**ASSIGNMENT-9**

**NAME : G.SAI RAJ**

**HT NO: 2403A52149**

**BATCH:** 24BTCAIAIB06

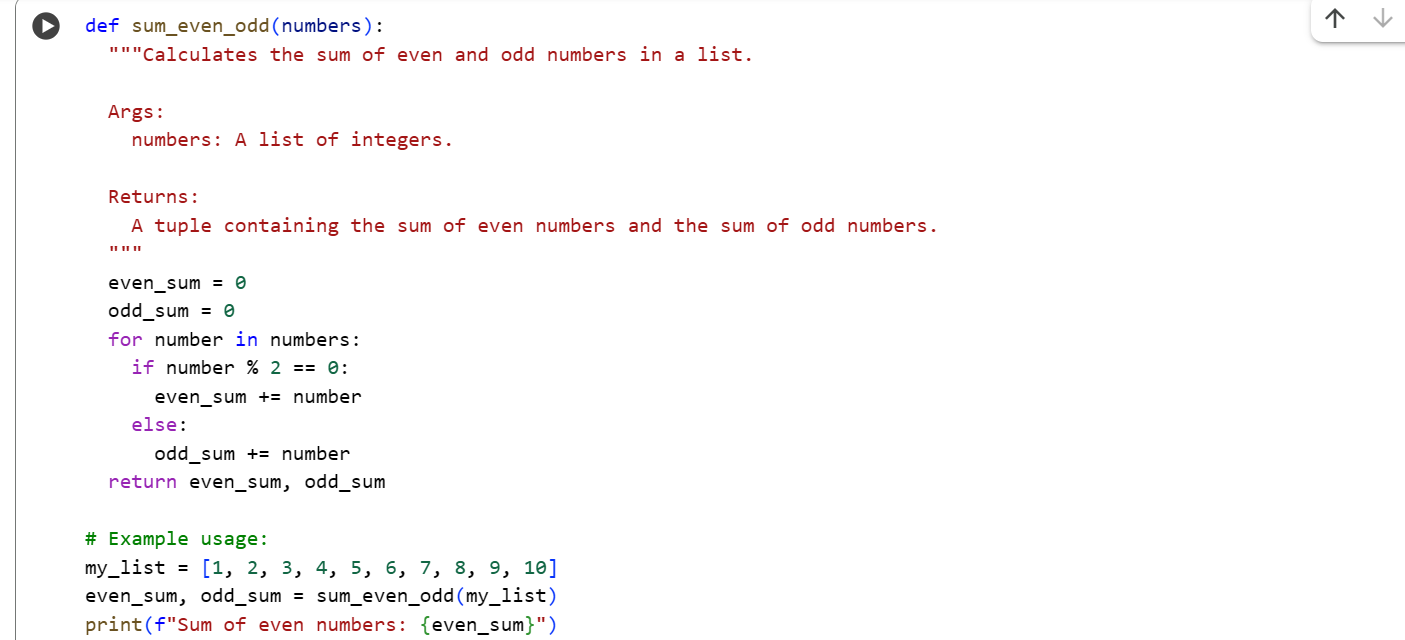
**TASK-1:**

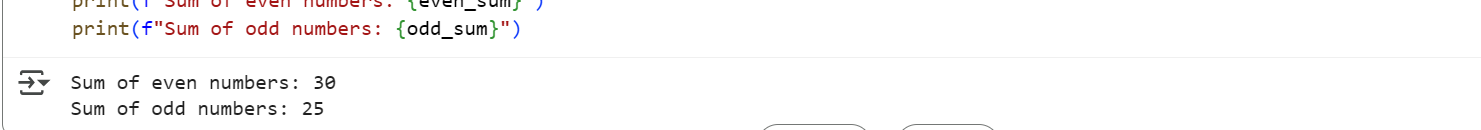
Task Description#1 Basic Docstring Generation

* Write python function to return sum of even and odd numbers in the given list.
* Incorporate manual docstring in code with Google Style
* Use an AI-assisted tool (e.g., Copilot, Cursor AI) to generate a docstring describing the function.
* Compare the AI-generated docstring with your manually written one.

