value pairs, similar to Python dictionaries. JSON values can be strings, numbers, objects, arrays, booleans, or null.

Import Json Statement

The **import json** statement in Python is used to bring the **json** module into your script or program. The **json** module is part of the Python standard library.

READ JSON FILES

In Python, you can use the **json** module to read JSON files. The **json** module provide a '**json.load()** ' method for this purpose: **json.load().**
Here is an Example of json.load():

```
# Import the json module to work with JSON data
# FILE_NAME is string containing the path to the JSON file
# Open the specified JSON file for reading ('r' stands for read mode)
# Use json.load() to deserialize the JSON data from the file into a Python object
import json
FILE_NAME = "example.json"
with open(FILE_NAME, "r") as file:
students = json.load(file)
```

Run

Write to a JSON FILE

Writing to a JSON file in Python typically involves using the **json.dump()** method from the **json** module. This function allows you to serialize a Python object (like a dictionary or a list) into a JSON-formatted string and then write that string to a file.

Here is an example of json.dump() method: (figure 2)

```
# Open the specified JSON file for writing ('w' stands for write mode)
# Use json.dump() to serialize the Python object ('students') and write it to
the file
with open(FILE_NAME, 'w') as file:
json.dump(students, file)
```

(figure 2)

Exception

In Python, an exception is an event that occurs during the execution of a program that disrupts the normal flow of instructions. When an exception occurs, the Python interpreter raises an exception, and if the exception is not handled, the program terminates with an error message.

Here is an example of Try-Except Blocks (figure 3)

```
# Try to open the specified JSON file for reading ('r' stands for read mode)
# Try to load the JSON data from the file
# Iterate through each item in the 'students' list
# Print information about each student
# Handle the FileNotFoundError exception
# Print a user-friendly error message
# Print additional technical error information
```

```
try:
with open(FILE_NAME, "r") as file:
students = json.load(file)
for item in students:
print(f'ID:{item["first_name"]}, name:{item["last_name"]}, course:{item["course"]}')
except FileNotFoundError as e:
print('Text file must exist before running the script.\n')
print('--Technical Error Message--')
print(e, e.__doc__, type(e), sep='\n')
```

(Figure 3)

Assignment 05

For this week's assignment, the task is to enhance a menu-driven program by implementing code that accomplishes the following tasks: reading data from a json file and displaying the data in the header, creating a dictionary, saving it to a json file and displaying all the data before the saving process.

Here is an Example of Python 3 Code: (figure 4)

```
# ------------------------------------------------------------------------------------------
# Title: Assignment05
# Desc: This assignment demonstrates using dictionaries, files, and exception handling
# Change Log: (Who, When, What)
#   JRideout,11/15/2023,Created Script
#   <Joseph Rideout>,<11/15/2023>, <Created Script>
# ------------------------------------------------------------------------------------------
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"

# Define the Data Variables and constants
students: list[dict[str, str]] = []  # a table of student data

# When the program starts, read the file data into a list of dictionaries
# Extract the data from the file
try:
    with open(FILE_NAME, "r") as file:
        students = json.load(file)
        for item in students:
            print(f'ID:{item["first_name"]}, name:{item["last_name"]}, course:{item["course"]}')
except FileNotFoundError as e:
    print('Text file must exist before running script.\n')
    print('--Technical Error Message--')
    print(e, e.__doc__, type(e), sep='\n')

# Present and Process the data
while True:
    # Present the menu of choices
    print(MENU)
    menu_choice = input("What would you like to do: ")

    # Input user data
    if menu_choice == "1":
        student_first_name = input("Enter the student's first name: ")
        student_last_name = input("Enter the student's last name: ")
        course_name = input("Please enter the name of the course: ")
        student_row: dict[str, str] = {
            'first_name': student_first_name,
            'last_name': student_last_name,
            'course': course_name
        }
        students.append(student_row)
```

```
        continue

    # Present the current data
    elif menu_choice == "2":
        print("-" * 50)
        for student in students:
            print(f"Student {student['first_name']} {student['last_name']} is enrolled in
{student['course']}")
        print("-" * 50)
        continue

    # Save the data to a file
    elif menu_choice == "3":
        try:
            with open(FILE_NAME, 'w') as file:
                json.dump(students, file)
            print("Data saved successfully.")
        except Exception as e:
            print(f"An error occurred: {e}")
            print(e, e.__doc__, type(e), sep='\n')
        continue

    # 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")
```

