|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **]SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE** | | | | | **DEPARTMENT OF COMPUTER SCIENCE ENGINEERING** | | | | |
| **ProgramName:**B. Tech | | | | **Assignment Type: Lab** | | | **AcademicYear:**2025-2026 | | |
| **CourseCoordinatorName** | | | | Venkataramana Veeramsetty | | | | | |
| **Instructor(s)Name** | | | | |  | | --- | | Dr. V. Venkataramana (Co-ordinator) | | Dr. T. Sampath Kumar | | Dr. Pramoda Patro | | Dr. Brij Kishor Tiwari | | Dr.J.Ravichander | | Dr. Mohammand Ali Shaik | | Dr. Anirodh Kumar | | Mr. S.Naresh Kumar | | Dr. RAJESH VELPULA | | Mr. Kundhan Kumar | | Ms. Ch.Rajitha | | Mr. M Prakash | | Mr. B.Raju | | Intern 1 (Dharma teja) | | Intern 2 (Sai Prasad) | | Intern 3 (Sowmya) | | NS\_2 ( Mounika) | | | | | | |
| **rCourseCode** | | | 24CS002PC215 | **CourseTitle** | | AI Assisted Coding | | | |
| **Year/Sem** | | | II/I | **Regulation** | | R24 | | | |
| **Date and Day**  **of Assignment** | | | Week5- Wednesday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicableto**  **Batches** | |  | | | |
| **AssignmentNumber:9.3**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***ExpectedTime***  ***to complete*** |  |
|  | 1 | Lab 8: Documentation Generation: Automatic documentation and code comments  **Lab Objectives:**   * To understand the importance of documentation and code comments in software development. * To explore how AI-assisted coding tools can generate meaningful documentation and inline comments. * To practice generating function-level and module-level docstrings automatically. * To evaluate the quality, accuracy, and limitations of AI-generated documentation. * To develop a small automated tool for documentation generation in Python..     **Lab Outcomes (LOs):**  After completing this lab, students will be able to:   * Apply AI-assisted coding tools to generate docstrings and inline comments for Python code. * Critically analyze AI-generated documentation for correctness, completeness, and readability. * Create structured documentation (function-level, module-level) following standard formats. * Design and implement a mini documentation generator tool to automate code commenting and docstring creation.   **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.   **Expected Outcome#1:** Students understand how AI can produce function-level documentation.  **Task Description#2 Automatic Inline Comments**   * Write 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 * Ask an AI tool to add inline comments explaining each line/step. * Compare the AI-generated comments with your manually written one.   **Expected Output#2:** Students critically analyze AI-generated code comments.  **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.   **Expected Output#3:** Students learn structured documentation for multi-function scripts  **Push documentation whole workspace as .md file in GitHub Repository**  **Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots** | | | | | | Week4 - Wednesday |  |

