AI Assisted Coding

Assignment 9.3

Name: Syed Murtaza

Hall ticket no: 2303A51259

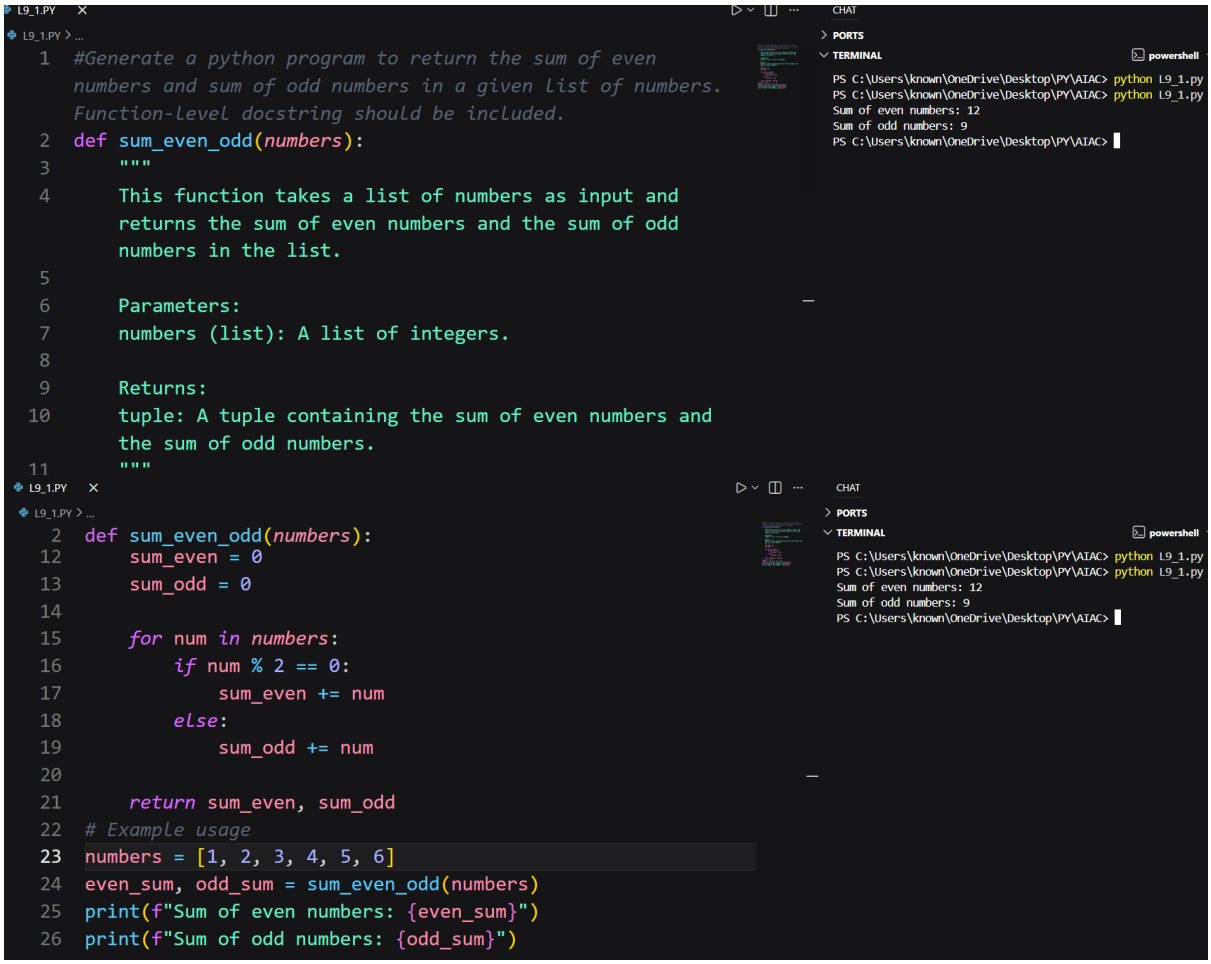
Batch no: 19

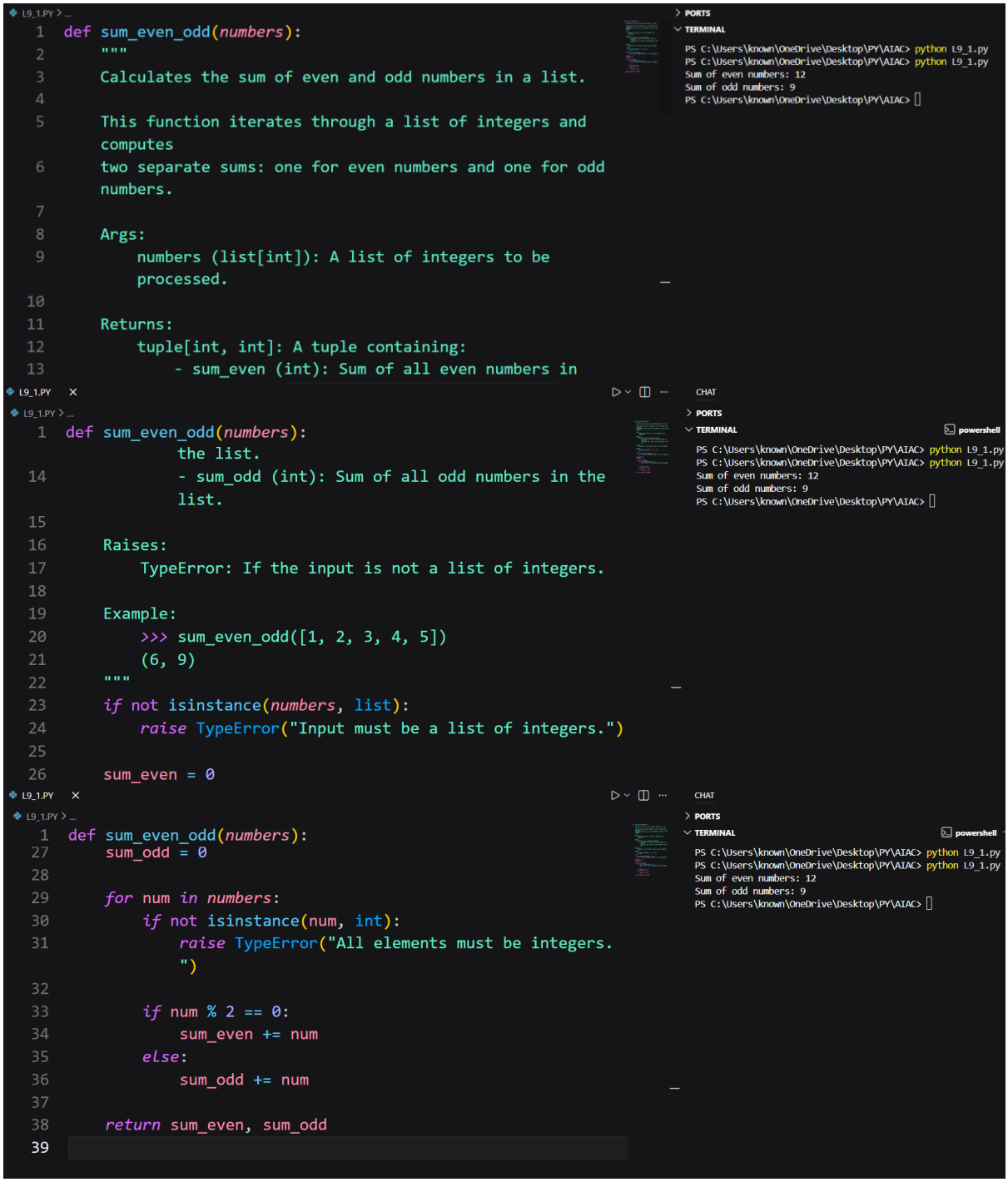
Task 1: Basic Docstring Generation

Prompt:

Write a Python function that returns the sum of even numbers and the sum of odd numbers in a given list.
Add a Google Style docstring manually and then generate a docstring using AI assistance for the same function.