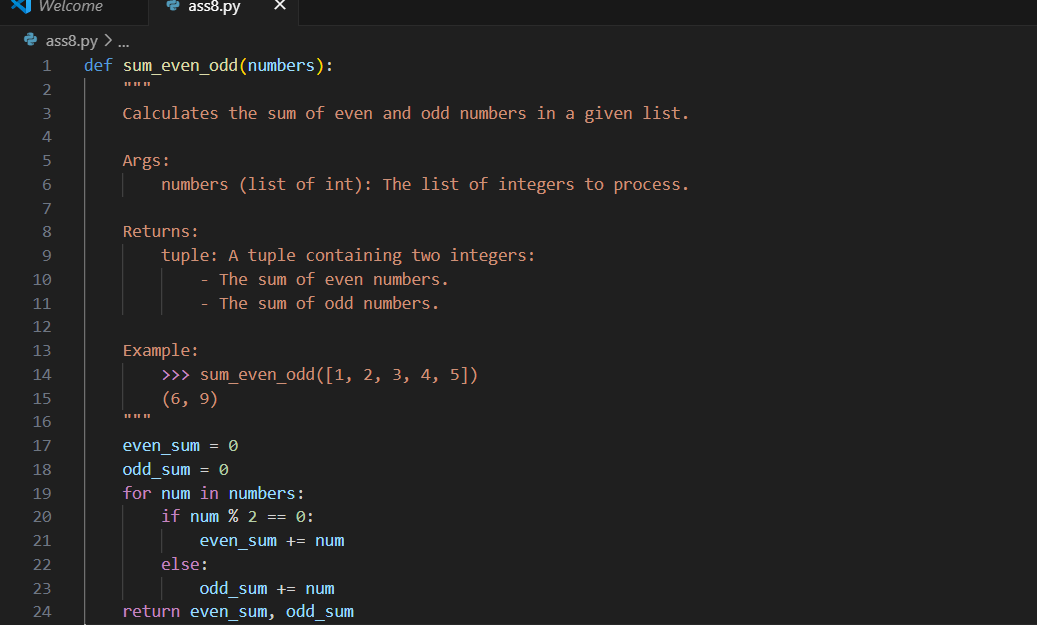
**CODE AND OUTPUT:**

**GEMINI:**

****

****

**COPILOT:**

****

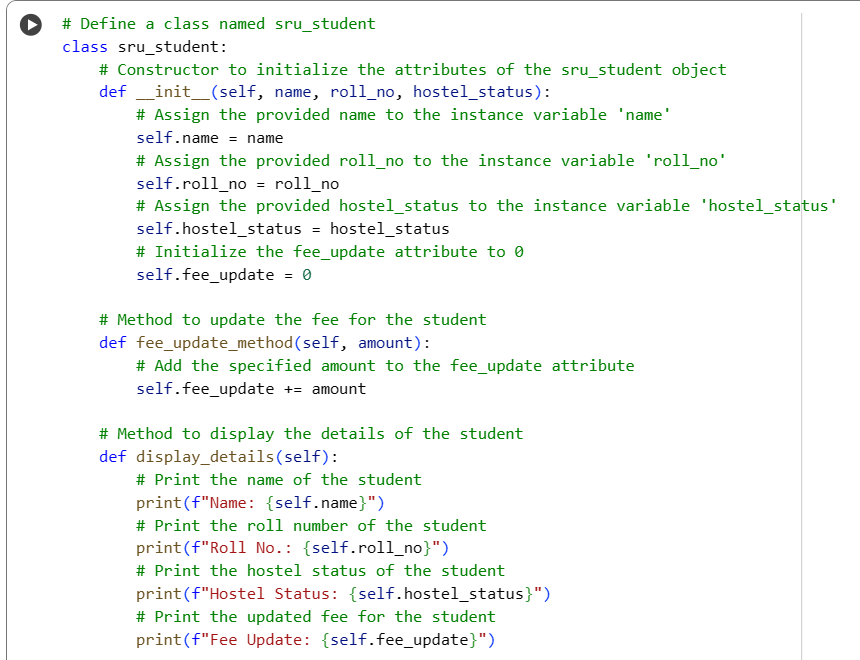
**TASK-2:**

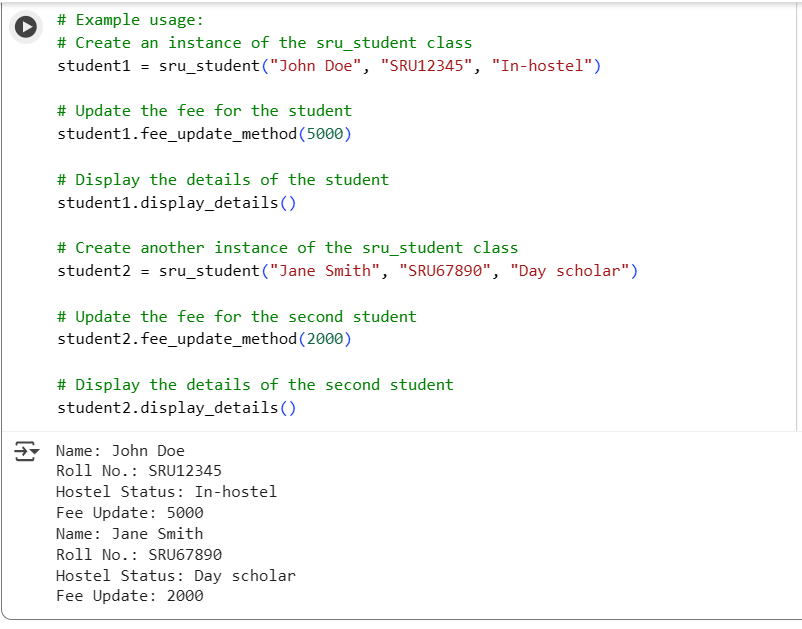
Prompt:

Generate 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/step.