Results: PyCharm / Command Prompt

PyCharm

```
ID:Joseph, name:Rideout, course: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: Cindey
Enter the student's last name: Campbell
Please enter the name of the course: 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 Joseph Rideout is enrolled in Python 100
-------------------------------------------------
```

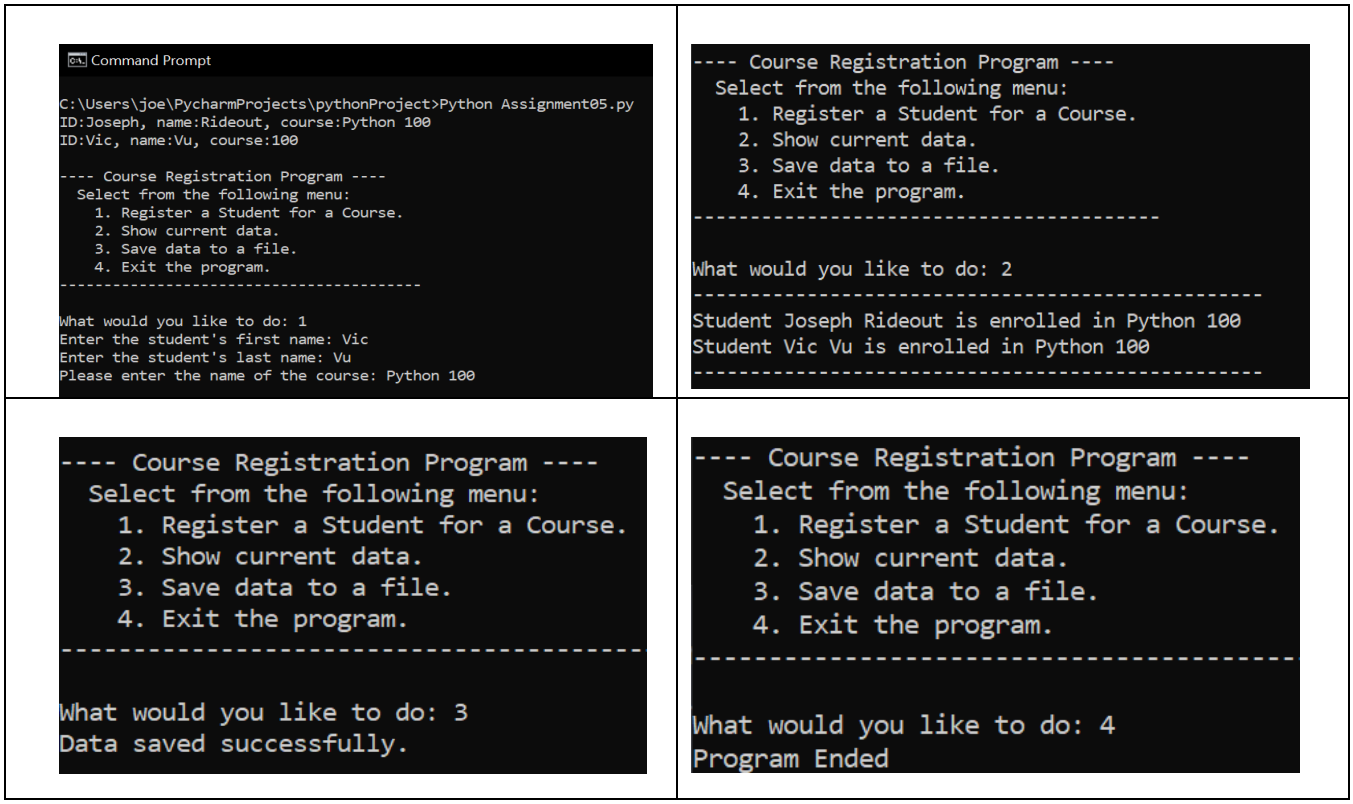
```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
----------------------------------------

What would you like to do: 3
Data saved successfully.
```

```
---- 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

Process finished with exit code 0
```



(Figure4)

Summary

This document serves as a comprehensive guide to data management in Python, introducing essential concepts and providing practical examples of Python snippets and programming techniques.
The knowledge gained from this week's assignment was an essential part of understanding Python 3 programming.