LAB-9.3

TOPIC :

Documentation Generation.

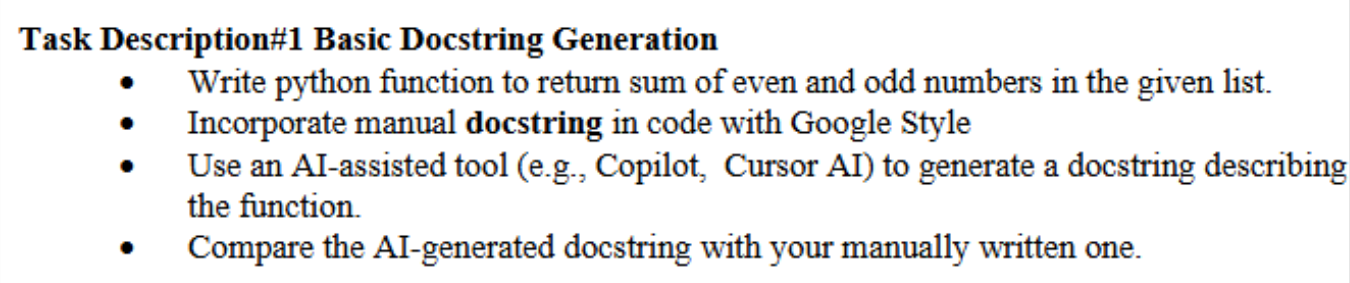
Name : K.Pardhasaradhi

Enrollment\_no : 2403a52001

Batch\_no : 01

Date: 16-09-2025

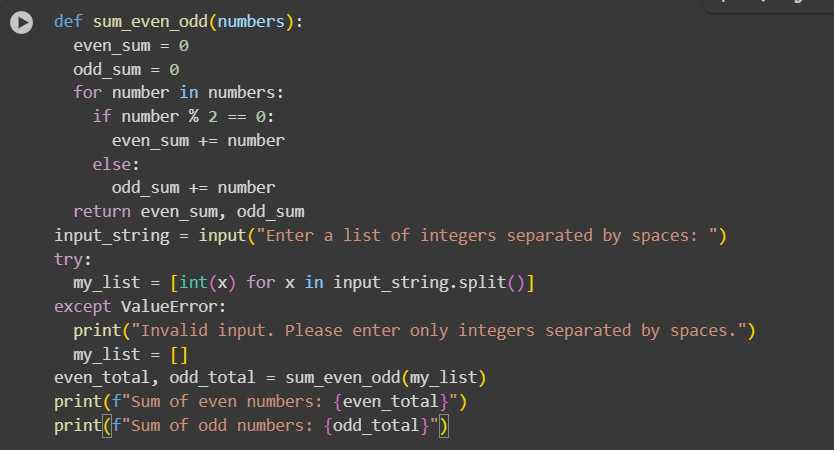
Task-01:



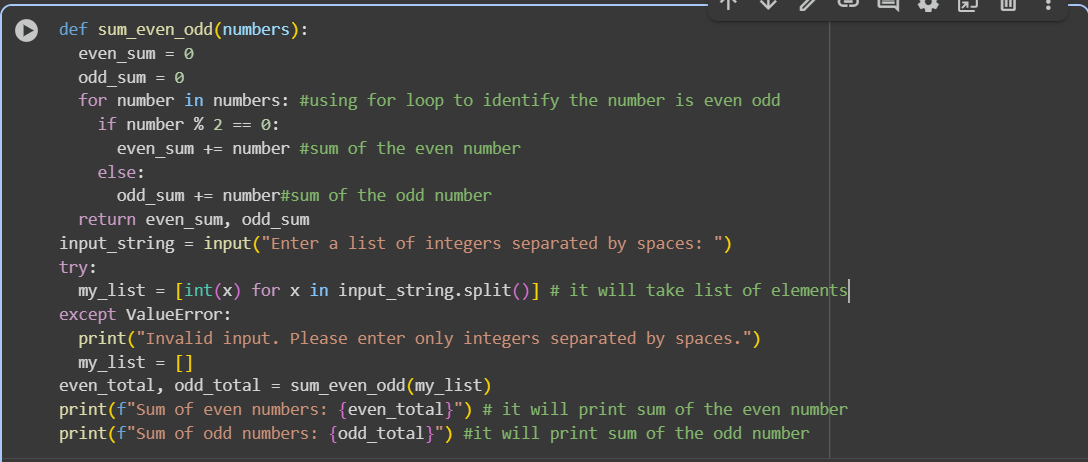
Used prompt :

Write a Python function that takes a list of integers and returns the sum of even and odd numbers separately. Add a docstring using Google-style format.

