# 2303A51797

# Batch : 26

Task 1: Auto-Generating Function Documentation in a Shared

Codebase

Scenario

You have joined a development team where several utility functions are

already implemented, but the code lacks proper documentation. New

team members are struggling to understand how these functions should

be used.

Task Description

You are given a Python script containing multiple functions without any

docstrings.

Using an AI-assisted coding tool:

• Ask the AI to automatically generate Google-style function

docstrings for each function

• Each docstring should include:

o A brief description of the function

o Parameters with data types

o Return values

o At least one example usage (if applicable)

Experiment with different prompting styles (zero-shot or context-based)

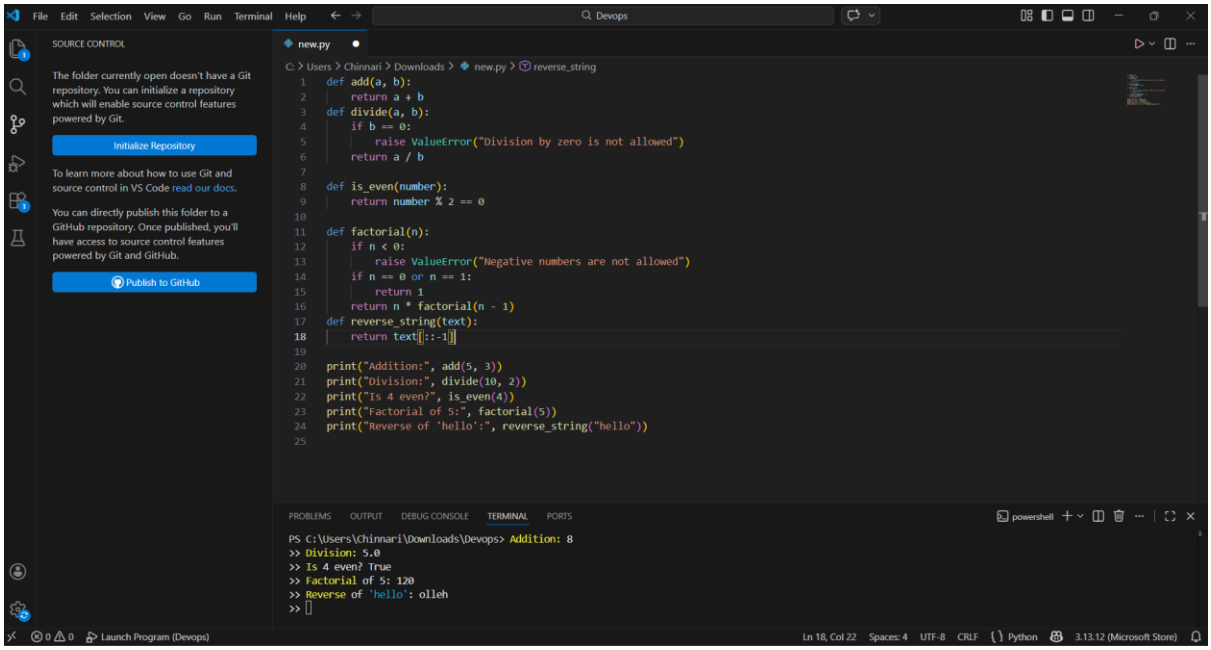
to observe quality differences.

Expected Outcome

• A Python script with well-structured Google-style docstrings

• Docstrings that clearly explain function behavior and usage

Improved readability and usability of the codebase  give code for ths question for vs code




Task 2: Enhancing Readability Through AI-Generated Inline Comments

Scenario

A Python program contains complex logic that works correctly but is difficult to understand at first glance. Future maintainers may find it hard to debug or extend this code.

