Ben Stafford

21 February 2024

IT FDN 110 A

Assignment 06

Assignment 6: Functions and Classes

Introduction

Assignment 6 focused on organizing and cleaning up the code with the assistance of Functions and Classes. These changes bring the script in line with the Separations of Concerns Pattern.

Rewriting Script to Include Classes and Functions

After filling out the header script, I commented out all the variables except those defined by the acceptance criteria. I began by creating the menu input/output functions, so I can test the rest of the options as I create them. I then created the FileProcessor class. The first function for reading the JSON file was mostly able to be copied and pasted from below. I edited the function slightly so that the variables were defined by the parameters rather than the commented out variables, and I added "return student_data" at the end. I encountered a bug where previous data was being overwritten, but I was able to work around this by appending the JSON data directly to the "students" variable. (**Figure 1**)

```
try:
    file = open(file_name, "r")
    student_data = json.load(file)
    for student in student_data:
        students.append(student)
        print(f'Student {student["FirstName"]} '
              f'{student["LastName"]} is enrolled in {student["CourseName"]}')

    file.close()
```

Figure 1: Final code for pulling JSON data

The function for writing the data to file was also quite similar to the existing script once I swapped out the variables for the defined parameters. For the IO class, I began with the error function, which was kept pretty basic so that the other functions could link to it in case of errors. The menu functions were also pretty straightforward; the output simply prints the menu UI and the input is a simple 1, 2, 3, 4 choice, with an error if other options are picked. The input function for student data was similarly mostly unchanged aside from the parameters. The result of all this is that the main body area is quite short and clean now.

Summary

At first, I found the intense reorganization of the code a bit tedious and unnecessary, but once I got to the final result, with how much cleaner it all was, I understood why we learn to do things this way from the start. It certainly makes sense to use a system that works when thinking about how much longer and more complex the code can get. Bringing it into alignment with universal systems of organization also makes it much more digestible and changeable by others.