

Lab assignment -9

Task-1:

Prompt: Write python function to return sum of even and odd numbers in the given list.).(generate a docstring format)

Code:

```
9.1.py > ...
1  def sum_even_odd(numbers):
2      even_sum = 0
3      odd_sum = 0
4      for number in numbers:
5          if number % 2 == 0:
6              even_sum += number
7          else:
8              odd_sum += number
9      return even_sum, odd_sum
10
11 my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
12 even_sum, odd_sum = sum_even_odd(my_list)
13 print(f"Sum of even numbers: {even_sum}")
14 print(f"Sum of odd numbers: {odd_sum}")
```

debugpy\launcher '49736' '--' 'C:\Aiassisted coding\9.1.py'

Sum of even numbers: 30

Sum of odd numbers: 25

PS C:\Aiassisted coding>

Used 1 reference

This code calculates the sum of even and odd numbers in a list:

- `def sum_even_odd(numbers):`
Defines a function that takes a list of numbers.
- `even_sum = 0` and `odd_sum = 0`
Initializes variables to store the sums.
- The `for` loop iterates through each number in the list.
If the number is even (`number % 2 == 0`), it adds to `even_sum`.
Otherwise, it adds to `odd_sum`.
- `return even_sum, odd_sum`
Returns both sums.
- `my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]`
Example list of numbers.
- `even_sum, odd_sum = sum_even_odd(my_list)`
Calls the function and stores the results.
- `print(f"Sum of even numbers: {even_sum}")`
Prints the sum of even numbers.
- `print(f"Sum of odd numbers: {odd_sum}")`
Prints the sum of odd numbers.

Code with inline comments:

```
9.1.2.py > ...
1  def sum_even_odd(numbers):
2      # Initialize sums for even and odd numbers
3      even_sum = 0
4      odd_sum = 0
5      # Iterate through each number in the list
6      for number in numbers:
7          if number % 2 == 0:
8              # Add to even_sum if the number is even
9              even_sum += number
10         else:
11             # Add to odd_sum if the number is odd
12             odd_sum += number
13     # Return both sums
14     return even_sum, odd_sum
15
16 # Example list of numbers
17 my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
18 # Call the function and store the results
19 even_sum, odd_sum = sum_even_odd(my_list)
20 # Print the sum of even numbers
21 print(f"Sum of even numbers: {even_sum}")
22 # Print the sum of odd numbers
23 print(f"Sum of odd numbers: {odd_sum}")
```

```
PS C:\Aiassisted coding> & 'C:\Users\Poojasree\AppData\Local\Programs\Python\Python313\python.exe' 'C:\Users\Poojasree\vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '53674' '--' 'C:\Aiassisted coding\9.1.2.py'
Sum of even numbers: 30
Sum of odd numbers: 25
PS C:\Aiassisted coding>
```

> Used 1 reference

This code calculates the sum of even and odd numbers in a list:

- `def sum_even_odd(numbers):`
Defines a function that takes a list of numbers.
- `even_sum = 0` and `odd_sum = 0`
Initializes variables to store the sums.
- The `for` loop iterates through each number in the list.
If the number is even (`number % 2 == 0`), it adds to `even_sum`.
Otherwise, it adds to `odd_sum`.
- `return even_sum, odd_sum`
Returns both sums.
- `my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]`
Example list of numbers.
- `even_sum, odd_sum = sum_even_odd(my_list)`
Calls the function and stores the results.
- `print(f"Sum of even numbers: {even_sum}")`
Prints the sum of even numbers.
- `print(f"Sum of odd numbers: {odd_sum}")`
Prints the sum of odd numbers.