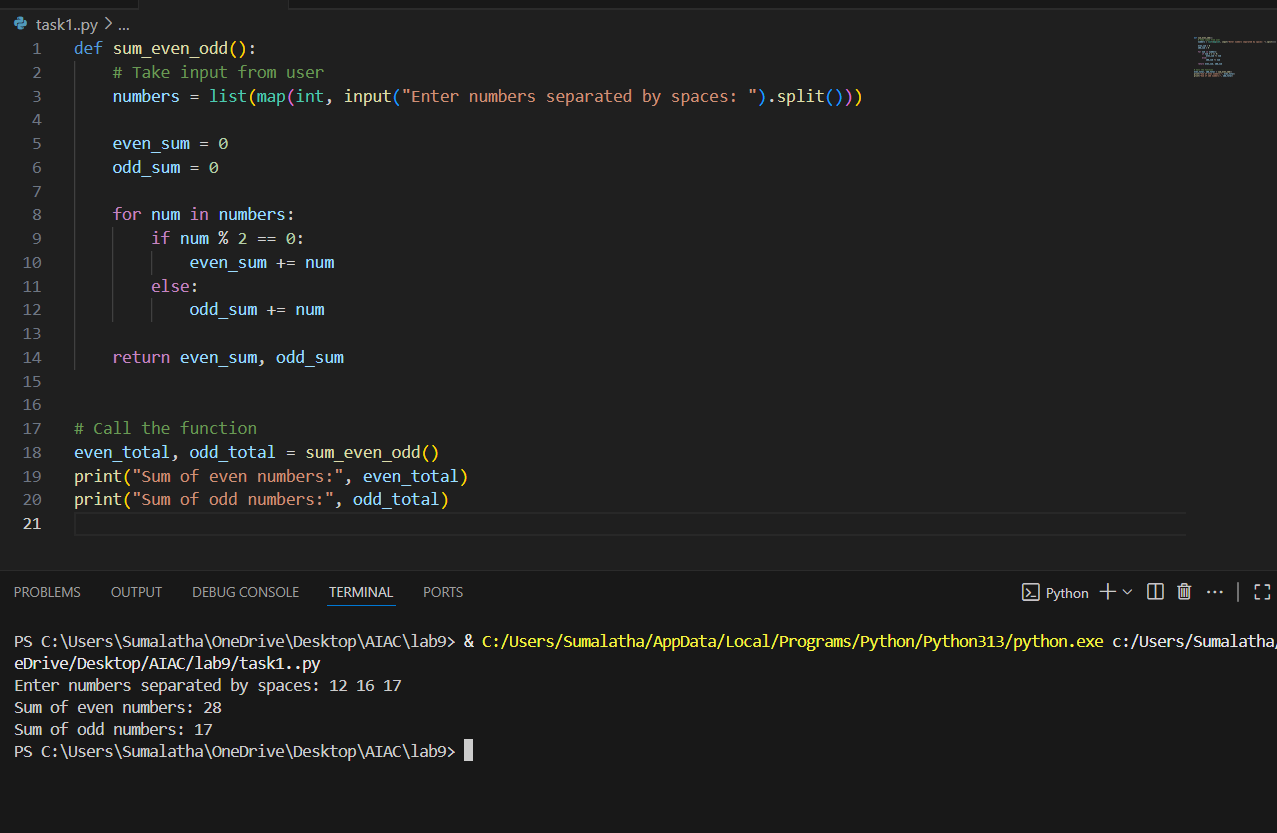
**Task Description#1 Basic Docstring Generation**

* 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.

**Expected Outcome#1:** Students understand how AI can produce function-level documentation

Prompt:

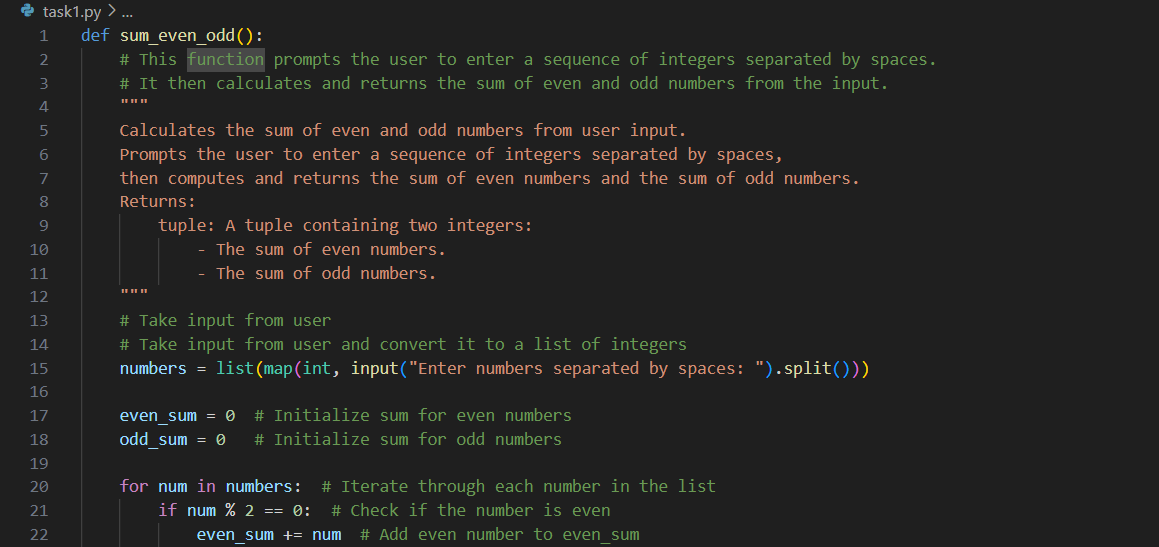
Write python function to return sum of even and odd numbers in the given list that takes input from the user.