Task Description

You are provided with a Python script containing:

• Loops

• Conditional logic

• Algorithms (such as Fibonacci sequence, sorting, or searching)

Use AI assistance to:

• Automatically insert inline comments only for complex or non-obvious logic

• Avoid commenting on trivial or self-explanatory syntax

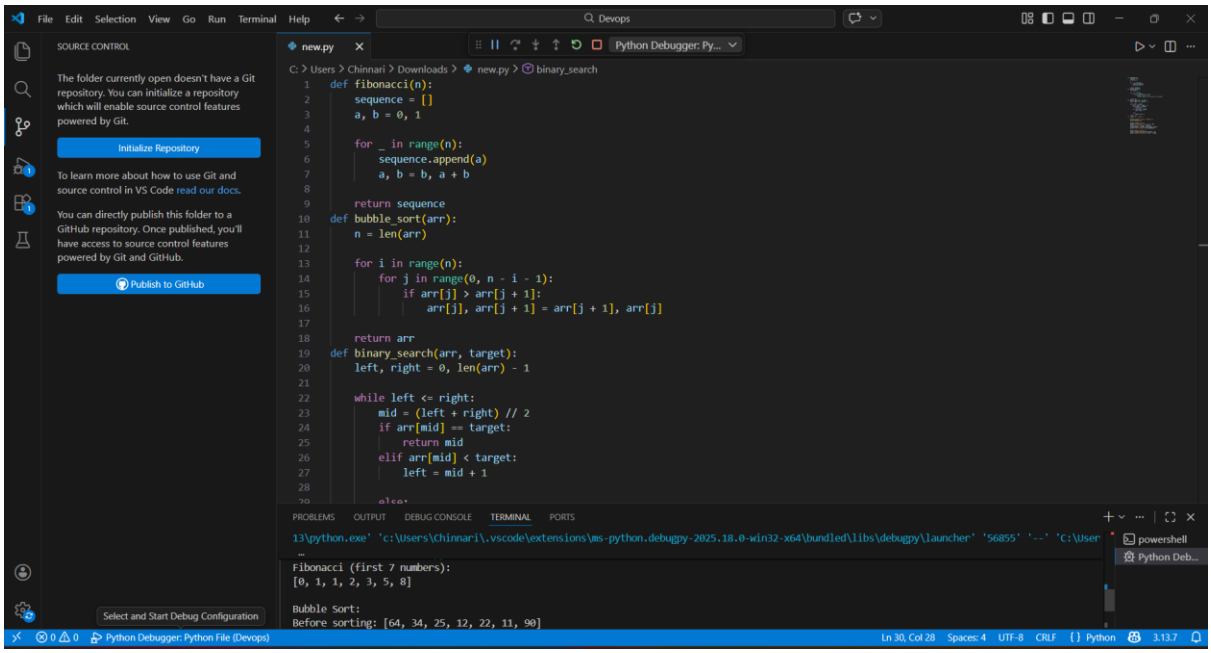
The goal is to improve clarity without cluttering the code.

