

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
ProgramName: B. Tech		Assignment Type: Lab	AcademicYear: 2025-2026
CourseCoordinatorName		Venkataramana Veeramsetty	
Instructor(s)Name		Dr. V. Venkataramana (Co-ordinator)	
		Dr. T. Sampath Kumar	
		Dr. Pramoda Patro	
		Dr. Brij Kishor Tiwari	
		Dr.J.Ravichander	
		Dr. Mohammand Ali Shaik	
		Dr. Anirodh Kumar	
		Mr. S.Naresh Kumar	
		Dr. RAJESH VELPULA	
		Mr. Kundhan Kumar	
		Ms. Ch.Rajitha	
		Mr. M Prakash	
		Mr. B.Raju	
		Intern 1 (Dharma teja)	
		Intern 2 (Sai Prasad)	
		Intern 3 (Sowmya)	
NS_2 ( Mounika)			
CourseCode	24CS002PC215	CourseTitle	AI Assisted Coding
Year/Sem	II/I	Regulation	R24
Date and Day of Assignment	Week4 - Wednesday	Time(s)	
Duration	2 Hours	Applicable to Batches	
AssignmentNumber: 9.3(Present assignment number)/24(Total number of assignments)			
Q.No.	Question	Expected Time to complete	
1	Lab 8: Documentation Generation: Automatic documentation and code comments  <b>Lab Objectives:</b> <ul style="list-style-type: none"> <li>To understand the importance of documentation and code comments in software development.</li> <li>To explore how AI-assisted coding tools can generate meaningful documentation and</li> </ul>	Week4 - Wednesday	

inline comments.

- To practice generating function-level and module-level docstrings automatically.
- To evaluate the quality, accuracy, and limitations of AI-generated documentation.
- To develop a small automated tool for documentation generation in Python..

#### Lab Outcomes (LOs):

After completing this lab, students will be able to:

- Apply AI-assisted coding tools to generate docstrings and inline comments for Python code.
- Critically analyze AI-generated documentation for correctness, completeness, and readability.
- Create structured documentation (function-level, module-level) following standard formats.
- Design and implement a mini documentation generator tool to automate code commenting and docstring creation.

#### 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.

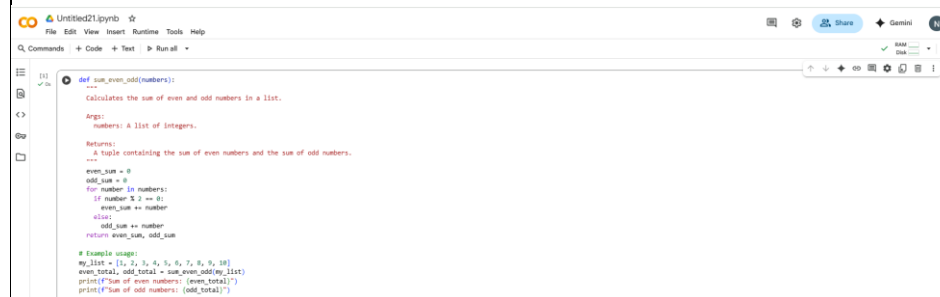
**Expected Outcome#1:** Students understand how AI can produce function-level documentation.

**PROMPT1:**write a python code to return the sum of even and odd numbers in the list using function

**PROMPT2:**take list numbers after the execution

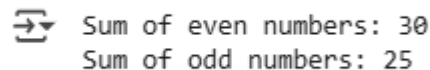
**Prompt3:**generate the docstring describing the code

#### Code:



```
def sum_even_odd(numbers):  
    """  
    Calculates the sum of even and odd numbers in a list.  
  
    Args:  
        numbers: A list of integers.  
  
    Returns:  
        A tuple containing the sum of even numbers and the sum of odd numbers.  
    """  
    even_sum = 0  
    odd_sum = 0  
    for number in numbers:  
        if number % 2 == 0:  
            even_sum += number  
        else:  
            odd_sum += number  
    return even_sum, odd_sum  
  
# Example usage:  
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
even_total, odd_total = sum_even_odd(my_list)  
print(f"Sum of even numbers: {even_total}")  
print(f"Sum of odd numbers: {odd_total}")
```

#### OUTPUT:



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

#### OBSERVATION:

In the above the code in the input I will be some integer numbers that contains all the numbers in the list. Then it will identify the which number is even and which number is off from the list then in the Output it print the even numbers in a list and also the odd

numbers in a list.