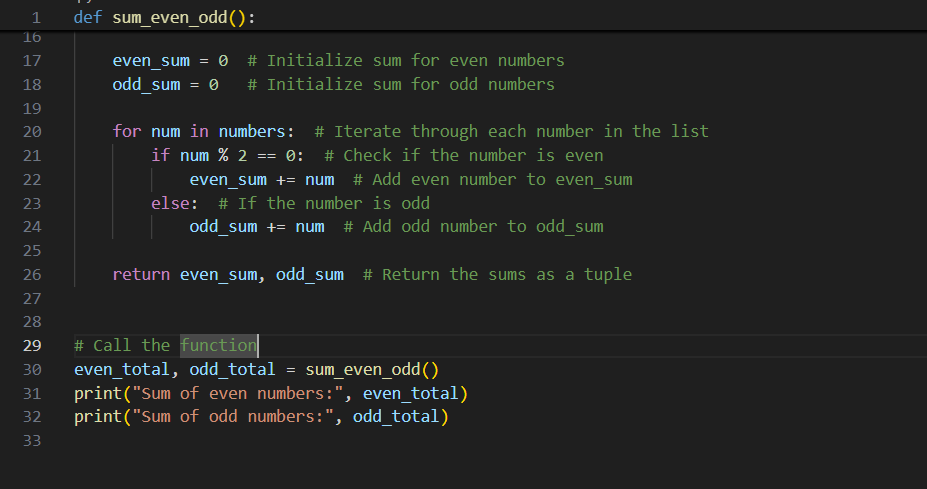


Manual document:

This program takes a list of numbers as input from the user. It calculates and returns the sum of even and odd numbers separately. The result is displayed showing the sum of even numbers and odd numbers.

Documented code:





Output:



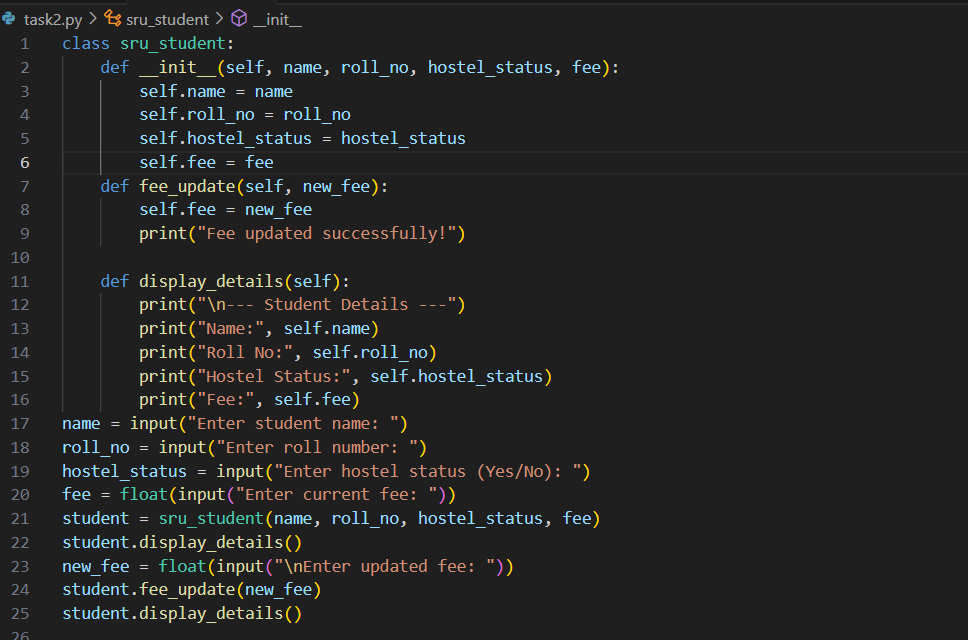
**Task Description#2 Automatic Inline Comments**

* Write 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
* Ask an AI tool to add inline comments explaining each line/step.
* Compare the AI-generated comments with your manually written one.

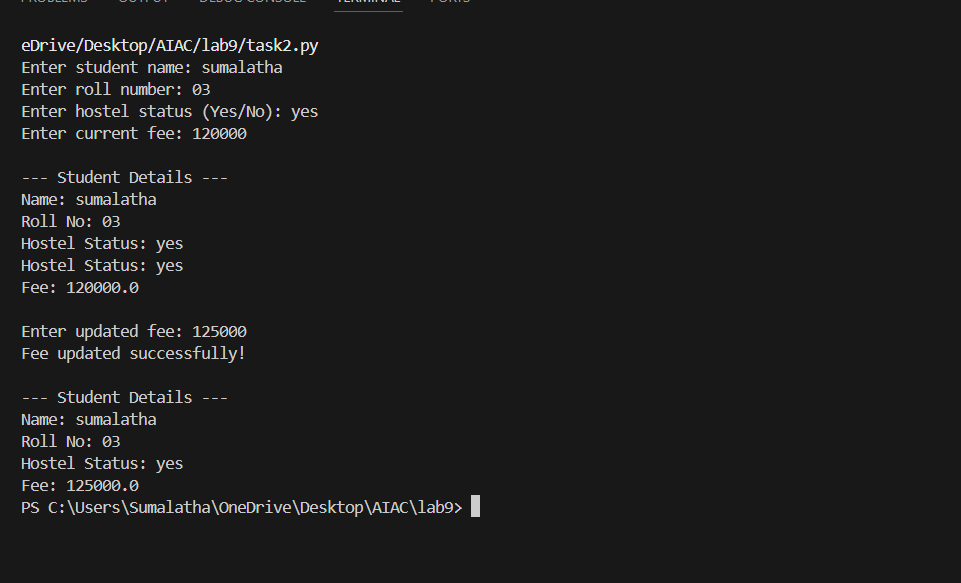
**Expected Output#2:** Students critically analyze AI-generated code comments.

Prompt: Write python program for sru\_student class with attributes like name, roll no., hostel\_status and fee\_update method and display\_details method and it should take input from user.

Code:



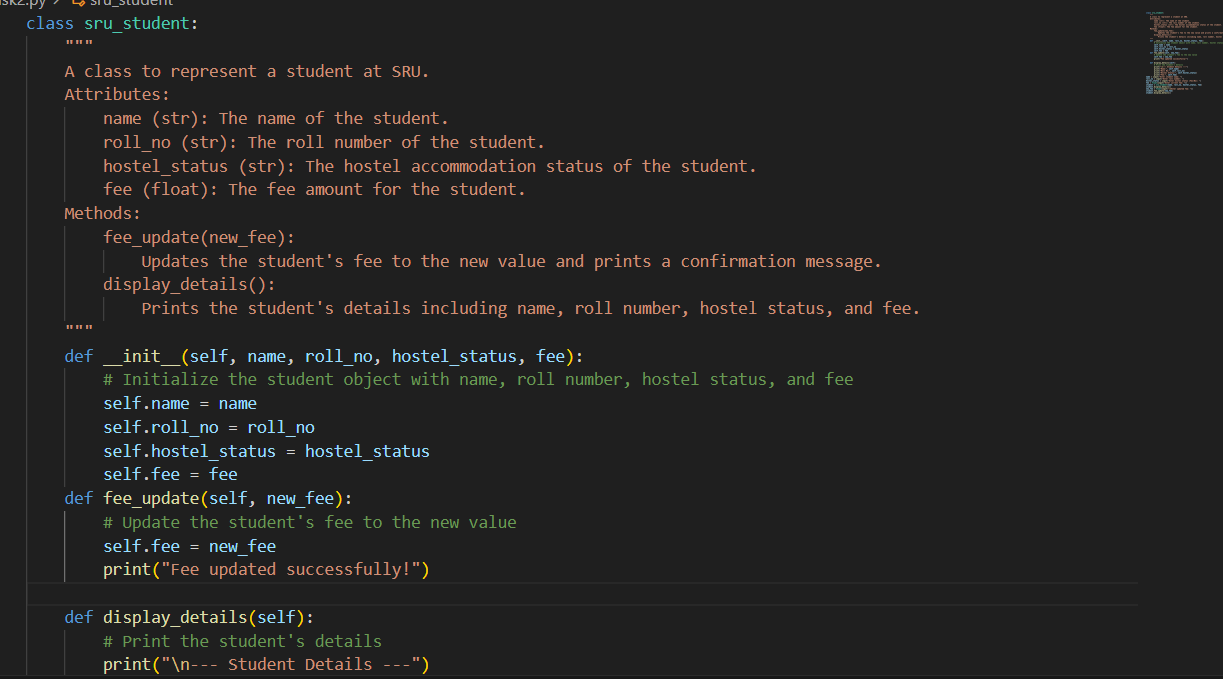
Output:

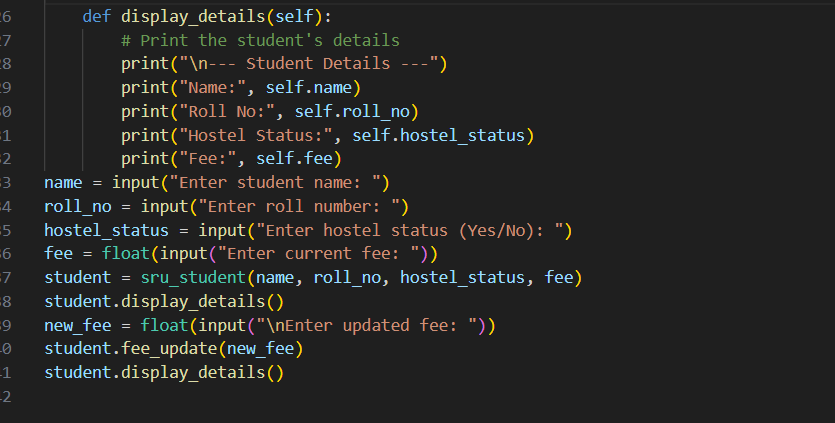


Manual document:

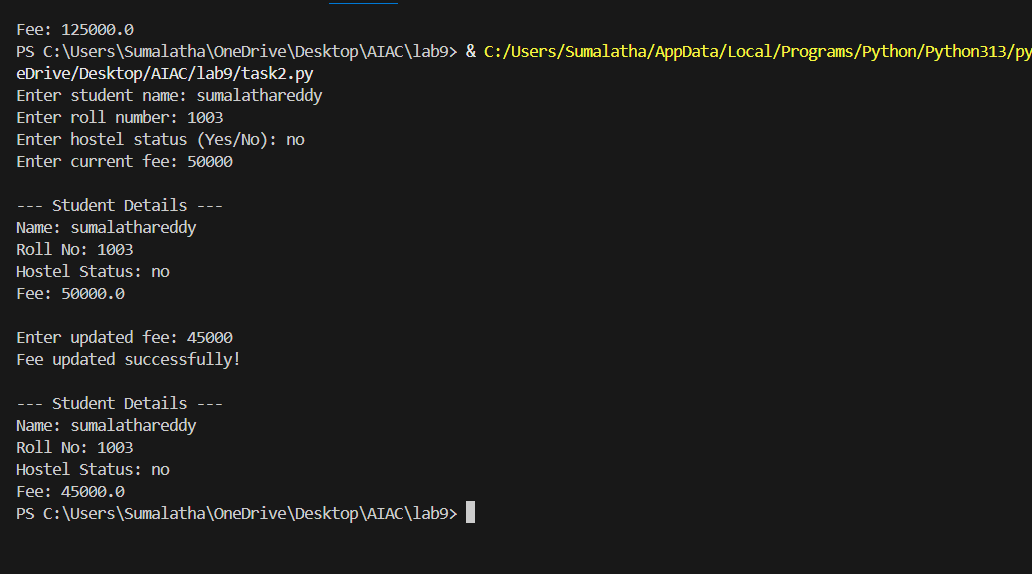
This program creates an sru\_student class to store student details like name, roll number, hostel status, and fee.It takes input from the user, displays the details, and allows updating the fee through the fee\_update method.  
The updated student details can be viewed using the display\_details method.

Documented code:





Output:



**Task Description#3**

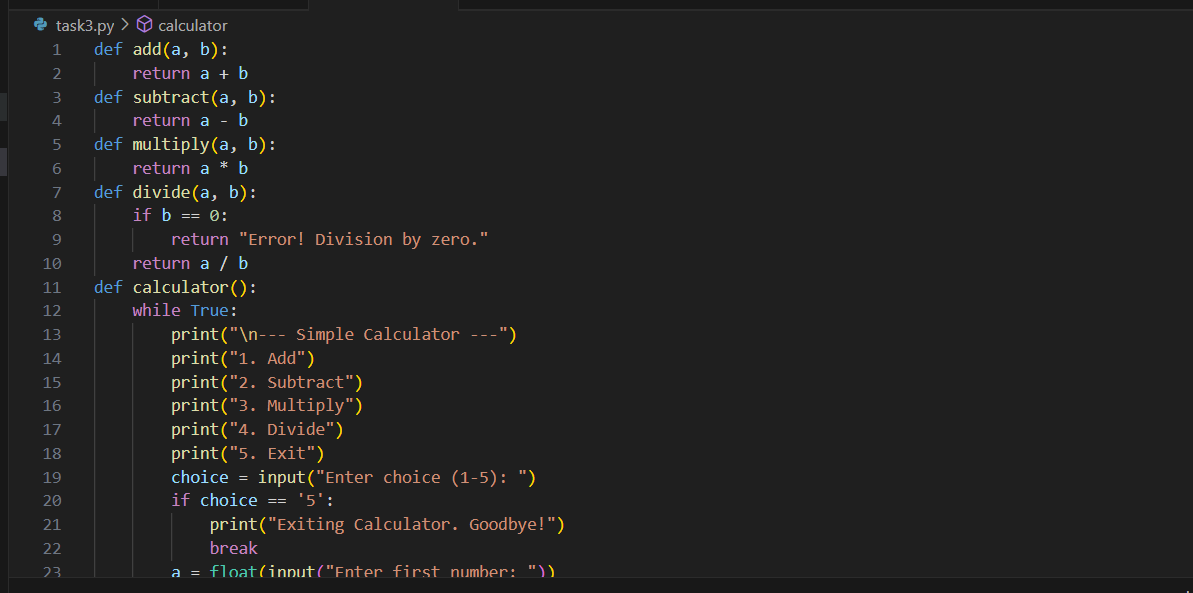
* Write a Python script with 3–4 functions (e.g., calculator: add, subtract, multiply, divide).
* Incorporate manual **docstring** in code with NumPy Style
* Use AI assistance to generate a module-level docstring + individual function docstrings.
* Compare the AI-generated docstring with your manually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

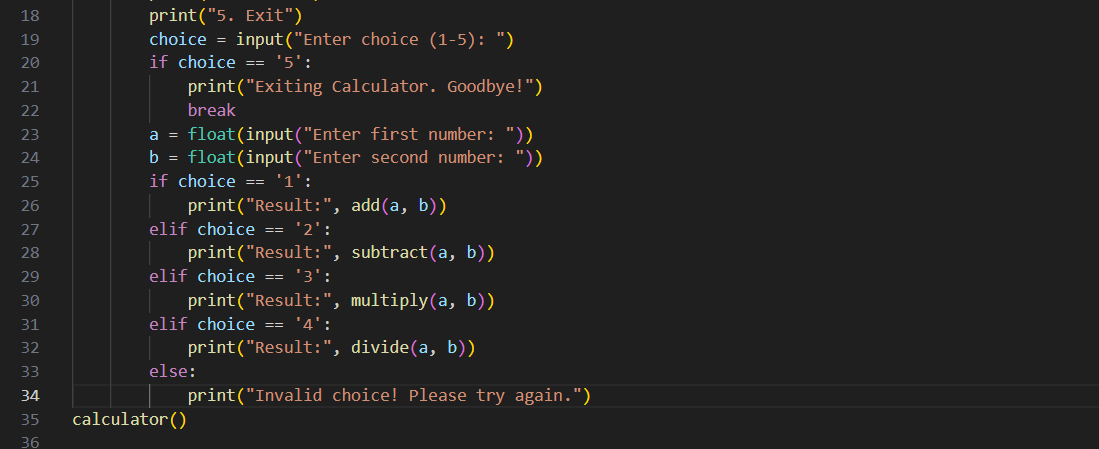
**Expected Output#3:** Students learn structured documentation for multi-function scripts

Prompt:

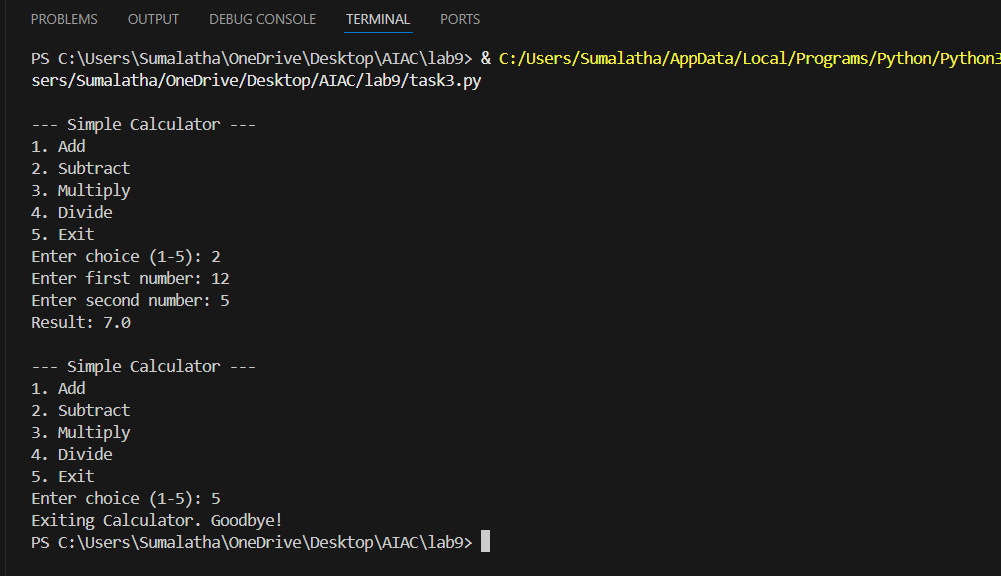
Write a Python script with 3–4 functions (e.g., calculator: add, subtract, multiply, divide) and take in input from the user and also ask user to exit.

Code:





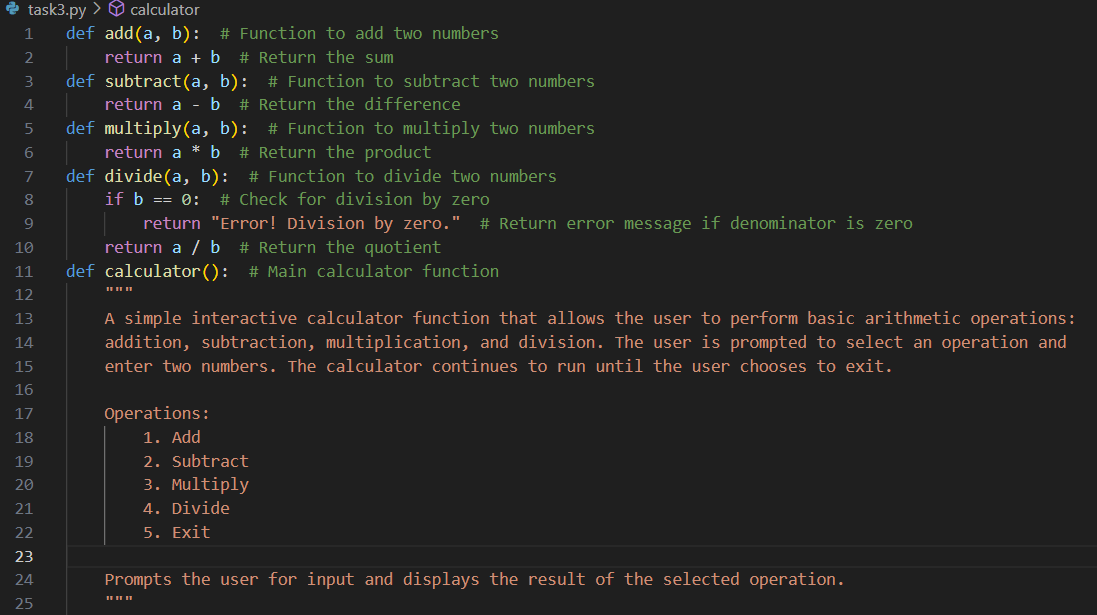
Output:

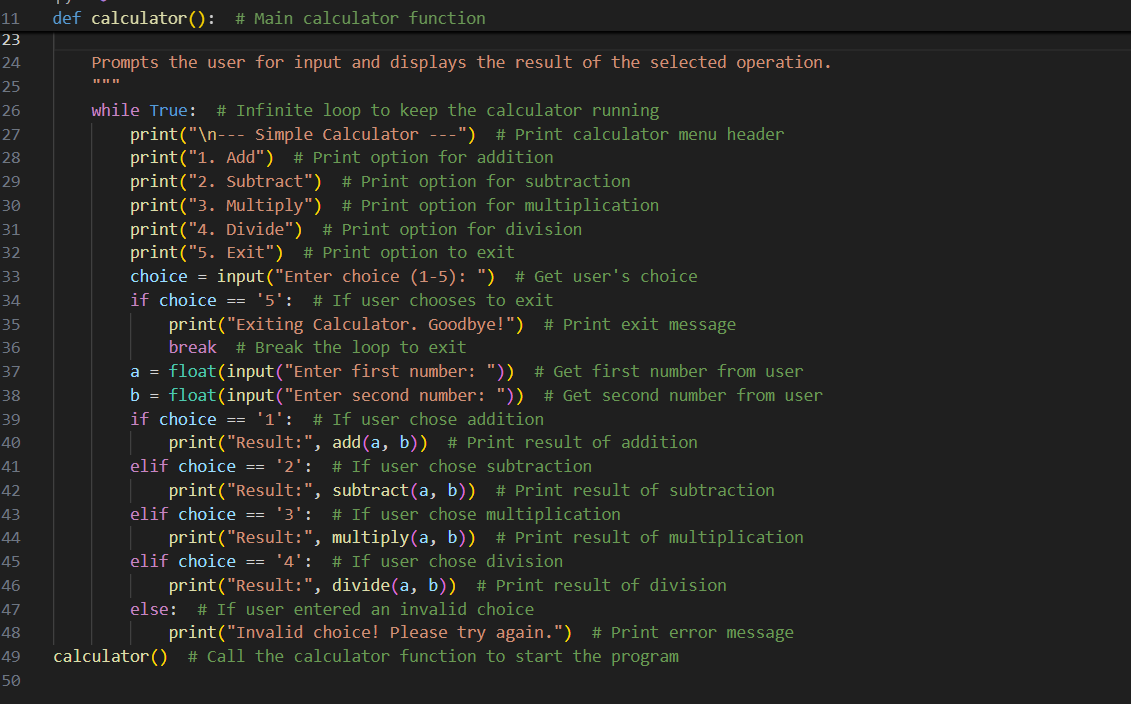


Manual document:

This program is a simple calculator with functions for addition, subtraction, multiplication, and division.  
It takes two numbers and an operation choice from the user, performs the calculation, and displays the result.

Documented code:





Output:

