Assignment 5: Advanced Collections and Error Handling

Rucha P. Nimbalkar

University of Washington

IT FDN 110: Fall 2024

Prof. Randall Root

Nov. 13, 2024

Abstract

Data can be stored as a collection of tuples, lists, dictionaries, or sets. Dictionary is a collection of data in term of keys and values. JSON also has a similar format and JSON files contain data in JSON format. Dictionaries and JSON data formatting makes it easier to not only save data but also access it.

Keywords: dictionary, key, value, JSON.

Assignment 5: Advanced Collections and Error Handling

Assignment 5 is the fifth coding assignment of the IT Fundamentals (IT FDN 110 A) course I am taking at University of Washington. The goal of this assignment is to help me understand the usage of dictionaries to save data and write in a JSON file. This the link to my GitHub account for Assignment 05.

In this assignment, a Python script gives the user a menu of choices. It is an upgrade of Assignment 4 wherein we are performing similar tasks, but we are saving the data in dictionary variables instead of list variables. Moreover, we are using JSON files wherein data is stored in JSON format, and we are also managing errors using error handling tools in Python.

Reflection

In this week, I learnt more about dictionaries. I first learnt about dictionaries in Module 04. Dictionaries have the format of Key and Value to store data, and they make data access easier. I also learnt about JSON format which is similar to dictionaries (Keys and values to store data) and I worked with JSON files. JSON stands for JavaScript Object Notation and JSON files hold JSON formatted data of key value pairs (Root, 2024).

Program Summary

I began the program using the starter file provided for Assignment 4. From my previous assignment, I have learnt to never change the code of another person without informing them and to never delete variables declared by another developer. I first updated the script header with my name (please refer fig. 1) and then I added the dictionary row variable (please see figure 2).

Figure 1

Assignment 05 Python Starter Code with appropriate updates

```
# ------------------------------------------------------------------------------------------ #
# Title: Assignment05
# Desc: This assignment demonstrates using dictionaries, files, and exception handling
# Change Log: (Who, When, What)
# <Rucha Nimbalkar>, <11/07/2024> <Update the starter file with my name and other details>
# ------------------------------------------------------------------------------------------ #
```

Figure 2

Assignment 05 Dictionary variable added to the code.