give the above code with comments

```
def sum_even_odd(numbers):
```

> Used 1 reference

9.1.2.py X

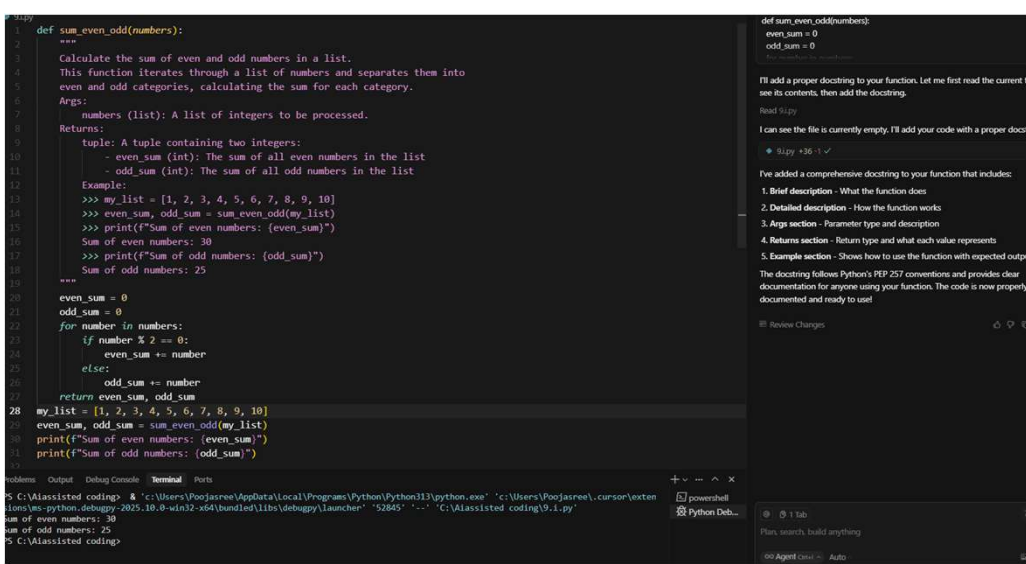
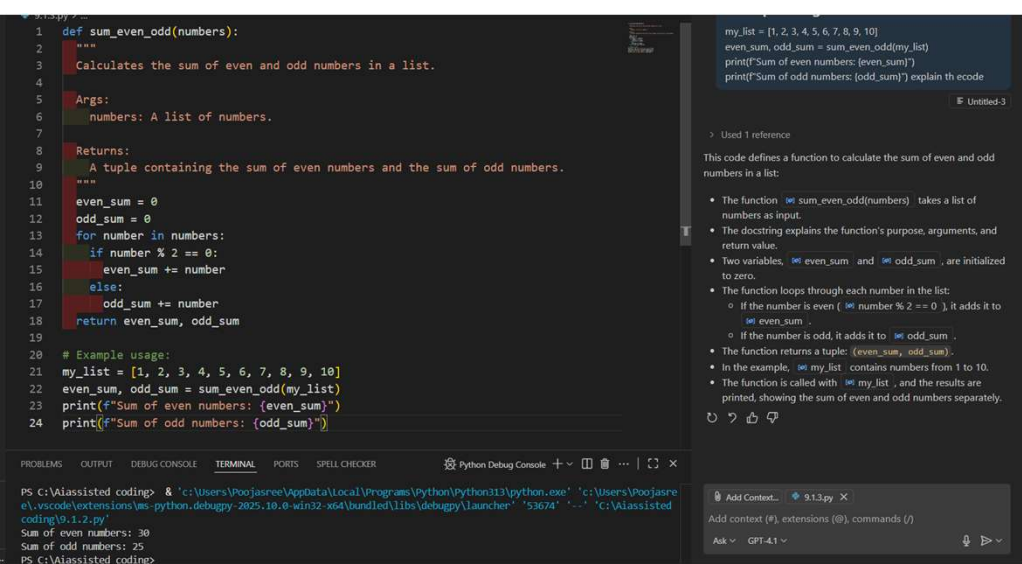
Add context (#), extensions (@), commands (/)

Ask GPT-4.1

Docstring format:

Using git hub copilot:

Using gemini ai:



Task-2:

Prompt: Write python program for **sru_student** class with attributes like name, roll no., hostel_status and **fee_update** method and **display_details** method.).(generate a docstring format)

Code:

```
1 class sru_student:
2     def __init__(self, name, roll_no, hostel_status):
3         self.name = name
4         self.roll_no = roll_no
5         self.hostel_status = hostel_status
6         self.fee_paid = False
7     def fee_update(self, status):
8         self.fee_paid = status
9     def display_details(self):
10        print(f"Name: {self.name}")
11        print(f"Roll No.: {self.roll_no}")
12        print(f"Hostel Status: {self.hostel_status}")
13        print(f"Fee Paid: {self.fee_paid}")
14    # Example usage
15    student1 = sru_student("Alice", 101, "Yes")
16    student1.fee_update(True)
17    student1.display_details()
18    student2 = sru_student("Bob", 102, "No")
19    student2.fee_update(False)
20    student2.display_details()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER

Python Debug Console + - □ □ ... |

```
e:\vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher '62105' '--' 'C:\AIassisted coding\9.2.1.py'
Name: Alice
Roll No.: 101
Hostel Status: Yes
Fee Paid: True
Name: Bob
Roll No.: 102
Hostel Status: No
Fee Paid: False
PS C:\AIassisted coding>
```

Code with inline comments:

```
1 class sru_student:
2     def __init__(self, name, roll_no, hostel_status):
3         self.name = name           # Store student's name
4         self.roll_no = roll_no     # Store student's roll number
5         self.hostel_status = hostel_status # Store hostel status (Yes/No)
6         self.fee_paid = False      # Initialize fee payment status as False
7
8     def fee_update(self, status):
9         self.fee_paid = status     # Update fee payment status
10
11    def display_details(self):
12        print(f"Name: {self.name}") # Print student's name
13        print(f"Roll No.: {self.roll_no}") # Print student's roll number
14        print(f"Hostel Status: {self.hostel_status}") # Print hostel status
15        print(f"Fee Paid: {self.fee_paid}") # Print fee payment status
16
17    # Example usage
18    student1 = sru_student("Alice", 101, "Yes") # Create first student object
19    student1.fee_update(True) # Update fee status to True (paid)
20    student1.display_details() # Display details of first student
21    student2 = sru_student("Bob", 102, "No") # Create second student object
22    student2.fee_update(False) # Update fee status to False (not paid)
23    student2.display_details() # Display details of second student
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER

Python Debug Console + - [] [x] [y] [z] [w] [v] [u] [t] [s] [r] [q] [p] [o] [n] [m] [l] [k] [j] [i] [h] [g] [f] [e] [d] [c] [b] [a] [z] [y] [x] [w] [v] [u] [t] [s] [r] [q] [p] [o] [n] [m] [l] [k] [j] [i] [h] [g] [f] [e] [d] [c] [b] [a]

```
e\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '62105' '--' 'C:\Aiassisted coding\9.2.1.py'
Name: Alice
Roll No.: 101
Hostel Status: Yes
Fee Paid: True
Name: Bob
Roll No.: 102
Hostel Status: No
Fee Paid: False
PS C:\Aiassisted coding>
```


Docstring format:

Using git hub copilot:

```
1 class sru_student:
2     """ Represents an SRU student with name, roll number, hostel status, and fee payment status.
3     Attributes:
4         name (str): Student's name.
5         roll_no (int): Student's roll number.
6         hostel_status (str): Hostel status (Yes/No).
7         fee_paid (bool): Fee payment status.
8     """
9     def __init__(self, name, roll_no, hostel_status):
10         """
11         Initializes a new sru_student instance.
12         Args:
13             name (str): Student's name.
14             roll_no (int): Student's roll number.
15             hostel_status (str): Hostel status (Yes/No)"""
16         self.name = name
17         self.roll_no = roll_no
18         self.hostel_status = hostel_status
19         self.fee_paid = False
20
21     def fee_update(self, status):
22         """
23         Updates the fee payment status.
24         Args:
25             status (bool): True if fee is paid, False otherwise. """
26         self.fee_paid = status
27     def display_details(self):
28         """ Displays the student's details."""
29         print(f"Name: {self.name}")
30         print(f"Roll No.: {self.roll_no}")
31         print(f"Hostel Status: {self.hostel_status}")
32         print(f"Fee Paid: {self.fee_paid}")
33 # Example usage
34 student1 = sru_student("Alice", 101, "Yes")
35 student1.fee_update(True)
36 student1.display_details()
37 student2 = sru_student("Bob", 102, "No")
38 student2.fee_update(False)
39 student2.display_details()
```

Using gemini ai:

```
class sru_student:
    """
    A class to represent a student at SRU (Shri Ramswaroop Memorial University).
    This class manages student information including personal details, hostel status,
    and fee payment status. It provides methods to update fee status and display
    student information.
    Attributes:
        name (str): The name of the student
        roll_no (int): The unique roll number of the student
        hostel_status (str): Whether the student stays in hostel ("Yes" or "No")
        fee_paid (bool): Whether the student has paid their fees (default: False)
    """
    def __init__(self, name, roll_no, hostel_status):
        """
        Initialize a new SRU student object.

        Args:
            name (str): The name of the student
            roll_no (int): The unique roll number assigned to the student
            hostel_status (str): Hostel accommodation status ("Yes" or "No")
        Example:
            >>> student = sru_student("Alice", 101, "Yes")
            >>> print(student.name)
            Alice
        """
        self.name = name
        self.roll_no = roll_no
        self.hostel_status = hostel_status
        self.fee_paid = False

    def fee_update(self, status):
        """
        Update the fee payment status of the student.

        Args:
            status (bool): True if fees are paid, False if not paid
        Example:
            >>> student = sru_student("Alice", 101, "Yes")
            >>> student.fee_update(True)
            >>> print(student.fee_paid)
            True
        """
        self.fee_paid = status

    def display_details(self):
        """
        Display all the details of the student.

        This method prints the student's name, roll number, hostel status,
        and fee payment status in a formatted manner.
        Example:
            >>> student = sru_student("Alice", 101, "Yes")
            >>> student.fee_update(True)
            >>> student.display_details()
            Name: Alice
            Roll No.: 101
            Hostel Status: Yes
            Fee Paid: True
        """
        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", 101, "Yes")
student1.fee_update(True)
student1.display_details()
student2 = sru_student("Bob", 102, "No")
student2.fee_update(False)
student2.display_details()
```

Task-3:

Prompt: Write a Python script with 3–4 functions (e.g., calculator: add, subtract, multiply, divide). (generate a docstring format)

Code:

```
1 def add(a, b):
2     return a + b
3
4 def subtract(a, b):
5     return a - b
6
7 def multiply(a, b):
8     return a * b
9
10 def divide(a, b):
11     if b == 0:
12         return "Error: Division by zero"
13     return a / b
14
15 print(add(10, 5))
16 print(subtract(10, 5))
17 print(multiply(10, 5))
18 print(divide(10, 5))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECK

```
PS C:\Aiassisted coding> & 'c:\Users\Poojasree\AppData\Local\Microsoft\Windows\apps\python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\python.exe'
15
5
50
2.0
PS C:\Aiassisted coding>
```

Code with inline comments:

```
1 def add(a, b):
2     return a + b # Return the sum of a and b
3
4 def subtract(a, b):
5     return a - b # Return the difference of a and b
6
7 def multiply(a, b):
8     return a * b # Return the product of a and b
9
10 def divide(a, b):
11     if b == 0:
12         return "Error: Division by zero" # Handle division by zero
13     return a / b # Return the quotient of a divided by b
14
15 # Example usage
16 print(add(10, 5)) # Output: 15
17 print(subtract(10, 5)) # Output: 5
18 print(multiply(10, 5)) # Output: 50
19 print(divide(10, 5)) # Output: 2.0
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER

```
PS C:\Aiassisted coding> & 'c:\Users\Poojasree\AppData\Local\Programs\Python\Python31
ns\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '61969' '--' '
15
5
50
2.0
PS C:\Aiassisted coding>
```


Docstring format:

Using git hub copilot:

```
1 def add(a, b):
2     """ Returns the sum of a and b.
3     Args:
4         a (int or float): First number.
5         b (int or float): Second number.
6     Returns:
7         int or float: The sum of a and b.
8     """
9     return a + b
10 def subtract(a, b):
11     """ Returns the difference of a and b.
12     Args:
13         a (int or float): First number.
14         b (int or float): Second number.
15     Returns:
16         int or float: The difference of a and b. """
17     return a - b
18 def multiply(a, b):
19     """ Returns the product of a and b.
20     Args:
21         a (int or float): First number.
22         b (int or float): Second number.
23     Returns:
24         int or float: The product of a and b. """
25     return a * b
26 def divide(a, b):
27     """ Returns the quotient of a divided by b.
28     Args:
29         a (int or float): Numerator.
30         b (int or float): Denominator.
31     Returns:
32         int, float, or str: The quotient if b is not zero, otherwise an error message. """
33     if b == 0:
34         return "Error: Division by zero"
35     return a / b #example usage
36 print(add(10, 5))
37 print(subtract(10, 5))
38 print(multiply(10, 5))
39 print(divide(10, 5))
```

Using gemini ai:

```
def add(a, b):
    """ Add two numbers together.
    This function takes two numeric values and returns their sum.
    Args:
        a (int or float): The first number to be added
        b (int or float): The second number to be added
    Returns:
        int or float: The sum of a and b
    Example:
        >>> add(10, 5)
        15
        >>> add(3.5, 2.5)
        6.0 """
    return a + b
def subtract(a, b):
    """ Subtract the second number from the first number.
    This function takes two numeric values and returns their difference.
    Args:
        a (int or float): The number to subtract from (minuend)
        b (int or float): The number to subtract (subtrahend)
    Returns:
        int or float: The difference of a and b (a - b)
    Example:
        >>> subtract(10, 5)
        5
        >>> subtract(3.5, 2.5)
        1.0 """
    return a - b
def multiply(a, b):
    """ Multiply two numbers together.
    This function takes two numeric values and returns their product.
    Args:
        a (int or float): The first number to be multiplied
        b (int or float): The second number to be multiplied
    Returns:
        int or float: The product of a and b
    Example:
        >>> multiply(10, 5)
        50
    Example:
        >>> multiply(3.5, 2.5)
        8.75 """
    return a * b
43 def divide(a, b):
44     """
45     Divide the first number by the second number.
46
47     This function takes two numeric values and returns their quotient.
48     It handles division by zero by returning an error message.
49
50     Args:
51         a (int or float): The dividend (number to be divided)
52         b (int or float): The divisor (number to divide by)
53
54     Returns:
55         int, float, or str: The quotient of a and b, or error message if b is 0
56
57     Example:
58         >>> divide(10, 5)
59         2.0
60         >>> divide(15, 3)
61         5.0
62         >>> divide(10, 0)
63         'Error: Division by zero'
64     """
65     if b == 0:
66         return "Error: Division by zero"
67     return a / b
68
69 print(add(10, 5))
70 print(subtract(10, 5))
71 print(multiply(10, 5))
72 print(divide(10, 5))
```