ASSIGNMENT-9.3

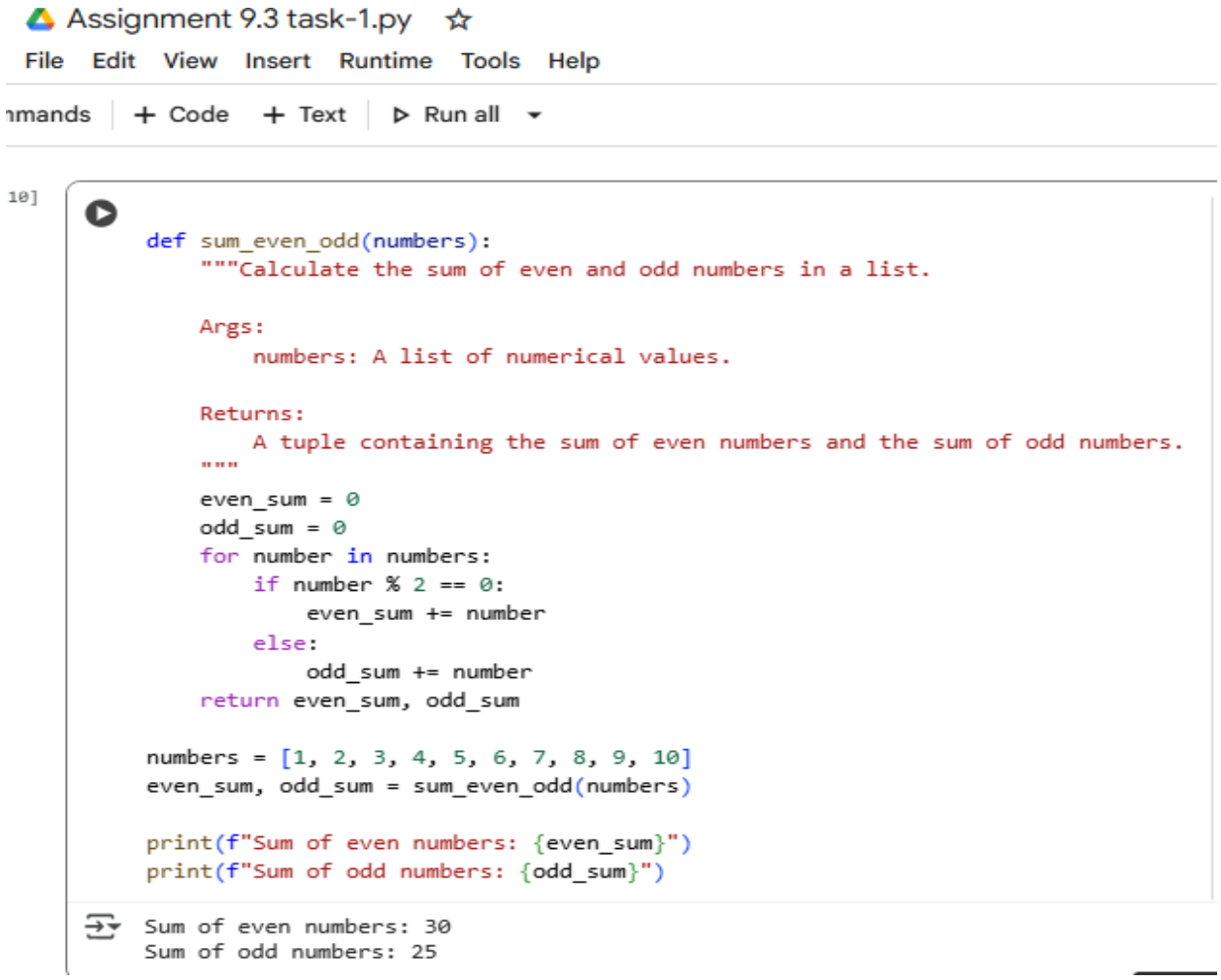
TASK-1:

Prompt:

#Write a python program to find sum of even, odd numbers using functions without any comments.

WITH GEMINI AI:

Code with Output:



Prompt:

#Write a python program to find sum of even, odd numbers using functions with inline comments.