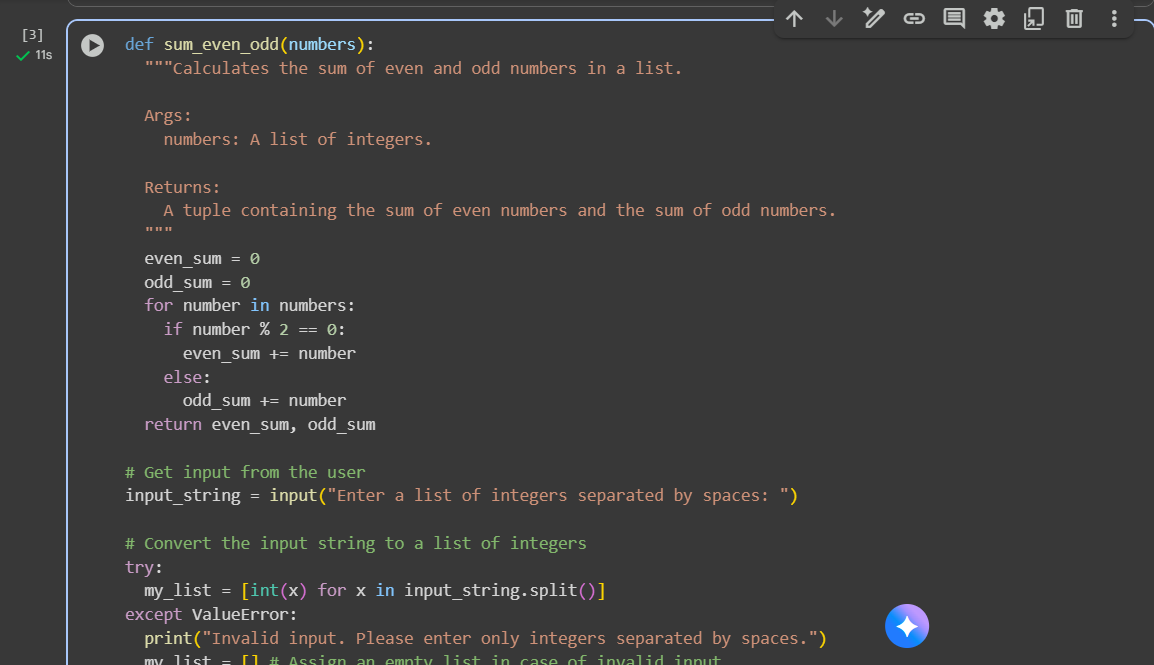
Code:

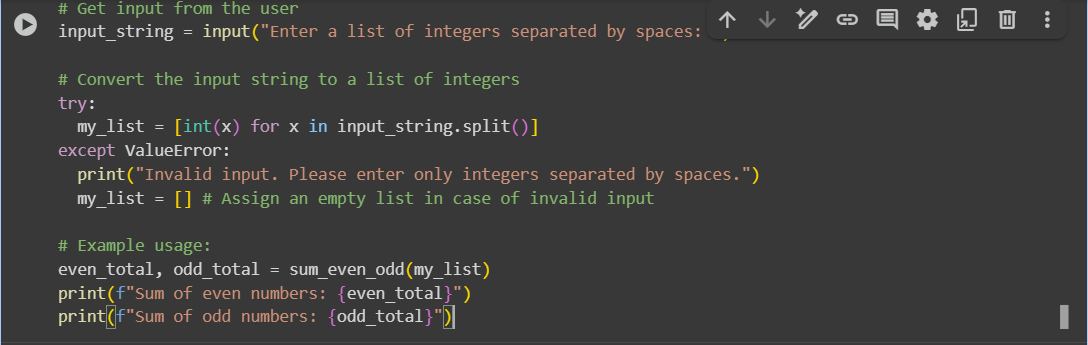


Comments entered by manually:

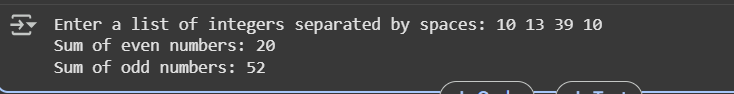


Comments generated by AI:



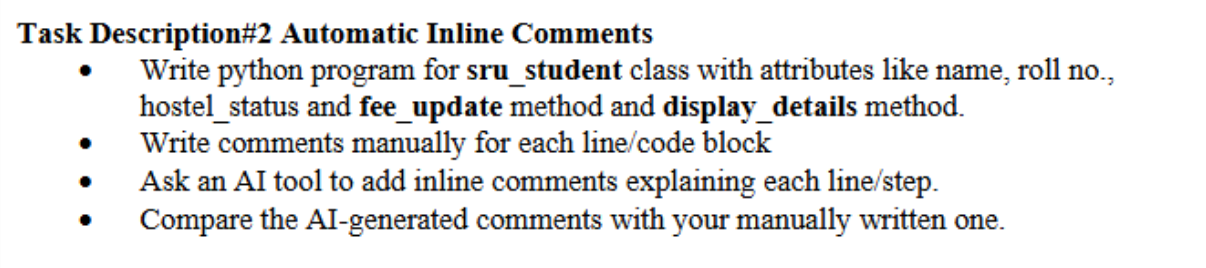


Output:



Observation: The comment generated by manually is very understandable and readable. But the comments generated by AI is very deep and take some time to understand.

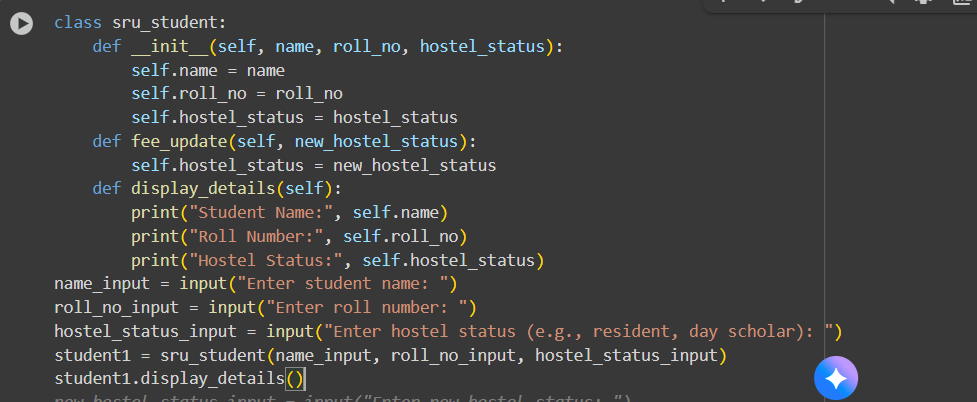
Task-02:



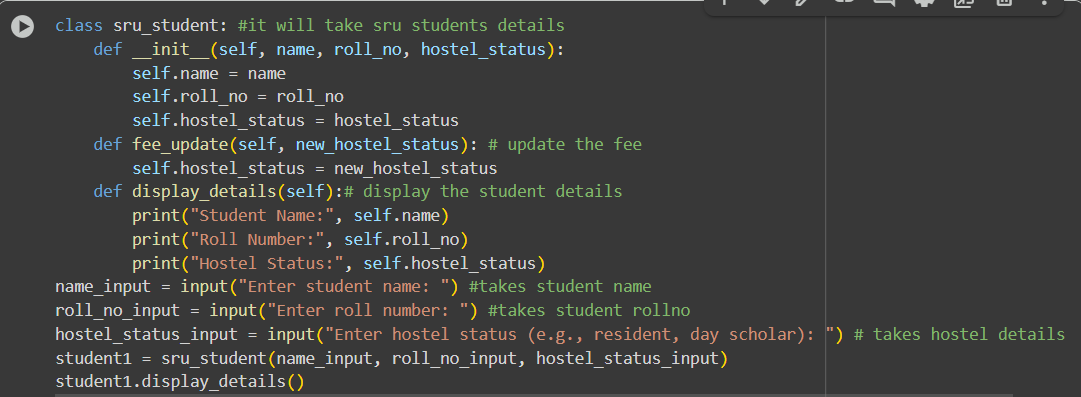
Used prompt :

Generate the python program for sru\_student class with attributes like name, roll no., hostel\_status and fee\_update method and display\_details method. Write comments manually for each line/code block. Input should be taken from the user.

