AI ASSISTED CODING

LAB ASSIGNMENT -9.3

Program : B-TECH

Specialization : AIML

Name of Student : AKULA MANOJ

Enrollment-No : 2403A52031

Batch No : 02

Date : 24-09-2025

TASK DESCRIPTION-1

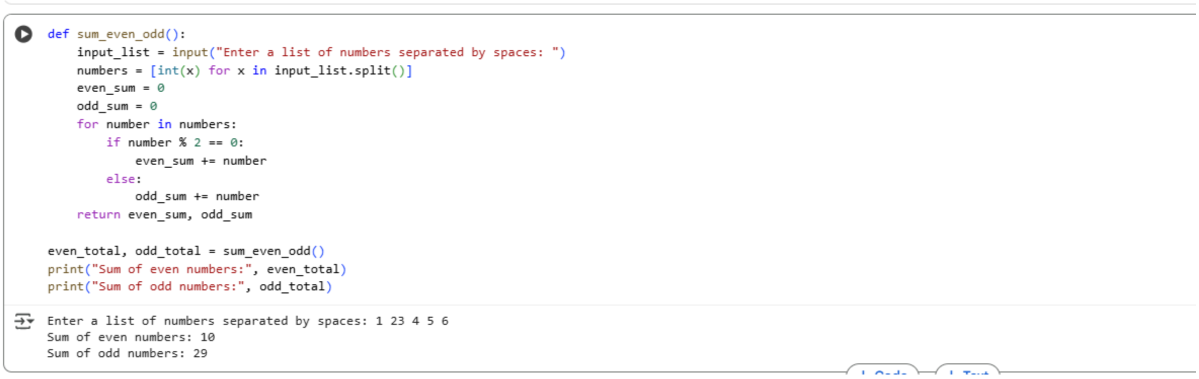
Basic Docstring Generation

- Write a Python function to return the 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

PROMPT:

Generate a Python function to return the sum of even and odd numbers in the given list. The list must be user input, provide code without comments and docstrings

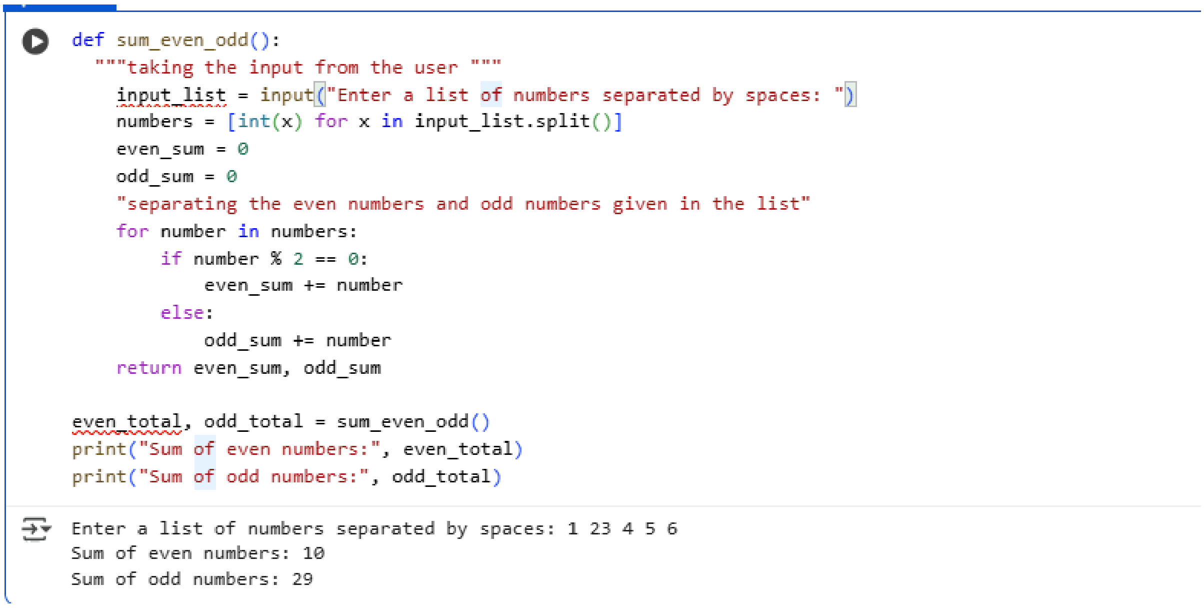
CODE:



```
def sum_even_odd():
    input_list = input("Enter a list of numbers separated by spaces: ")
    numbers = [int(x) for x in input_list.split()]
    even_sum = 0
    odd_sum = 0
    for number in numbers:
        if number % 2 == 0:
            even_sum += number
        else:
            odd_sum += number
    return even_sum, odd_sum

even_total, odd_total = sum_even_odd()
print("Sum of even numbers:", even_total)
print("Sum of odd numbers:", odd_total)
```

```
Enter a list of numbers separated by spaces: 1 23 4 5 6
Sum of even numbers: 10
Sum of odd numbers: 29
```

CODE WITH MANUAL COMMENTS:



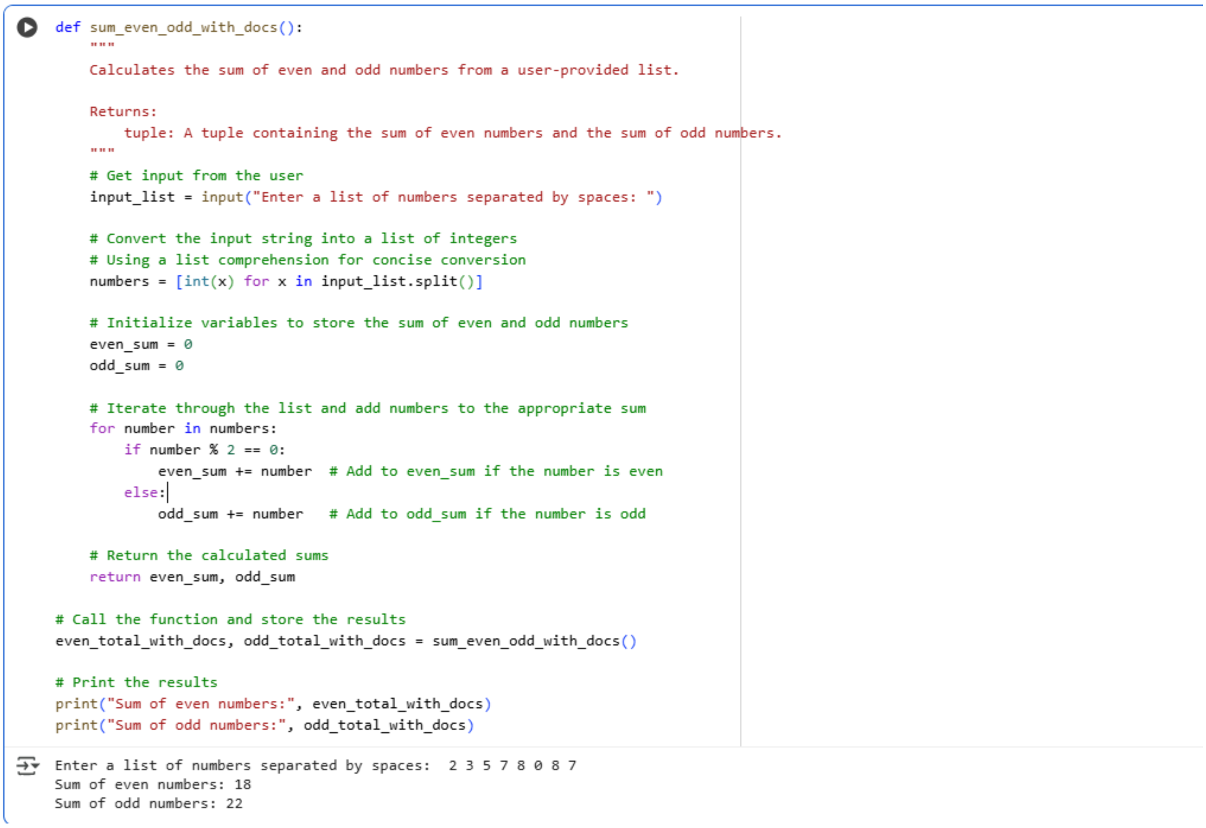
```
def sum_even_odd():
  """taking the input from the user """
    input_list = input("Enter a list of numbers separated by spaces: ")
    numbers = [int(x) for x in input_list.split()]
    even_sum = 0
    odd_sum = 0
    "separating the even numbers and odd numbers given in the list"
    for number in numbers:
        if number % 2 == 0:
            even_sum += number
        else:
            odd_sum += number
    return even_sum, odd_sum

even_total, odd_total = sum_even_odd()
print("Sum of even numbers:", even_total)
print("Sum of odd numbers:", odd_total)
```

```
Enter a list of numbers separated by spaces: 1 23 4 5 6
Sum of even numbers: 10
Sum of odd numbers: 29
```

PROMPT:

Generate a Python function to return the sum of even and odd numbers in the given list. The list must be user input, provide code with comments and docstrings

GENERATED CODE WITH AI COMMENTS:



```
def sum_even_odd_with_docs():
    """
    Calculates the sum of even and odd numbers from a user-provided list.

    Returns:
        tuple: A tuple containing the sum of even numbers and the sum of odd numbers.
    """
    # Get input from the user
    input_list = input("Enter a list of numbers separated by spaces: ")

    # Convert the input string into a list of integers
    # Using a list comprehension for concise conversion
    numbers = [int(x) for x in input_list.split()]

    # Initialize variables to store the sum of even and odd numbers
    even_sum = 0
    odd_sum = 0

    # Iterate through the list and add numbers to the appropriate sum
    for number in numbers:
        if number % 2 == 0:
            even_sum += number  # Add to even_sum if the number is even
        else:
            odd_sum += number   # Add to odd_sum if the number is odd

    # Return the calculated sums
    return even_sum, odd_sum

# Call the function and store the results
even_total_with_docs, odd_total_with_docs = sum_even_odd_with_docs()

# Print the results
print("Sum of even numbers:", even_total_with_docs)
print("Sum of odd numbers:", odd_total_with_docs)
```

```
Enter a list of numbers separated by spaces:  2 3 5 7 8 0 8 7
Sum of even numbers: 18
Sum of odd numbers: 22
```

DIFFERENCES:

The comments I wrote are simple and understandable, whereas the comments provided by the AI are more advanced. However, I liked the comments I gave.

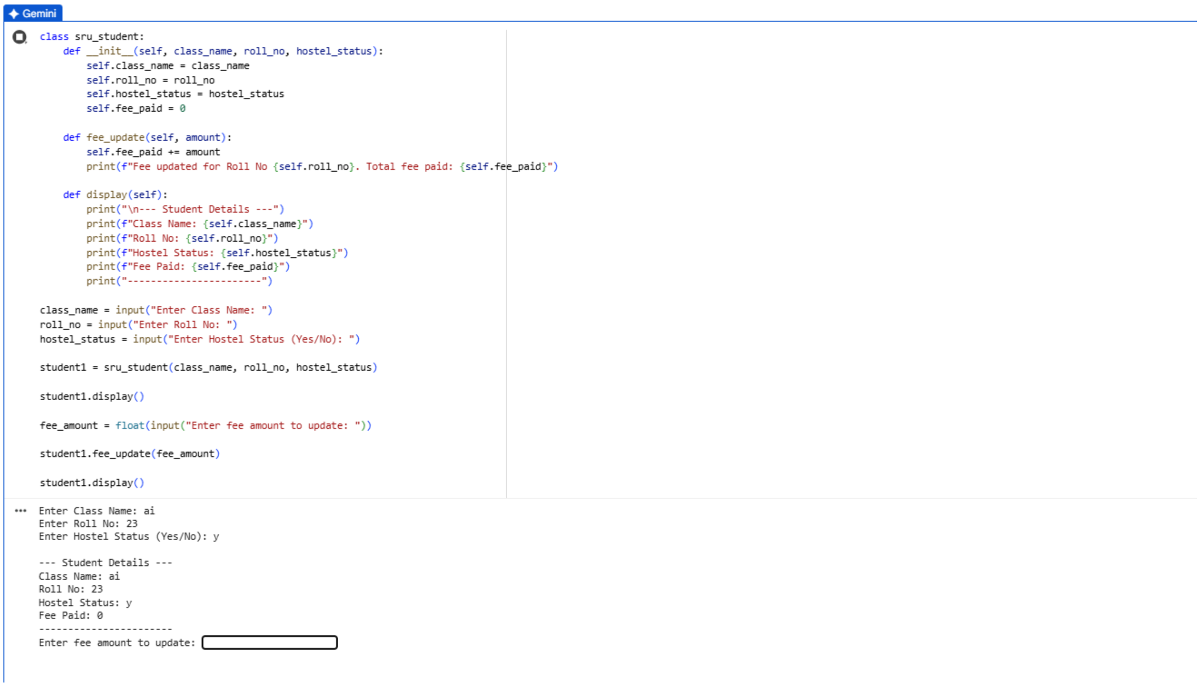
TASK DESCRIPTION-2:-

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

PROMPT: Write a Python program for the sru_student class with attributes like class name, roll no,hostel_status and fee_update method and display method.all the values and all the attributes must be entered by the user manually.without any comments**.**

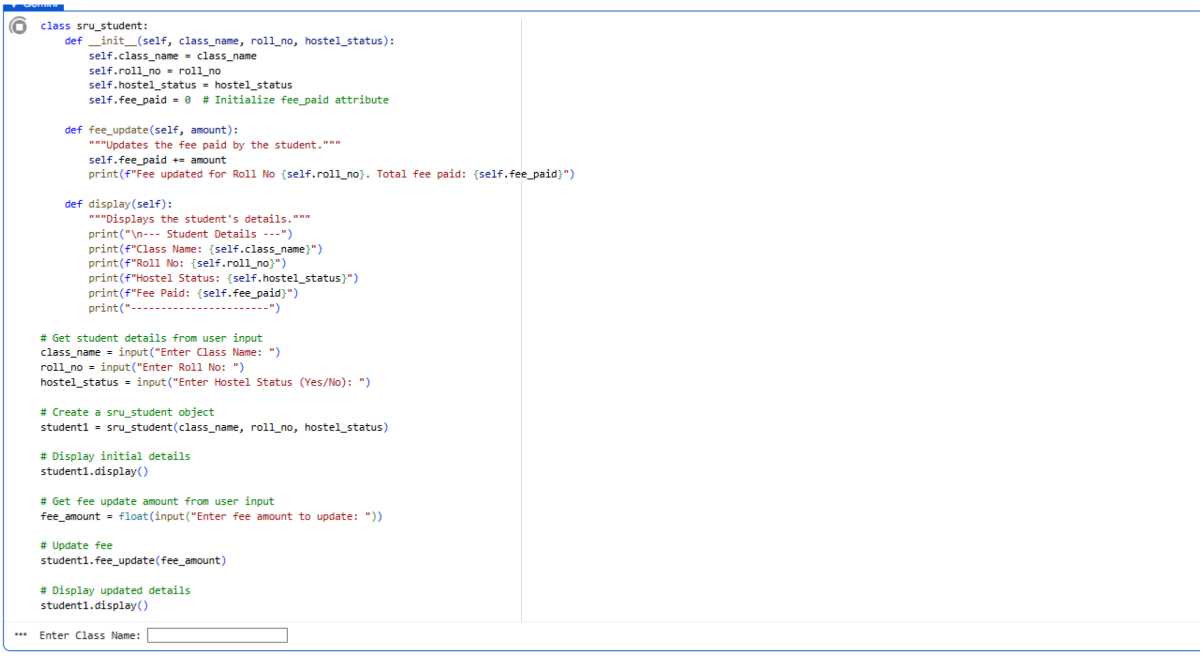
CODE WITHOUT ANY COMMENTS:



```python
class sru_student:
    def __init__(self, class_name, roll_no, hostel_status):
        self.class_name = class_name
        self.roll_no = roll_no
        self.hostel_status = hostel_status
        self.fee_paid = 0

    def fee_update(self, amount):
        self.fee_paid += amount
        print(f"Fee updated for Roll No {self.roll_no}. Total fee paid: {self.fee_paid}")

    def display(self):
        print("\n--- Student Details ---")
        print(f"Class Name: {self.class_name}")
        print(f"Roll No: {self.roll_no}")
        print(f"Hostel Status: {self.hostel_status}")
        print(f"Fee Paid: {self.fee_paid}")
        print("----------------------")

class_name = input("Enter Class Name: ")
roll_no = input("Enter Roll No: ")
hostel_status = input("Enter Hostel Status (Yes/No): ")

student1 = sru_student(class_name, roll_no, hostel_status)

student1.display()

fee_amount = float(input("Enter fee amount to update: "))

student1.fee_update(fee_amount)

student1.display()
```

```
Enter Class Name: ai
Enter Roll No: 23
Enter Hostel Status (Yes/No): y

--- Student Details ---
Class Name: ai
Roll No: 23
Hostel Status: y
Fee Paid: 0
----------------------
Enter fee amount to update:
```

CODE WITH MANUAL COMMENTS:



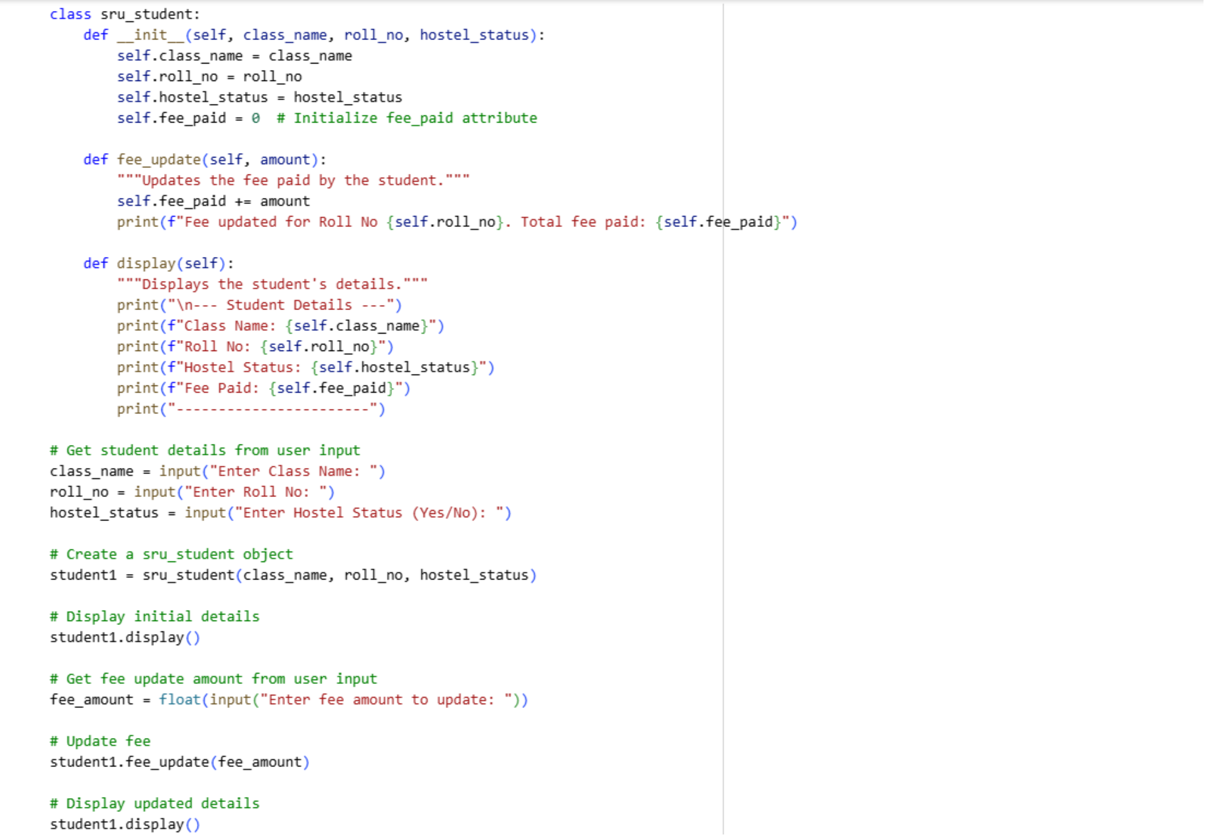
```python
class sru_student:
    def __init__(self, class_name, roll_no, hostel_status):
        self.class_name = class_name
        self.roll_no = roll_no
        self.hostel_status = hostel_status
        self.fee_paid = 0  # Initialize fee_paid attribute

    def fee_update(self, amount):
        """Updates the fee paid by the student."""
        self.fee_paid += amount
        print(f"Fee updated for Roll No {self.roll_no}. Total fee paid: {self.fee_paid}")

    def display(self):
        """Displays the student's details."""
        print("\n--- Student Details ---")
        print(f"Class Name: {self.class_name}")
        print(f"Roll No: {self.roll_no}")
        print(f"Hostel Status: {self.hostel_status}")
        print(f"Fee Paid: {self.fee_paid}")
        print("----------------------")

# Get student details from user input
class_name = input("Enter Class Name: ")
roll_no = input("Enter Roll No: ")
hostel_status = input("Enter Hostel Status (Yes/No): ")

# Create a sru_student object
student1 = sru_student(class_name, roll_no, hostel_status)

# Display initial details
student1.display()

# Get fee update amount from user input
fee_amount = float(input("Enter fee amount to update: "))

# Update fee
student1.fee_update(fee_amount)

# Display updated details
student1.display()
```

```
Enter Class Name:
```

PROMPT:

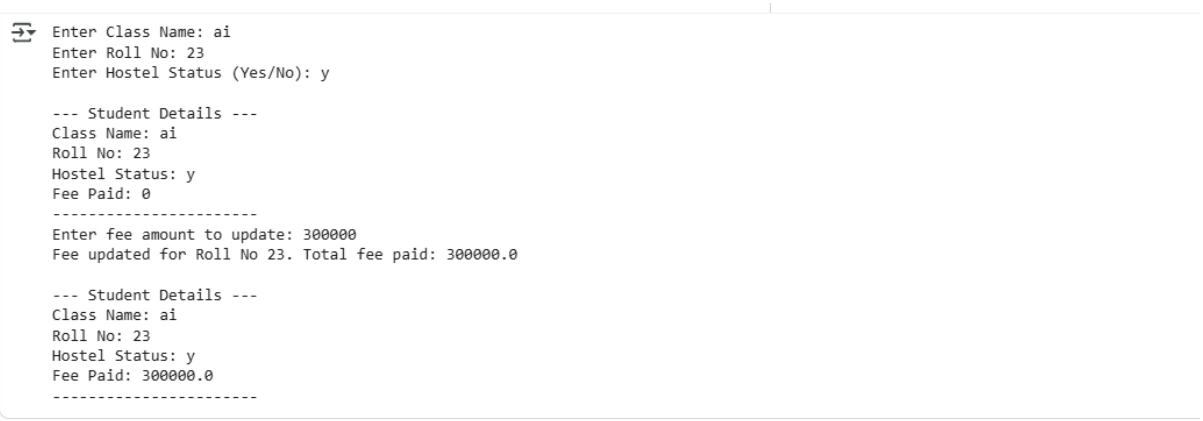
Write a Python program for the sru_student class with attributes like class name, roll no,hostel_status and fee_update method and display method.all the values and all the attributes must be entered by the user manually.

CODE GENERATED BY AI WITH COMMENTS:



```
class sru_student:
    def __init__(self, class_name, roll_no, hostel_status):
        self.class_name = class_name
        self.roll_no = roll_no
        self.hostel_status = hostel_status
        self.fee_paid = 0  # Initialize fee_paid attribute

    def fee_update(self, amount):
        """Updates the fee paid by the student."""
        self.fee_paid += amount
        print(f"Fee updated for Roll No {self.roll_no}. Total fee paid: {self.fee_paid}")

    def display(self):
        """Displays the student's details."""
        print("\n--- Student Details ---")
        print(f"Class Name: {self.class_name}")
        print(f"Roll No: {self.roll_no}")
        print(f"Hostel Status: {self.hostel_status}")
        print(f"Fee Paid: {self.fee_paid}")
        print("----------------------")

# Get student details from user input
class_name = input("Enter Class Name: ")
roll_no = input("Enter Roll No: ")
hostel_status = input("Enter Hostel Status (Yes/No): ")

# Create a sru_student object
student1 = sru_student(class_name, roll_no, hostel_status)

# Display initial details
student1.display()

# Get fee update amount from user input
fee_amount = float(input("Enter fee amount to update: "))

# Update fee
student1.fee_update(fee_amount)

# Display updated details
student1.display()
```

OUTPUT:



```
Enter Class Name: ai
Enter Roll No: 23
Enter Hostel Status (Yes/No): y

--- Student Details ---
Class Name: ai
Roll No: 23
Hostel Status: y
Fee Paid: 0
----------------------
Enter fee amount to update: 300000
Fee updated for Roll No 23. Total fee paid: 300000.0

--- Student Details ---
Class Name: ai
Roll No: 23
Hostel Status: y
Fee Paid: 300000.0
----------------------
```

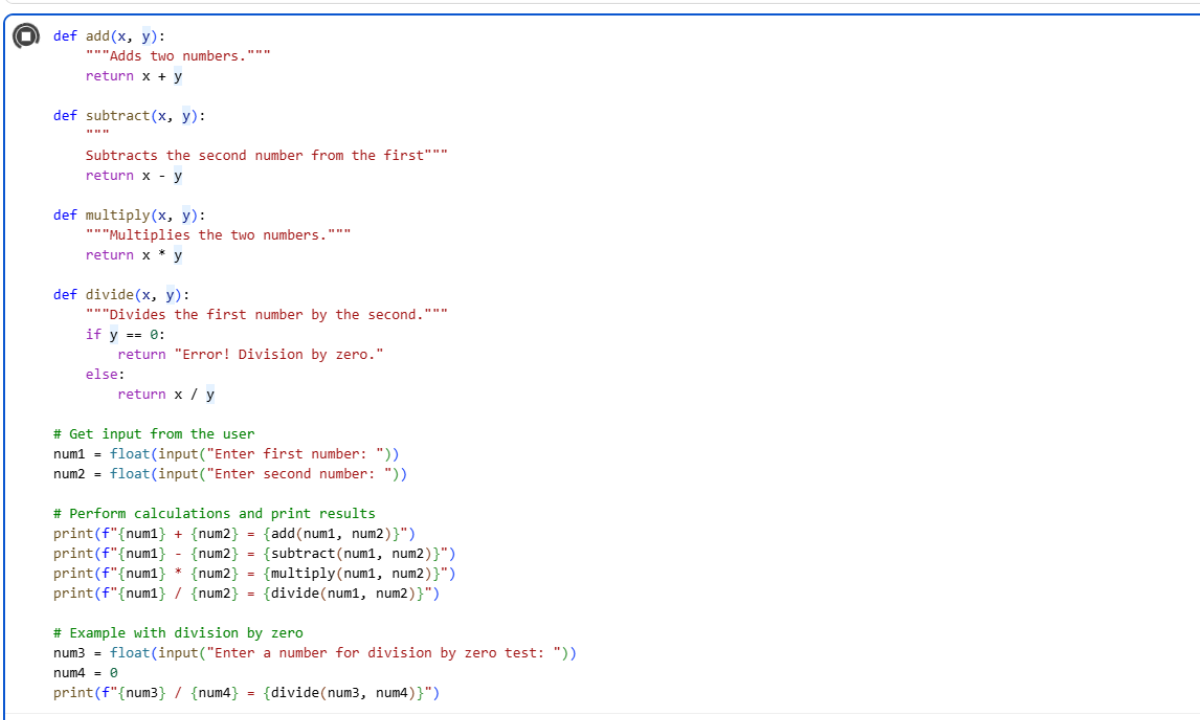
TASK DESCRIPTION-3:-

- 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

PROMT:-

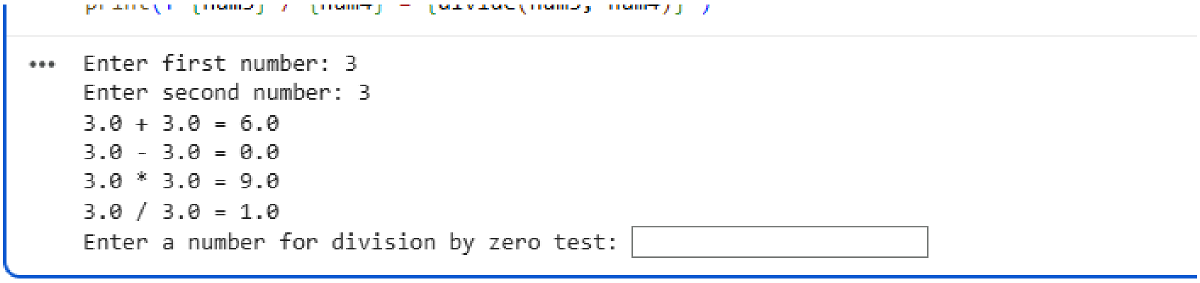
Write a Python script with few functions (e.g., calculator: add, subtract, multiply, divide).the input from the user Incorporate manual docstring in code with NumPy Style With no comments and docstrings

CODE WITH MANUAL COMMENTS:-



```
def add(x, y):
    """Adds two numbers."""
    return x + y

def subtract(x, y):
    """
    Subtracts the second number from the first"""
    return x - y

def multiply(x, y):
    """Multiplies the two numbers."""
    return x * y

def divide(x, y):
    """Divides the first number by the second."""
    if y == 0:
        return "Error! Division by zero."
    else:
        return x / y

# Get input from the user
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))

# Perform calculations and print results
print(f"{num1} + {num2} = {add(num1, num2)}")
print(f"{num1} - {num2} = {subtract(num1, num2)}")
print(f"{num1} * {num2} = {multiply(num1, num2)}")
print(f"{num1} / {num2} = {divide(num1, num2)}")

# Example with division by zero
num3 = float(input("Enter a number for division by zero test: "))
num4 = 0
print(f"{num3} / {num4} = {divide(num3, num4)}")
```

OUTPUT:-

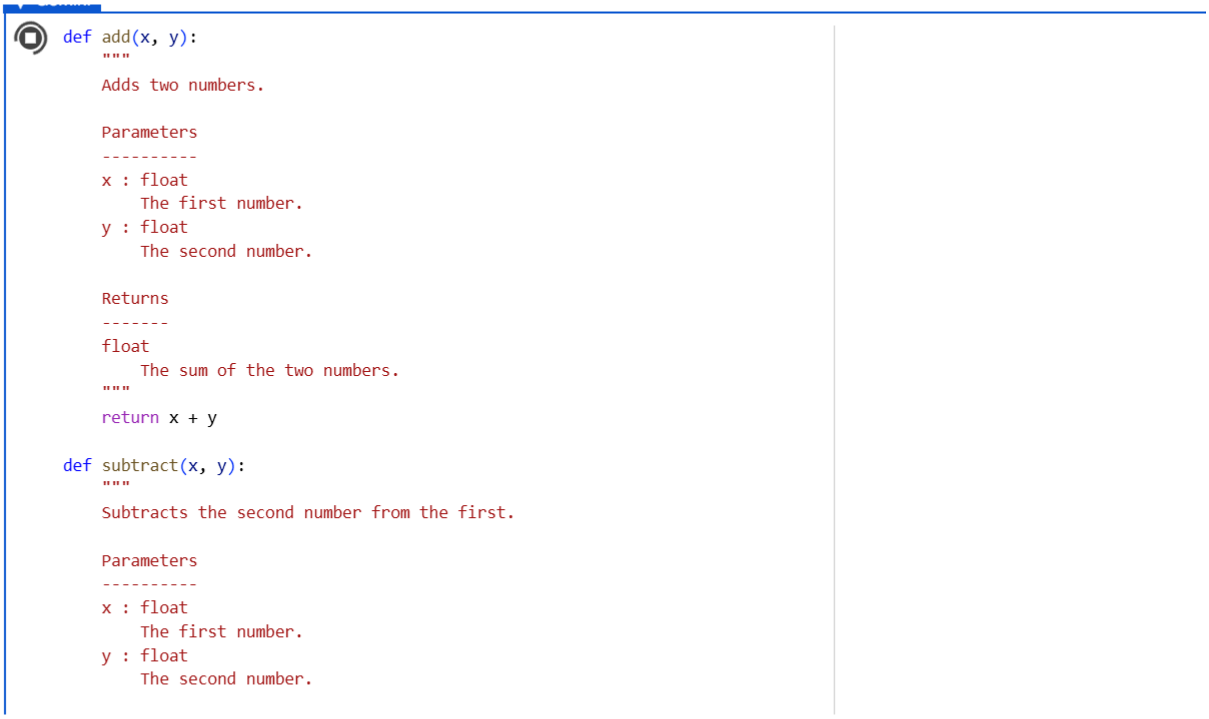


```
Enter first number: 3
Enter second number: 3
3.0 + 3.0 = 6.0
3.0 - 3.0 = 0.0
3.0 * 3.0 = 9.0
3.0 / 3.0 = 1.0
Enter a number for division by zero test:
```

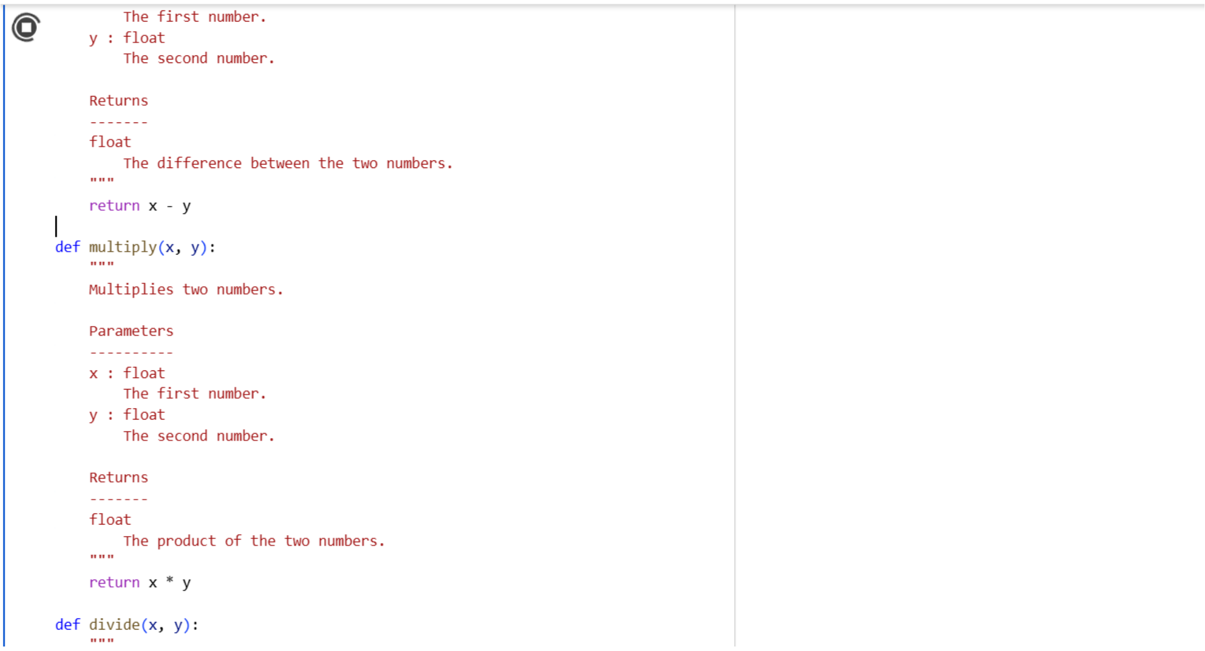
PROMPT:

Write a Python script with few functions (e.g., calculator: add, subtract, multiply, divide).the input from the user Incorporate manual docstring in code with NumPy Style

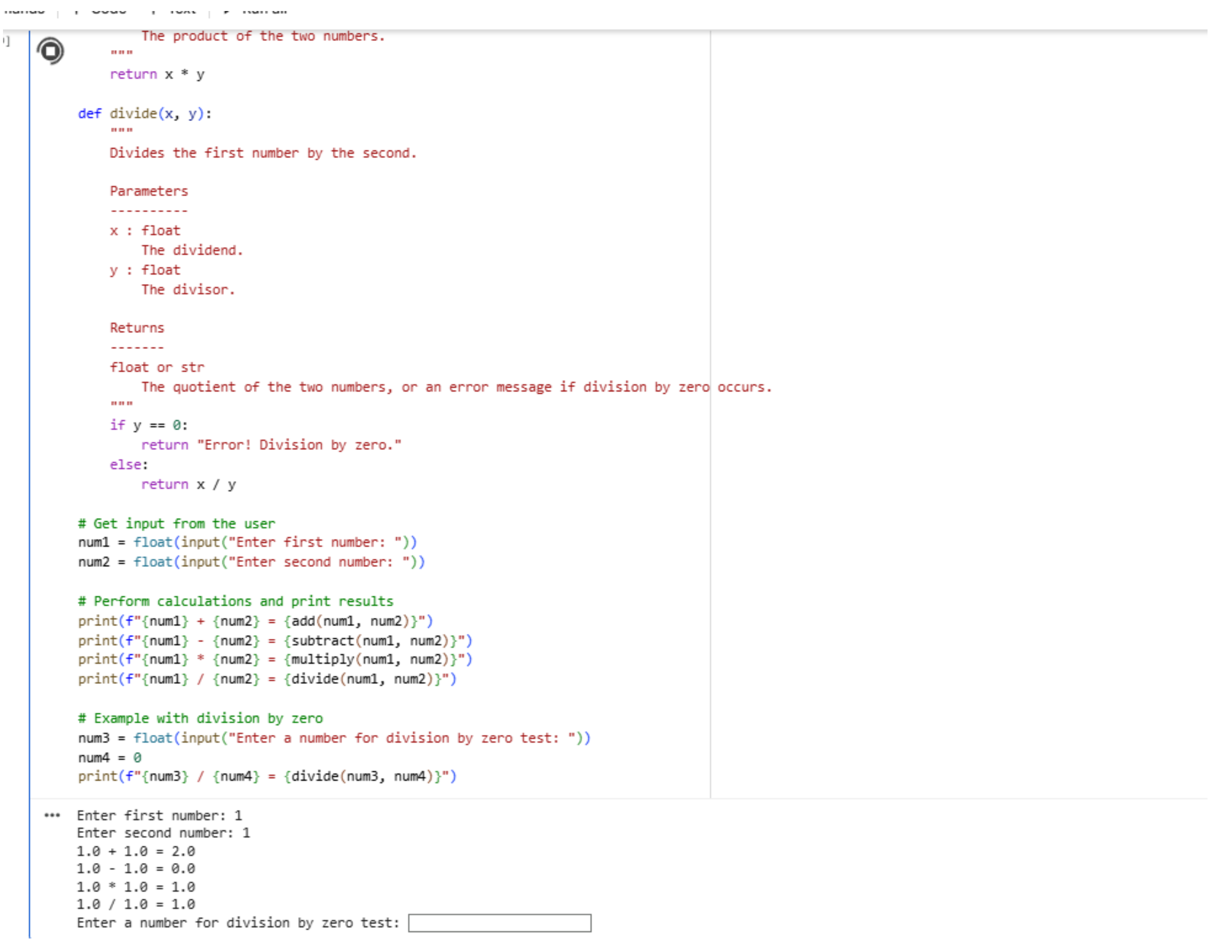
CODE WITH AI-GENERATED COMMENTS:-



```python
def add(x, y):
    """
    Adds two numbers.

    Parameters
    ----------
    x : float
        The first number.
    y : float
        The second number.

    Returns
    -------
    float
        The sum of the two numbers.
    """
    return x + y

def subtract(x, y):
    """
    Subtracts the second number from the first.

    Parameters
    ----------
    x : float
        The first number.
    y : float
        The second number.
```



```python
        The first number.
    y : float
        The second number.

    Returns
    -------
    float
        The difference between the two numbers.
    """
    return x - y

def multiply(x, y):
    """
    Multiplies two numbers.

    Parameters
    ----------
    x : float
        The first number.
    y : float
        The second number.

    Returns
    -------
    float
        The product of the two numbers.
    """
    return x * y

def divide(x, y):
    """
```



```python
        The product of the two numbers.
    """
    return x * y

def divide(x, y):
    """
    Divides the first number by the second.

    Parameters
    ----------
    x : float
        The dividend.
    y : float
        The divisor.

    Returns
    -------
    float or str
        The quotient of the two numbers, or an error message if division by zero occurs.
    """
    if y == 0:
        return "Error! Division by zero."
    else:
        return x / y

# Get input from the user
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))

# Perform calculations and print results
print(f"{num1} + {num2} = {add(num1, num2)}")
print(f"{num1} - {num2} = {subtract(num1, num2)}")
print(f"{num1} * {num2} = {multiply(num1, num2)}")
print(f"{num1} / {num2} = {divide(num1, num2)}")

# Example with division by zero
num3 = float(input("Enter a number for division by zero test: "))
num4 = 0
print(f"{num3} / {num4} = {divide(num3, num4)}")
```

```
Enter first number: 1
Enter second number: 1
1.0 + 1.0 = 2.0
1.0 - 1.0 = 0.0
1.0 * 1.0 = 1.0
1.0 / 1.0 = 1.0
Enter a number for division by zero test:
```

OBSERVATION:

As per my Observation.Both docstrings looking good and has meaning as per the code that is been given,In my docstring it has simple English letters and words where as in Gemini code ,it has used some Terminology.

Apart from that Everything is good when we compare the docstrings .Externally We took help to get module level docstring from GEMINI.

THANK YOU