Code with output:

```
def sum_even_odd(numbers): # Define a function that takes a list of numbers
    """Calculate the sum of even and odd numbers in a list.

    Args:
        numbers: A list of numerical values.

    Returns:
        A tuple containing the sum of even numbers and the sum of odd numbers.
    """

    even_sum = 0 # Initialize a variable for the sum of even numbers
    odd_sum = 0 # Initialize a variable for the sum of odd numbers
    for number in numbers: # Iterate through each number in the list
        if number % 2 == 0: # Check if the number is even
            even_sum += number # Add the number to the even_sum
        else: # If the number is not even (it's odd)
            odd_sum += number # Add the number to the odd_sum
    return even_sum, odd_sum # Return the sum of even and odd numbers

numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] # Create a list of numbers
even_sum, odd_sum = sum_even_odd(numbers) # Call the function and store the results

print(f"Sum of even numbers: {even_sum}") # Print the sum of even numbers
print(f"Sum of odd numbers: {odd_sum}") # Print the sum of odd numbers
```

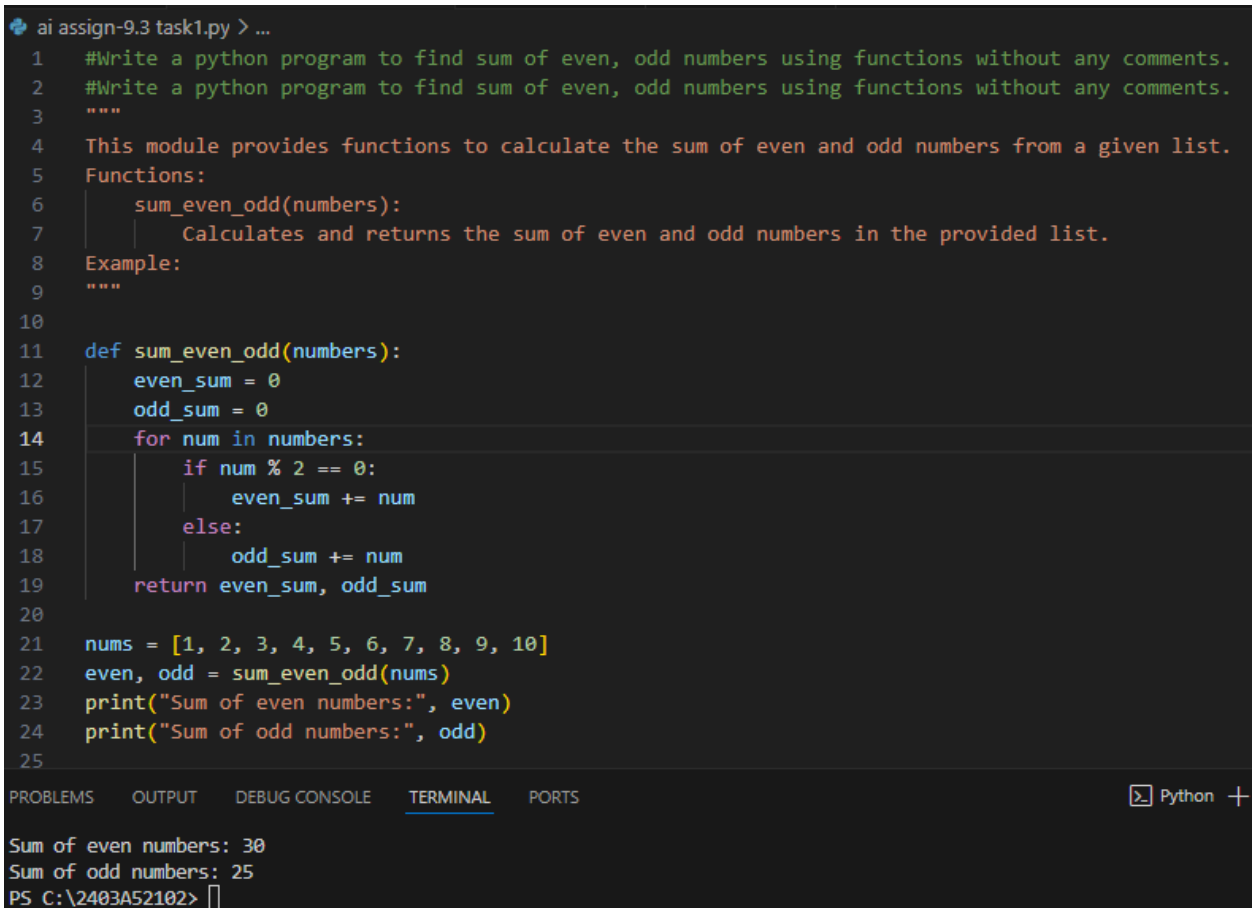
```
Sum of even numbers: 30
Sum of odd numbers: 25
```

WITH GITHUB COPILOT:

Prompt:

#Write a python program to find sum of even, numbers using functions without any comments.

