Nathan Castek

February 14th, 2024

IT FDN 110 A

Assignment05

Advanced Collections and Error Handling

Introduction

In this assignment, I reviewed various articles and videos detailing the usage of dictionaries, working with JSON files, how to handle errors, and how to use GitHub for configuration management. Then, I created a simple python script that prompts a user to make a selection from a menu of options. Depending on the selection, the program will execute the appropriate command. In this assignment, I utilized: constants, variables, conditional operators, the input function, the print function, type hints, while loops, lists, dictionaries, error handling, and worked with JSON files. Furthermore, I practiced testing my code from the integrated development environment PyCharm and from the terminal.

Research

I started out the assignment by reading the various articles and watching the videos for this week's module and assignment. I learned about dictionaries, error handling, working with JSON files, and hot to utilize GitHub for configuration management.

Programming

In the next part of the assignment I created a simple python program that prompts a user to make a selection from a menu of options and then performs a specific action based on the selection the user made.

Header

I started out my program by revising the header file provided in the assignment starter file. I removed the professors name and replaced it with my name and date. The resultant header can be seen below.

```
# ------------------------------------------------------------------------
-------------- #
# Title: Assignment05
# Desc: This assignment demonstrates using dictionaries, files, and exception
handling
# Change Log: (Who, When, What)
#   NCastek,2/12/2024,Created Script
# ------------------------------------------------------------------------
-------------- #
```

Defining Imports, Constants, and Variables

I then defined the imports, constants, and variables for the program. In this assignment I knew I had to read and write to a JSON file. Therefore, I needed to import json in order to use the json functions. Next, I defined the constants and variables. The constants and variables were detailed in the assignment, so I ensured the names were identical to the assignment instructions. The resultant code can be seen below.

```
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
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 = ""  # Holds the json data
file = None  # Holds a reference to an opened file.
menu_choice: str = ""  # Hold the choice made by the user.
student_data: dict = {}  # one row of student data
students: list = []  # a table of student data
```

Read JSON File

After revising the header, defining the constants, and variables, I set up the code to read in the Enrollments.json file. I also built in error handling to this operation using the try/except/finally error handling method. I started out using the "try" keyword and opening the Enrollments.json file in the read mode. Then, I used the json.load function to load the file data and store it in the variable students. I knew when reading this file, the most likely error I was about to encounter was the "FileNotFoundError." I planned for this by creating an "except" block that excepted the error "FileNotFoundError." When it encountered this error, it printed out a series of messages to notify the user about what was going on. Furthermore, I made another "except" block to catch any other exceptions the program might encounter when trying to read this file. Lastly, I used the "finally" block to close the file, if the file was already not closed. Below is the resultant code:

```
try:
    file = open(FILE_NAME, "r")
    students = json.load(file)
except FileNotFoundError as e:
    print("Enrollments.json file must exist before running this script.\n")
    print("Built in python error info: ")
    print(e, e.__doc__, type(e), sep='\n')
except Exception as e:
    print("There was a non-specific error!\n")
    print("-- Technical Error Message --")
    print(e, e.__doc__, type(e), sep='\n')
finally:
    if not file.closed:
        file.close()
```

Menu Selection

After reading in the JSON file, I set up the code to obtain user menu selection. I knew I always wanted the menu selection to pop up first until the user hit menu option '4'. Therefore, I decided to create a "while" loop that would always be "True" until the user selected menu option '4'.

I started out the menu selection with "while True:" and then indented under that loop I printed the constant "MENU". Next, I prompted the user for input using the input function and stored their input in the variable "menu_selection". The resultant code can be seen below:

```
while (True):

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

If/Else Statements

After obtaining user's input for menu selection, I created a series of if/elif/else statements for all the options the user could have picked. My first block of code was: if menu_choice == "1". In this scenario I prompted the user for their name and course name using the input function and storing the information in their respective variables. I also defined what I wanted in the dictionary "student_data" by specifying the three key/value pairs I wanted in the dictionary. Then, I appended student_data to the "students" variable. Furthermore, I added in some error handling to check to see if the user input an alphabetic name. If the user did not, the code would raise a ValueError and print out an error message. Lastly, the block also excepted all other exceptions and printed out an error message as well. The resultant code can be seen below:

```
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's first name should only contain
alphabetic characters.")

        student_last_name = input("Enter the student's last name: ")
        if not student_last_name.isalpha():
            raise ValueError("Student's last name should only contain
alphabetic characters.")