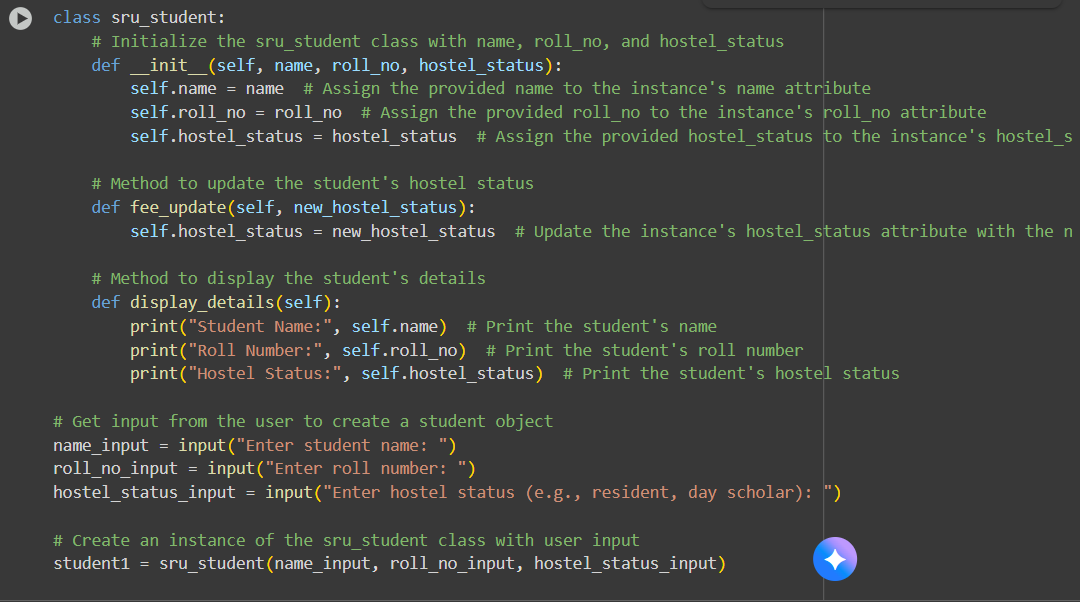
Code:

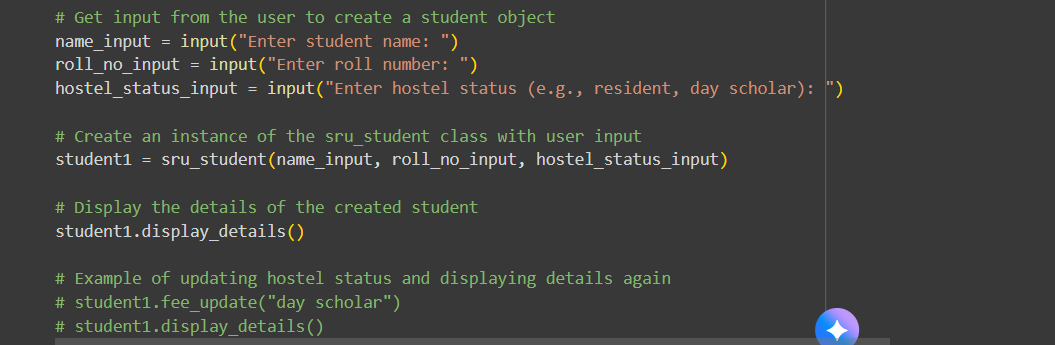


Comments entered by manually:

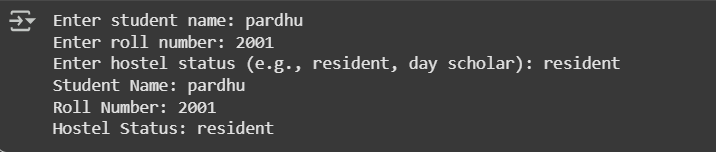


Comments generated by AI:



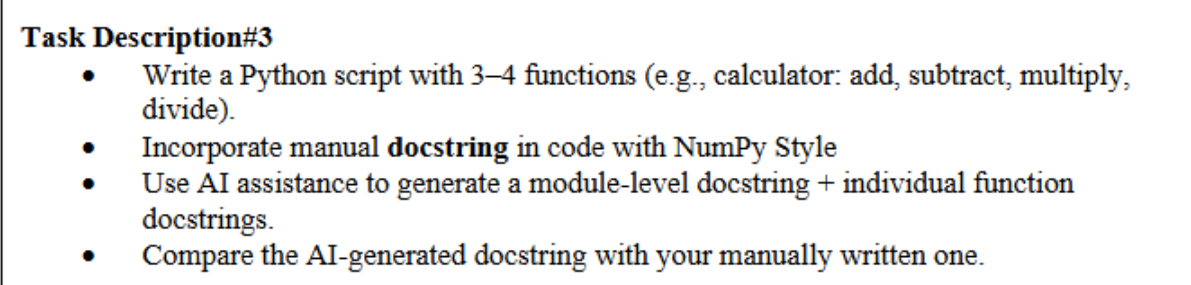


Output:



Observation: The comment generated by manually is very understandable and readable. But the comments generated by AI is very deep and take some time to understand.

