Crisandra Wilkie

2/26/25

Foundations of Programming: Python

Assignment 5

Introduction

Module 5 focused on advanced data collections and error handling. This included working with dictionaries, JSON (JavaScript Object Notation) files and modules. This assignment was similar to assignment 4. The objective was still to register students in a course, display messages about a student's registration, and to save new student registration information to a file. What was added was the use of dictionaries for data processing and error or exception handling using a Try-Except construct. Another added element was source control using GitHub.

Unchanging elements of the script

- Header: includes a title, description of script, change log, which holds the author, date and purpose.
- Create and define variables and constants.
- Pseudo-code, which outlines the steps to reach the objective.

Working with JSON

JSON files consist of key-value pairs, where the keys are strings enclosed in double quotes and values can be strings, numbers, objects, arrays, Booleans, or null. JSON is flexible and lends for more complexity. At the start of this script I imported the JSON module to help read from and write to a JSON file. Starting with the enrollments JSON file with one row of data, I was able to read in the existing data, and as in previous weeks, had the option to add registration data and append it to the JSON file. Reading in the file was similar to reading in a csv file, as shown below. Similarly, writing to a JSON file is similar to writing to a csv file, except that it uses a dump() function, also shown below.

```
file = open(FILE_NAME, "r")
students = json.load(file)
file.close()
```

Figure 1. Read in JSON file

```
file = open(FILE_NAME, "w")
json.dump(students, file)
file.close()
```

Figure 2. Write to JSON file

Working with dictionaries

I was able to create the table of registered students using a list of dictionaries. A dictionary can be thought of as a row of data and the column names as keys. Dictionaries use key-value pairs, this ultimately is a more intuitive way to retrieve specific fields. In this script, I used “TitleCasing” to name keys. Dictionaries use {} opposed to lists that use []. The dictionary, “student_data”, included the FirstName, LastName, and Course.

Structured error handling

Bugs can sometimes be introduced to a program leading to errors and potentially breaking your program. The try-except construct is a way of error handling. This gives the ability provide user-friendly error messages, it also allows Python to automatically move to another set of statements where you can customize handling an error.

In this script there were some readily identifiable areas prone for errors. I added a try-except construct to the read statement. Potential errors identified include, file not found error, in which the user would be informed that the text file must exist before running the script. I also included a generic expression, this message states there was a non-specific error, shown below.

```
try:
    file = open(FILE_NAME, "r")
    students = json.load(file)
    file.close()
except FileNotFoundError as e:
    print("Text file must exist before running this script!\n")
    print("-- Technical Error Message -- ")
    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 file.closed == False:
        file.close()
```

Figure 3. Error handling

The next identified potential area for errors was in collecting first and last names. I added a try-except construct to let the user know that the first and last names could not contain a number. Again, this construct included the generic expression, there was a non-specific error.

The third readily identifiable spot for errors is in writing the data to the JSON file. Here, I added a try-except construct that would remind the user to, “please check that the data is a valid JSON format. And the added generic expression, there was a non-specific error, shown below.

Testing the script

In addition to running in PyCharm, this script was also run using the command prompt window. This was achieved by opening the command prompt window and changing the directory to where this script lives. Once in the proper directory I pointed to the script and was able to successfully run it.

GitHub

GitHub is a cloud-based file sharing platform that allows code hosting and collaboration. This platform allows for version control, access control, pull requests and code reviews, among other useful features. I started by creating a GitHub account, once I created an account, I created a repository that will be used for this and future assignments. My GitHub repository can be found at the following link: https://github.com/cris-wilkie/Python110Wi25.git

Conclusion

This week’s objective was to register students in a course. This objective was accomplished by using dictionaries, JSON modules and files, and some error handling. Running this script allows a user to read in the existing file, add new students, and add any new students to the existing file. Sharing code files is a common practice. Cloud file sharing is one way to share and collaborate, GitHub is a multifaceted platform that allows code and file hosting.