James Copsey

11/13/2024

ITFDN110

Assignment05

https://github.com/jcopsey/IntroToProg-Python-Mod05

Assignment 05 – Advanced Collections and Error Handling

Introduction

Assignment 05 builds upon the script we used for Assignment 04, however, this time we're using a .json file to store the data about our registrations. By importing the json module, we're able to use functions like json.load() and json.dump().

Using these functions makes handling data much easier because we don't have to worry about the formatting when using json.dump(). The function json.dump() does it for us.

Try and Except

In Assignment 05 we were tasked with using try and except to catch errors and handle them in a structured way.

One example from my scripts is that if the initial file read of the .json file does not work, it will print a message that the file was not found, and then it will go and create that file.

Here is a snippet of that code:

```
# When the program starts, read the file data into a list of lists (table)
# Extract the data from the file
try:
    file = open(FILE_NAME, 'r')
    students = json.load(file)
    file.close()
except FileNotFoundError as e:
    print(f"{FILE_NAME} was not found. Creating {FILE_NAME}")
    file = open(FILE_NAME, 'w') # If the file doesn't exist, create it
except Exception as e:
    print("There was an error opening the file.")
    print(e, e.__doc__)
finally:
    if not file.closed:
        file.close()
```

The file that gets created is completely empty, but that's okay! When we use json.dump(), it formats and new data we write to it in json format.

Here is a snippet from where we call json.dump():

```
    # Save the data to a file
    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']} "
                    f"{student['LastName']} "
                    f"is enrolled in {student['CourseName']}."
                ))
        except Exception as e:
            print(f"There was an error writing to {FILE_NAME}.")
        continue
```

The above snippet also shows an example of using double parentheses so that we can split our f-string into multiple f-strings on multiple lines.

I was attempting to use the line continuation character, but PyCharm suggested using print(()) with f-strings. Using an IDE that is specifically built for the language you are writing in comes with many advantages.

Summary

We built upon the script we created last week and streamlined the process of adding new information to files using functions from an imported module. Generally, we added error handling to tell our program what to do if a specific event happened. Specifically, we ensured that the first name, last name, and course name inputs where alphanumeric and that the .json file was created if it couldn't be found.