AI-ASSISTED-CODING

Assignment-9.3

2403A51276 Mohammed Ibrahim Hussain

Task Description#1:

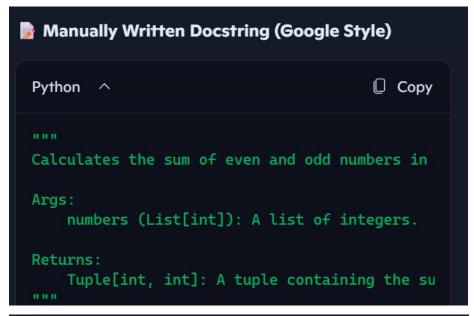
Basic Docstring Generation

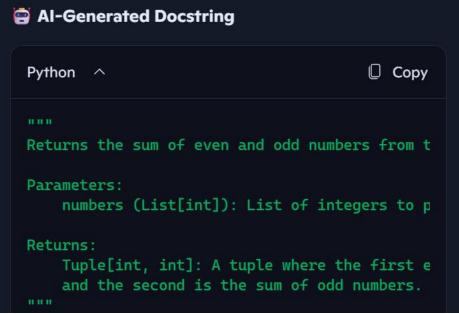
• Write python function to return sum of even and odd numbers in the given list.

```
def sum even odd(numbers):
 2
 3
        Calculates the sum of even and odd
            numbers in a given list.
 4
 5
        Args:
 6
            numbers (List[int]): A list of
                integers.
 7
 8
        Returns:
            Tuple[int, int]: A tuple
 9
                containing the sum of even
                numbers and the sum of odd
                numbers.
10
11
        even_sum = sum(num for num in numbers
            if num \% 2 == 0)
        odd_sum = sum(num for num in numbers
12
            if num % 2 != 0)
        return even_sum, odd_sum
```

• 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.

Aspect	Manual Docstring	Al-Generated Docstring
Clarity	Clear and concise	Also clear, slightly more conversational
Structure	Strict Google Style	Mostly Google Style, but uses "Parameters"
Detail	Precise and formal	Slightly more casual but informative
Terminology	Uses "Args" and "Returns"	Uses "Parameters" instead of "Args"

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.

```
class sru_student:
 2
 3
        def __init__(self, name, roll_no, hostel_status):
4
            self.name = name # Student's name
 5
            self.roll_no = roll_no # Student's roll number
6
            self.hostel_status = hostel_status # Whether the student stays
7
            self.fee_paid = False # Initial fee status set to unpaid
8
9
        def fee_update(self, status):
10 -
11
            self.fee_paid = status # Update fee status based on input
12
13
14 -
        def display_details(self):
15
            print(f"Name: {self.name}") # Print student's name
16
            print(f"Roll No: {self.roll_no}") # Print roll number
17
            print(f"Hostel Status: {self.hostel_status}") # Print hostel
18
            print(f"Fee Paid: {'Yes' if self.fee paid else 'No'}") # Print
```

• Write comments manually for each line/code block

```
class sru student:
 2
3 -
        def __init__(self, name, roll_no, hostel_status):
4
            self.name = name # Assign the student's name
5
            self.roll_no = roll_no # Assign the student's roll number
6
            self.hostel_status = hostel_status # Assign hostel status (Yes
 7
            self.fee_paid = False # Initialize fee status to unpaid
8
 9 -
        def fee_update(self, status):
            self.fee_paid = status # Set the fee status to the given value
10
11
        def display_details(self):
12 -
13
            print(f"Name: {self.name}") # Display the student's name
            print(f"Roll No: {self.roll_no}") # Display the roll number
14
15
            print(f"Hostel Status: {self.hostel_status}") # Display hostel
16
            print(f"Fee Paid: {'Yes' if self.fee_paid else 'No'}") #
```

• Compare the AI-generated comments with your manually written one

Line/Block	Manual Comment	Al-Generated Comment
class sru_student:	No comment	"Define a class to represent a student"
init method	"Constructor to initialize"	"Initialize student attributes" (implied)
self.name = name	"Student's name"	"Assign the student's name"
self.fee_paid = False	"Initial fee status set to unpaid"	"Initialize fee status to unpaid"
fee_update method	"Method to update fee status"	"Set the fee status to the given value"
display_details method	"Method to display student details"	"Display the student's" (per line)

Task Description#3:

• Write a Python script with 3-4 functions (e.g., calculator: add, subtract, multiply,

```
2 # Manual Module Docstring
        4
         This module provides basic arithmetic operations: addition, subtraction,
         multiplication, and division. Each function takes two numeric inputs and
          returns the result of the operation.
        8
        9 - Functions:
       10
             add(a, b)
       11
       12
             multiply(a, b)
       13
       14
       15
       16 - def add(a, b):
       17
       18
       19
       20
              Parameters
       21
       22
divide). 23
```

```
47
        float or int
48
            The result of a - b.
49
50
        return a - b
51
52 def multiply(a, b):
53
54
        Multiply two numbers.
55
56
        Parameters
57
58
        a : float or int
           The first number.
59
60
       b : float or int
61
            The second number.
62
63
        Returns
64
65
        float or int
66
            The product of a and b.
67
68
        return a * b
23
             The first number.
24
         b : float or int
25
             The second number.
26
27
         Returns
28
29
         float or int
30
            The sum of a and b.
31
32
         return a + b
33
34 def subtract(a, b):
35
36
         Subtract one number from another.
37
38
         Parameters
39
40
         a : float or int
41
             The number to subtract from.
42
         b : float or int
43
             The number to subtract.
44
45
         Returns
```

```
70 def divide(a, b):
 71
 72
        Divide one number by another.
 73
 74
        Parameters
 75
 76
        a : float or int
 77
            The numerator.
 78
        b : float or int
 79
            The denominator.
 80
 81
        Returns
 82
 83
        float
 84
            The result of a / b.
 85
 86
        Raises
 87
        ValueError
 88
 89
            If b is zero.
 90
 91
        if b == 0:
 92
            raise ValueError("Cannot divide by zero.")
91 -
          if b == 0:
92
                raise ValueError("Cannot divide by zero.")
```

```
92 raise ValueError("Cannot divide by zero.")
93 return a / b
```

• Incorporate manual docstring in code with

Numpy

```
2 A simple calculator module that provides functions to perform basic arithmetic
   addition, subtraction, multiplication, and division.
 3
 4
 5
 6 def add(a, b):
        """Returns the sum of two numbers."""
 8
        return a + b
 9
10 def subtract(a, b):
        """Returns the difference between two numbers."""
11
12
        return a - b
13
14 def multiply(a, b):
        """Returns the product of two numbers."""
15
16
        return a * b
17
18 def divide(a, b):
19
        """Returns the result of dividing a by b. Raises an error if b is zero."""
20 -
            raise ValueError("Cannot divide by zero.")
21
        return a / b
22
```

• Use AI assistance to generate a module-level docstring + individual function docstrings.

```
1
    A basic calculator module that provides functions for performing
2
        arithmetic operations.
3
    This module includes functions to add, subtract, multiply, and divide
4
        two numeric values.
    Each function accepts two inputs and returns the result of the
 5
        corresponding operation.
6
 7
    Functions
8
9
    add(a, b)
10
        Returns the sum of two numbers.
11
12
    subtract(a, b)
        Returns the difference between two numbers.
13
14
15
   multiply(a, b)
16
        Returns the product of two numbers.
17
18
    divide(a, b)
19
        Returns the quotient of two numbers, with error handling for
            division by zero.
```

• Compare the AI-generated docstring with your manually written one

Feature	Manual Docstring (NumPy Style)	Al-Generated Docstring
Format	Structured with headers (Parameters , Returns)	Simple one-liners
Detail Level	Explicit types, error handling, and descriptions	Concise and readable
Consistency	Uniform across all functions	Varies slightly in phrasing
Style Compliance	Fully adheres to NumPy Style	More casual, not style-specific
Module Docstring	Lists functions and purpose	General overview only