        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}.")

    except ValueError as e:
        print(e)
        print("--- Technical Error Message ---")
        print(e.__doc__)
        print(e.__str__())
    except Exception as e:
        print("There was a non-specific error!\n")
        print("--- Technical Error Message ---")
        print(e, e.__doc__, type(e), sep='\n')

    continue
```

My next block of code was for if the user selected menu option two. I utilized an "elif" statement because I knew there were more options to select from, so did not want to use an else statement here. I then used a for loop to loop through all the elements in the students list and print out each student. Lastly, I used the word "continue" to make sure the "while" loop at the beginning of the program started over again. The resultant code can be seen below:

```
elif menu_choice == "2":

    # Process the data to create and display a custom message
    print("-"*50)
    for student in students:
        print(f"Student {student["FirstName"]} {student["LastName"]} is
enrolled in {student["CourseName"]}")
    print("-"*50)
    continue
```

My next block of code was for if the user selected menu option three. I utilized an “elif” statement because I knew there were more options to select from, so did not want to use an else statement here. Furthermore, I utilized a “try” block to catch any errors that occurred during this block. I used the “open” command to open the constant “FILE_NAME” in the write mode and stored that in “file”. Next, I used the json.dump function to “dump” all the students from the file. Then, I used a series of “except” block to catch a TypeError exception and any other Exception. Lastly, I used a “finally” block to close the file if it was already not closed.

After closing the file, I created a “for” loop to loop through each student and print out their information. The resultant code can be seen below:

```
elif menu_choice == "3":
    try:
        file = open(FILE_NAME, "w")
        json.dump(students, file)
        file.close()
        print("The following data was saved to file!")
        for student in students:
            print(f"Student {student['FirstName']} {student['LastName']} is
enrolled in {student["CourseName"]}")
    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 not file.closed:
            file.close()
    continue
```

My next block of code was for if the user selected menu option four. I utilized an “elif” statement because I knew there were more options to select from, so did not want to use an else statement here. In this block I used the word “break” to break out of the “while” loop at the beginning of the program. The resultant code can be seen below:

```
elif menu_choice == "4":
    break  # out of the loop
```

My last block of code was used as an input error catching block. The user was supposed to input 1, 2, 3, or 4. However, there is a chance that they could have entered something else. Therefore, I created an “else” statement that would catch any user input that was not 1, 2, 3, or 4. In this block I simply used the print function to notify the user that they did not enter a valid option.

```
else:
    print("Please only choose option 1, 2, or 3")
```

Testing

After completing my program I tested it by running it from PyCharn and the terminal. I first ran my program from PyCharm and below you can see my user input’s and what was printed to the screen.

Note: I first made the entry:

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

in the Enrollments.json file so the program could read in the file.

---- 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

---- 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: Joe
Enter the student's last name: Garis
Please enter the name of the course: Python 100
You have registered Joe Garis 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 Joe Garis 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 Joe Garis 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: 1
Enter the student's first name: Sam
Enter the student's last name: Lewis
Please enter the name of the course: Python 100
You have registered Sam Lewis 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 Joe Garis is enrolled in Python 100
Student Sam Lewis 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 Joe Garis is enrolled in Python 100
Student Sam Lewis 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: 1
```

```
Enter the student's first name: 5
Student's first name should only contain alphabetic characters.
--- Technical Error Message ---
Inappropriate argument value (of correct type).
Student's first name should only contain alphabetic characters.

---- 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: mike
Enter the student's last name: 5
Student's last name should only contain alphabetic characters.
--- Technical Error Message ---
Inappropriate argument value (of correct type).
Student's last name should only contain alphabetic characters.

---- 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 Joe Garis is enrolled in Python 100
Student Sam Lewis 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

Next, I tested my program in the terminal and saw identical results to when I used PyCharm.

Summary

In this assignment, I reviewed various articles and videos detailing the usage of dictionaries, working with JSON files, how to handle errors, and how to use GitHub for configuration management. Then, I created a simple python script that prompts a user to make a selection from a menu of options. Depending on the selection, the program will execute the appropriate command. In this assignment, I utilized: constants, variables, conditional operators, the input function, the print function, type hints, while loops, lists, dictionaries, error handling, and worked with JSON files. Furthermore, I practiced testing my code from the integrated development environment PyCharm and from the terminal.