Adnan Dawood

May 21, 2024

Foundations of Programming: Python

Assignment 06

The link to the repository in GitHub: https://github.com/adnandawood/IntroToProg-Python-Mod06

Functions

Introduction:

In Python programming, efficiency, readability, and maintainability are paramount. As scripts evolve from small-scale prototypes to complex applications, employing advanced techniques becomes essential for ensuring robustness and scalability. Among these techniques, leveraging functions, classes, and the separation of concerns programming pattern stand out as fundamental strategies for enhancing script quality and developer productivity.

In this assignment, we'll delve into three common techniques for improving Python scripts: functions, classes, and the separation of concerns programming pattern. Each of these techniques plays a distinct role in streamlining code structure, promoting reusability, and fostering code maintainability. By mastering these concepts, Python developers can unlock the full potential of their scripts, enabling them to tackle larger projects with confidence and efficiency.

Drafting the Code:
This Python code is for managing student course registrations. Let's break it down step by step:
  1- Imports:
import json: Imports the JSON module for handling JSON data.

  2- Class Definitions:
FileProcessor: Handles file input/output operations.
IO: Handles input/output operations such as displaying menu, taking user input, and showing student data.

  3- FileProcessor Methods:
read_data_from_file(file_name: str, student_data: list): Reads data from a JSON file and loads it into a list.
write_data_to_file(file_name: str, student_data: list): Writes data from a list to a JSON file.

4- IO Methods:

output_error_messages(message: str, error: Exception = None): Outputs error messages along with exception details.

output_menu(menu: str): Outputs the menu to the console.

input_menu_choice(): Prompts the user to input a menu choice.

output_student_courses(student_data: list): Outputs the student data to the console.

input_student_data(student_data: list): Prompts the user to input student data and adds it to the list.

5- Constants and Variables:

MENU: Menu options displayed to the user.

FILE_NAME: Name of the file to store student enrollments.

student_data: List to store student data.

menu_choice: Variable to store the user's menu choice.

6- Main Execution:

Reads data from the file into student_data.

Enters a while loop to repeatedly display the menu and process user input until the user chooses to exit.

Depending on the user's choice (registering a student, displaying data, saving data, or exiting), appropriate methods from FileProcessor and IO classes are called.

7- End of Program:

Once the user chooses to exit (menu_choice == "4"), the program breaks out of the loop and displays "Program Ended".
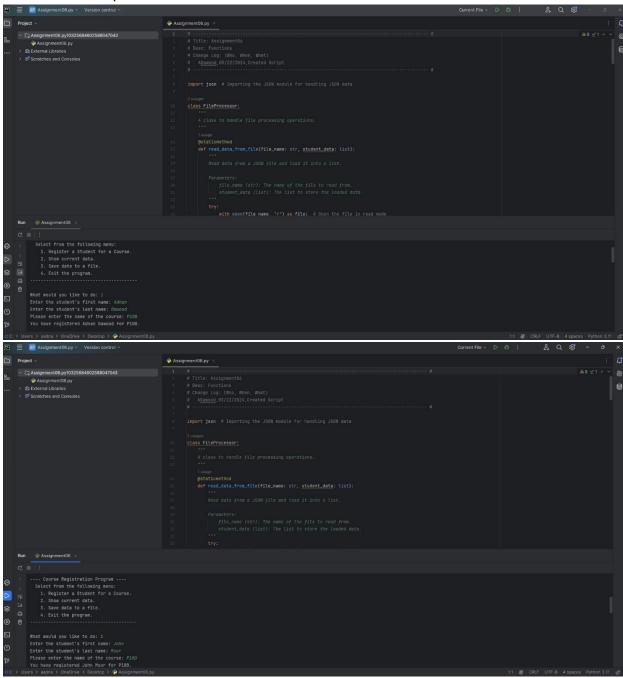
This program provides an interface for registering students for courses, displaying enrolled students, and saving the data to a JSON file.

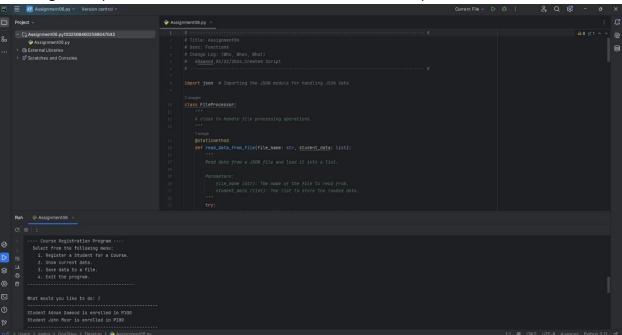**Testing the script and the findings:**

I used PyCharm to evaluate my script. Also, the script was evaluated in terminal.

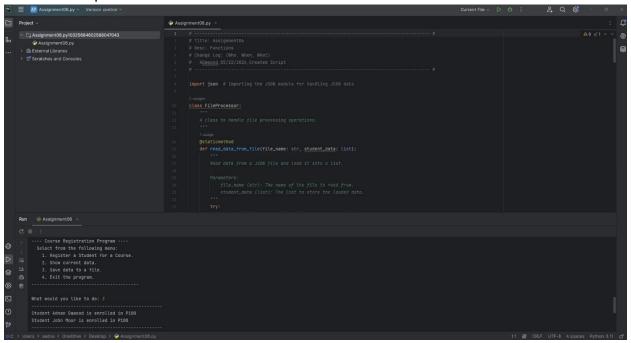### 1- Here is how my script looks like and its output in PyCharm:

a- Running the program and entering 1 for option one "Register a Student for a Course" as shown in the picture below:
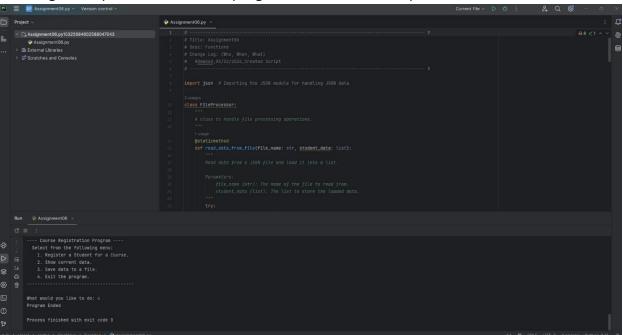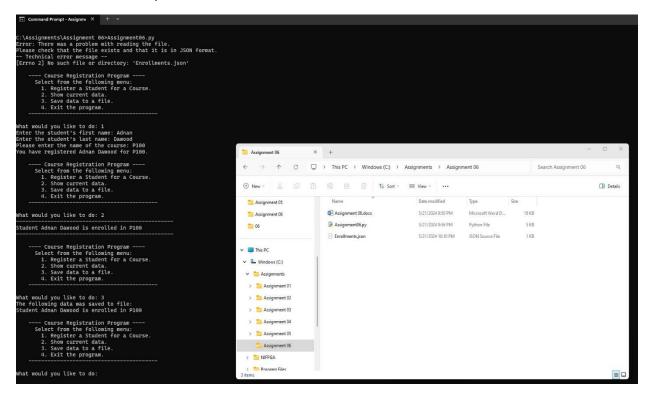
b- Entering 2 for option two "Show current data" as shown in the picture below:

c- Entering 3 for option three "Save data to a file (check the right side of the picture)" as shown in the picture below:

d- Entering 4 for option four "Exit the program" as shown in the picture below:

2- Here is how my script looks like and its output in terminal:

a- Running the program in terminal and entering 1 for option one "Register a Student for a Course" as shown in the picture below:

b- Entering 2 for option two "Show current data" as shown in the picture below:

c- Entering 3 for option three "Save data to a file (check the right side of the picture)" as shown in the picture below:

d- Entering 4 for option four "Exit the program" as shown in the picture below:

Summary:

Improving Python scripts often involves employing advanced techniques like functions, classes, and the separation of concerns programming patterns. Functions allow for modularizing code by encapsulating specific tasks, promoting code reuse and readability. Classes provide a blueprint for creating objects with attributes and methods, enabling the organization of related functionality into cohesive units. Additionally, the separation of concerns programming pattern emphasizes dividing code into distinct modules, each responsible for a specific aspect of functionality, leading to clearer code structure and easier maintenance. By mastering these techniques, Python developers can enhance script quality, scalability, and maintainability, ultimately empowering them to tackle complex projects with efficiency and confidence.