AI ASSISTED CODING

Rohith reddy **2303A51111**

BATCH – 03 **20 – 02 – 2026**

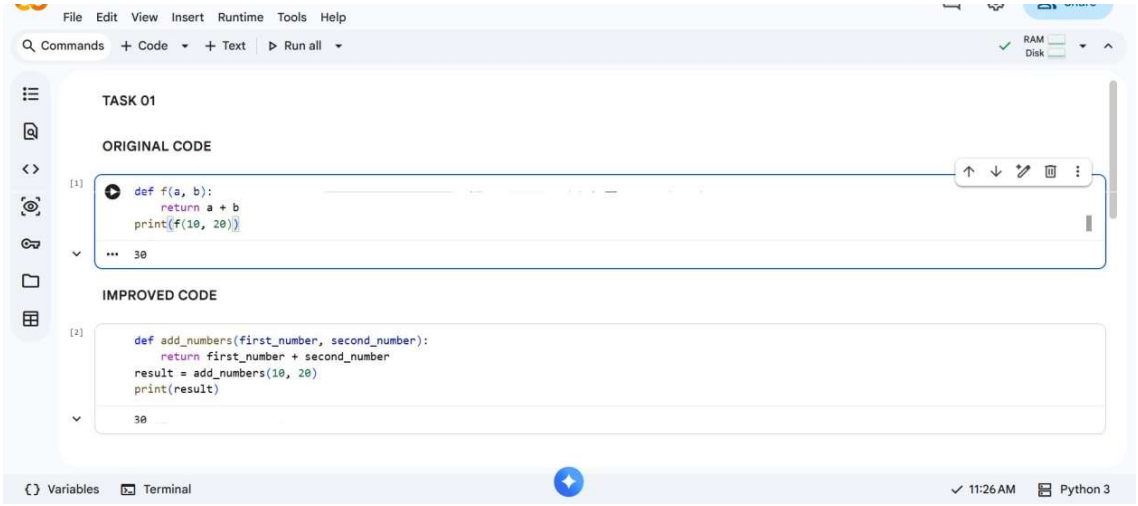
ASSIGNMENT – 10.5

LAB – 10.5 : Code Review and Quality : Using AI to improve code Quality and Readability.

Task – 01: Variable Naming Issues.

Prompt: Review the following Python code and improve it by replacing unclear function and variable names with meaningful and descriptive names. Refactor the code to follow PEP 8 standards and improve readability and maintainability without changing its functionality.

Code & Output:



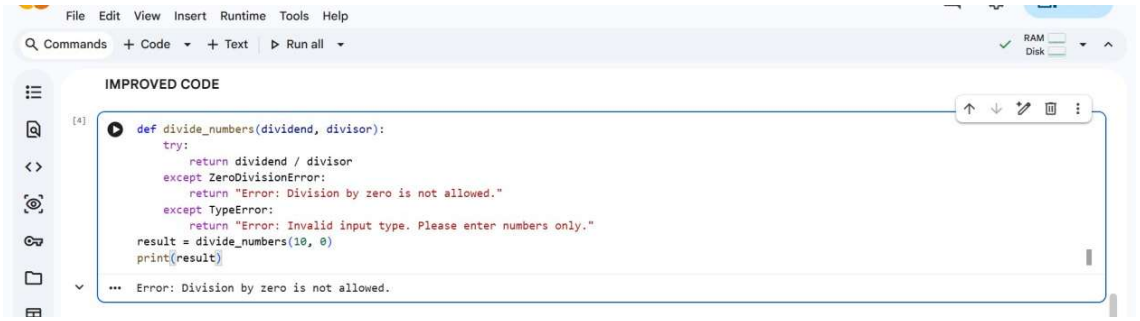
Explanation :

The original code used unclear function and variable names, making it difficult to understand its purpose. AI improved the code by using meaningful names and adding structure, which enhances readability and maintainability.

Task – 02 : Missing Error Handling.

Prompt: Review the following Python code and improve it by adding proper error handling. Handle possible exceptions such as division by zero and invalid input types. Refactor the code to follow PEP 8 standards, use meaningful variable names, and provide clear, user-friendly error messages without changing the core functionality.

Code & Output :



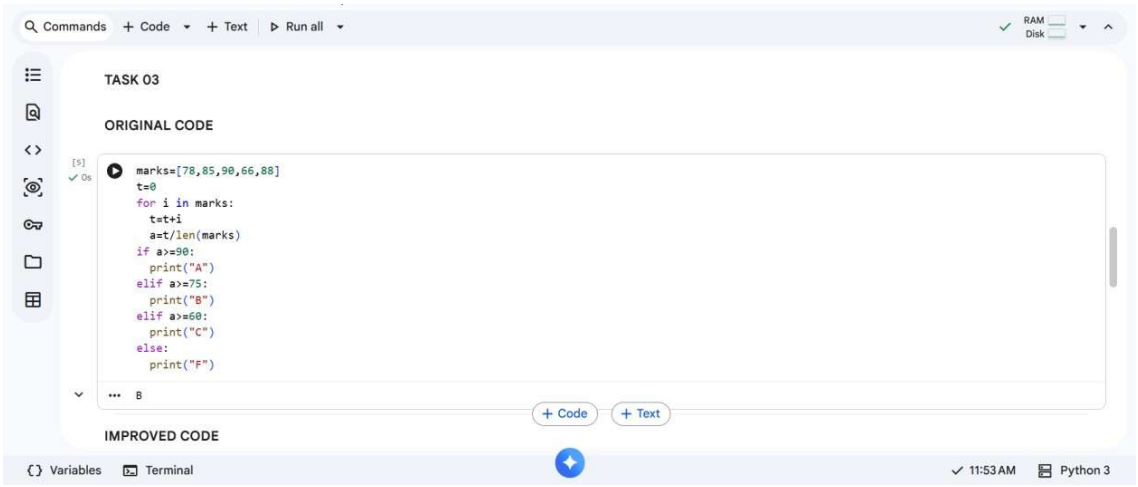
Explanation :

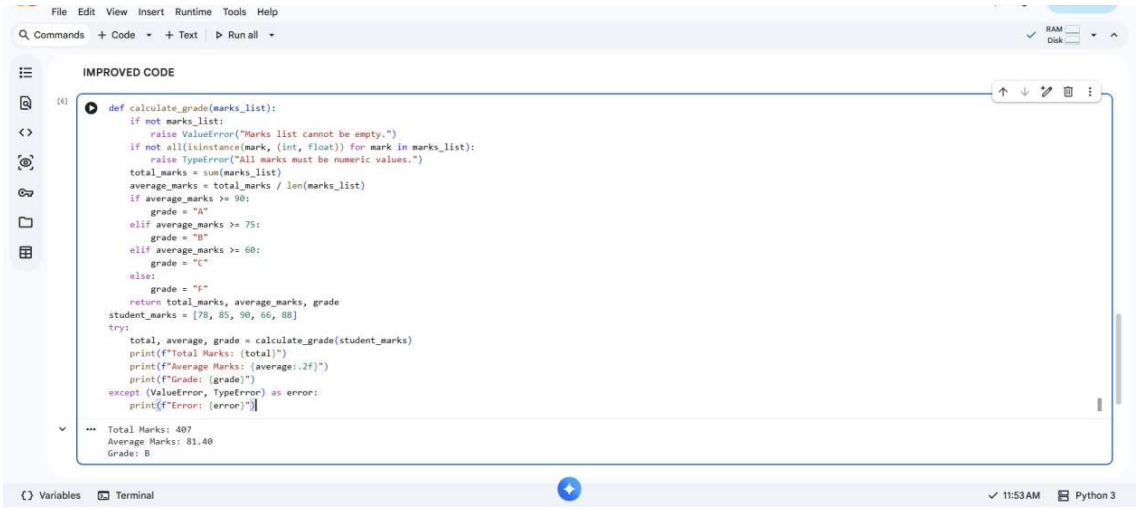
The original code does not handle runtime errors like division by zero, which can cause the program to crash. The improved version adds exception handling to manage errors gracefully and display clear, user-friendly messages. This enhances program reliability, robustness, and overall code quality.

Task – 03 : Student Marks Processing System.

Prompt : Review the following Python program and refactor it to improve readability, structure, and code quality. Follow PEP 8 standards, use meaningful variable and function names, and convert the logic into reusable functions. Add proper input validation, error handling, comments, and a clear docstring. Do not change the core functionality.

Code & Output :





Explanation :

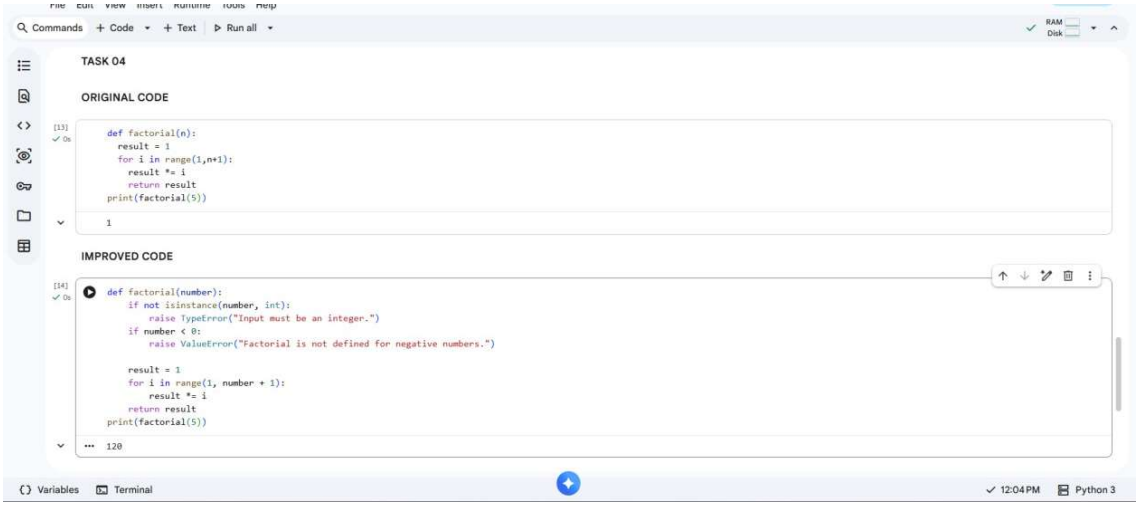
The original program had poor variable naming, no function structure, and lacked input validation, making it difficult to maintain and understand. The refactored version follows PEP 8 standards, uses meaningful names, and organizes the logic into reusable functions with proper validation. This improves readability, maintainability, and overall code quality.

Task – 04: Use AI to add docstrings and inline comments to the following Function.

Prompt : Review the following Python function and enhance it by adding a proper docstring and meaningful inline comments. Ensure the

documentation explains the purpose, parameters, return value, and possible exceptions. Follow PEP 8 standards and improve readability without changing the core functionality.

Code & Output :



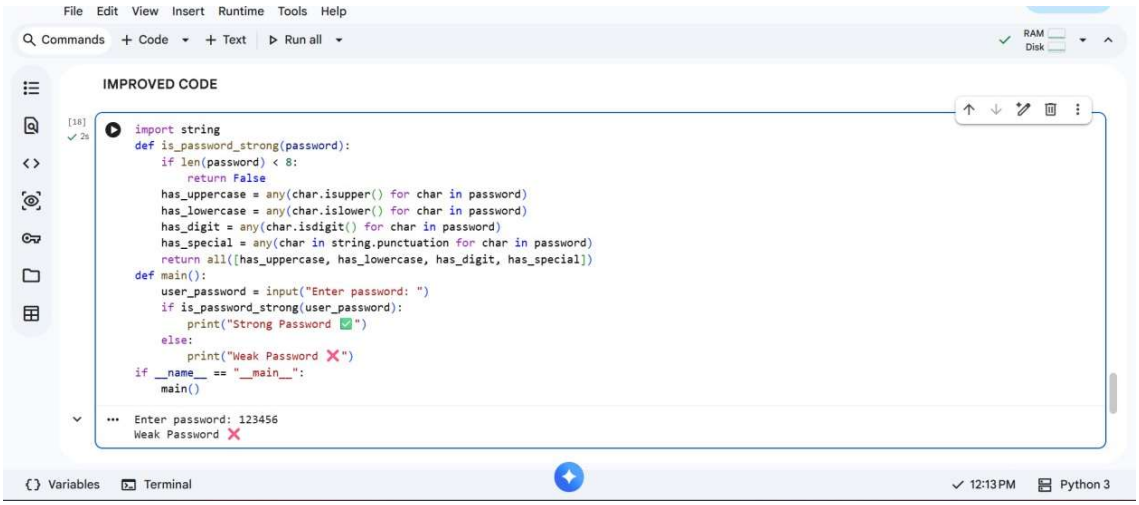
Explanation :

The original function lacked documentation and comments, making it harder to understand its purpose and logic. The improved version adds a clear docstring and inline comments, explaining the functionality, parameters, and return value. This enhances readability, maintainability, and adherence to coding best practices.

Task – 05 : Password Validation System.

Prompt : Refactor the code using meaningful function names, PEP 8 standards, and include a proper docstring with inline comments.

Code & Output :



Explanation :

Maintainability & Reusability

- Password logic inside a function
- Can reuse in web apps, login systems
- Easy to update rules

Password Security Rules

Rule	Why It Improves Security
Minimum Length	Prevents short brute-force attacks
Uppercase	Increases complexity
Lowercase	Improves character variation
Digit	Adds numeric complexity
Special Character	Maximizes entropy

Maintainability & Reusability

Original	Enhanced
✗ Cannot reuse validation logic	✓ **is_password_strong()** reusable
✗ Hard to extend	✓ Easy to add new rules
✗ No separation of concerns	✓ Logic separated from input/output

The enhanced version is more **maintainable** and **scalable.**

Original	Enhanced
Single condition	Multiple structured rules
No function	Modular function
No comments	Docstring + inline comments
Poor naming	Clear naming