Efren Enriquez

08/07/2024

IT FDN 110A

Assignment 06

https://github.com/jefrene10/IntroToProg-Python-Mod06

Assignment 06 – Functions

Introduction

This week in IT FDN 110A the topic of functions was introduced. In this paper there will be a brief overview about some of the concepts learned during the module. Then the process used to write the script for this assignment will be further elaborated on throughout the paper. The content this week focused on various ways to organize code. This will include the use of variables, functions, parameters, classes, and incorporating the Separation of Concerns Patter (SoC). Finally discussing the initial code setup including setting a menu constant and defining variables, main body code, and testing of the code.

Concepts Learned

This week new concepts were introduced and drove the point of organizing code. This includes the use of functions, parameters, classes, and SoC.

Functions and Parameters

As the code gets more and more complex as time goes on, it is important to learn how to organize and use professional coding practices. One of the ways to advance code is through the use of functions. The use of functions in this class focuses on modularity and reusability. Functions themselves can allow the user to breakdown a large program and make it smaller. This will allow for each function to focus on a specific task, making code east to understand and maintain. With that functions can be used multiple times throughout a program. It is important to write functions after the variables are defined. Functions can be called multiple times throughout the code and keeps the program tidy.

Global and local variables were talked about in the notes, however another way to access data using global variables in functions is to pass the data into the function by creating parameters. Defining parameters allows for encapsulation, flexibility, and predictable behavior. Parameters encapsulates data needed for the function within its parameter to operate correctly. By calling the functions in various contexts, it is more flexible and reusable. Predicable behavior makes function's behavior more predictable as it doesn't depend on other global variables that could be changed.

Classes and SoC

Continuing on, another way to organize code is through the use of classes. Classes are a helpful way of grouping functions, variables, and constants by the name of the class. This way of organizing the code creates a modular structure making it easy to maintain and manage the code. Learning how to use

classes is a fundamental concept as classes provide a way to organize the code by grouping making the code more structure and readable. Static classes are also utilized for the case the function code never changes it will become “static”. This is indicated by using the @staticmethod decorator and Python will allow us to use the class functions directly, instead of creating an object first.

The separation of concerns (SoC) pattern is another fundamental software design principle that helps to maintain, scale, and allow for readability of code. This is done by breaking code down into distinct, self-contained components, each responsible for a specific aspect of the code. The “concern” refers to specific parts of the code’s functionality like the user interface logic, data storage/processing, input validation, and so much more. SoC is integral as it makes the code more maintainable and easier to work with. The focus of SoC in this course is to work on presentation, logic, and data storage concerns. It was taught that to organize the code, three primary layers are found within the program: 1. The Data Layer, 2. The Processing Layer, and 3. The Presentation Layer.

Writing the Program

This weeks assignment built upon Assignment 05 but added new concepts learned such as advanced data collections such as functions, parameters, classes, and SoC. To start the Assignment06.py and Assignment05-Starter.py file was utilized as a reference and to practice using others code.

Constants and Variables

First the program constants were defined and were not changed throughout the program. Two constants were defined, MENU and FILE_NAME. MENU was a string that displayed the menu for a Course Registration Program and four selections: 1. Register a Student for a Course, 2. Show current data, 3. Save data to a file, and 4. Exit the program. FILE_NAME was created and set to a value of “Enrollments.json” to aid in creating the file with a correct name.

Next two global variables were created for this program. These variables include: students and menu_choice. There were other local variables that were utilized within the different functions created.

Classes and Functions

Two classes were created, the FileProcessor and the IO. These two classes had many nested functions allowing for good organization. All functions created incorporates SoC to help with keeping track of the changes. The FileProcessor had two functions the read_data_from_file and write_data_to_file. The first function reads the file data from the Json file and presents error messages. While the later writes data to the file as done previously in other labs. Within the IO class there were five different functions: 1. output_error_messages, 2. output_menu, 3. input_menu_choice, 4. output_student_courses, and 5. input_student_data. The first function presents custom error messages to the user. The second function presents the menu to the user. The third function allows for the user to select a choice. The fourth function allows for the correct printing of the data. Finally, the fifth function allows for the student to input data when prompted.

Main Body

The main body of the code is much cleaner and organized as the lines of code are now nested within classes and functions. It was important to correctly write the different if and elif statements with the correct syntax as the smallest errors caused the code to stop. The while loop allows for repetitive

selections and inputs while the various if statements execute the different functions created performing the different tasks required.

Testing the Program

To successfully test the program, the program needed to do various things successfully. First was to take the users input for the first, last, and course name and then displaying it correctly (Choice 1 and 2). Then once done, saving the data to the .json file. Finally it is important that the code record, display, and save multiple registrations. The different error handling code integrated into the program was tested as a numerical value was inputted for the first and last name, no file being present to read, and errors while saving to the file. This was done successfully through executing the code both in PyCharm and the Command Shell.

Summary

In conclusion, the necessary tasks for Assignment 06 were completed. Tasks 1 through 3 were done to aid in learning and understanding the material for the week. Task 4 was completed through creating the script Assignment06.py. Finally task 5 was completed by writing this paper. Task 6 and 7 will be done following the completion of the documentation through posting files on GitHub. This week it was fun to learn more and build upon previous topics. This week the there was lots of content to learn and built upon previous knowledge at the same time. The program that was made this week was challenging but fun to make. To create the program this week the starter file was utilized and revised as this was encouraged. Learning how to use json files was interesting and building upon dictionary knowledge was helpful. The introduction of error handling was a great topic as it will help to ensure future programs will successfully run and could be shared and collaborated on. It was important to comment along the way as this was my feedback from the previous week. I was able to successfully implement new learning in achieving the correct outputs for this week's assignments.