

# AI Assisted Coding

## Lab Assignment 9.3

Name: J SUJITH

Hall Ticket no: 2303A51327

Batch No: 20

### Task -1:

**Prompt:** Generate a Google-style Python docstring for a function named sum\_even\_odd(numbers) that takes a list of integers as input and returns a tuple containing the sum of even numbers and the sum of odd numbers. Include description, arguments, return values, and example usage.

```
Assignment-9.3.py ●
Assignment-9.3.py > sru_student > _init_
1 #Task-1:
2 def sum_even_odd(numbers):
3     """Calculates the sum of even and odd numbers in a list.
4     Args:
5         numbers (list): A list of integers to process.
6     Returns:
7         tuple: A tuple containing (sum_of_evens, sum_of_odds).
8     Example:
9         >>> sum_even_odd([1, 2, 3, 4, 5, 6])
10        (12, 9)"""
11
12    sum_even = sum(num for num in numbers if num % 2 == 0)
13    sum_odd = sum(num for num in numbers if num % 2 != 0)
14    return sum_even, sum_odd
15
16 # Test the function
17 if __name__ == "__main__":
18     test_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
19     even_sum, odd_sum = sum_even_odd(test_list)
20     print(f"Sum of even numbers: {even_sum}")
21     print(f"Sum of odd numbers: {odd_sum}")
```

### OUTPUT:

```
PS C:\Users\chara\OneDrive\Desktop\Ai-Assisted Coding>
ignment-9.3.py"
Sum of even numbers: 30
Sum of odd numbers: 25
```

### Justification:

The prompt clearly specifies the function name, input type, output format, and required Google-style documentation. This helps the AI generate accurate, structured, and complete docstrings. A well-defined prompt reduces ambiguity and improves the quality of AI-generated documentation.

## Task 2:

**Prompt:** Add clear and meaningful inline comments to the following Python class sru\_student explaining each attribute, constructor, and method functionality. Ensure comments improve readability for beginner developers.

```
Assignment-9.3.py ●
Assignment-9.3.py > ...
22 # Task-2:
23 # Define a class to manage SRU student details
24 class sru_student:
25     """A class to manage student information and fees."""
26     # Constructor to initialize student details
27     def __init__(self, name, roll_no, hostel_status):
28         # Assign student name
29         self.name = name
30         # Assign roll number
31         self.roll_no = roll_no
32         # Store hostel status (Yes/No)
33         self.hostel_status = hostel_status
34         # Initialize fee amount to zero
35         self.fee = 0
36     # Method to update student fee
37     def fee_update(self, amount):
38         # Add given amount to existing fee
39         self.fee += amount
40     # Method to display student details
41     def display_details(self):
42         # Print student name
43         print("Name: " + self.name)
44         # Print roll number
45         print("Roll Number: " + str(self.roll_no))
46         # Print hostel status
47         print("Hostel Status: " + self.hostel_status)
48         # Print total fee
49         print("Fee: " + str(self.fee))
50     # Test the class
51 if __name__ == "__main__":
52     # Create student object
53     student = sru_student("Alice", 101, "Yes")
54     # Update Fee amount
55     student.fee_update(5000)
56     # Display student details
57     student.display_details()
```

## Output:

```
PS C:\Users\chara\OneDrive\Desktop\Ai-Assisted Coding>
ignment-9.3.py"
Name: Alice
Roll Number: 101
Hostel Status: Yes
Fee: 5000
```

**Justification:** The prompt clearly asks the AI to generate inline comments explaining the class structure and methods. Specifying readability and beginner understanding helps produce simple and meaningful comments. This ensures the generated comments improve code clarity and maintainability.

## Task-3:

**Prompt:** Generate a module-level docstring and NumPy-style function-level docstrings for a Python calculator module containing add, subtract, multiply, and divide functions. Include description, parameters, returns, and usage details.

```
Assignment-9.3.py X
Assignment-9.3.py > ...
59 # Task-3:
60 """
61 Calculator Module
62 =====
63 A lightweight calculator module providing basic arithmetic operations.
64 This module is designed for use across multiple projects and provides
65 clean, well-documented functions for common mathematical calculations.
66 Functions
67 -----
68 add : Add two numbers
69 subtract : Subtract two numbers
70 multiply : Multiply two numbers
71 divide : Divide two numbers with error handling
72 Examples
73 -----
74 >>> add(5, 3)
75 8
76 >>> divide(10, 2)
77 5.0
78 """
79 def add(a, b):
80     """
81     Add two numbers.
82     Parameters
83     -----
84     a : float
85         The first number.
86     b : float
87         The second number.
88     Returns
89     -----
90     float
91         The sum of a and b.
92     Examples
93     -----
94     >>> add(5, 3)
95     8
96     """
97     return a + b
98 def subtract(a, b):
99     """
100     Subtract two numbers.
101     Parameters
102     -----
```

```
◆ Assignment-9.3.py ×
◆ Assignment-9.3.py > ...
98     def subtract(a, b):
103         a : float
104             | The minuend.
105         b : float
106             | The subtrahend.
107     Returns
108     -----
109     float
110         | The difference of a and b.
111     Examples
112     -----
113     >>> subtract(10, 4)
114     6
115     """
116     return a - b
117     def multiply(a, b):
118         """
119         Multiply two numbers.
120         Parameters
121         -----
122         a : float
123             | The first factor.
124         b : float
125             | The second factor.
126     Returns
127     -----
128     float
129         | The product of a and b.
130     Examples
131     -----
132     >>> multiply(4, 5)
133     20
134     """
135     return a * b
136     def divide(a, b):
137         """
138         Divide two numbers.
139         Parameters
140         -----
141         a : float
142             | The dividend.
143         b : float
144             | The divisor.
145     Returns
146     -----
```

```
◆ Assignment-9.3.py ×
◆ Assignment-9.3.py > ...
136     def divide(a, b):
137         """
138             float
139                 | The quotient of a and b.
140             Raises
141             -----
142             ValueError
143                 | If b is zero.
144
145             Examples
146             -----
147             >>> divide(10, 2)
148             5.0
149             """
150
151             if b == 0:
152                 raise ValueError("Cannot divide by zero")
153             return a / b
154
155             # Test the functions
156             if __name__ == "__main__":
157                 print(f"Add: {add(10, 5)}")
158                 print(f"Subtract: {subtract(10, 5)}")
159                 print(f"Multiply: {multiply(10, 5)}")
160                 print(f"Divide: {divide(10, 5)}")
```

**Output:**

```
PS C:\Users\chara\OneDrive\Desktop\Ai-Assisted Coding>
ignment-9.3.py"
Add: 15
Subtract: 5
Multiply: 50
Divide: 2.0
```

**Justification:**

The prompt clearly specifies the documentation style, module purpose, and required functions, helping the AI generate structured and accurate documentation. Mentioning NumPy style ensures standardized formatting. This improves readability and consistency across multiple functions.