AI- Ass-9.3

Name:T.Shivani

Ht.No:2303A51312

Batch:05

Prompt: #You are developing a utility function that processes numerical lists and must be properly

#documented for future maintenance.

#Write a Python function to return the sum of even numbers and sum of odd numbers in a

#given list

#• Manually add a Google Style docstring to the function

#• Use an AI-assisted tool (Copilot / Cursor AI) to generate a function-level docstring

#• Compare the AI-generated docstring with the manually written docstring

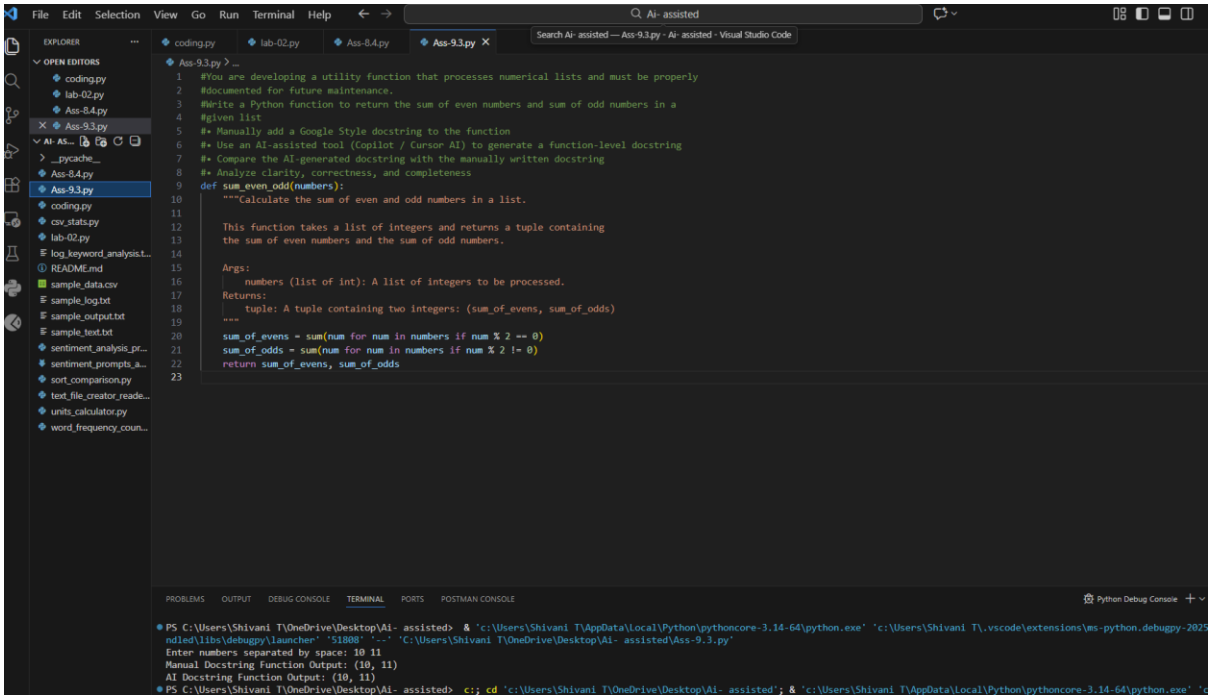
#• Analyze clarity, correctness, and completeness

Code: def sum_even_odd(numbers):

```
    """Calculate the sum of even and odd numbers in a list.

    This function takes a list of integers and returns a tuple containing
    the sum of even numbers and the sum of odd numbers.

    Args:
        numbers (list of int): A list of integers to be processed.
    Returns:
        tuple: A tuple containing two integers: (sum_of_evens, sum_of_odds)
    """
    sum_of_evens = sum(num for num in numbers if num % 2 == 0)
    sum_of_odds = sum(num for num in numbers if num % 2 != 0)
return sum_of_evens, sum_of_odds
```



Prompt: Task 2: Automatic Inline Comments

#You are developing a student management module that must be easy to understand for new

developers.

#Write a Python program for an sru_student class with the following:

#Attributes: name, roll_no, hostel_status

Methods: fee_update() and display_details()

Manually write inline comments for each line or logical block

Use an AI-assisted tool to automatically add inline comments

#Compare manual comments with AI-generated comments

Identify missing, redundant, or incorrect AI comments

Expected Output

• Python class with manually written inline comments

• AI-generated inline comments added to the same code

• Comparative analysis of manual vs AI comments

• Critical discussion on strengths and limitations of AI-generated comments

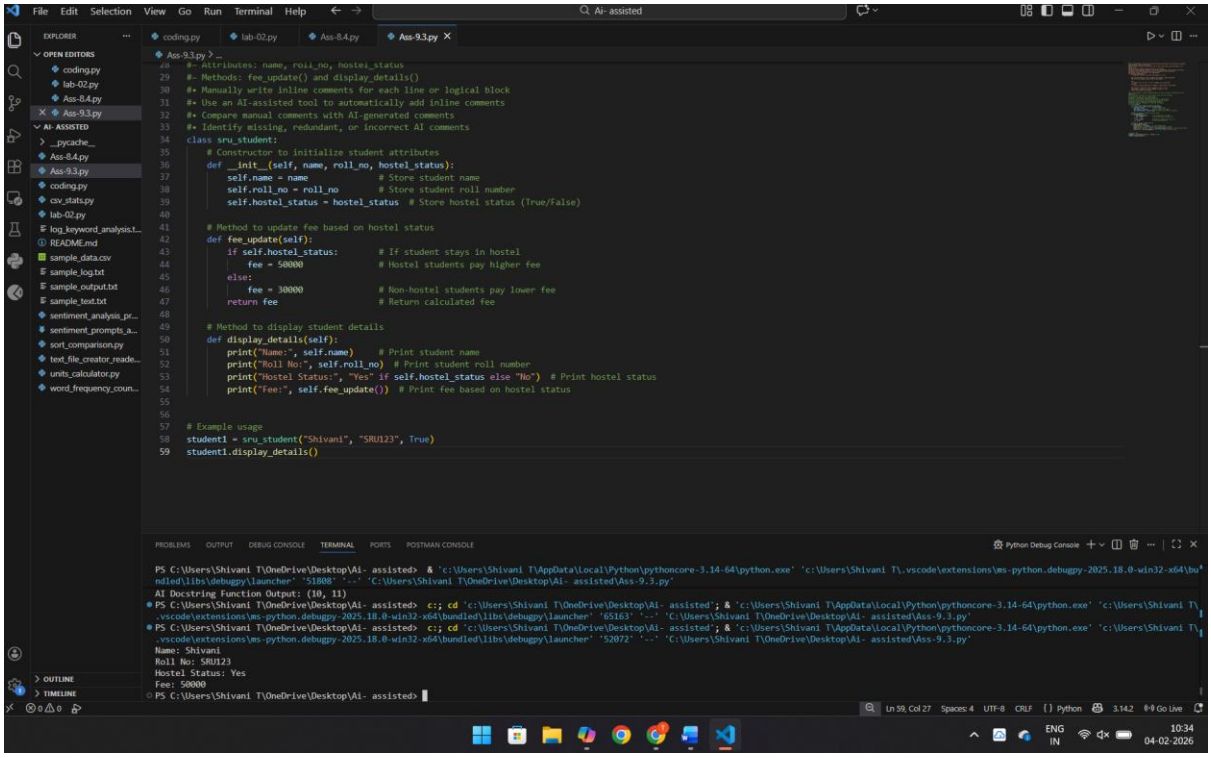
Code: class sru_student:

```
    # Constructor to initialize student attributes
    def __init__(self, name, roll_no, hostel_status):
        self.name = name             # Store student name
        self.roll_no = roll_no       # Store student roll number
        self.hostel_status = hostel_status  # Store hostel status (True/False)

    # Method to update fee based on hostel status
    def fee_update(self):
        if self.hostel_status:       # If student stays in hostel
            fee = 50000              # Hostel students pay higher fee
        else:
            fee = 30000              # Non-hostel students pay lower fee
        return fee                   # Return calculated fee

    # Method to display student details
    def display_details(self):
        print("Name:", self.name)     # Print student name
        print("Roll No:", self.roll_no)  # Print student roll number
        print("Hostel Status:", "Yes" if self.hostel_status else "No")  # Print hostel status
        print("Fee:", self.fee_update())  # Print fee based on hostel status

# Example usage
student1 = sru_student("Shivani", "SRU123", True)
student1.display_details()
```



Prompt: Task 3: Module-Level and Function-Level Documentation

#You are building a small calculator module that will be shared across multiple projects and requires structured documentation.

Write a Python script containing 3–4 functions (e.g., add, subtract, multiply, divide)

#Manually write NumPy Style docstrings for each function

#Use AI assistance to generate:

#A module-level docstring

#Individual function-level docstrings

Compare AI-generated docstrings with manually written ones

Evaluate documentation structure, accuracy, and readability

Code: """

calculator.py

==========

A simple calculator module providing basic arithmetic operations.

```
This module can be reused across multiple projects where basic
mathematical functions are required.
"""

def add(a, b):
    """
    Add two numbers.

    Parameters
    ----------
    a : float or int
        First number.
    b : float or int
        Second number.

    Returns
    -------
    float or int
        The sum of `a` and `b`.

    Examples
    --------
    >>> add(2, 3)
    5
    """
    return a + b
```

```
def subtract(a, b):
    """
    Subtract one number from another.

    Parameters
    ----------
    a : float or int
        First number.
    b : float or int
        Second number to subtract from `a`.

    Returns
    -------
    float or int
        The result of `a - b`.

    Examples
    --------
    >>> subtract(5, 2)
    3
    """
    return a - b


def multiply(a, b):
    """
    Multiply two numbers.
```

```
    Parameters
    ----------
    a : float or int
        First number.
    b : float or int
        Second number.

    Returns
    -------
    float or int
        The product of `a` and `b`.

    Examples
    --------
    >>> multiply(4, 3)
    12
    """
    return a * b


def divide(a, b):
    """
    Divide one number by another.

    Parameters
    ----------
    a : float or int
        Numerator.
```

b : float or int

    Denominator.

Returns

----

float

    The result of `a / b`.

Raises

ZeroDivisionError

    If `b` is zero.

Examples

>>> divide(10, 2)

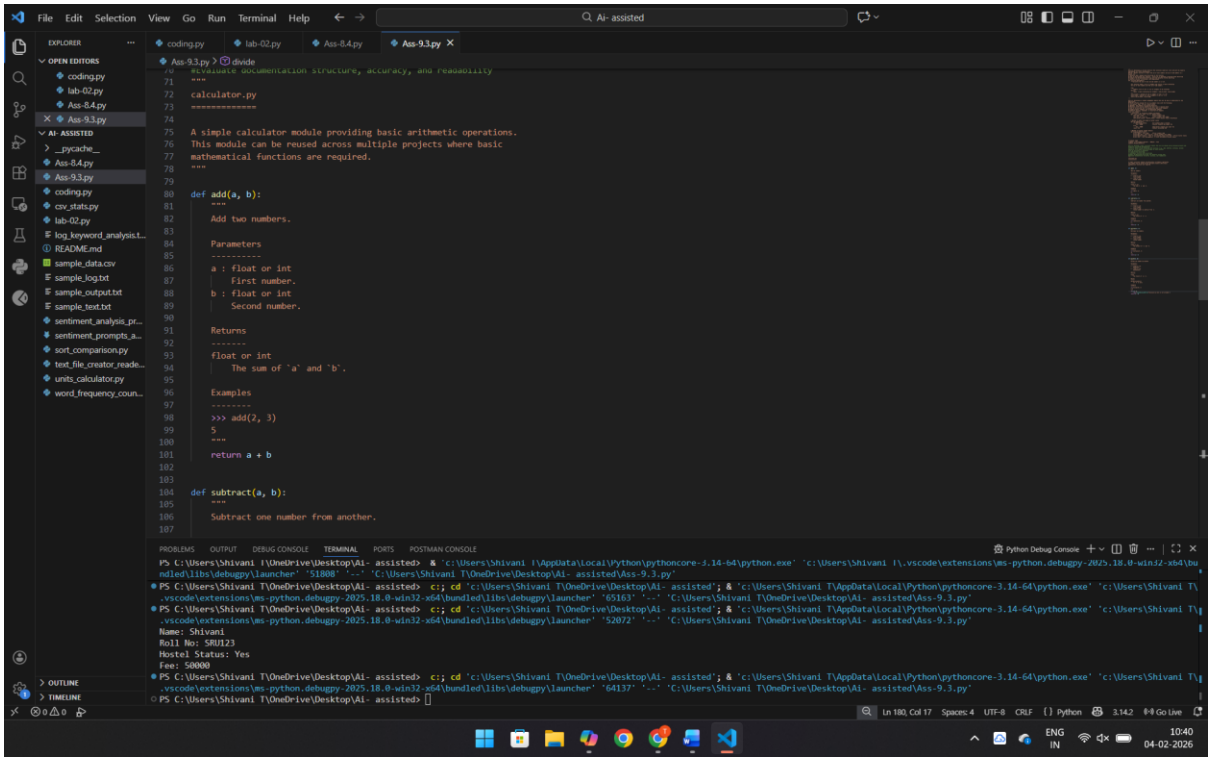
5.0

"""

if b == 0:

    raise ZeroDivisionError("Division by zero is not allowed.")

return a / b


File Edit Selection View Go Run Terminal Help
Ai- assisted
EXPLORER
OPEN EDITORS
coding.py
lab-02.py
Ass-8.4.py
Ass-9.3.py
AI- ASSISTED
__pycache__
Ass-8.4.py
Ass-9.3.py
coding.py
csv_stats.py
lab-02.py
log_keyword_analysis.t...
README.md
sample_data.csv
sample_log.txt
sample_output.txt
sample_text.txt
sentiment_analysis_pr...
sentiment_prompts_a...
sort_comparison.py
text_file_creator_reade...
units_calculator.py
word_frequency_coun...
OUTLINE
TIMELINE
coding.py
lab-02.py
Ass-8.4.py
Ass-9.3.py
Ass-9.3.py > divide
#Evaluate documentation structure, accuracy, and readability
"""
calculator.py
=============
A simple calculator module providing basic arithmetic operations.
This module can be reused across multiple projects where basic
mathematical functions are required.
"""
def add(a, b):
"""
Add two numbers.
Parameters
----
a : float or int
First number.
b : float or int
Second number.
Returns
----
float or int
The sum of `a` and `b`.
Examples

>>> add(2, 3)
5
"""
return a + b
def subtract(a, b):
"""
Subtract one number from another.
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
POSTMAN CONSOLE
Python Debug Console
PS C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted> & 'c:\Users\Shivani T\AppData\Local\Python\pythoncore-3.14-64\python.exe' 'c:\Users\Shivani T\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '51808' '--' 'C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted\Ass-9.3.py'
PS C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted> c:; cd 'c:\Users\Shivani T\OneDrive\Desktop\Ai- assisted'; & 'c:\Users\Shivani T\AppData\Local\Python\pythoncore-3.14-64\python.exe' 'c:\Users\Shivani T\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '65163' '--' 'C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted\Ass-9.3.py'
PS C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted> c:; cd 'c:\Users\Shivani T\OneDrive\Desktop\Ai- assisted'; & 'c:\Users\Shivani T\AppData\Local\Python\pythoncore-3.14-64\python.exe' 'c:\Users\Shivani T\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '52072' '--' 'C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted\Ass-9.3.py'
Name: Shivani
Roll No: SRU123
Hostel Status: Yes
Fee: 50000
PS C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted> c:; cd 'c:\Users\Shivani T\OneDrive\Desktop\Ai- assisted'; & 'c:\Users\Shivani T\AppData\Local\Python\pythoncore-3.14-64\python.exe' 'c:\Users\Shivani T\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '64137' '--' 'C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted\Ass-9.3.py'
PS C:\Users\Shivani T\OneDrive\Desktop\Ai- assisted>
Ln 180, Col 17
Spaces: 4
UTF-8
CRLF
Python
3.14.2
Go Live
ENG
IN
10:40
04-02-2026