**Procedural Programming:**

**Introduction:**

The provided solution demonstrates the application of paradigms, specifically procedural programming and data manipulation using pandas, to process and analyze student performance data. The solution utilizes the Python programming language and the pandas library to perform calculations and generate insights from the data provided. This report discusses how these paradigms were applied in the solution and evaluates their effectiveness.

**Procedural Programming Paradigm:**

The solution primarily follows a procedural programming paradigm, where the focus is on executing a series of steps to accomplish a task. Here are the key aspects of procedural programming employed in the solution:

**A. Step-by-step Execution:**

The solution is structured in a sequential manner, with each step being executed in the order they are written. It starts by defining the GPA scale, then reads the CSV file using pandas, calculates letter grades for each module, computes GPAs for each student, and finally calculates the overall GPA.

**B. Control Flow and Conditionals:**

The solution uses conditional statements, specifically the if-else construct, to assign letter grades based on the marks obtained by each student. The conditional statements allow for different outcomes based on the ranges defined in the GPA scale.

**D. Iteration:**

The solution iterates over the module columns to calculate the letter grade for each module. It uses a *for loop* to perform the same set of operations on each module.

The procedural programming paradigm breaks down the solution into manageable steps and leverages control flow and iteration constructs to handle different scenarios efficiently.

**Conclusion:**

The application of paradigms in the solution, including procedural programming and data manipulation using pandas, has been effective in processing and analyzing student performance data. Procedural programming provides a clear and organized approach to solving the problem, breaking it down into sequential steps and utilizing control flow constructs for decision-making.

**Object Orientated Programming (OOP):**

**Introduction:**

This code includes elements of Object-Oriented Programming (OOP) in C. Although C is not a pure object-oriented language, it is possible to implement certain concepts of OOP using struct and function compositions.

In the given code, the structures GPA\_Scale, Module, and Student represent objects with their respective attributes and behaviors. These structures encapsulate data and related functions.

**The code also demonstrates the following key concepts of OOP:**

**Encapsulation:** The structures (GPA\_Scale, Module, and Student) encapsulate data and related functions together, keeping them organized and self-contained.

**Abstraction:** The structures hide their internal implementation details from the outside world. The user interacts with these structures through well-defined functions (calculate\_letter\_grade, calculate\_gpa, calculate\_highest\_scoring\_module, and process\_data) without needing to know the underlying implementation.

**Data Hiding:** The member variables of the structures (GPA\_Scale, Module, and Student) are declared with the private access modifier by default in C, preventing direct access from outside the structure. Functions like calculate\_letter\_grade and calculate\_gpa provide an interface to modify and access the data in a controlled manner.

**Modularity:** The code is divided into smaller functions, each responsible for a specific task. This promotes modularity, reusability, and easier maintenance.

**NB.:** It's important to note that this code does not utilize concepts such as inheritance or polymorphism, which are commonly associated with OOP.

**Data Manipulation using pandas (Extra Procedural)**:

The solution utilizes the pandas library to manipulate and analyze tabular data. Here are the key instances of data manipulation using pandas in the solution:

**A. Reading the CSV file:**

The pandas read\_csv function is used to read the input CSV file and create a DataFrame. This allows for easy access and manipulation of the data in a tabular format.

**B. Creating New Columns:**

The solution creates new columns in the DataFrame to store the calculated letter grades for each module. It uses the apply method along with a lambda function to apply the grading logic to each mark in the module columns.

C. **Calculating GPAs:**

The solution calculates the GPA for each student by applying the GPA scale to the letter grades for each module. It uses pandas' applymap method to map the letter grades to their corresponding GPA values and then calculates the mean GPA across all modules for each student.

**D. Calculating Overall GPA:**

The solution calculates the overall GPA by taking the mean of all the calculated GPAs for each student.

The use of pandas simplifies data manipulation tasks, allowing for concise and efficient code. The library provides powerful functions and methods that enable operations on tabular data in an intuitive manner.

Data manipulation using pandas allows for easy handling and analysis of tabular data, simplifying complex operations and enhancing code readability. Overall, the solution demonstrates a structured and efficient approach to processing and analyzing student performance data, leveraging the strengths of procedural programming and pandas for effective problem-solving.