Code & Output:





Explanation:

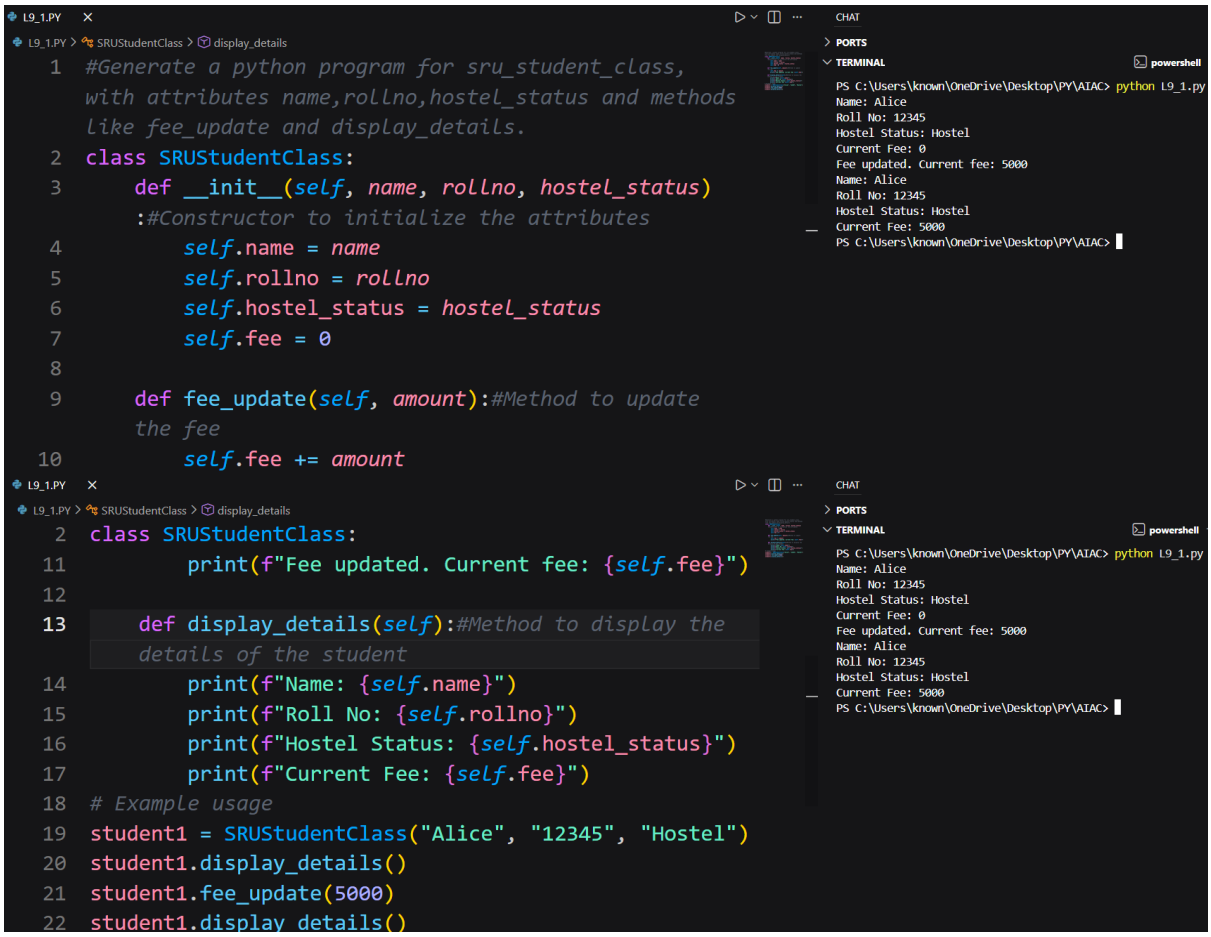
The manual docstring provides a more descriptive explanation of the function's purpose and return structure. It clearly states the argument type and explains the output in sentence form. The AI-generated docstring is shorter and more concise but still correct. However, it lacks deeper descriptive clarity compared to the manual version. This comparison highlights that AI-generated documentation is accurate but sometimes less detailed, requiring human refinement for completeness.