### Task Description#2 Automatic Inline Comments

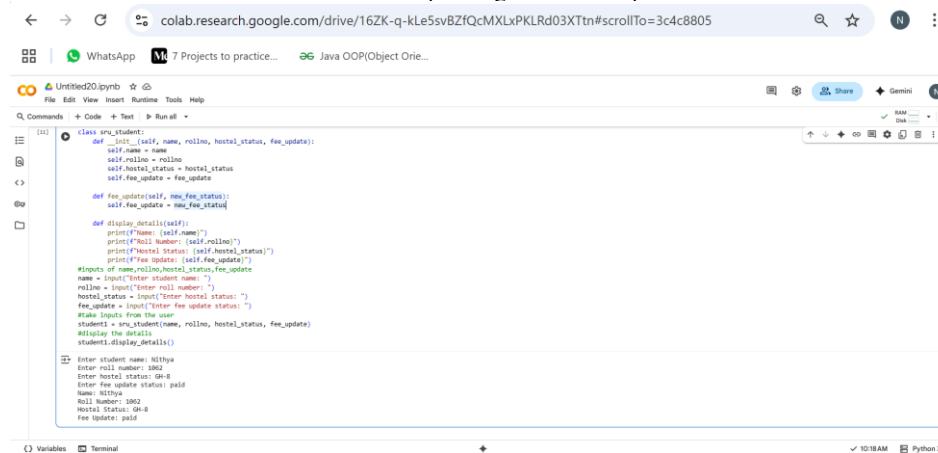
- Write python program for **sru\_student** class with attributes like name, roll no., hostel\_status and **fee\_update** method and **display\_details** method.
- Write comments manually for each line/code block
- Ask an AI tool to add inline comments explaining each line/step.
- Compare the AI-generated comments with your manually written one.

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

PROMPT1: write a python code class sru\_student with attributes name, rollno, hostel\_status, fee\_update method and diplay\_details method.

PROMPT2: Take inputs from the user.

PROMPT3: Please add inline comments explaining each line/step of the code



```
[11] class sru_student:
    def __init__(self, name, rollno, hostel_status, fee_update):
        self.name = name
        self.rollno = rollno
        self.hostel_status = hostel_status
        self.fee_update = fee_update

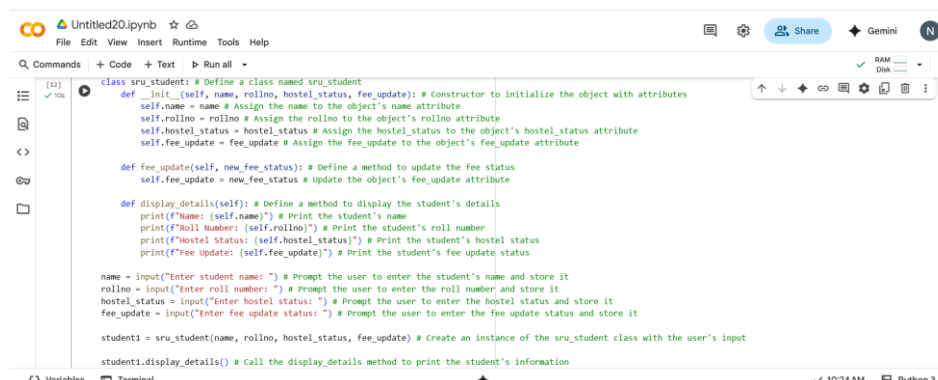
    def fee_update(self, new_fee_status):
        self.fee_update = new_fee_status

    def display_details(self):
        print("Name: (self.name)")
        print("Roll Number: (self.rollno)")
        print("Hostel Status: (self.hostel_status)")
        print("Fee Update: (self.fee_update)")

#inputs of name,rollno,hostel_status,fee_update
name = input("Enter student name: ")
rollno = input("Enter roll number: ")
hostel_status = input("Enter hostel status: ")
fee_update = input("Enter fee update status: ")
#take inputs from the user
student1 = sru_student(name, rollno, hostel_status, fee_update)
#display the details
student1.display_details()

Enter student name: Nithya
Enter roll number: 1062
Enter hostel status: GH-8
Enter fee update status: paid
Name: Nithya
Roll Number: 1062
Hostel Status: GH-8
Fee Update: paid
```

