**AI ASSISTED CODING **

*Lab 10: Code Review and Quality: Using AI to improve code quality and readability*

Name: D.Nagamrutha

Roll no: 2503A51L06

**Task Description#1 AI-Assisted Code Review (Basic Errors)**

* Write python program as shown below.
* A screenshot of a computer program

  AI-generated content may be incorrect.Use an AI assistant to review and suggest corrections

**Code Generated :**

A screenshot of a computer program

AI-generated content may be incorrect.

**Output :**

A screenshot of a computer

AI-generated content may be incorrect.

**Corrections made in the code:**

* The loop for i in range(1, n): is incorrect. It should be range(1, n + 1) to include n itself.
* Variable x = 0 in calcFact() is never used.
* In main(), num is defined, but passed to calcFact() as num, yet the function is called before num is declared as a parameter.
* The variable t = 10 is declared but unused.
* The if FACT > 10: is valid, but t being declared above has no effect.
* Use proper naming conventions (e.g., fact instead of FACT).
* Ensure consistent indentation and spacing for readability.

**Task Description#2 Automatic Inline Comments**

* Write the Python code for Fibonacci as shown below and execute.
* Ask AI to improve variable names, add comments, and apply PEP8 formatting (cleaned up).
* Students evaluate which suggestions improve readability most. one.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.**Code Generated:**

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.

**Observation:**

This Python program generates the Fibonacci series up to a specified number of terms. It defines a function generate\_fibonacci(n\_terms) that initializes the first two numbers (0 and 1) and uses a loop to compute the next values by summing the last two numbers, storing them in a list. The main() function sets the number of terms (10 in this case), calls the generator, and prints the series. While the logic works, there is a small issue: because the loop runs with the condition count <= n\_terms, the program produces one extra term, resulting in n\_terms + 1 values instead of exactly n\_terms. Changing the condition to count < n\_terms would fix this.

**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

A screenshot of a computer program

AI-generated content may be incorrect.

* Use AI assistance to generate a module-level docstring + individual function docstrings.
* Compare the AI-generated docstring with your manually written one.

A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.

**Observation:**

The manual docstrings in your code are simple and give a one-line explanation of each function, but they lack details such as parameter types and return values. In contrast, AI-generated docstrings using the NumPy style are more structured and professional, clearly defining parameters, return types, and edge cases, which improves readability and maintainability. While manual docstrings are sufficient for small scripts, NumPy-style docstrings make the code easier to understand and more useful in larger or collaborative projects.