LAB ASSIGNMENT-9.3

NAME: m.krishnam raju

H.T.No: 2403A52203

Task Description#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.

Expected Outcome#1: Students understand how AI can produce function-level documentation

even_and_odd > sum_even_odd

```python
def sum_even_odd(numbers):
    """
    Calculates the sum of even and odd numbers in a given list.

    Args:
        numbers (list of int): A list of integers to be processed.

    Returns:
        tuple: A tuple containing two integers:
            - The sum of even numbers in the list.
            - The sum of odd numbers in the list.

    Example:
        >>> sum_even_odd([1, 2, 3, 4, 5])
        (6, 9)
    """
    even_sum = 0
    odd_sum = 0
    for num in numbers:
        if num % 2 == 0:
            even_sum += num
        else:
            odd_sum += num
    return even_sum, odd_sum


if __name__ == "__main__":
    user_input = input("Enter numbers separated by spaces: ")
    numbers = [int(x) for x in user_input.strip().split()]
    even_sum, odd_sum = sum_even_odd(numbers)
    print("Sum of even numbers:", even_sum)
    print("Sum of odd numbers:", odd_sum)
```

```
d_odd"
Enter numbers separated by spaces: 1 2 3 4 5 6 7 8 9
Sum of even numbers: 20
Sum of odd numbers: 25
PS C:\Users\M Bharath\Downloads\AI>
```

Manual Docstring:

"""

Calculates the sum of even and odd numbers in the provided input list.

input:

numbers (list of int): A list of integers.

Returns:

tuple: A tuple containing :

- The sum of even numbers.

- The sum of odd numbers.

"""

AI Generated Docstring:

"""

Calculates the sum of even and odd numbers in a given list.

Args:

numbers (list of int): A list of integers to be processed.

Returns:

tuple: A tuple containing two integers:

- The sum of even numbers in the list.

- The sum of odd numbers in the list.

Example:

>>> sum_even_odd([1, 2, 3, 4, 5])

(6, 9)

"""

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.

Expected Output#2: Students critically analyze AI-generated code comments.

Manual Comments:

sru_student.py > ...

```python
#class to store student details
class sru_student:
    #constructor to initialize student details
    def __init__(self, name, roll_no, hostel_status):
        self.name = name
        self.roll_no = roll_no
        self.hostel_status = hostel_status
        self.fee_paid = False
    #method to update fee status
    def fee_update(self, status):
        self.fee_paid = status
    #method to display student details
    def display_details(self):
        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'}")

if __name__ == "__main__":
    student1 = sru_student("Manvith", "2403A52216", "Yes")
    student1.fee_update(True)
    name = input("Enter student name: ")
    roll_no = input("Enter roll number: ")
    hostel_status = input("Hostel status (Yes/No): ")
    fee_paid_input = input("Fee paid? (Yes/No): ")
    fee_paid = fee_paid_input.strip().lower() == "yes"

    student2 = sru_student(name, roll_no, hostel_status)
    student2.fee_update(fee_paid)
    student2.display_details()
```

```
PS C:\Users\M Bharath\Downloads\AI> & "
Enter student name: Manvith
Enter roll number: 2403A52216
Hostel status (Yes/No): Yes
Fee paid? (Yes/No): Yes
Name: Manvith
Roll No.: 2403A52216
Hostel Status: Yes
Fee Paid: Yes
PS C:\Users\M Bharath\Downloads\AI>
```

AI Generated Comments:

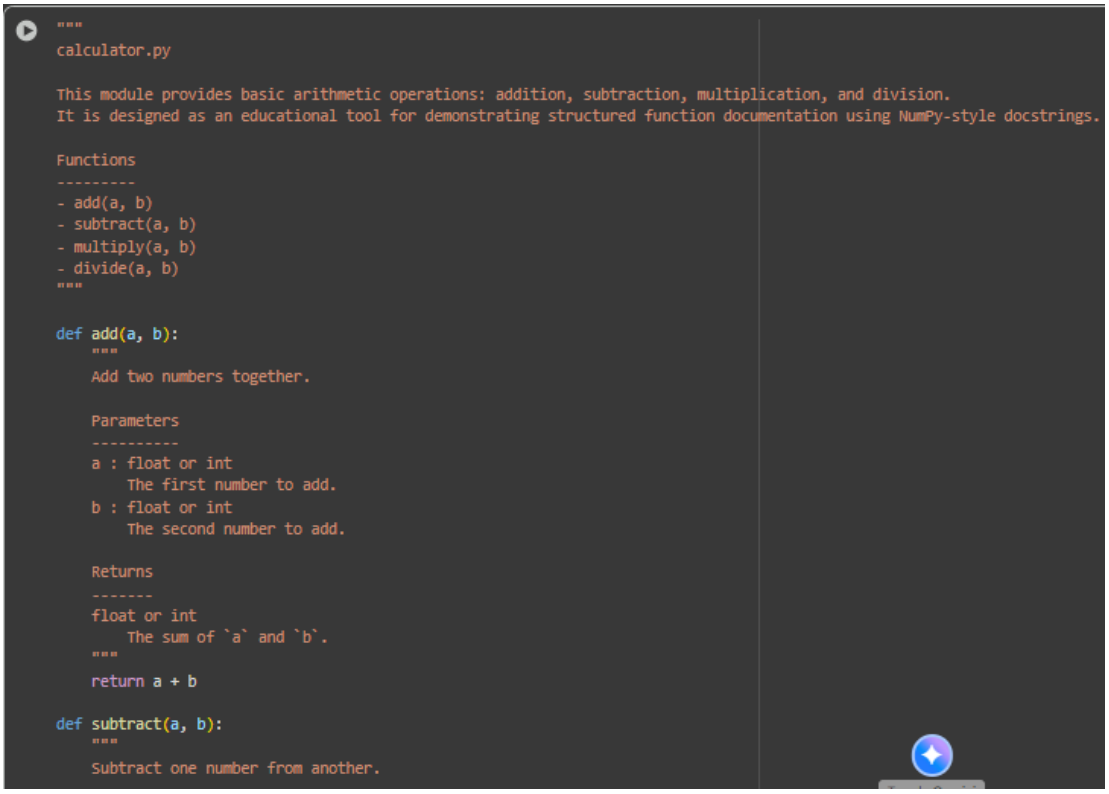
sru_student.py > ...

```python
class sru_student:
    # Constructor to initialize student details and fee status
    def __init__(self, name, roll_no, hostel_status):
        self.name = name
        self.roll_no = roll_no
        self.hostel_status = hostel_status
        self.fee_paid = False  # Default fee status is not paid

    # Method to update the fee payment status
    def fee_update(self, status):
        self.fee_paid = status

    # Method to display all details of the student
    def display_details(self):
        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'}")

if __name__ == "__main__":
    # Example student with hardcoded details
    student1 = sru_student("Manvith", "2403A52216", "Yes")
    student1.fee_update(True)

    # Take dynamic input from user for a new student
    name = input("Enter student name: ")
    roll_no = input("Enter roll number: ")
    hostel_status = input("Hostel status (Yes/No): ")
    fee_paid_input = input("Fee paid? (Yes/No): ")
    fee_paid = fee_paid_input.strip().lower() == "yes"

    # Create a new student object with user input and update fee status
    student2 = sru_student(name, roll_no, hostel_status)
    student2.fee_update(fee_paid)
```

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.

Expected Output#3: Students learn structured documentation for multi-function scripts



```
"""
calculator.py

This module provides basic arithmetic operations: addition, subtraction, multiplication, and division.
It is designed as an educational tool for demonstrating structured function documentation using NumPy-style docstrings.

Functions
---------
- add(a, b)
- subtract(a, b)
- multiply(a, b)
- divide(a, b)
"""

def add(a, b):
    """
    Add two numbers together.

    Parameters
    ----------
    a : float or int
        The first number to add.
    b : float or int
        The second number to add.

    Returns
    -------
    float or int
        The sum of `a` and `b`.
    """
    return a + b

def subtract(a, b):
    """
    Subtract one number from another.
```

```
        Parameters
        ----------
        a : float or int
            The number from which to subtract.
        b : float or int
            The number to subtract.

        Returns
        -------
        float or int
            The result of `a - b`.
        """
        return a - b

    def multiply(a, b):
        """
        Multiply two numbers.

        Parameters
        ----------
        a : float or int
            The first factor.
        b : float or int
            The second factor.

        Returns
        -------
        float or int
            The product of `a` and `b`.
        """
        return a * b

    def divide(a, b):
        """
        Divide one number by another.

        Parameters
```

```
        ----------
        a : float or int
            The numerator.
        b : float or int
            The denominator. Must not be zero.

        Returns
        -------
        float
            The result of `a / b`.

        Raises
        ------
        ZeroDivisionError
            If `b` is zero.
        """
        if b == 0:
            raise ZeroDivisionError("Cannot divide by zero.")
        return a / b

    if __name__ == "__main__":
        print("Simple Calculator")
        print("Available operations: add, subtract, multiply, divide")
        op = input("Enter operation: ").strip().lower()
        try:
            a = float(input("Enter first number: "))
            b = float(input("Enter second number: "))
            if op == "add":
                result = add(a, b)
            elif op == "subtract":
                result = subtract(a, b)
            elif op == "multiply":
                result = multiply(a, b)
            elif op == "divide":
                result = divide(a, b)
            else:
                print("Invalid operation.")
                exit(1)
            print(f"Result: {result}")
        except Exception as e:
            print(f"Error: {e}")
```

```
Simple Calculator
Available operations: add, subtract, multiply, divide
Enter operation: divide
Enter first number: 8
Enter second number: 0
Error: Cannot divide by zero.
```