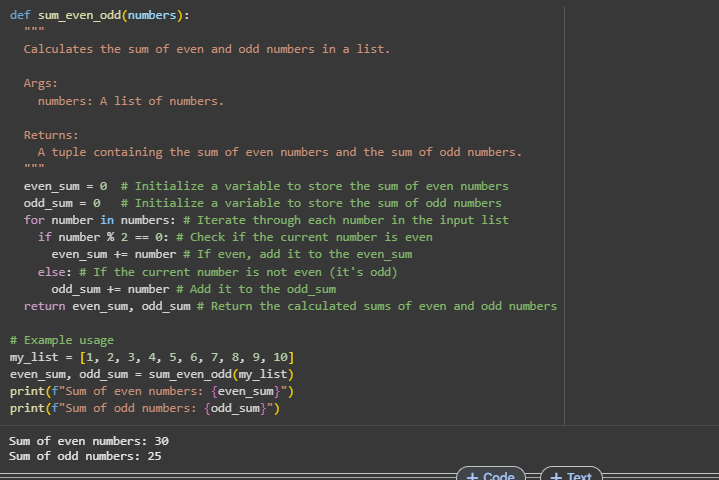
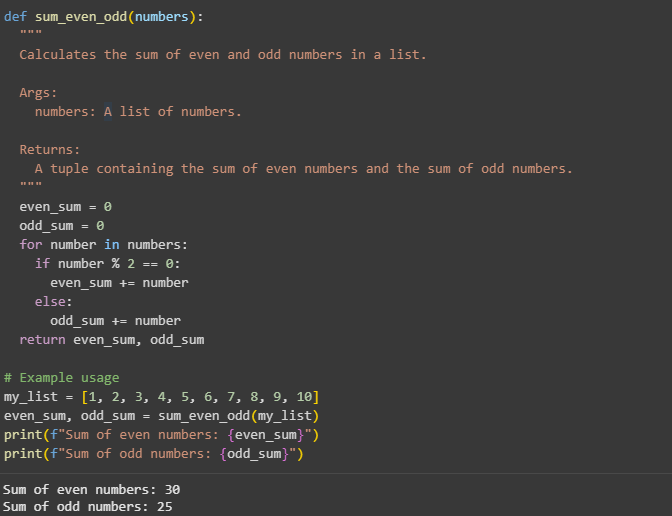
**AI ASSISSTED CODE – 9**

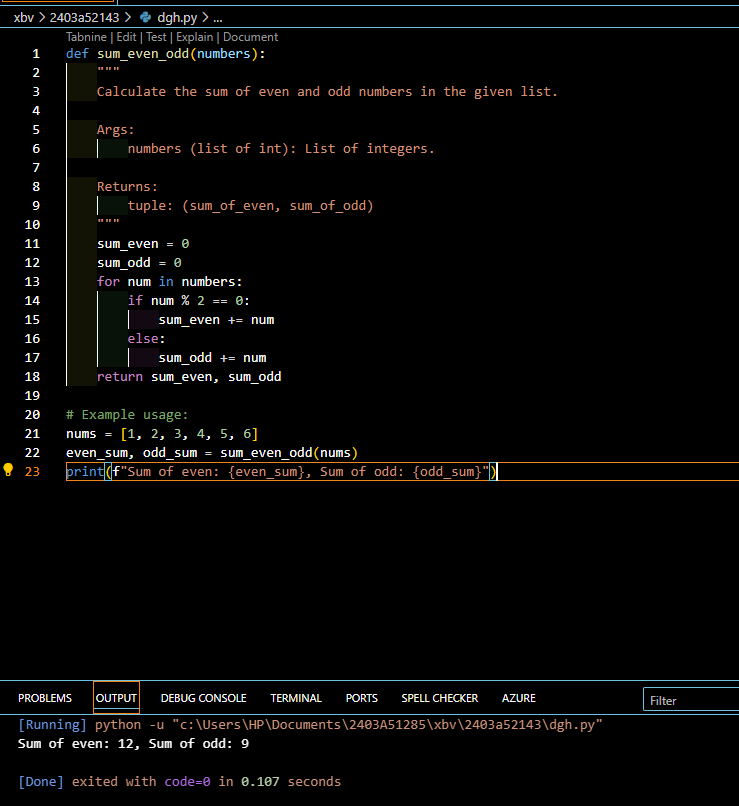
**TASK-1:**

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

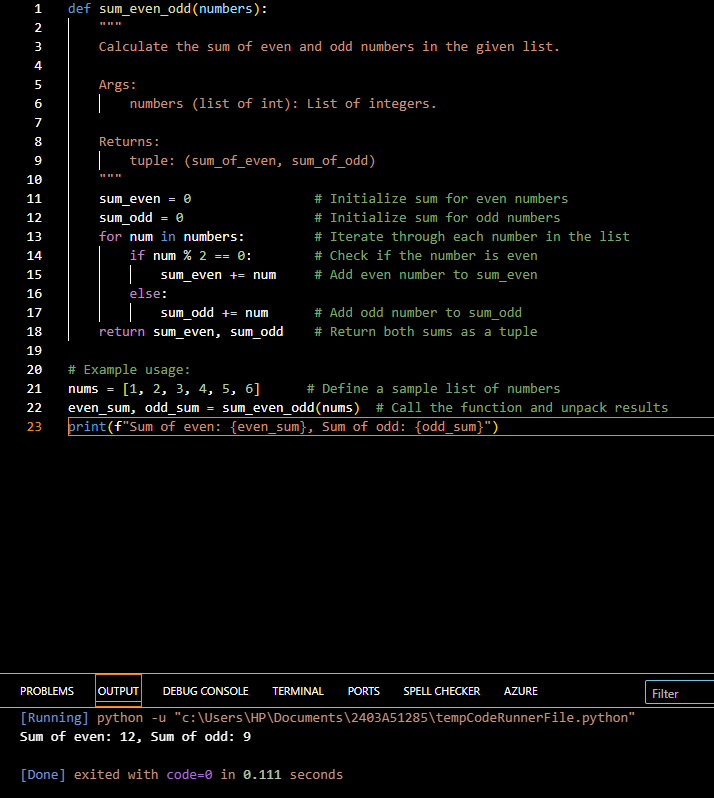
****

**COMMENTS:**

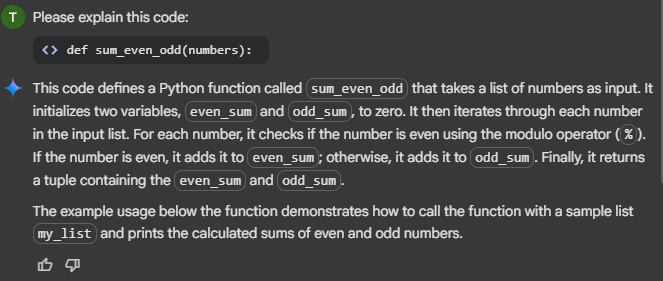
****

****

**COMMENTS:**

****

**CODE EXPLANATION:**

****

**TASK-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

class sru\_student:

  """

  Represents a student at SRU with attributes for name, roll number, and hostel status.

  Includes methods to update fees and display student details.

  """

  def \_\_init\_\_(self, name, roll\_no, hostel\_status):

    """

    Initializes a new sru\_student object.

    Args:

      name (str): The name of the student.

      roll\_no (str): The roll number of the student.

      hostel\_status (bool): True if the student is a hosteller, False otherwise.

    """

    self.name = name

    self.roll\_no = roll\_no

    self.hostel\_status = hostel\_status

    self.fees\_paid = 0 # Initialize fees paid to 0

  def fee\_update(self, amount):

    """

    Updates the amount of fees paid by the student.

    Args:

      amount (float): The amount of fees to add to the fees paid.

    """

    if amount > 0:

      self.fees\_paid += amount

      print(f"Fees updated for {self.name}. Total fees paid: {self.fees\_paid}")

    else:

      print("Invalid amount. Please provide a positive value for fee update.")

  def display\_details(self):

    """

    Displays the details of the student.

    """

    print("\n--- Student Details ---")

    print(f"Name: {self.name}")

    print(f"Roll Number: {self.roll\_no}")

    print(f"Hostel Status: {'Hosteller' if self.hostel\_status else 'Day Scholar'}")

    print(f"Fees Paid: {self.fees\_paid}")

    print("-----------------------")

# Example Usage:

student1 = sru\_student("Alice Smith", "SRU12345", True)

student2 = sru\_student("Bob Johnson", "SRU67890", False)

student1.display\_details()

student2.display\_details()

student1.fee\_update(5000)

student2.fee\_update(3000)

student1.fee\_update(2000) # Another fee update for student1

student1.display\_details()

student2.display\_details()

**COMMENTS:**

class sru\_student:

  """

  Represents a student at SRU with attributes for name, roll number, and hostel status.

  Includes methods to update fees and display student details.

  """

  def \_\_init\_\_(self, name, roll\_no, hostel\_status):

    """

    Initializes a new sru\_student object.

    Args:

      name (str): The name of the student.

      roll\_no (str): The roll number of the student.

      hostel\_status (bool): True if the student is a hosteller, False otherwise.

    """

    self.name = name  # Assign the provided name to the object's name attribute

    self.roll\_no = roll\_no # Assign the provided roll number to the object's roll\_no attribute

    self.hostel\_status = hostel\_status # Assign the provided hostel status to the object's hostel\_status attribute

    self.fees\_paid = 0 # Initialize fees paid to 0 for a new student

  def fee\_update(self, amount):

    """

    Updates the amount of fees paid by the student.

    Args:

      amount (float): The amount of fees to add to the fees paid.

    """

    if amount > 0: # Check if the update amount is positive

      self.fees\_paid += amount # Add the valid amount to the fees\_paid

      print(f"Fees updated for {self.name}. Total fees paid: {self.fees\_paid}") # Print a confirmation message

    else:

      print("Invalid amount. Please provide a positive value for fee update.") # Print an error message for invalid input

  def display\_details(self):

    """

    Displays the details of the student.

    """

    print("\n--- Student Details ---") # Print a header for student details

    print(f"Name: {self.name}") # Print the student's name

    print(f"Roll Number: {self.roll\_no}") # Print the student's roll number

    print(f"Hostel Status: {'Hosteller' if self.hostel\_status else 'Day Scholar'}") # Print the hostel status (Hosteller or Day Scholar)

    print(f"Fees Paid: {self.fees\_paid}") # Print the total fees paid by the student

    print("-----------------------") # Print a footer for student details

# Example Usage:

student1 = sru\_student("Alice Smith", "SRU12345", True) # Create the first student object

student2 = sru\_student("Bob Johnson", "SRU67890", False) # Create the second student object

student1.display\_details() # Display details of the first student

student2.display\_details() # Display details of the second student

student1.fee\_update(5000) # Update fees for the first student

student2.fee\_update(3000) # Update fees for the second student

student1.fee\_update(2000) # Another fee update for the first student

student1.display\_details() # Display updated details of the first student

**OUTPUT:**

--- Student Details ---

Name: Alice Smith

Roll Number: SRU12345

Hostel Status: Hosteller

Fees Paid: 0

-----------------------

--- Student Details ---

Name: Bob Johnson

Roll Number: SRU67890

Hostel Status: Day Scholar

Fees Paid: 0

-----------------------

Fees updated for Alice Smith. Total fees paid: 5000

Fees updated for Bob Johnson. Total fees paid: 3000

Fees updated for Alice Smith. Total fees paid: 7000

--- Student Details ---

Name: Alice Smith

Roll Number: SRU12345

Hostel Status: Hosteller

Fees Paid: 7000

-----------------------

--- Student Details ---

Name: Bob Johnson

Roll Number: SRU67890

Hostel Status: Day Scholar

Fees Paid: 3000

------

class SRUStudent:

    def \_\_init\_\_(self, name, roll\_no, hostel\_status):

        self.name = name                      # Student's name

        self.roll\_no = roll\_no                # Student's roll number

        self.hostel\_status = hostel\_status    # Hostel status (True/False or string)

        self.fee\_paid = False                 # Fee payment status

    def fee\_update(self, status):

        """

        Update the fee payment status.

        Args:

            status (bool): True if fee is paid, False otherwise.

        """

        self.fee\_paid = status

    def display\_details(self):

        """

        Display the student's details.

        """

        print(f"Name: {self.name}")

        print(f"Roll No.: {self.roll\_no}")

        print(f"Hostel Status: {self.hostel\_status}")

        print(f"Fee Paid: {'Yes' if self.fee\_paid else 'No'}")

# Example usage:

student1 = SRUStudent("Alice", 101,"Yes")

student1.fee\_update(True)

student1.display\_details()

**COMMENTS:**

class SRUStudent:

    def \_\_init\_\_(self, name, roll\_no, hostel\_status):

        self.name = name                      # Assign the student's name

        self.roll\_no = roll\_no                # Assign the student's roll number

        self.hostel\_status = hostel\_status    # Assign the student's hostel status

        self.fee\_paid = False                 # Initialize fee payment status as False

    def fee\_update(self, status):

        """

        Update the fee payment status.

        Args:

            status (bool): True if fee is paid, False otherwise.

        """

        self.fee\_paid = status                # Update the fee payment status

    def display\_details(self):

        """

        Display the student's details.

        """

        print(f"Name: {self.name}")           # Print the student's name

        print(f"Roll No.: {self.roll\_no}")    # Print the student's roll number

        print(f"Hostel Status: {self.hostel\_status}")  # Print the hostel status

        print(f"Fee Paid: {'Yes' if self.fee\_paid else 'No'}")  # Print fee payment status

# Example usage:

student1 = SRUStudent("Alice", 101,"Yes")     # Create a student object

student1.fee\_update(True)                     # Update fee status to paid

student1.display\_details()                    # Display student details

**OUTPUT:**

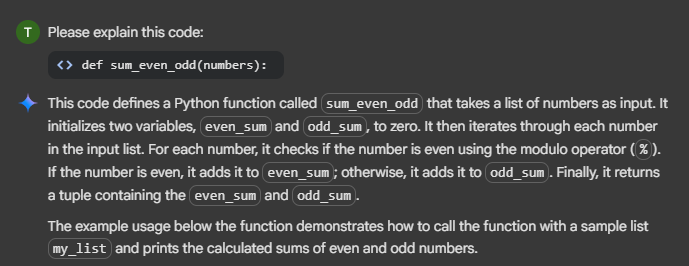
Name: Alice

Roll No.: 101

Hostel Status: Yes

Fee Paid: Yes

**CODE EXPLANATION:**

****

**TASK-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

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 two numbers."""

  return x \* y

def divide(x, y):

  """Divides the first number by the second. Returns an error message if the divisor is zero."""

  if y == 0:

    return "Error: Division by zero is not allowed."

  return x / y

# Example usage:

num1 = 10

num2 = 5

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)}")

print(f"{num1} / 0 = {divide(num1, 0)}") # Example of division by zero

**COMMENTS:**

def add(x, y):

  """Adds two numbers."""

  return x + y # Return the sum of x and y

def subtract(x, y):

  """Subtracts the second number from the first."""

  return x - y # Return the difference between x and y

def multiply(x, y):

  """Multiplies two numbers."""

  return x \* y # Return the product of x and y

def divide(x, y):

  """Divides the first number by the second. Returns an error message if the divisor is zero."""

  if y == 0: # Check if the divisor is zero

    return "Error: Division by zero is not allowed." # Return an error message if division by zero

  return x / y # Return the result of the division

# Example usage:

num1 = 10 # Define the first number

num2 = 5  # Define the second number

print(f"{num1} + {num2} = {add(num1, num2)}") # Call add function and print the result

print(f"{num1} - {num2} = {subtract(num1, num2)}") # Call subtract function and print the result

print(f"{num1} \* {num2} = {multiply(num1, num2)}") # Call multiply function and print the result

print(f"{num1} / {num2} = {divide(num1, num2)}") # Call divide function and print the result

print(f"{num1} / 0 = {divide(num1, 0)}") # Example of division by zero, call divide and print the result

**OUTPUT:**

10 + 5 = 15

10 - 5 = 5

10 \* 5 = 50

10 / 5 = 2.0

10 / 0 = Error: Division by zero is not allowed.

def add(a, b):

    """Return the sum of a and b."""

    return a + b

def subtract(a, b):

    """Return the difference of a and b."""

    return a - b

def multiply(a, b):

    """Return the product of a and b."""

    return a \* b

def divide(a, b):

    """Return the quotient of a divided by b. Handles division by zero."""

    if b == 0:

        return "Error: Division by zero"

    return a / b

# Example usage:

x = 10

y = 5

print("Add:", add(x, y))

print("Subtract:", subtract(x, y))

print("Multiply:", multiply(x, y))

print("Divide:", divide(x, y))

**COMMENTS:**

def add(a, b):

    """Return the sum of a and b."""

    return a + b                # Add a and b, return the result

def subtract(a, b):

    """Return the difference of a and b."""

    return a - b                # Subtract b from a, return the result

def multiply(a, b):

    """Return the product of a and b."""

    return a \* b                # Multiply a and b, return the result

def divide(a, b):

    """Return the quotient of a divided by b. Handles division by zero."""

    if b == 0:                  # Check if b is zero to avoid division error

        return "Error: Division by zero"  # Return error message if b is zero

    return a / b                # Divide a by b, return the result

# Example usage:

x = 10                         # Assign value 10 to x

y = 5                          # Assign value 5 to y

print("Add:", add(x, y))       # Print result of add function

print("Subtract:", subtract(x, y))  # Print result of subtract function

print("Multiply:", multiply(x, y))  # Print result of multiply function

print("Divide:", divide(x, y))      # Print result of divide function

**OUTPUT:**

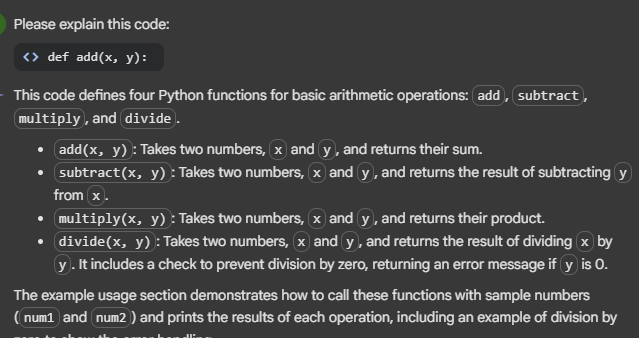
Add: 15

Subtract: 5

Multiply: 50

Divide: 2.0

**CODE EXPLANATION:**

****