Katherine Sather

11/19/2024

Foundations of Programming: Python

Assignment No. 6

Github URL

Assignment No. 6

# Introduction

While working through Module 6, I had the opportunity to dive deep into more programming concepts, focusing on functions and the importance of organizing your code. We covered the separation of concerns pattern, showing how these ideas make code cleaner and easier to maintain. I can see how, with this process, we are frontloading the bulk of our code from the main body of the script to the top instead. The main body then becomes a pattern of calling functions. Additionally, in the same way a written assignment has specific parts (an introduction, headings, sections, conclusion), the script file has a common layout, too: header, imports, the definition of variables and constants, functions, and the main body.

In this assignment, I’ll describe my experience using these concepts to build a program that includes classes, functions, and the separation of concerns.

# Getting started

I started by updating the script header to include my name and the date. I commented out the variables that will no longer be needed, and made sure to include the two variables referenced in the instructions for assignment six:

* **menu\_choice: str** is set to empty string.
* **students: list** : list is set to and empty list

The instructions asked for two new classes with docstrings. I set up the first class, “FileProcessor,” under a block called Processing. I then placed the second class, “IO,” in a block titled presentation. This is in accordance with the separation of concerns method, a key design principle that improves code maintainability and readability by dividing it into modular, self-contained components. I did find it slightly redundant to label these sections so similarly to the class names.

For each class, I included descriptive docstrings, which describe what the code does, its purpose, and how to use it. This helps others understand the code better (I can also see how this would help me personally if I revisited the code after some time and needed a reminder or two). Docstrings are written inside triple quotes (`"""`) and are typically the first statement included in the section of code.

So far, this work was very similar to our labs from Module 6. The next step was managing functions.

# Functions

The “FileProcessor” class will handle reading and writing data, so I added a function here to read the file automatically when the program starts: read\_data\_from\_file(file\_name: str, student\_data: list). Inside the parentheses are my parameters.

The instructions request a static function, which is a function that lives inside a class – but doesn’t use the class or object. It’s another way of organizing code. I added docstrings for my function that provide context and detail about the date and purpose. In this same processing class, I added a function to write data to the JSON file.

At this point, I realized I needed to switch up my order of work. I couldn’t include try/except code for the “FileProcessor” functions without first adding a new function to output error messages in the IO class. So that is what I worked on next. What followed was a long process of copying and pasting similar code from previous module six labs into my final assignment, and making necessary adjustments.

While I understand the value of clean, minimal code – I have some anxiety about deleting content altogether, even if it’s already commented out. So at first, I was leaving many comments and commenting out *a lot of code.* However, when I showed this to a colleague who works in engineering, they were repulsed by the amount of comments and said my code was “very hard to read.” Lesson learned! I cleaned up my file.

I encountered some challenges along the way and ultimately leaned on the instructor’s videos to complete this assignment. I encountered this additional hiccup:

Once I added all my functions to their appropriate classes, and called the functions in the main body, I expected everything to run successfully. But instead I got this error just above elif menu\_choice == “4” (See figure 1).

A screenshot of a computer program

Description automatically generated

***(Figure 1: Debugging my final assignment)***

I could not figure out how to fix this on my own, and asked ChatGPT, which suggested this code (see figure 2):

A computer code with colorful text

Description automatically generated with medium confidence

***(Figure 2: Implementing an exception calling the function output\_error\_messages.)***

After this change, my code ran with no problems, but I’m still mixed up … as I thought the error treatment in my function itself would handle this. Moving forward, I’ll continue balancing independent problem-solving with strategic use of resources like ChatGPT, but ChatGPT doesn’t always have all the answers.

# Summary

Studying Module 6 has been an eye-opening experience, reinforcing the value of functions and proper code organization in programming. From understanding modularity and reusability to implementing the separation of concerns pattern, I’ve learned how these principles simplify development and improve maintainability.