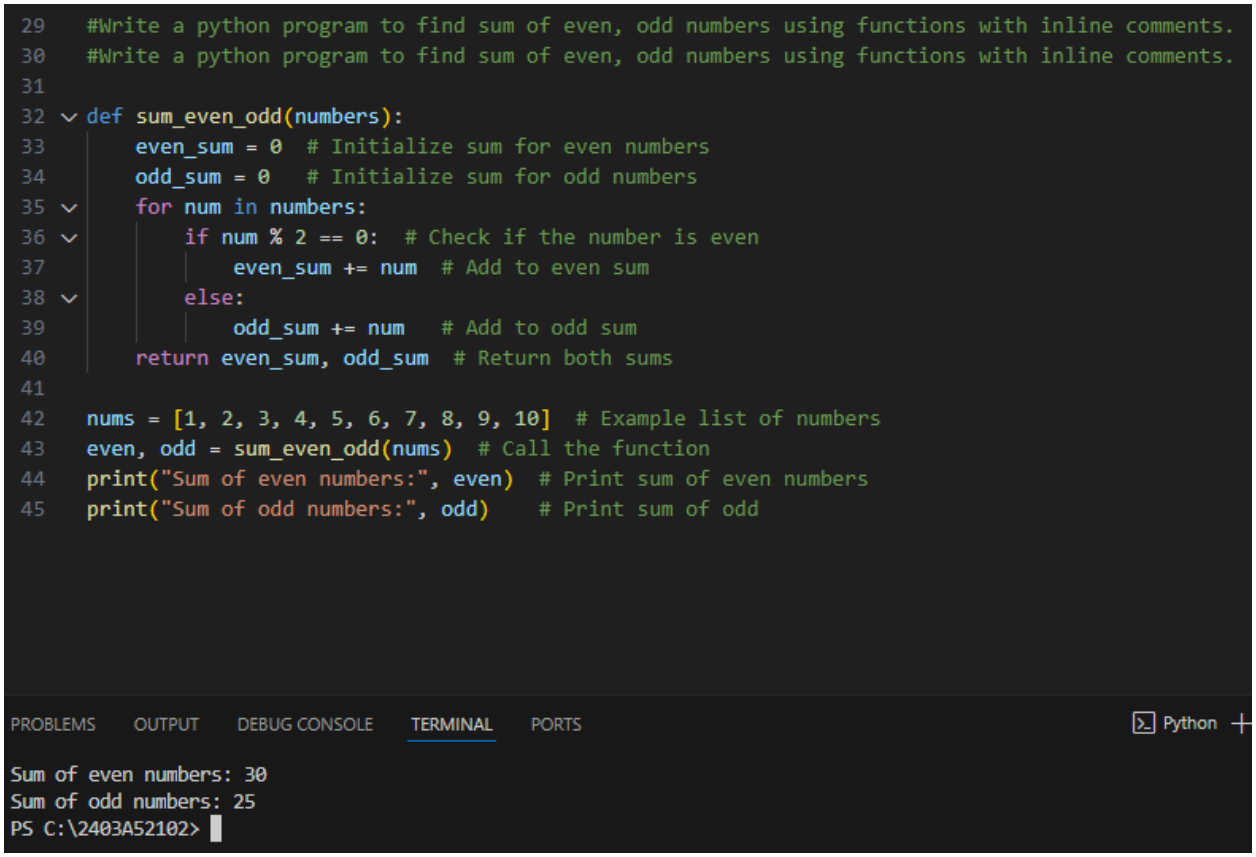
Code with Output:



Prompt:

#Write a python program to find sum of even, numbers using functions with inline comments.

Code with Output:



```python
#Write a python program to find sum of even, odd numbers using functions with inline comments.
#Write a python program to find sum of even, odd numbers using functions with inline comments.

def sum_even_odd(numbers):
    even_sum = 0  # Initialize sum for even numbers
    odd_sum = 0   # Initialize sum for odd numbers
    for num in numbers:
        if num % 2 == 0:  # Check if the number is even
            even_sum += num  # Add to even sum
        else:
            odd_sum += num   # Add to odd sum
    return even_sum, odd_sum  # Return both sums

nums = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  # Example list of numbers
even, odd = sum_even_odd(nums)  # Call the function
print("Sum of even numbers:", even)  # Print sum of even numbers
print("Sum of odd numbers:", odd)    # Print sum of odd
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                Python +

Sum of even numbers: 30
Sum of odd numbers: 25
PS C:\2403A52102>
```

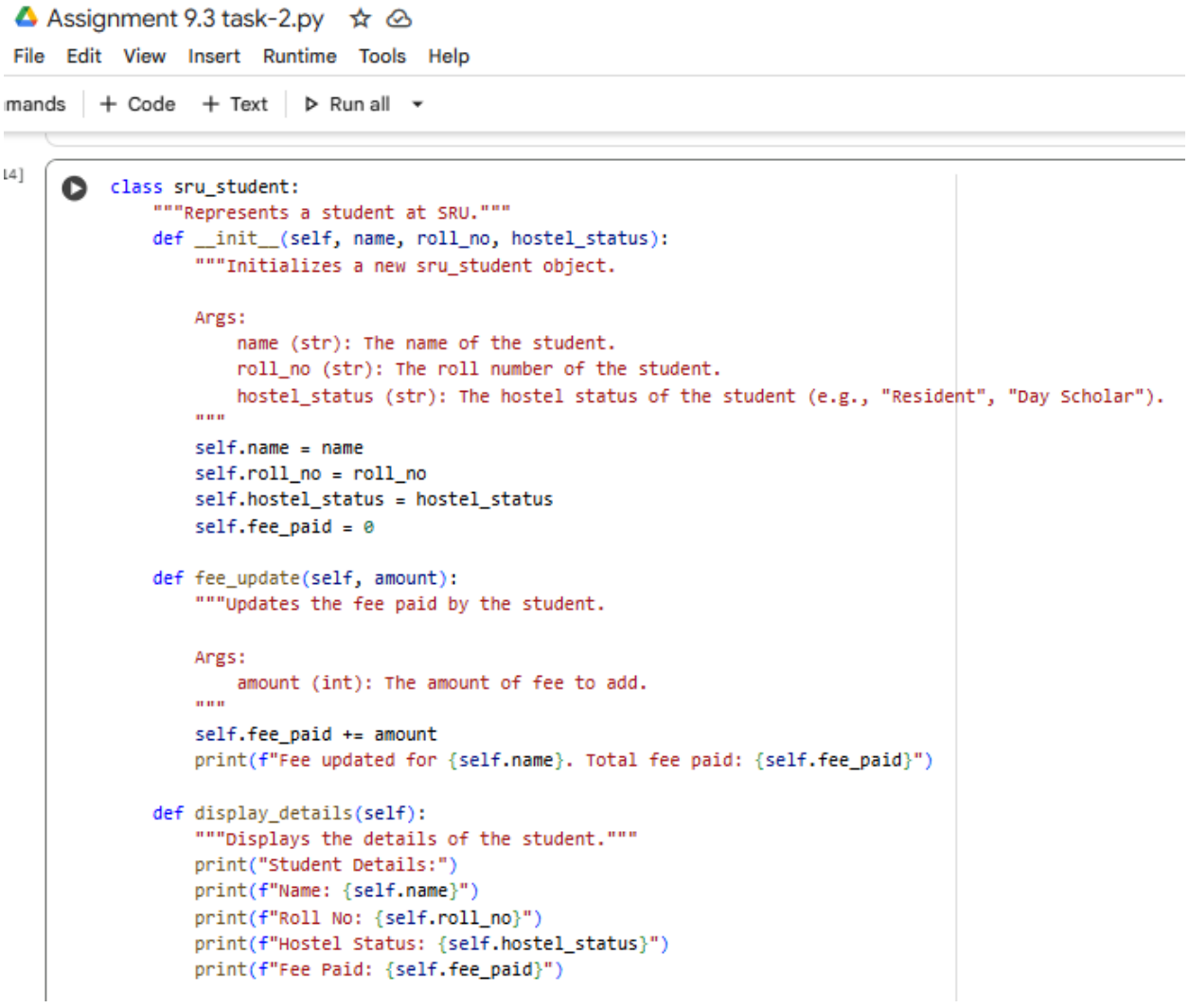
TASK-2:

Prompt:

#Write python program for class named sru_student with attributes like name, roll no., hostel_status and fee_update method and display_details method without any comments.

WITH GEMINI AI:

Code with Output:




```python
class sru_student:
    """Represents a student at SRU."""
    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 (str): The hostel status of the student (e.g., "Resident", "Day Scholar").
        """
        self.name = name
        self.roll_no = roll_no
        self.hostel_status = hostel_status
        self.fee_paid = 0

    def fee_update(self, amount):
        """Updates the fee paid by the student.

        Args:
            amount (int): The amount of fee to add.
        """
        self.fee_paid += amount
        print(f"Fee updated for {self.name}. Total fee paid: {self.fee_paid}")

    def display_details(self):
        """Displays the details of the student."""
        print("Student Details:")
        print(f"Name: {self.name}")
        print(f"Roll No: {self.roll_no}")
        print(f"Hostel Status: {self.hostel_status}")
        print(f"Fee Paid: {self.fee_paid}")
```

```
# Example usage:
student1 = sru_student("Alice", "SRU123", "Resident")
student1.display_details()
student1.fee_update(5000)
student1.display_details()
```

```
Student Details:
Name: Alice
Roll No: SRU123
Hostel Status: Resident
Fee Paid: 0
Fee updated for Alice. Total fee paid: 5000
Student Details:
Name: Alice
Roll No: SRU123
Hostel Status: Resident
Fee Paid: 5000
```

Prompt:

#Write python program for class named sru_student with attributes like name, roll no., hostel_status and fee_update method and display_details method with inline comments.

Code with output:

```
class sru_student: # Define a class named sru_student
    def __init__(self, name, roll_no, hostel_status): # Constructor method to initialize the student object
        self.name = name # Initialize the name attribute
        self.roll_no = roll_no # Initialize the roll_no attribute
        self.hostel_status = hostel_status # Initialize the hostel_status attribute
        self.fee_paid = 0 # Initialize the fee_paid attribute to 0

    def fee_update(self, amount): # Method to update the fee paid by the student
        self.fee_paid += amount # Add the amount to the fee_paid
        print(f"Fee updated for {self.name}. Total fee paid: {self.fee_paid}") # Print a confirmation message

    def display_details(self): # Method to display the student's details
        print("Student Details:") # Print a header for the 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 student's hostel status
        print(f"Fee Paid: {self.fee_paid}") # Print the total fee paid

# Example usage:
student1 = sru_student("Alice", "SRU123", "Resident") # Create an instance of the sru_student class for Alice
student1.display_details() # Display details for student1
student1.fee_update(5000) # Update the fee paid for student1
student1.display_details() # Display details for student1 again
```

Output:

```
Student Details:
Name: Alice
Roll No: SRU123
Hostel Status: Resident
Fee Paid: 0
Fee updated for Alice. Total fee paid: 5000
Student Details:
Name: Alice
Roll No: SRU123
Hostel Status: Resident
Fee Paid: 5000
```

WITH GITHUB COPILOT:

Prompt:

#Write python program for class named sru_student with attributes like name, roll no., hostel_status and fee_update method and display_details method without any comments.

Code with output:

ai assign-9.3 task2.py > ...

```python
#Write python program for class named sru_student with attributes like name, roll no., hostel_status and fee_update method and display_details method
#without any comments.
"""
Class sru_student represents a student with attributes for name, roll number, hostel status, and fee paid.
Attributes:
    name (str): Name of the student.
    roll_no (str): Roll number of the student.
    hostel_status (str): Hostel accommodation status ("Yes" or "No").
    fee_paid (float): Amount of fee paid by the student.
Methods:
    fee_update(amount):
        Adds the specified amount to the student's fee_paid.
    display_details():
        Prints the student's details including name, roll number, hostel status, and fee paid.
"""

class sru_student:
    def __init__(self, name, roll_no, hostel_status, fee_paid):
        self.name = name
        self.roll_no = roll_no
        self.hostel_status = hostel_status
        self.fee_paid = fee_paid

    def fee_update(self, amount):
        self.fee_paid += amount

    def display_details(self):
        print("Name:", self.name)
        print("Roll No.:", self.roll_no)
        print("Hostel Status:", self.hostel_status)
        print("Fee Paid:", self.fee_paid)

```

```
33    student1 = sru_student("Savika", "2403A52102", "Yes", 5000)
34    student1.display_details()

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

Name: Savika
Roll No.: 2403A52102
Hostel Status: Yes
Fee Paid: 5000
PS C:\2403A52102> []
```

Prompt:

#Write python program for class named sru_student with attributes like name, roll no., hostel_status and fee_update method and display_details method with inline comments.

Code with output:

```
36
37  #Write python program for class named sru_student with attributes like name, roll no., hostel_status and fee_update method and display_details
38  # with inline comments.
39  """
40  Class sru_student represents a student with attributes for name, roll number, hostel status, and fee paid.
41  Attributes:
42      name (str): Name of the student.
43      roll_no (str): Roll number of the student.
44      hostel_status (str): Hostel accommodation status ("Yes" or "No").
45      fee_paid (float): Amount of fee paid by the student.
46  Methods:
47      fee_update(amount):
48          Adds the specified amount to the student's fee_paid.
49      display_details():
50          Prints the student's details including name, roll number, hostel status, and fee paid.
51  """
52  class sru_student:
53      def __init__(self, name, roll_no, hostel_status, fee_paid):
54          # Initialize student attributes
55          self.name = name
56          self.roll_no = roll_no
57          self.hostel_status = hostel_status
58          self.fee_paid = fee_paid
59
60      def fee_update(self, amount):
61          # Update the fee_paid by adding the given amount
62          self.fee_paid += amount
63
64      def display_details(self):
65          # Display all details of the student
66          print("Name:", self.name)
67          print("Roll No.:", self.roll_no)
68          print("Hostel Status:", self.hostel_status)
69          print("Fee Paid:", self.fee_paid)
70
```

```
64          def display_details(self):
65              # Display all details of the student
66              print("Name:", self.name)
67              print("Roll No.:", self.roll_no)
68              print("Hostel Status:", self.hostel_status)
69              print("Fee Paid:", self.fee_paid)
70
71      # Create a student object with initial details
72      student1 = sru_student("Savika", "2403A52102", "Yes", 5000)
73      # Update the fee paid by the student
74      student1.fee_update(2000)
75      # Display the student's details
76      student1.display_details()

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

sign-9.3 task2.py"
Name: Savika
Roll No.: 2403A52102
Hostel Status: Yes
Fee Paid: 5000
```

TASK-3:

Prompt:

#Write a python program to generate a calculator using functions like add, sub, multiply, divide without any comments.

WITH GEMINI COPILOT:

Code with Output:

```
[3]
19s
def add(x, y):
    return x + y

def subtract(x, y):
    return x - y

def multiply(x, y):
    return x * y

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

print("Select operation:")
print("1.Add")
print("2.Subtract")
print("3.Multiply")
print("4.Divide")
```

```
while True:
    choice = input("Enter choice(1/2/3/4): ")

    if choice in ('1', '2', '3', '4'):
        num1 = float(input("Enter first number: "))
        num2 = float(input("Enter second number: "))

        if choice == '1':
            print(num1, "+", num2, "=", add(num1, num2))

        elif choice == '2':
            print(num1, "-", num2, "=", subtract(num1, num2))

        elif choice == '3':
            print(num1, "*", num2, "=", multiply(num1, num2))

        elif choice == '4':
            print(num1, "/", num2, "=", divide(num1, num2))
        break
    else:
        print("Invalid Input")
```

```
Select operation:
1.Add
2.Subtract
3.Multiply
4.Divide
Enter choice(1/2/3/4): 5
Invalid Input
Enter choice(1/2/3/4): 1
Enter first number: 1
Enter second number: 2
1.0 + 2.0 = 3.0
```

Prompt:

#Write a python program to generate a calculator using functions like add, sub, multiply, divide with inline comments.

Code with Output:

```python
def add(x, y): # Function to add two numbers
    """Adds two numbers and returns the result."""
    return x + y

def subtract(x, y): # Function to subtract two numbers
    """Subtracts the second number from the first and returns the result."""
    return x - y

def multiply(x, y): # Function to multiply two numbers
    """Multiplies two numbers and returns the result."""
    return x * y

def divide(x, y): # Function to divide two numbers
    """Divides the first number by the second and returns the result.
    Returns an error message if the second number is zero."""
    if y == 0: # Check if the denominator is zero
        return "Error! Division by zero."
    else:
        return x / y

print("Select operation:") # Print the available operations
print("1.Add")
print("2.Subtract")
print("3.Multiply")
print("4.Divide")
```

```python
while True: # Start a loop to get user input
    choice = input("Enter choice(1/2/3/4): ") # Get the user's choice of operation

    if choice in ('1', '2', '3', '4'): # Check if the choice is valid
        num1 = float(input("Enter first number: ")) # Get the first number from the user
        num2 = float(input("Enter second number: ")) # Get the second number from the user

        if choice == '1': # If the choice is 1 (Add)
            print(num1, "+", num2, "=", add(num1, num2)) # Perform addition and print the result

        elif choice == '2': # If the choice is 2 (Subtract)
            print(num1, "-", num2, "=", subtract(num1, num2)) # Perform subtraction and print the result

        elif choice == '3': # If the choice is 3 (Multiply)
            print(num1, "*", num2, "=", multiply(num1, num2)) # Perform multiplication and print the result

        elif choice == '4': # If the choice is 4 (Divide)
            print(num1, "/", num2, "=", divide(num1, num2)) # Perform division and print the result
        break # Exit the loop after a valid operation is performed
    else:
        print("Invalid Input") # Print an error message for invalid input
```

```
Select operation:
1.Add
2.Subtract
3.Multiply
4.Divide
Enter choice(1/2/3/4): 5
Invalid Input
Enter choice(1/2/3/4): 4
Enter first number: 10
Enter second number: 0
10.0 / 0.0 = Error! Division by zero.
```

Prompt:

#Write a python program to generate a calculator using functions like add, sub, multiply, divide without any comments.

WITH GITHUB COPILOT:

Code with Output:

ai assign-9.3 task3.py > ...

```python
#Write a python program to generate a calculator using functions like add, sub, multiply, divide without any comments.
"""
A simple calculator program with functions for addition, subtraction, multiplication, and division.
Functions:
    add(a, b): Returns the sum of a and b.
    sub(a, b): Returns the difference of a and b.
    multiply(a, b): Returns the product of a and b.
    divide(a, b): Returns the quotient of a divided by b, or an error message if b is zero.
Prompts the user to input two numbers and displays the results of the four operations.
"""
#Write a python program to generate a calculator using functions like add, sub, multiply, divide without any comments.

def add(a, b):
    return a + b

def sub(a, b):
    return a - b

def multiply(a, b):
    return a * b

def divide(a, b):
    if b != 0:
        return a / b
    else:
        return "Division by zero error"

x = float(input("Enter first number: "))
y = float(input("Enter second number: "))
print("Addition:", add(x, y))
print("Subtraction:", sub(x, y))
print("Multiplication:", multiply(x, y))
print("Division:", divide(x, y))
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Enter first number: 3
Enter second number: 3
Addition: 6.0
Subtraction: 0.0
Multiplication: 9.0
Division: 1.0
PS C:\2403A52102>
```

Prompt:

#Write a python program to generate a calculator using functions like add, sub, multiply, divide with inline comments.

Code with Output:

```python
#Write a python program to generate a calculator using functions like add, sub, multiply, divide with inline comments.
#Write a python program to generate a calculator using functions like add, sub, multiply, divide with inline comments.

def add(a, b):
    return a + b  # Return the sum

def sub(a, b):
    return a - b  # Return the difference

def multiply(a, b):
    return a * b  # Return the product

def divide(a, b):
    if b != 0:
        return a / b  # Return the quotient
    else:
        return "Division by zero error"  # Handle division by zero

x = float(input("Enter first number: "))  # Input first number
y = float(input("Enter second number: ")) # Input second number

print("Addition:", add(x, y))             # Display addition result
print("Subtraction:", sub(x, y))          # Display subtraction result
print("Multiplication:", multiply(x, y))  # Display multiplication result
print("Division:", divide(x, y))          # Display
```

OUTPUT:

```
Enter first number: 25
Enter second number: 26
Addition: 51.0
Subtraction: -1.0
Multiplication: 650.0
Division: 0.9615384615384616
PS C:\2403A52102>
```