George Nuesca

November 22, 2023

IT FDN 100 A

Assignment 06

https://github.com/uwgmn/IntroToProg-Python-Mod06

Using Classes and Functions

Introduction

This paper discusses the implementation of a menu program using classes and functions. A significant amount of the code was written from previous exercises and restructured to realize functions. Those functions were then classified into processing and input/output layers to create a beneficial pattern with regards to a Separation of Concerns (SoC) approach, as outlined in the *Mod06-Notes*, which further identified code activities for usage in the main body of the script.

Functional Coding

Since a significant amount of code was previously written, a major part of the exercise was organizing it into specific functions to be used in the main body of the program. The approach to programming was executed somewhat sequentially by defining a single function, testing it in the code, then moving onto the next function. While there are several improvements regarding exception handling to make, those parts of the code were not altered to avoid further complicating the review.

The easiest code to initially experiment with in a function had to do with displaying information. This was placed at the beginning of the code for the *output_menu* function, which used a parameter to pass in a string constant. This code was then tested in the pre-existing while loop of the main body to display menu items for the user to select.

Similar approaches were taken for the other functions to ensure the results of their execution performed appropriately. In some cases, the function returned data to be used in other aspects of the code. While adding the other functions was more of a way to organize the code, the *output_error_message* function significantly simplified the code by its reuse in seven function locations.

Classes

For this assignment the use of classes is used to demonstrate organization. While not necessary for this short amount of code, it added a way to separate the functions into a more modular structure, and into a SoC pattern. **Figure 1** shows the collapsed organization of functions in the *FileProcessor* classes. Additionally, using the @staticmethod decorator helps us avoid creating an additional object to execute functions.

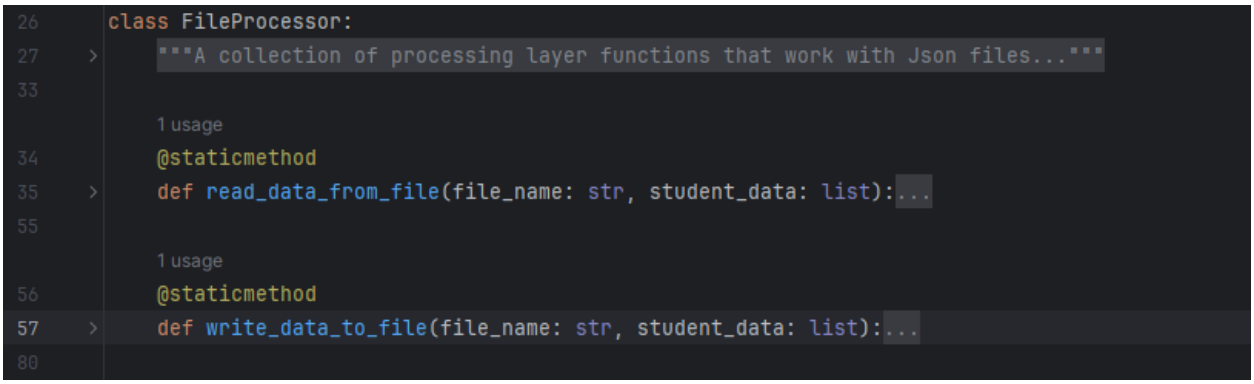

Figure 1: FileProcessor class with collapsed functions

Summary

This assignment demonstrated the use of functions and classes to effectively organize code and promote reuse, especially for more complex programs. Overall, the exercise also exhibits functions and their local variables are easier to track than with a large set of global variables, especially if a program becomes very large. The actual implementation of the assignment was straightforward with little complexity in the code.

As stated in the **Functional Coding** section, the sequential approach of implementing the program by individual functions provided a concise way to debug code, of which there were primarily only syntax errors. Some care was used when substituting variables for parameters to prevent confusion between local and global variables, with little issues arising.