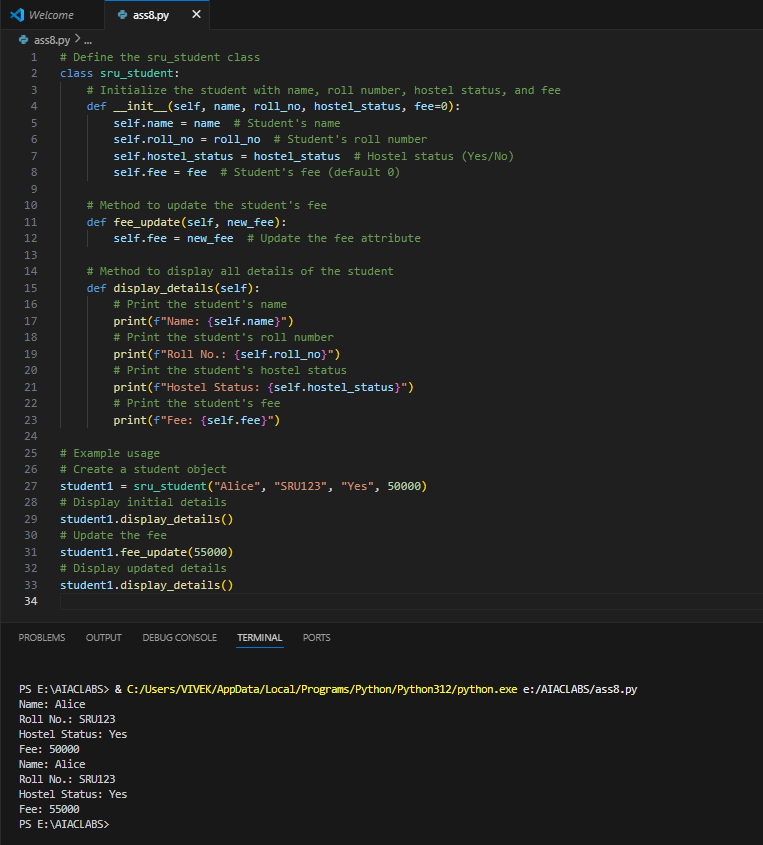
**CODE AND OUPUT:**

**GEMINI:**

****

****

**COPILOT:**

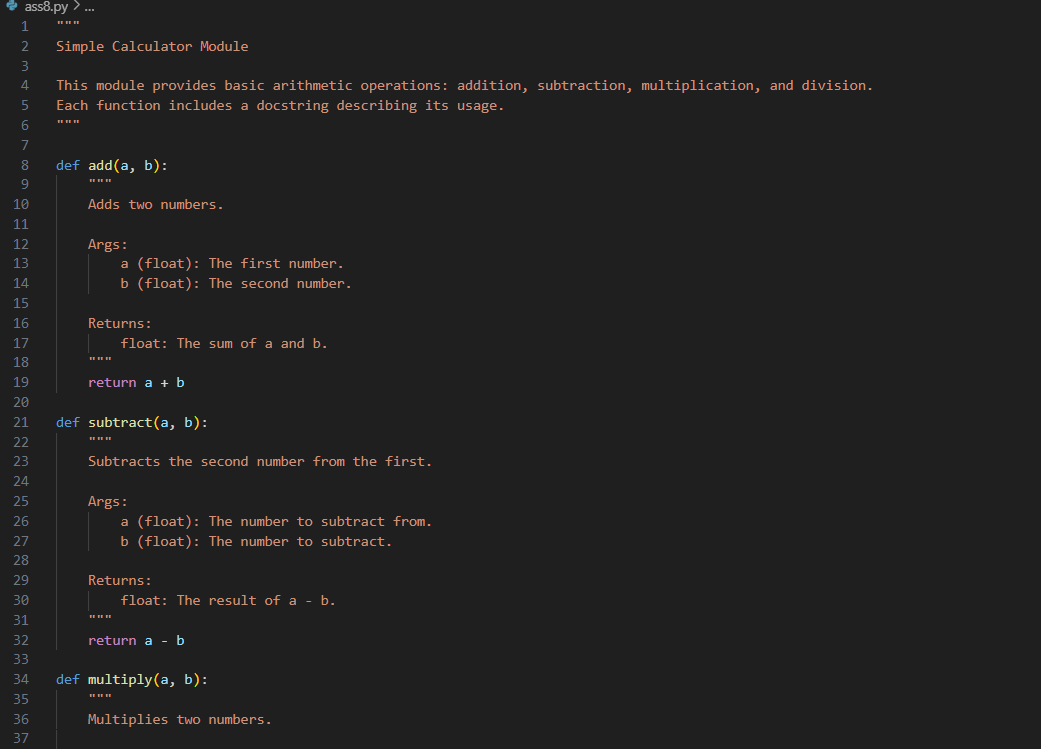
****

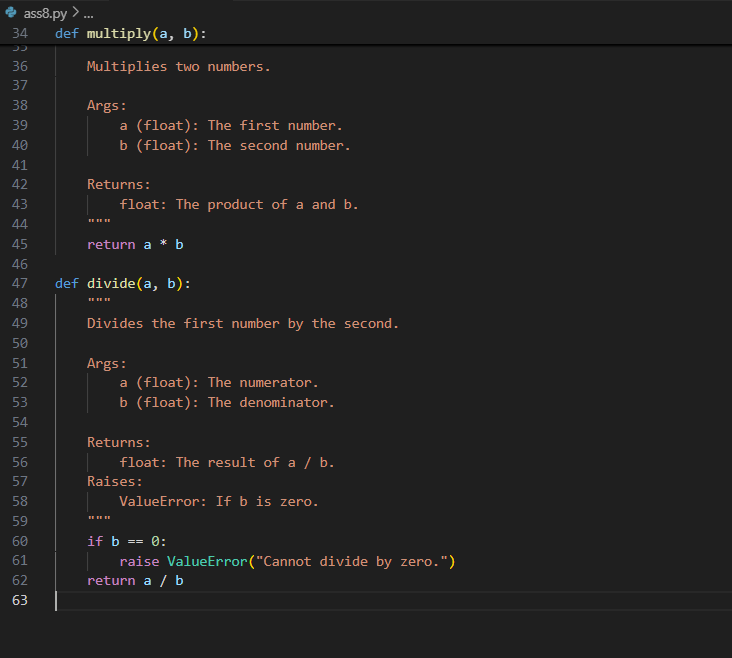
**TASK-3:**

GitHub Copilot:

Prompt:

Generate a Python script with 3–4 functions (e.g., calculator: add, subtract, multiply, divide). Include module-level docstring + individual function docstrings.