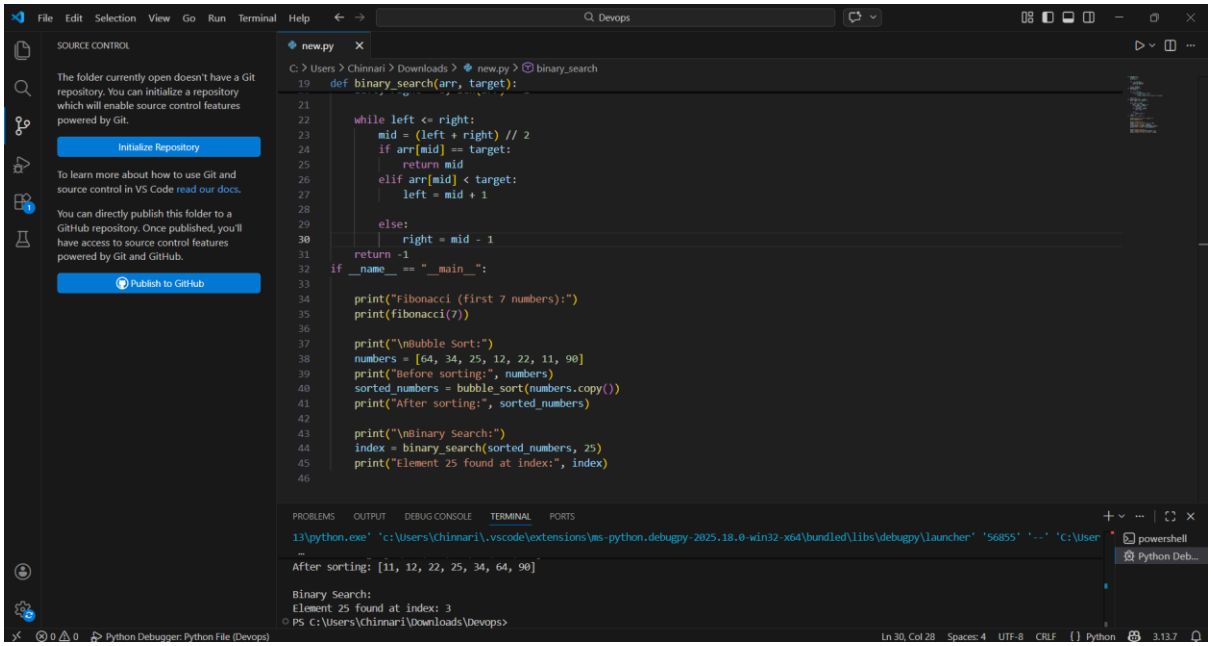
Expected Outcome

• A Python script with concise, meaningful inline comments

• Comments that explain why the logic exists, not what Python syntax does

• Noticeable improvement in code readability  i want code for this with output




Task 3: Generating Module-Level Documentation for a Python

Package

Scenario

Your team is preparing a Python module to be shared internally (or uploaded to a repository). Anyone opening the file should immediately understand its purpose and structure.

Task Description

Provide a complete Python module to an AI tool and instruct it to automatically generate a module-level docstring at the top of the file that includes:

• The purpose of the module

• Required libraries or dependencies

• A brief description of key functions and classes

• A short example of how the module can be used

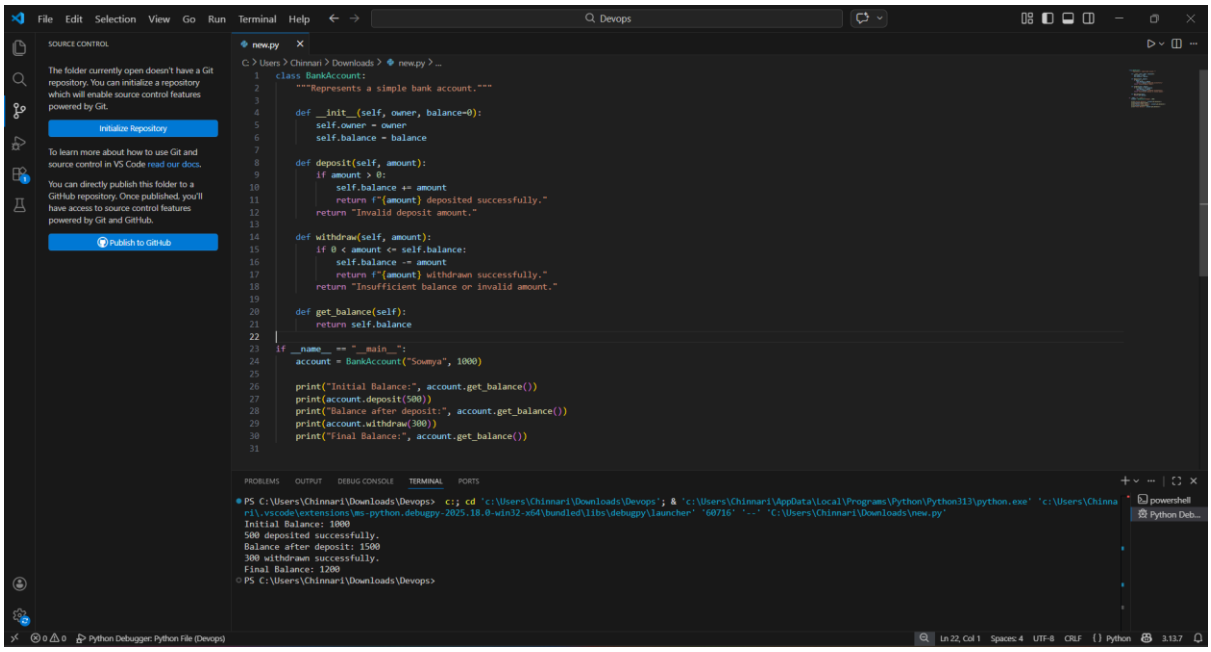
Focus on clarity and professional tone

Expected Outcome

• A well-written multi-line module-level docstring

• Clear overview of what the module does and how to use it

• Documentation suitable for real-world projects or repositories




Task 4: Converting Developer Comments into Structured Docstrings

Scenario

In a legacy project, developers have written long explanatory comments inside functions instead of proper docstrings. The team now wants to standardize documentation.

