Greg McCaslin

February 24, 2025

IT FDN 110 A

Assignment 05

<https://github.com/gmc5/UW_Python_Winter_2025/commit/992b0c675fdc5e07e591442d66fae1c5acfb7648/Assignment05.py>

Creating Assignment 05

# Introduction

This week, I am continuing using some of the loop functions that were used last week. New for this week is the use of dictionaries and JSON. So, the big change for this assignment was changing from files to dictionaries and from CSV to JSON. To better track changes, I commented out items and added new lines instead of modifying them (didn’t always happen).

# Getting Started in Programming My 5th Assignment

I created an A05 folder where my assignment files reside (Figure 1.1). I also quickly compared the assignment04 starter code to my assignment03 code.

A screenshot of a computer screen

AI-generated content may be incorrect.

**Figure 1.1: Snapshot of assignment artifacts**

Installing the Header

As directed in the homework module, I installed the header with the required information. I used hashtags to comment on the information, so it does not get executed as code. (figure 1.2)

*# ------------------------------------------------------------------------------------------ #  
# Title: Assignment03  
# Desc: This assignment demonstrates using conditional logic and looping  
# Change Log: (Who, When, What)  
# GMcCaslin,2/24/2025,Created Script  
# <Your Name Here>,<Date>, <Activity>  
# ------------------------------------------------------------------------------------------ #*

**Figure 1.2: Header information entered into the script**

# Importing of JSON Library

It was recommended that when working with JSON, to import it’s library at the beginning of the script. This way, the library will already be available when called upon. (figure 1.3)

A black background with white text

AI-generated content may be incorrect.

**Figure 1.3: Importing of JSON library**

# Setting the Constant Values

For the code, I set all required constant values with their type (which should never change). The big change for the constants was the changing of the file from CSV to JSON.(figure 1.4)

*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.  
-----------------------------------------   
'''  
FILE\_NAME: str = "Enrollments.json"*

**Figure 1.4: Constants defined in script**

Define the Data Variables

New for this assignment, student\_data: list was changed to student\_data: dict. (figure 1.5)

*student\_first\_name: str = ''  
student\_last\_name: str = ''  
course\_name: str = ''  
#student\_data: list=[]  
student\_data: dict= {}   
students: list = []   
file = None  
menu\_choice: str*

**Figure 1.5: The variables used in the script**

# Reading of and checking if file is correct

In place of the loop that was used in assignment 04, the “try”, “except”, and “finally” function was used. This was used for informing the user if the file could not be found. (figure 1.6)

*try:  
 file = open(FILE\_NAME, "r")  
 students=json.load(file)  
except FileNotFoundError as e:  
 print("This file doesn't exist")  
except Exception as e:  
 print("There was an error opening the file")  
 print(e, e.\_\_doc\_\_)  
finally:  
 print("Closing file")  
 file.close()*

**Figure 1.6: Checking if file is the correct one**

# Menu selection of user entering student data with error checking

Utilizing the conditional “if”, “try”, “if not”, “except”, and “continue” functions, the user input was check to ensure it met required conditions (in this case the input was are only letter characters). The “raise” function is used to help provide a statement about the error. The user is presented with a generic statement of the type of error they caused. (figure 1.7)

if menu\_choice == "1":   
 try:  
 student\_first\_name = input("Enter the student's first name: ")  
 if not student\_first\_name.isalpha():  
 raise ValueError('The first name can only be letter characters')  
 student\_last\_name = input("Enter the student's last name: ")  
 if not student\_last\_name.isalpha():  
 raise ValueError('The last name can only be letter characters')  
 course\_name = input("Please enter the name of the course: ")  
 student\_data = {"FirstName":student\_first\_name,"LastName":student\_last\_name,"CourseName":course\_name}#dictionary  
 students.append(student\_data)#dictionary to the list  
 print(f"You have registered {student\_first\_name} {student\_last\_name} for {course\_name}.")  
 except ValueError as e:  
 print("User entered invalid value. Continuing...")  
 continue

**Figure 1.7: User input, error checking, and feedback**

# Print file data including the key-value pairs

Previously, lists were used to print out the data that is currently in memory. This script now utilizes a elif condition (checking if “2” was selected) and loop around the key-value pairs and prints both attributes. (figure 1.8)

*elif menu\_choice == "2":  
 for student in students:  
 print(student["FirstName"],student["LastName"],student["CourseName"])  
 continue*

**Figure 1.8: Conditional statement**

# Print Data Entered by the User

The user enters student data and the student data stored in the file, which will be printed. (figure 1.9)

*elif menu\_choice == "2":  
 print("\nThe list of current data is:")  
 for student\_data in students:  
 print(f"Student: {student\_data[0]}, {student\_data[1]}, {student\_data[2]}")  
 continue*

**Figure 1.9: Print of student data**

# Creating, Writing, and Saving JSON File

This portion of the script creates the JSON file, writes the data to it, and then saves it. (figure 1.10)

*elif menu\_choice == "3":  
 print("Opening file")  
 with open(FILE\_NAME,"w") as file\_obj:  
 for student\_data in students:  
 file\_obj.write(f"{student\_data[0]},{student\_data[1]},{student\_data[2]}\n")  
 print("Writing file")  
 file\_obj.close()  
 print("Closed file\n")  
 print(f"You have registered {student\_first\_name} {student\_last\_name} for {course\_name}.")  
 continue*

**Figure 1.10: Creating, Writing, and Saving JSON File**

# Breaking the Loop and Exiting the Program

For this conditional statement of the script, if the user selects “4”, the “break” stops and exits the loop. (figure 1.11)

*elif menu\_choice == "4":  
 break # out of the loop*

*else:  
 print("Please only choose option 1, 2, or 3")  
  
print("Program Ended")*

**Figure 1.11: Breaking the Loop**

# Running from the Command Prompt

Running from the command prompt, I first showed the existing data in the file. (figure 1.12)

A computer screen with white text

AI-generated content may be incorrect.

**Figure 1.12: Existing student data in the file**

Next, I added another student enrollment and showed it was in memory. (figure 1.13)

A computer screen with white text

AI-generated content may be incorrect.

**Figure 1.13: Enrolled another student data to file**

After saving the file to the JSON, I confirmed that the added enrollment was in the file. (figure 1.14)

A screen shot of a computer program

AI-generated content may be incorrect.

**Figure 1.14: All enrollment shown in JSON file**

Creating GitHub Account

Below is the screenshot of the creation of my GitHub account. Figure 1.15

A screenshot of a computer

AI-generated content may be incorrect.

**Figure 1.15 GitHub account creation**

Below is the initial saving of my Python assignment 05 script. Figure 1.16

A screenshot of a computer

AI-generated content may be incorrect.

**Figure 1.16 GitHub uploaded assignment**

Here is the URL of the folder <https://github.com/gmc5/UW_Python_Winter_2025>**.** (figure 1.17)

A screenshot of a computer

AI-generated content may be incorrect.

**Figure 1.17 GitHub URL of assignment 05 folder**

Summary

Using JSON definitely required less code than for what was previously done. I need to further review the use of dictionaries and key-value pairs. It seems that this might be used a lot in future projects that I might be working with.