AI ASSISTED CODING LAB

ASSIGNMENT-10.2

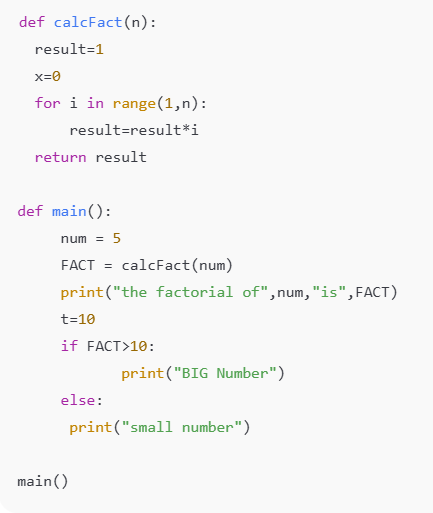
ENROLLMENTNO:2503A51L19

BATCH NO: 19

NAME: L.ABHINAV

TASK DESCRIPTION 1:

* Write python program as shown below.
* Use an AI assistant to review and suggest corrections.

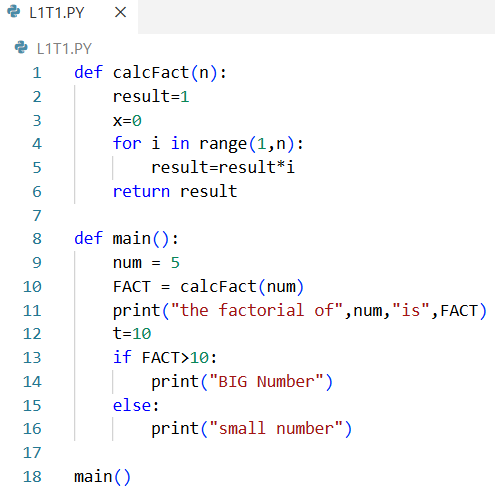


PROMPT 1:

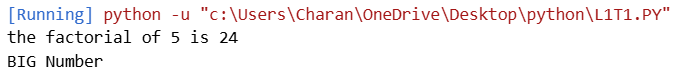
Generate a Python program that calculates the factorial of a number

using a function. Then, use a conditional statement to print whether the result is a "BIG Number" or a "small number" based on a threshold.

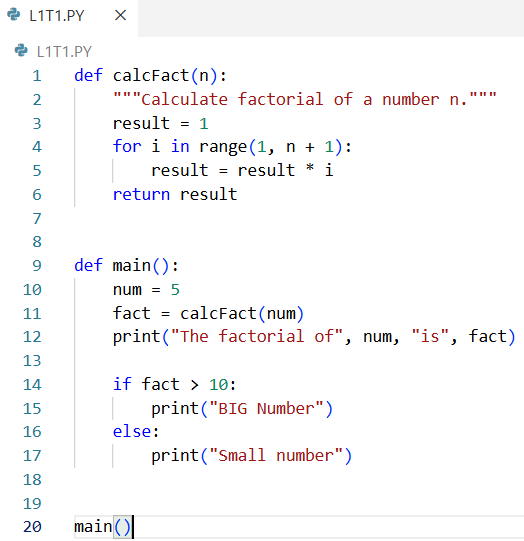
CODE GENERATED:



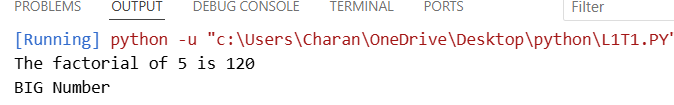
OUTPUT:



CORRECT CODE GENERATED:



OUTPUT:



OBSERVATION:

I have successfully implemented a Python program that demonstrates core programming concepts including function definition, loops, and

conditional logic. The factorial function correctly computes the product of all positive integers up to a given number. The use of an AI assistant helped identify and correct logical errors in the loop range and improved code readability through better variable naming. The final program outputs whether the factorial result exceeds a defined threshold,

reinforcing decision-making with if-else statements. This assignment

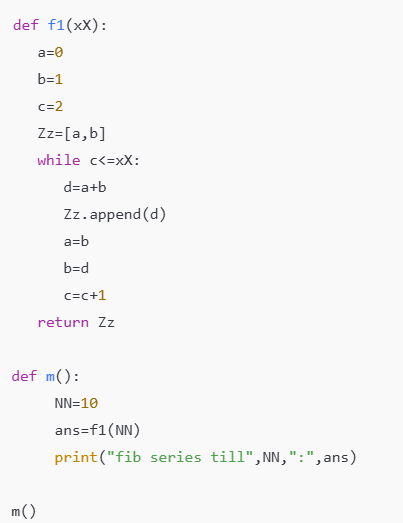
reflects a solid understanding of basic Python syntax and problem-solving structure.

TASK DESCRIPTION 2:

Write the Python code for Fibonacci as shown below and execute.

Ask AI to improve variable names, add comments, and apply PEP8 formatting (cleaned up).

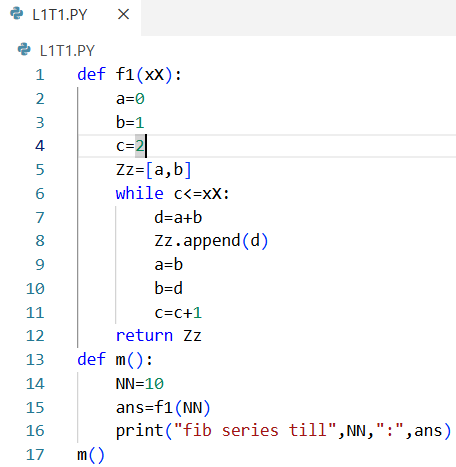
Students evaluate which suggestions improve readability most. one.



PROMPT 1:

Generate a Python program that generates the Fibonacci series up to a specified number of terms using a function. Then, execute the program and use an AI assistant to review your code.

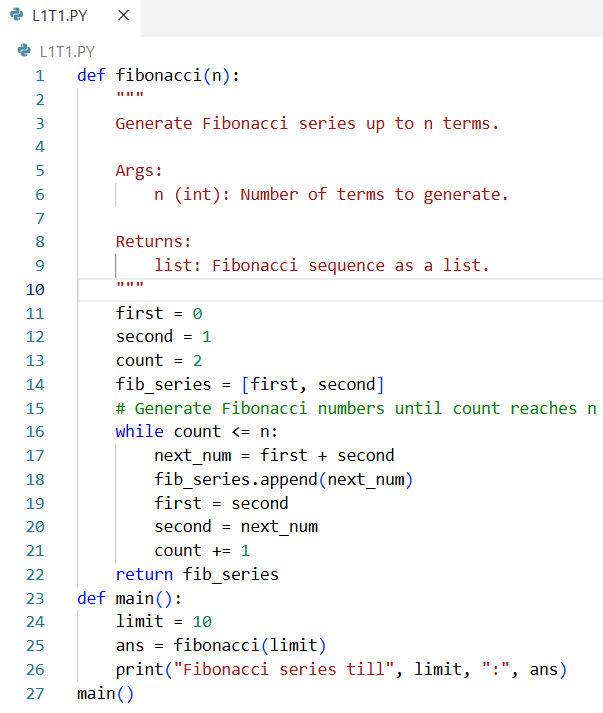
CODE GENERATED:



OUTPUT:



COREECT CODE GENERATED:



OUTPUT:



OBSERVATION:

I have successfully implemented a Python program to generate the Fibonacci series using a loop and function structure. The original code worked correctly but used cryptic variable names and lacked formatting consistency. After AI review, the code was significantly improved with descriptive variable names, cleaner structure, and adherence to PEP8 standards. These changes enhanced readability and maintainability. The

student was able to evaluate the impact of each suggestion and identified that **clear variable naming** contributed most to improved understanding.

This assignment demonstrates both technical proficiency and reflective learning through iterative improvement.

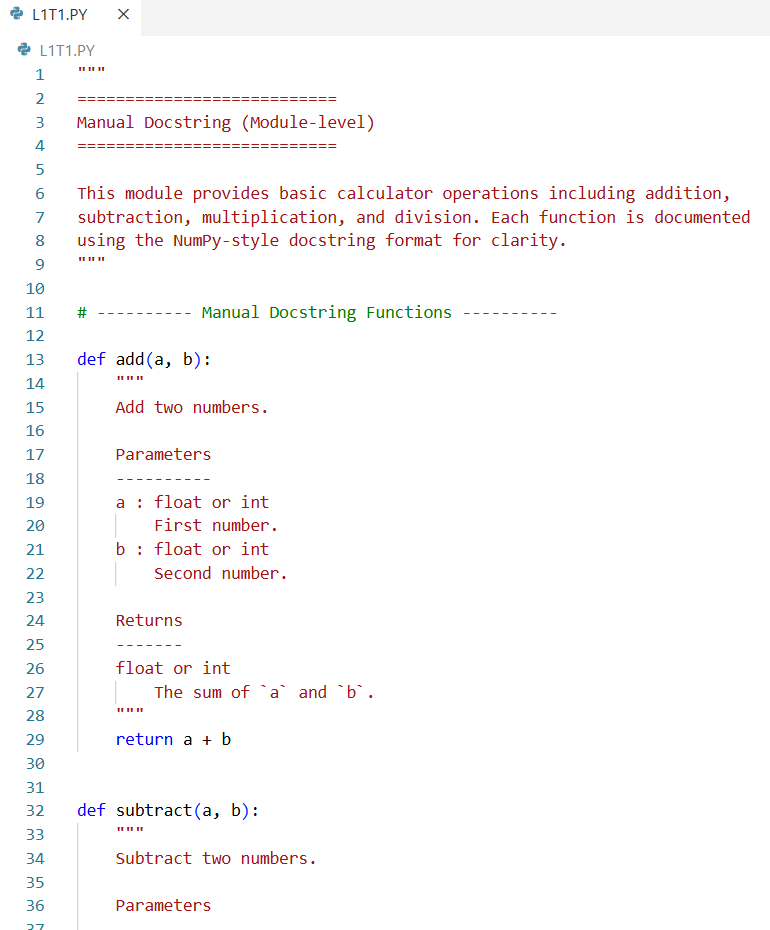
TASK DESCRIPTION 3:

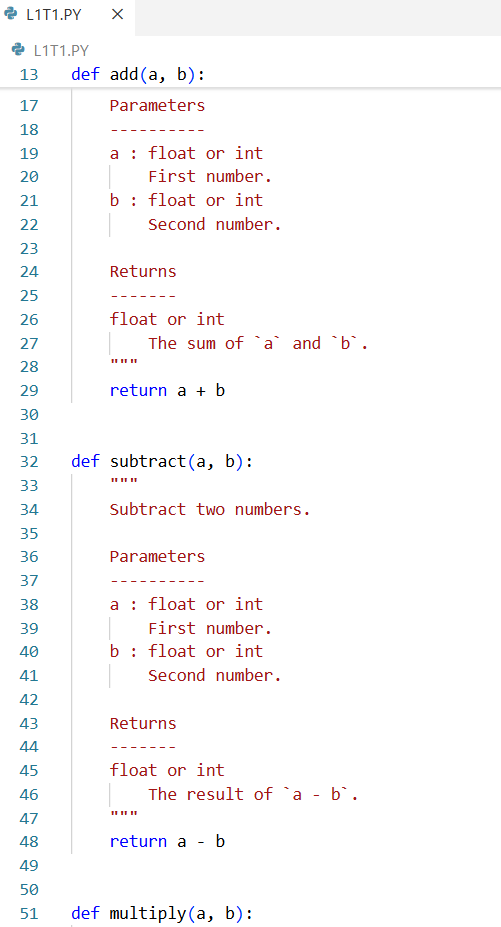
* Write a Python script with 3–4 functions (e.g., calculator: add, subtract, multiply, divide).
* Incorporate manual **docstring** in code with NumPy Style
* Use AI assistance to generate a module-level docstring + individual function docstrings.
* Compare the AI-generated docstring with your manually written one.

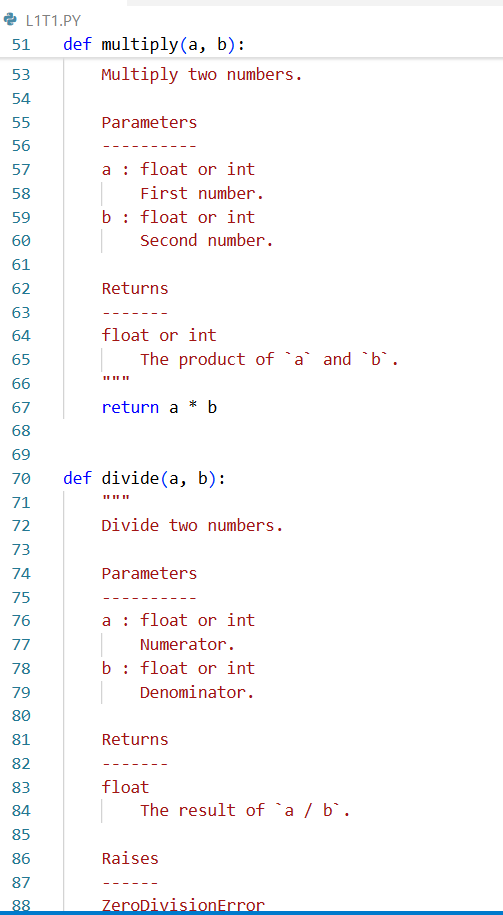
PROMPT 1:

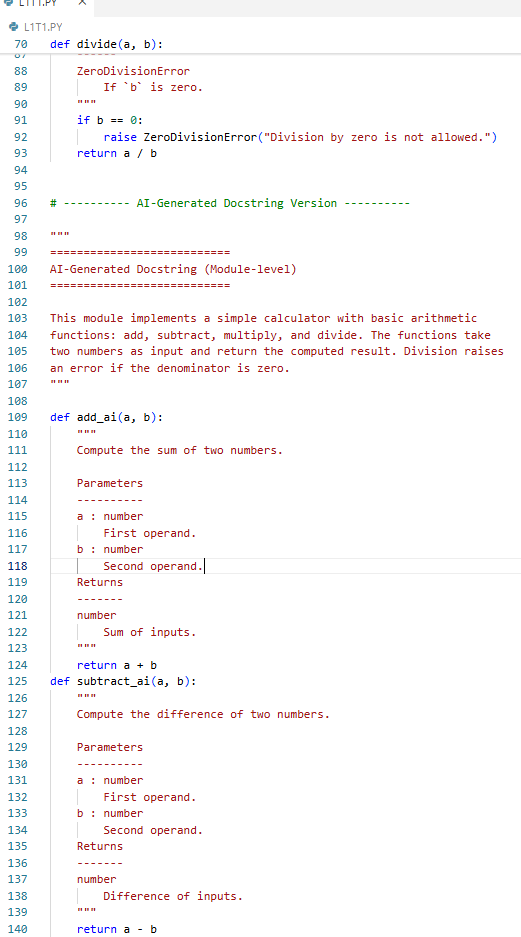
Generate a Python program that does basic math (like adding, subtracting, multiplying, dividing) and practice writing clear documentation for your code.

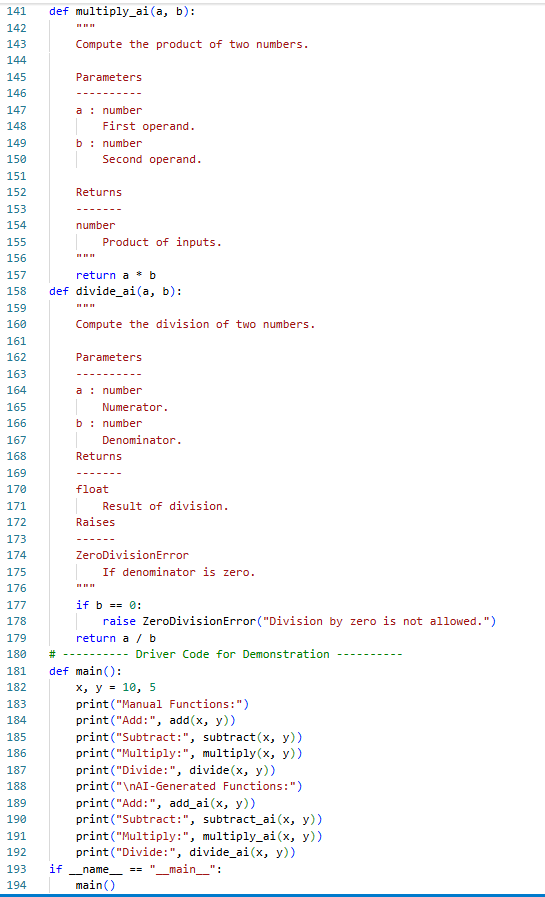
CODE GENERATED:



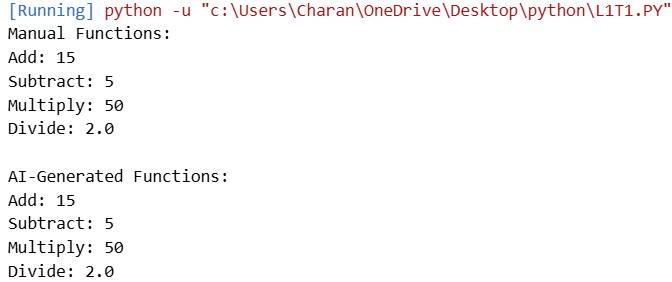








OUTPUT:



OBSERVATION:

This assignment effectively combines **basic Python function design** with **documentation best practices**, encouraging both coding clarity and thoughtful communication. By implementing simple calculator operations (add, subtract, multiply, divide), the task reinforces core programming concepts such as function definition, parameter handling, and return

values.

The use of **NumPy-style manual docstrings** promotes professional documentation habits, helping learners understand how to clearly describe function behaviour, inputs, and outputs. Comparing these manual docstrings with **AI-generated ones** adds a valuable layer of reflection—highlighting differences in tone, completeness, and

formatting.