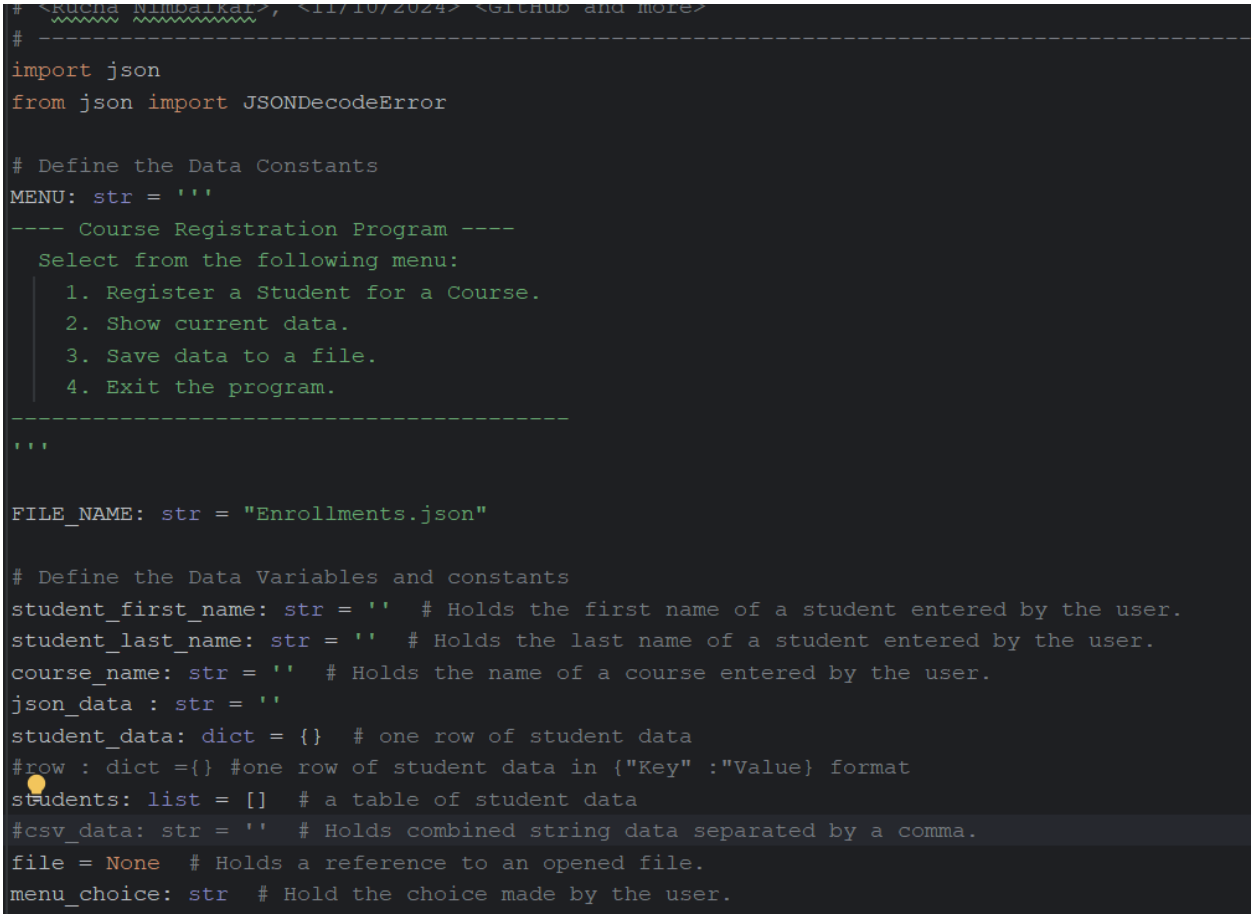
```
# ------------------------------------------------------------------------------------------ #
# Title: Assignment05
# Desc: This assignment demonstrates using dictionaries, files, and exception handling
# Change Log: (Who, When, What)
# <Rucha Nimbalkar>, <11/07/2024> <Update the starter file with my name and other details>
# <Rucha Nimbalkar>, <11/08/2024> <Add dictionary variable row{}
# ------------------------------------------------------------------------------------------ #

# 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.csv"

# 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: list = []  # one row of student data
row : dict ={} #one row of student data in {"Key" :"Value} format
students: list = []  # a table of student data
csv_data: str = ''  # Holds combined string data separated by a comma.
file = None  # Holds a reference to an opened file.
menu_choice: str  # Hold the choice made by the user.
```

After completing the module lessons, I added code to import JSON, and JSONDECODE ERROR for using JSON file and methods, and managing exceptions respectively. I also added a

new variable like json_data. I changed the file name extension from “Enrollments.csv” to “Enrollments.json”. I updated the students variable datatype to dictionary and then I commented out old variables that I believe I won’t be using in this code. This is something I learnt in the live session with Professor Root as mentioned previously. Assuming I am using somebody else’s code, I will not remove anything unless absolutely necessary and just comment code that is not used and add meaningful comments to explain why certain section of the code are commented out (Please see figure 3).

Figure 3

Assignment 05 New variables are added and unused variables are commented out



```
# <Rucha Nimbalkar>, <11/10/2024> <GitHub and more>
# ------------------------------------------------------------------------------------------
import json
from json import JSONDecodeError

# 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.
----------------------------------------
'''

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.
json_data : str = ''
student_data: dict = {}  # one row of student data
#row : dict ={} #one row of student data in {"Key" :"Value} format
students: list = []  # a table of student data
#csv_data: str = ''  # Holds combined string data separated by a comma.
file = None  # Holds a reference to an opened file.
menu_choice: str  # Hold the choice made by the user.
```

Then I added the code for reading the contents of "Enrollments.json" file into the students variable which is a two-dimensional list table (a list of dictionary rows) along with the exceptions for file not found error and other errors as shown in figure 4.

Figure 4

Assignment 05 Read the file contents when program starts

```
try:
    file = open(FILE_NAME,"r") #open the json file in read mode
    json_data = json.load(file) #Now json_data contains the parsed JSON data as a Python list of dictionaries
    for item in json_data:
        student_first_name = item["FirstName"]
        student_last_name = item["LastName"]
        course_name = item["CourseName"]
        student_data ={"FirstName" :student_first_name,"LastName" :student_last_name,"CourseName" : course_name}
        students.append(student_data)
    file.close() #close the file
except FileNotFoundError as e: #Handle FileNotFoundError
    print("File not found!")
    print("---Technical Error Message")
    print(e, e.__doc__, type(e), sep="\n")
    print("Creating new file , since file does not exist.")
    file = open(FILE_NAME, "w")
    json.dump(students, file)
except JSONDecodeError as e: #HandleJSONDecodeError
    print("Data in file is invalid! Resetting it.")
    file = open(FILE_NAME, "w")
    json.dump(students, file)
    print(e, e.__doc__, type(e), sep="\n")
except Exception as e:#Handle general (unexpected non-specific) error
    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() #close the file if it is not closed.
```

Then I started working on menu_choice “3” wherein the user input should be saved to the JSON file. in write mode using the open() function. The contents of the students variable are written to the JSON file using the dump () function. After that I added lines to show what was saved in the file. I also added exception handling to my code to avoid errors in writing that I learnt in the demo videos.

Figure 5

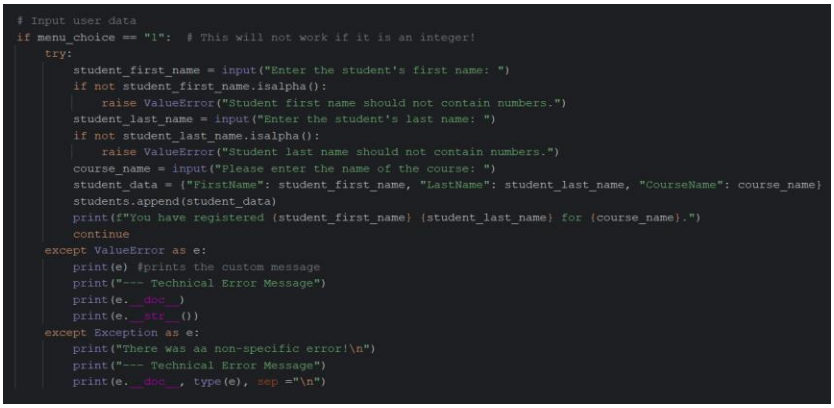
Assignment 05 Write in the file and display what was saved in the file

```
# Save the data to a file
elif menu_choice == "3":
    '''file = open(FILE_NAME, "w") #This is the code from the starter file and it is commented because it is not relevant to this assignment.
    for student in students:
        csv_data = f"{student[0]},{student[1]},{student[2]}\n"
        file.write(csv_data)
    file.close()'''
    try:
        file = open(FILE_NAME, "w")
        json.dump(students, file)
        file.close()
        print("The following data was saved to file!")
        print("-" * 60)
        for student in students:
            print(f" 'FirstName' :{student["FirstName"]},'LastName': {student["LastName"]},'CourseName':{student["CourseName"]}")
        print("-" * 60)
        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()
```

After that I started working on menu_choice “1” to add exceptions for non-alphabetic input for first name and last name as shown in figure 6.

Figure 6

Assignment 05 Handle exceptions for inappropriate input for student name

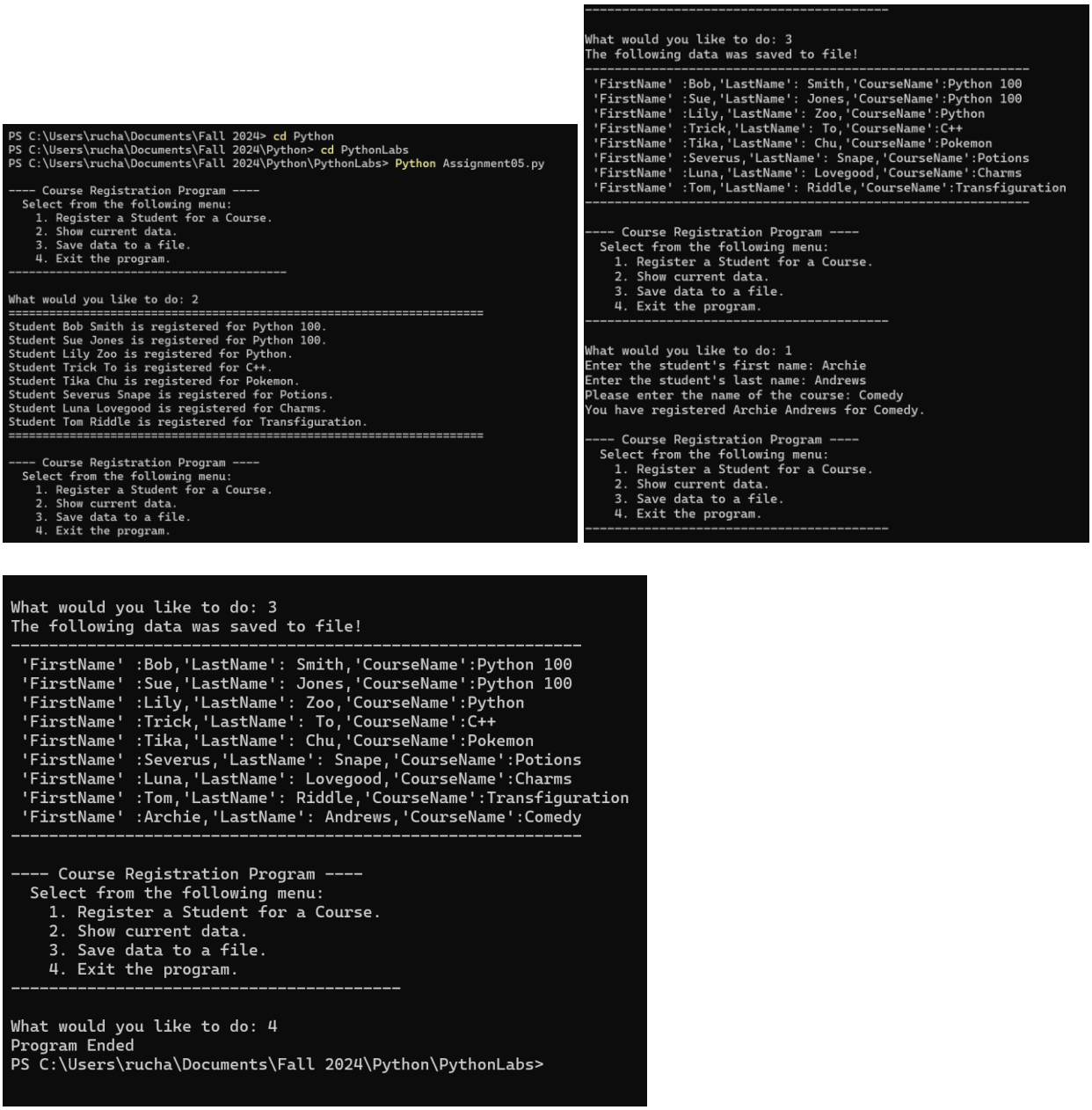


```
# Input user data
if menu_choice == "1":  # This will not work if it is an integer!
    try:
        student_first_name = input("Enter the student's first name: ")
        if not student_first_name.isalpha():
            raise ValueError("Student first name should not contain numbers.")
        student_last_name = input("Enter the student's last name: ")
        if not student_last_name.isalpha():
            raise ValueError("Student last name should not contain numbers.")
        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
    except ValueError as e:
        print(e) #prints the custom message
        print("--- Technical Error Message")
        print(e.__doc__)
        print(e.__str__())
    except Exception as e:
        print("There was aa non-specific error!\n")
        print("--- Technical Error Message")
        print(e.__doc__, type(e), sep ="\n")
```

Then I ran the code in the command terminal to verify whether I received the same output, and it ran successfully as shown in figure 7.

Figure 7

Assignment 05 Output in Command Terminal



After that I open the “Enrollments.json” file to check if data was actually saved in the file. I found that it was all saved in the file in the JSON format.

References

Randall, R.(n.d.). *IT Fundamentals 110 A* [MOOC]. University of Washington. Foundations of Programming (Python) - UW Professional & Continuing Education