Task-03:

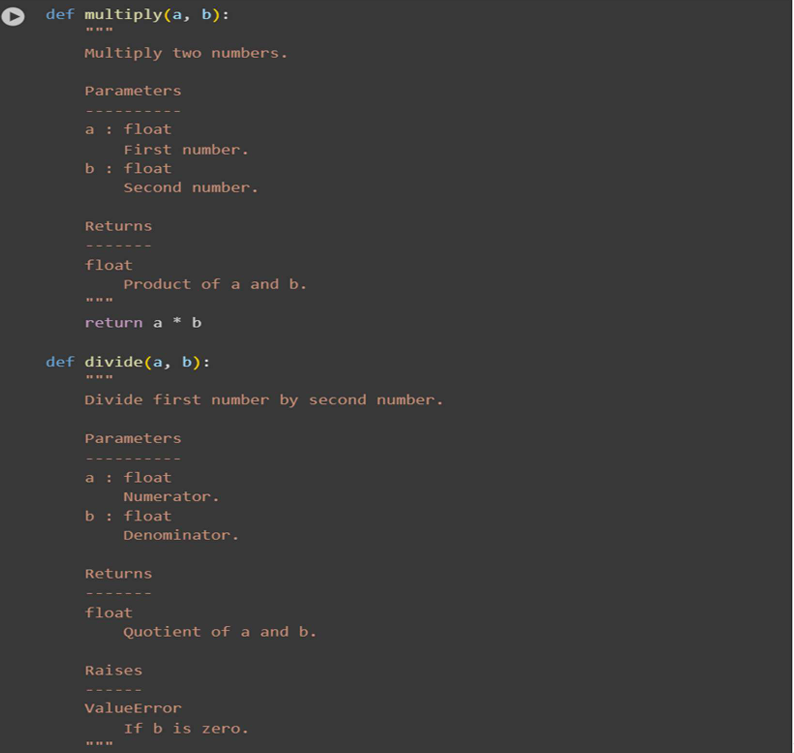


Used prompt :

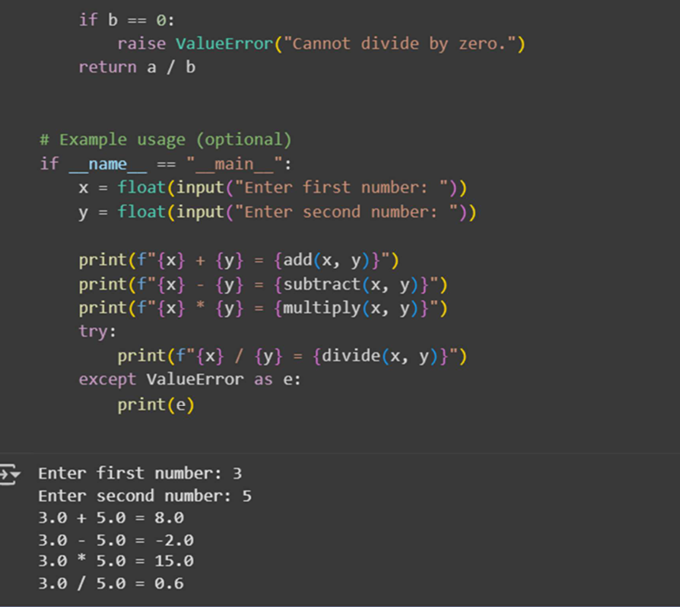
Write a Python script with four calculator functions: add, subtract, multiply, and divide.

Add manual docstrings to each function using the NumPy style. Then generate a module-level docstring and function docstrings

using AI.







Comments generated by AI:



Observation: The manual input code is simple and easy to understand but lacks error handling for invalid inputs. The AI-generated code adds input validation and organizes the logic inside a main() function, making it more robust and maintainable. Overall, the AI version improves user experience and code structure without adding much complexity.