Name: Niki Way
Date: 02/26/2025
Course: IT FDN 110 A Wi 25: Foundations Of Programming: Python

Working with JSON

Introduction

For this assignment, I created a Python program that builds on last week's assignment in that, I created a program for students' registration for a Python course, but this week, in addition to the constants, variables, print statements, loop, programming menus, and conditional logic, I used data processing by using dictionaries, using JSON, and exception handling. In this paper, I will go over my process how the script was created and tested as well as my thoughts on the results.

The constants and variables

To start of this program, I started as I normally do by defining the constants and variables for this program. Like all the previous assignments I defined the constants and variables in the consistent format with the exception of my "MENU" constant. I updated the formatting per comments on my last weeks assignment. I used triple quotations around the "MENU" constant. The other change that I made was that the "FILE_NAME" was set to "Enrollments.json"; introducing the new file type. Additionally, I had new variable type in the "Dictionary" type. I set the lists to empty by using {}. (Figure 1)

```
# 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
students: list = []  # a table of student data
json_data = ''
file = None  # Holds a reference to an opened file.
menu_choice: str  # Hold the choice made by the user.
```

Figure 1: Constants and Variables set

Creating two-dimensional list

Part of the processing requirements is that when the program starts the contents of “Enrollments.json” are read automatically into a two-dimensional list table. To achieve this, I opened the file via open() and used “r” to express that I want to read the file. I started off by importing the json file. From there I used the json.load(file) method to parse the JSON data from the "Enrollments.json" file into a Python list of dictionaries (students). This part of the code contains structured error handling by using a Try-Except block in order to run this part of the script the user will need to have a created file prior to trying to open and read it. If the user does not have this file created then they will received an error message. Additionally, there is another error message that will occur if the user tries to close the file when it has not been open. (Figure 2)

```
# When the program starts, read the file data into a list of lists (table)
import json
# Extract the data from the file
try:
    file = open("Enrollments.json", "r")
    students = json.load(file)
    file.close()
except FileNotFoundError as e:
    print('Text file must exist before running this script!]\n')
    print("Built-In Python error info: ")
    print(e, e.__doc__, type(e), sep='\n')
finally:
    if file.closed == False:
        file.close()
    print("Text file must be open before running this script!]\n")
    print("Built-In Python error info:")
    print(e, e.__doc__, type(e), sep='\n')
```

Figure 2: Two-dimensional list of current data with structured error handling

Creating the loop

For this program just like last week, I used a while loop and built my program within the loop using the tab key for consistent indentation. I used the while true statement to contain my loop. Finally, I printed the MENU constant, prompted the user for input with the use of variable input line for “menu_choice”. (Figure 3)

```
# Present and Process the data
while (True):

    # Present the menu of choices
    print(MENU)
    menu_choice = input("What would you like to do: ")
```

Figure 3: Created Loop and Printed Menu Choice

Menu Selection

Using if/elif to display different outcomes based on the user's selection. This menu selection follows the same format has previously been submitted. With option one being for the user inputted information, option two to display the current data, option three is to save the data to the file, and option four is to exit the loop. There are new additions to the code in the form of structured error handling by using a Try-Except block.

In option one, there are two Try-Except block, one for "student_first_name" and one for "student_last_name." Each one is an error message about the input from the user. It explains to the user that their responses can only be in alpha characters. If the user uses anything other than alpha characters then this error message will appear. (Figure 4)

In option three, there is two different Try-Except block. One of the error message is to inform the user that the data being loaded in to the JSON file is not in the correct format. One is for the user to make sure that the file is open before is can be closed.(Figure 5)

```python
if menu_choice == "1":  # This will not work if it is an integer!
    try:
        student_first_name = input("Enter the student's first name: ")
        student_last_name = input("Enter the student's last name: ")
        if not student_first_name.isalpha():
            raise Exception('The first name should not contain numbers.')
    except ValueError as e:
        print(e)
        print('---Technical Error Message---')
        print(e.__doc__)
        print(e.__str__())
        if not student_last_name.isalpha():
            raise Exception('The last name should not contain numbers.')
    except Exception as e:
        print(e)
        print('---Technical Error Message---')
        print(e.__doc__)
        print(e.__str__())
    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} {student_last_name} for {course_name}.")
    continue
```

Figure 4: Option one with Try-Except Block

```python
# Save the data to a file
elif menu_choice == "3":
    try:
        file = open("Enrollments.json", "w")
        json.dump(students,file)
        file.close()
    except TypeError as e:
        print("Please check that the data is a vaild JSON format\n")
        print("--Technical Error Message --")
        print(e,e.__doc__,type(e),sep='\n')
    finally:
        if file.closed == False:
            file.close()
        print("Text file must be open before running this script!]\n")
        print("Built-In Pythong error info:")
        print(e, e.__doc__, type(e), sep='\n')
    print("The following data was saved to file!")
    for student in students:
    print(f"Student {student["FirstName"]} {student["LastName"]} is enrolled in {student["CourseName"]}")
    continue
```

Figure 5: Option three with Try-Except Block

Testing

I was able to obtain all of the following outcomes to meet the requirements for testing. For option one in the program: requests the user's input for a student's first, last name, and course name and allows users to enter multiple registrations.(Figure 6) For option two, the program displays current data. For option three, the program saves the user's input that was displayed in option two to a JSON file previously created and displays all data in that file. (Figure 7) I confirmed the file was created and saved correctly by opening the file in the Text.app. (Figure 8) Finally, I was able to exit the loop and program ceased running when option four was selected. (Figure 7) Finally this program was tested for the same outcome in the Terminal.app with the same results. (Figures 9-10)

```
---- 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: Niki
Enter the student's last name: Way
Please enter the name of the course: Python 100
You have registered Niki Way 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: 1
Enter the student's first name: Sue
Enter the student's last name: Jones
Please enter the name of the course: Python 100
You have registered Sue Jones for Python 100.
```

Figure 6: Option one repeated for multiple student registrations

```
What would you like to do: 2
-------------------------------------------------
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Niki Way is enrolled in Python 100
Student Niki Way is enrolled in Python 100
Student Vic Vu is enrolled in Python 100
Student Niki Way is enrolled in Python 100
Student Sue Jones 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 Niki Way is enrolled in Python 100
Student Niki Way is enrolled in Python 100
Student Vic Vu is enrolled in Python 100
Student Niki Way is enrolled in Python 100
Student Sue Jones 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
```

Photos

Figure 7: Option two, three, and four outcomes

Enrollments.json


```
[{"FirstName": "Bob", "LastName": "Smith", "CourseName": "Python 100"}, {"FirstName": "Sue",
"LastName": "Jones", "CourseName": "Python 100"}, {"FirstName": "Niki", "LastName": "Way",
"CourseName": "Python 100"}, {"FirstName": "Niki", "LastName": "Way", "CourseName": "Python
100"}, {"FirstName": "Vic", "LastName": "Vu", "CourseName": "Python 100"}, {"FirstName": "Niki",
"LastName": "Way", "CourseName": "Python 100"}, {"FirstName": "Sue", "LastName": "Jones",
"CourseName": "Python 100"}]
```


Figure 8: JSON File with data

```
Last login: Sat Feb 22 14:15:39 on console
nikiway@Nikis-MacBook-Pro ~ % cd documents/python/PythonCourse/A05
nikiway@Nikis-MacBook-Pro A05 % python3 Assignment05.py

---- 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: Niki
Enter the student's last name: Way
Please enter the name of the course: Python 100
You have registered Niki Way 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: 1
Enter the student's first name: Sally
Enter the student's last name: Stevens
Please enter the name of the course: Python 100
You have registered Sally Stevens 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 Niki Way is enrolled in Python 100
Student Niki Way is enrolled in Python 100
Student Vic Vu is enrolled in Python 100
Student Niki Way is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Niki Way is enrolled in Python 100
Student Sally Stevens is enrolled in Python 100
-------------------------------------------------------
```

Figure 9: Terminal test and outcomes for options one and two

```
---- 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 Niki Way is enrolled in Python 100
Student Niki Way is enrolled in Python 100
Student Vic Vu is enrolled in Python 100
Student Niki Way is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Niki Way is enrolled in Python 100
Student Sally Stevens 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
nikiway@Nikis-MacBook-Pro A05 %
```

Figure 10: Terminal test and outcomes for options three and four

Source Control

Both the scripting file and the knowledge document have been uploaded to GitHub. The link to the repository is below:

https://github.com/anway0123/IntroToProg-Python

Summary

This assignment proved to a bit easy for me to understand than last weeks I seem to have grasped the concept of dictionaries easier that I did with lists. That very well could have been because I already understood lists prior to learning how dictionaries functioned and while they are similar I can definitely see the uses of each. I do think that the dictionary knowledge seems to be sticking better for me.