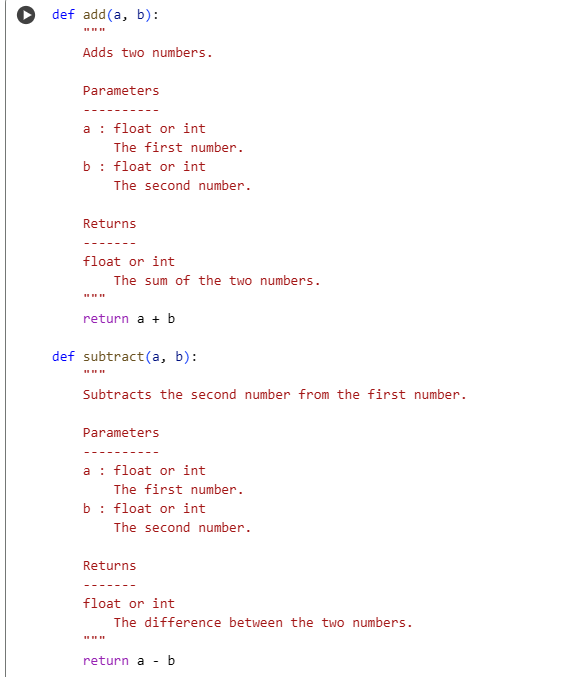


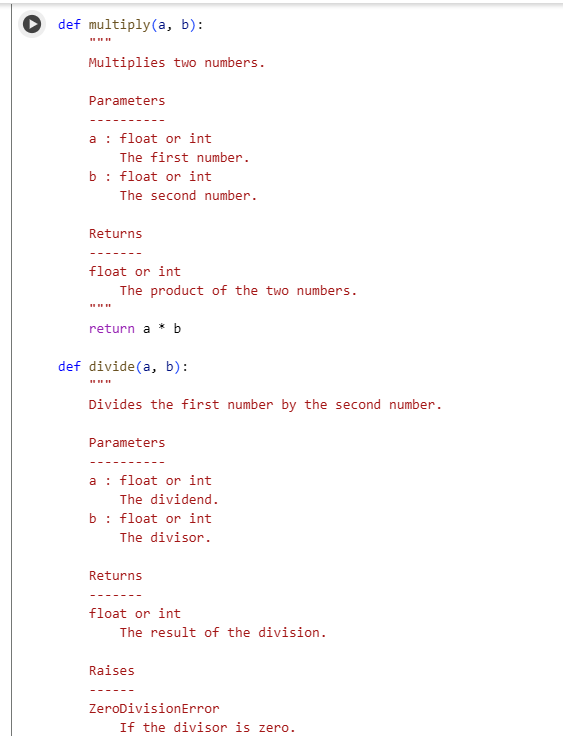
Gemini:

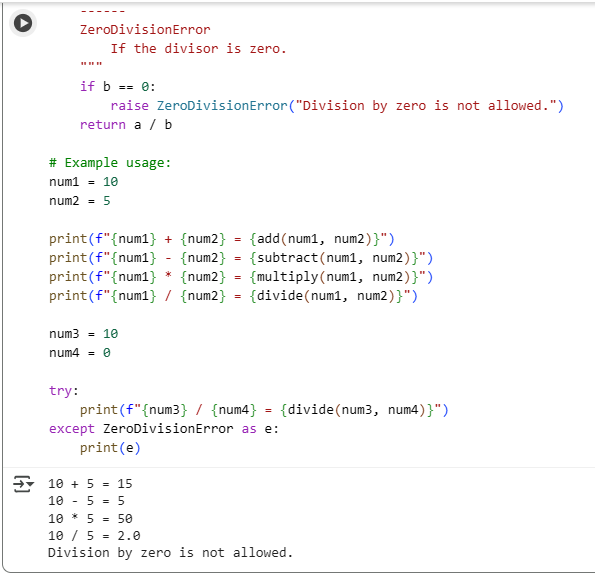
Prompt:

Generate a Python script with 3–4 functions (e.g., calculator: add, subtract, multiply, divide) . Write docstring in code with NumPy Style

Code and Output:

****

****

****

**Comparison: Google-style vs NumPy-style**

**Readability:**

* ***Google-style* is cleaner and easier for beginners to read.**
* ***NumPy-style* is more structured, preferred in scientific computing.**

**Parameters:**

* ***Google-style*: Uses Args: with simple name-type-description format.**
* ***NumPy-style*: Uses a Parameters section with aligned formatting.**

**Return values:**

* ***Google-style*: Returns: with brief description.**
* ***NumPy-style*: Returns block includes type and detailed explanation.**

**Exceptions:**

* **Both styles support Raises: or Raises section, but NumPy-style is more detailed.**

**Examples:**

* ***Google-style*: Examples: section uses inline >>> syntax.**
* ***NumPy-style*: Examples section is more formal, under its own heading.**

**Usage:**

* ***Google-style*: Common in web apps, general Python projects.**
* ***NumPy-style*: Standard in scientific libraries (NumPy, Pandas, etc.).**

**Tooling:**

* **Both are compatible with Sphinx; *NumPy-style* works best with napoleon extension for auto-doc generation.**