### With ai added inline comments



```
[12] class sru_student: # Define a class named sru_student
    def __init__(self, name, rollno, hostel_status, fee_update): # constructor to initialize the object with attributes
        self.name = name # Assign the name to the object's name attribute
        self.rollno = rollno # Assign the rollno to the object's rollno attribute
        self.hostel_status = hostel_status # Assign the hostel_status to the object's hostel_status attribute
        self.fee_update = fee_update # Assign the fee_update to the object's fee_update attribute

    def fee_update(self, new_fee_status): # Define a method to update the fee status
        self.fee_update = new_fee_status # update the object's fee_update attribute

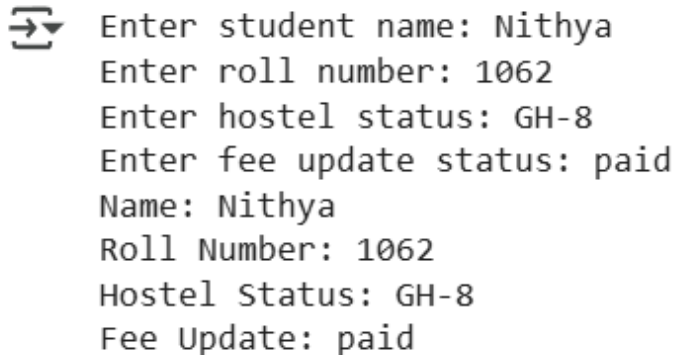
    def display_details(self): # Define a method to display the student's details
        print("Name: (self.name)") # Print the student's name
        print("Roll Number: (self.rollno)") # Print the student's roll number
        print("Hostel Status: (self.hostel_status)") # Print the student's hostel status
        print("Fee Update: (self.fee_update)") # Print the student's fee update status

name = input("Enter student name: ") # Prompt the user to enter the student's name and store it
rollno = input("Enter roll number: ") # Prompt the user to enter the roll number and store it
hostel_status = input("Enter hostel status: ") # Prompt the user to enter the hostel status and store it
fee_update = input("Enter fee update status: ") # Prompt the user to enter the fee update status and store it

student1 = sru_student(name, rollno, hostel_status, fee_update) # Create an instance of the sru_student class with the user's input
student1.display_details() # Call the display_details method to print the student's information

Enter student name: Nithya
Enter roll number: 1062
Enter hostel status: GH-8
Enter fee update status: paid
Name: Nithya
Roll Number: 1062
Hostel Status: GH-8
Fee Update: paid
```

### OUTPUT:



```
Enter student name: Nithya
Enter roll number: 1062
Enter hostel status: GH-8
Enter fee update status: paid
Name: Nithya
Roll Number: 1062
Hostel Status: GH-8
Fee Update: paid
```

### OBSERVATION:

From the above code I observed that I given the code to print the name, rollno, hostel, fee\_update from that it collected the inputs and given the output. And also I written the comments by my own without ai as shown In first screenshoot. Coming to 2<sup>nd</sup> screenshoot it given by ai with inline commenst of each line or also we can say given the code of commenting with the each step.

### 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.

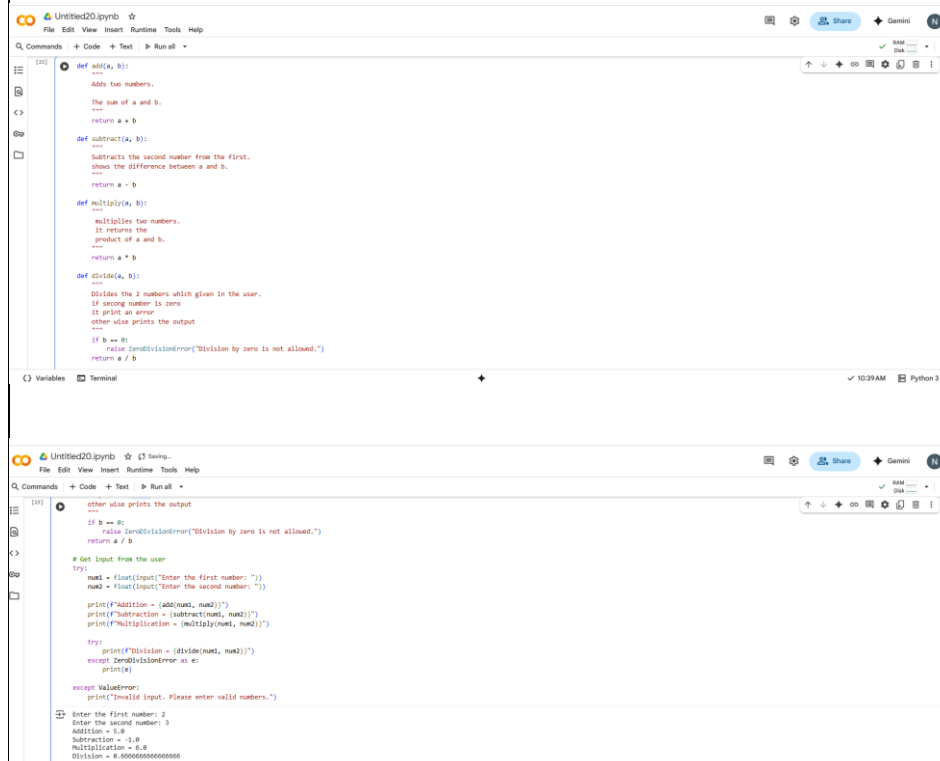
**Expected Output#3:** Students learn structured documentation for multi-function scripts