Task Description

You are given a Python script where functions contain detailed inline comments explaining their logic.

Use AI to:

• Automatically convert these comments into structured Google-

style or NumPy-style docstrings

• Preserve the original meaning and intent of the comments

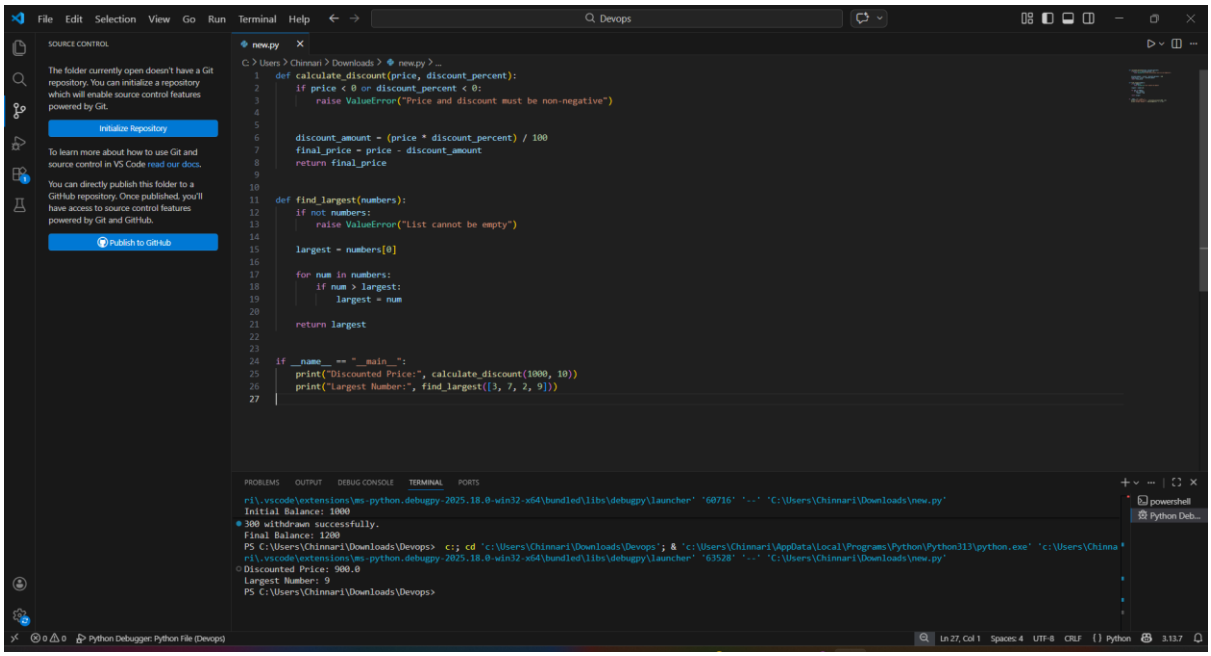
• Remove redundant inline comments after conversion

Expected Outcome

• Functions with clean, standardized docstrings

• Reduced clutter inside function bodies

• Improved consistency across the codebase  i want code for this with output




Task 5: Building a Mini Automatic Documentation Generator

Scenario

Your team wants a simple internal tool that helps developers start

documenting new Python files quickly, without writing documentation

from scratch.

Task Description

Design a small Python utility that:

• Reads a given .py file

• Automatically detects:

o Functions

o Classes

• Inserts placeholder Google-style docstrings for each detected

function or class

AI tools may be used to assist in generating or refining this utility.
Note: The goal is documentation scaffolding, not perfect

documentation.

Expected Outcome

• A working Python script that processes another .py file

• Automatically inserted placeholder docstrings

• Clear demonstration of how AI can assist in documentation

automation  i want code for this with output