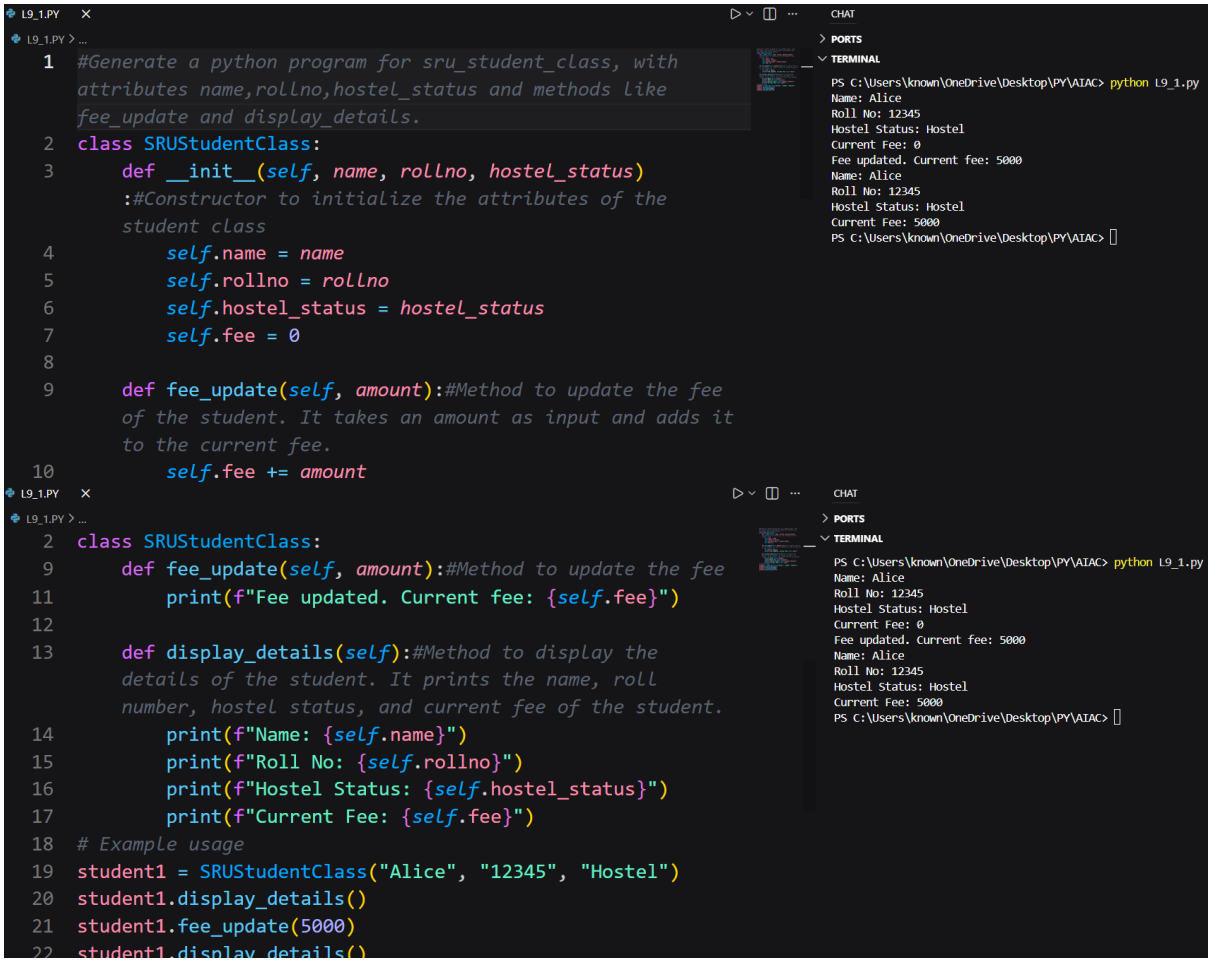
Task 2: Automatic Inline Comments

Prompt:

Generate a Python class named sru_student with attributes name, roll_no, and hostel_status, and methods fee_update() and display_details(). Add inline comments automatically.

Code & Output:





Explanation:

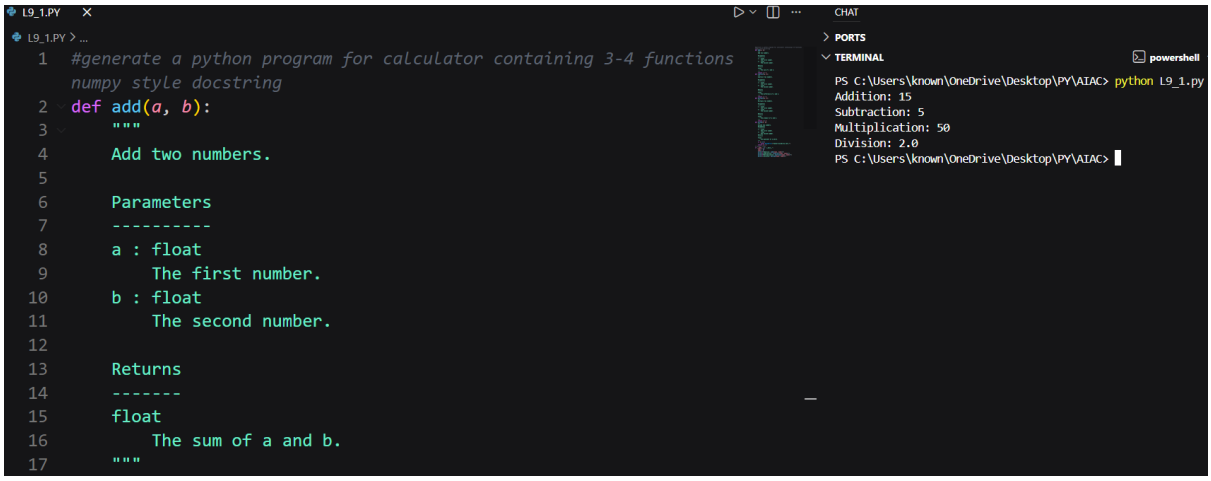
Manual comments are precise and written with developer intent. AI-generated comments are also correct but slightly generic. AI explains obvious lines but may miss contextual explanations. This shows that AI can accelerate documentation but human review is necessary to ensure meaningful and non-redundant comments.

Task 3: Module-Level and Function-Level Documentation

Prompt:

Generate a Python calculator module with functions add, subtract, multiply, and divide. Add NumPy-style docstrings manually and then generate module-level and function-level documentation using AI assistance.

Code & Output:

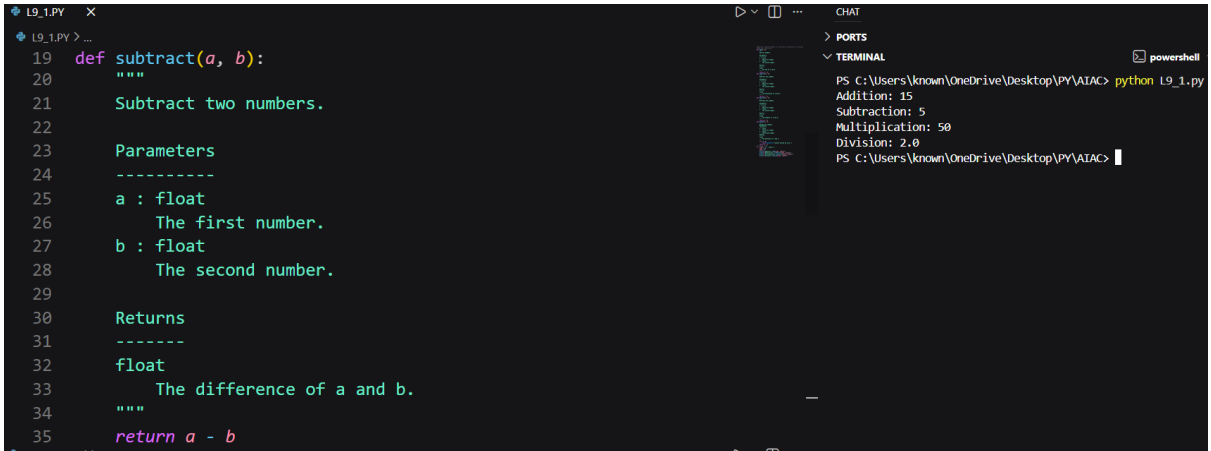


```python
#generate a python program for calculator containing 3-4 functions numpy style docstring
def add(a, b):
    """
    Add two numbers.

    Parameters
    ----------
    a : float
        The first number.
    b : float
        The second number.

    Returns
    -------
    float
        The sum of a and b.
    """
```

```
PS C:\Users\known\OneDrive\Desktop\PY\AIAC> python L9_1.py
Addition: 15
Subtraction: 5
Multiplication: 50
Division: 2.0
PS C:\Users\known\OneDrive\Desktop\PY\AIAC>
```

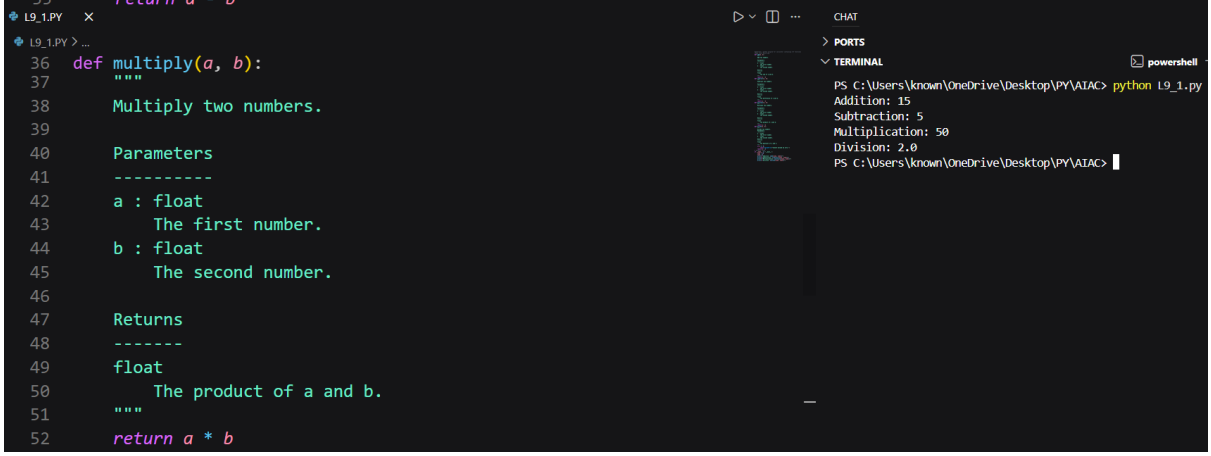


```python
def subtract(a, b):
    """
    Subtract two numbers.

    Parameters
    ----------
    a : float
        The first number.
    b : float
        The second number.

    Returns
    -------
    float
        The difference of a and b.
    """
    return a - b
```

```
PS C:\Users\known\OneDrive\Desktop\PY\AIAC> python L9_1.py
Addition: 15
Subtraction: 5
Multiplication: 50
Division: 2.0
PS C:\Users\known\OneDrive\Desktop\PY\AIAC>
```

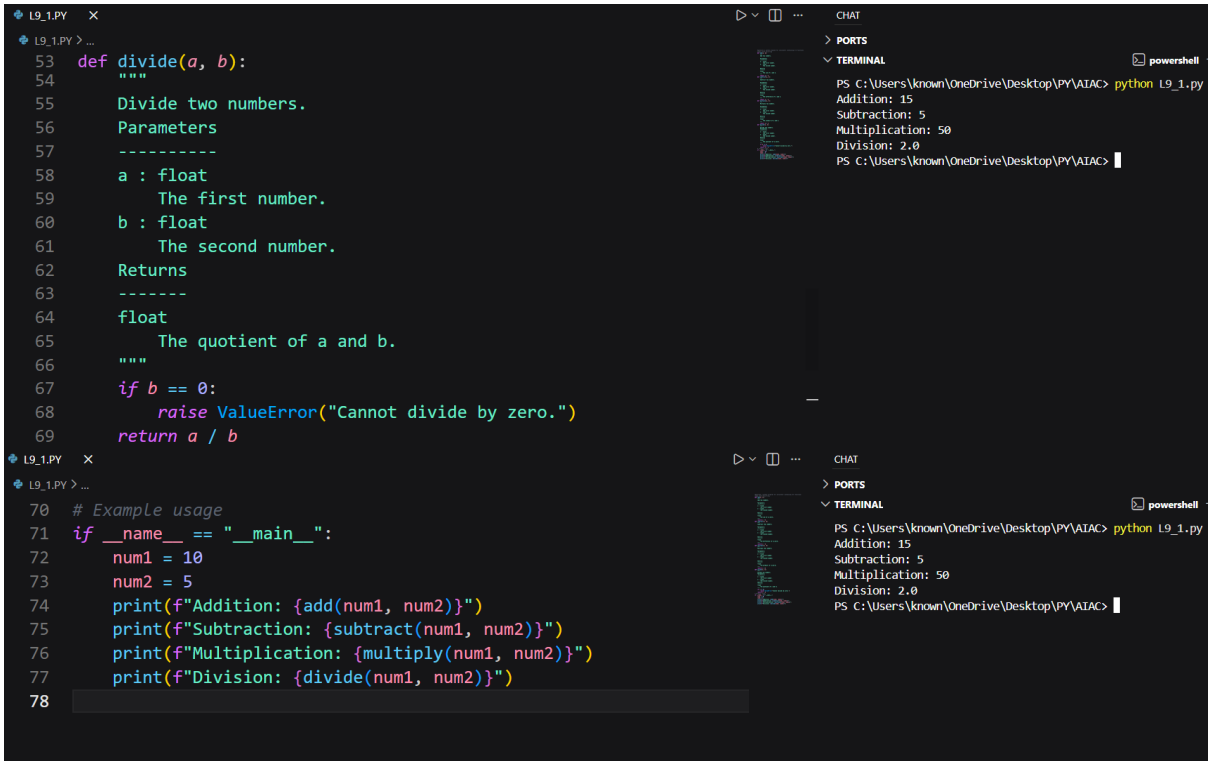


```python
def multiply(a, b):
    """
    Multiply two numbers.

    Parameters
    ----------
    a : float
        The first number.
    b : float
        The second number.

    Returns
    -------
    float
        The product of a and b.
    """
    return a * b
```

```
PS C:\Users\known\OneDrive\Desktop\PY\AIAC> python L9_1.py
Addition: 15
Subtraction: 5
Multiplication: 50
Division: 2.0
PS C:\Users\known\OneDrive\Desktop\PY\AIAC>
```



Explanation:

Manual NumPy-style docstrings follow a structured scientific documentation format with sections for parameters and returns. AI-generated documentation is concise and suitable for module overviews but lacks deep parameter-level detailing. AI performs well in summarization, while manual documentation provides stronger technical clarity.

Final Conclusion:

This lab demonstrated the role of AI in generating documentation and comments automatically. AI-assisted tools significantly reduce documentation effort and improve consistency. However, human review remains essential to ensure clarity, correctness, and contextual relevance in software documentation.