**PROMPT1:** write a python code to add,subtract,multiply,divide using functions.

**Writing in manual docstring**

**PROMPT2:** Form the above code generate the a module-level docstring + individual function docstrings.

**Code**



The image displays two screenshots of a Python IDE (Untitled20.py) showing code for a calculator. The top screenshot shows the initial code with manual docstrings for four functions: add, subtract, multiply, and divide. The bottom screenshot shows the same code after AI assistance, where the module-level docstring and individual function docstrings have been generated and formatted.

```
[20] def add(a, b):  
    """  
    Adds two numbers.  
    The sum of a and b.  
    """  
    return a + b  
  
def subtract(a, b):  
    """  
    Subtracts the second number from the first.  
    shows the difference between a and b.  
    """  
    return a - b  
  
def multiply(a, b):  
    """  
    multiplies two numbers.  
    It returns the  
    product of a and b.  
    """  
    return a * b  
  
def divide(a, b):  
    """  
    Divides the 2 numbers which given in the user.  
    if second number is zero  
    it print an error  
    other else prints the output  
    """  
    if b == 0:  
        raise ZeroDivisionError("Division by zero is not allowed.")  
    return a / b
```

```
[20] """  
other else prints the output  
"""  
if b == 0:  
    raise ZeroDivisionError("Division by zero is not allowed.")  
return a / b  
  
# Get input from the user  
try:  
    num1 = float(input("Enter the first number: "))  
    num2 = float(input("Enter the second number: "))  
    print("Addition = (add(num1, num2))")  
    print("Subtraction = (subtract(num1, num2))")  
    print("Multiplication = (multiply(num1, num2))")  
  
    try:  
        print("Division = (divide(num1, num2))")  
    except ZeroDivisionError as e:  
        print(e)  
  
except ValueError:  
    print("Invalid Input. Please enter valid numbers.")  
  
Enter the first number: 2  
Enter the second number: 3  
Addition = 5.0  
Subtraction = 1.0  
Multiplication = 6.0  
Division = 0.6666666666666667
```

**From ai generated module-level docstring + individual function docstrings.**

```
def add(a, b):
    """
    This module provides basic arithmetic functions: addition, subtraction,
    multiplication, and division.
    """
    """
    Add two numbers.

    Args:
        a: The first number.
        b: The second number.

    Returns:
        The sum of a and b.
    """
    return a + b

def subtract(a, b):
    """
    Subtract the second number from the first.

    Args:
        a: The number to subtract from.
        b: The number to subtract.

    Returns:
        The difference between a and b.
    """
    return a - b

def multiply(a, b):
    """
    Multiplies two numbers.
    """
    return a * b

def divide(a, b):
    """
    Divides the first number by the second.

    Args:
        a: The dividend.
        b: The divisor.

    Returns:
        The result of the division.

    Raises:
        ZeroDivisionError: If the divisor (b) is zero.
    """
    if b == 0:
        raise ZeroDivisionError("Division by zero is not allowed.")
    return a / b

# Get input from the user
try:
    num1 = float(input("Enter the first number: "))
    num2 = float(input("Enter the second number: "))
    print("Addition = (add(num1, num2))")
    print("Subtraction = (subtract(num1, num2))")
    print("Multiplication = (multiply(num1, num2))")

    try:
        print("Division = (divide(num1, num2))")
    except ZeroDivisionError as e:
        print(e)
except ValueError:
    print("Invalid input. Please enter valid numbers.")
```

## OUTPUT:

```
➡ Enter the first number: 2
Enter the second number: 3
Addition = 5.0
Subtraction = -1.0
Multiplication = 6.0
Division = 0.6666666666666666
```

## OBSERVATION:

From the above task I observed that I written the code with manually comments .the code is to take the two numbers as first number and second number and prints the addition,subtraction,multiplication and division of the two numbers. Then I given my code to generate the module-level docstring + individual function docstrings. Then it given in the understandable and step by step comments .

Push documentation whole workspace as .md file in GitHub Repository

**Note:** Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots