Assignment 9.3(Ai Assisted Coding)

Htno:2303a51305

Btno:05

Task 1:

Basic Docstring Generation

Scenario

You are developing a utility function that processes numerical lists and must be properly documented for future maintenance.

Requirements

• Write a Python function to return the sum of even numbers and sum of odd numbers in a given list

• Manually add a Google Style docstring to the function

• Use an AI-assisted tool (Copilot / Cursor AI) to generate a function-level docstring

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

• Analyze clarity, correctness, and completeness

Expected Output

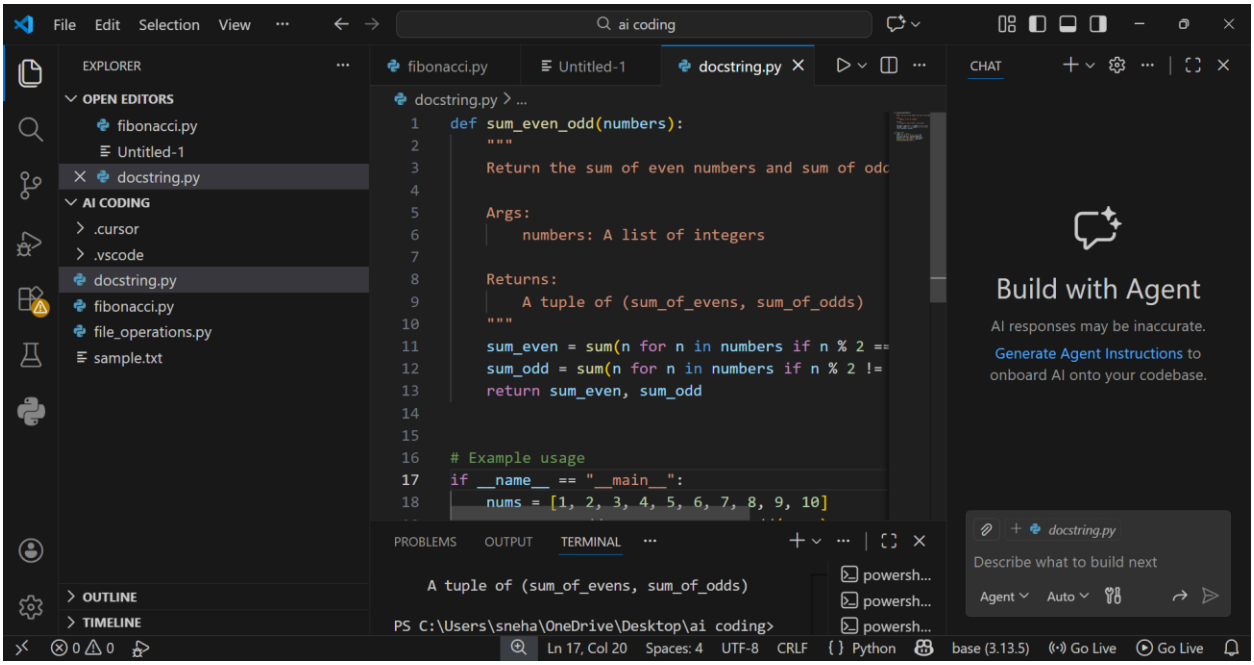
• Python function with manual Google-style docstring

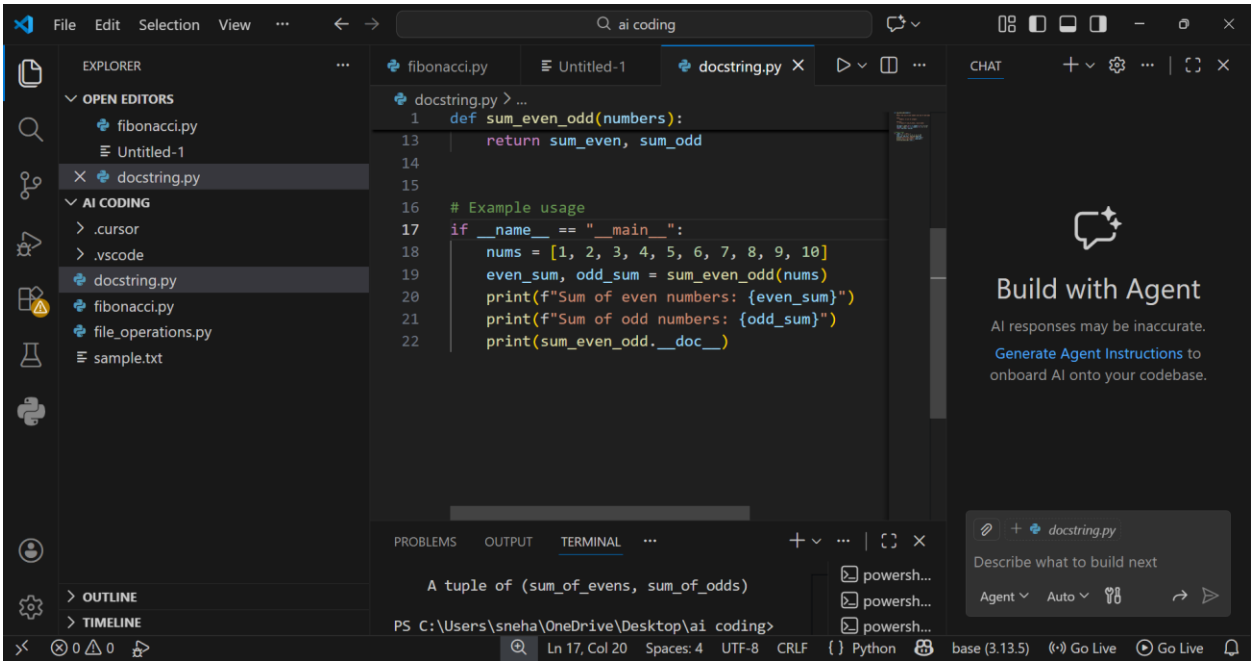
• AI-generated docstring for the same function

• Comparison explaining differences between manual and AI-generated documentation

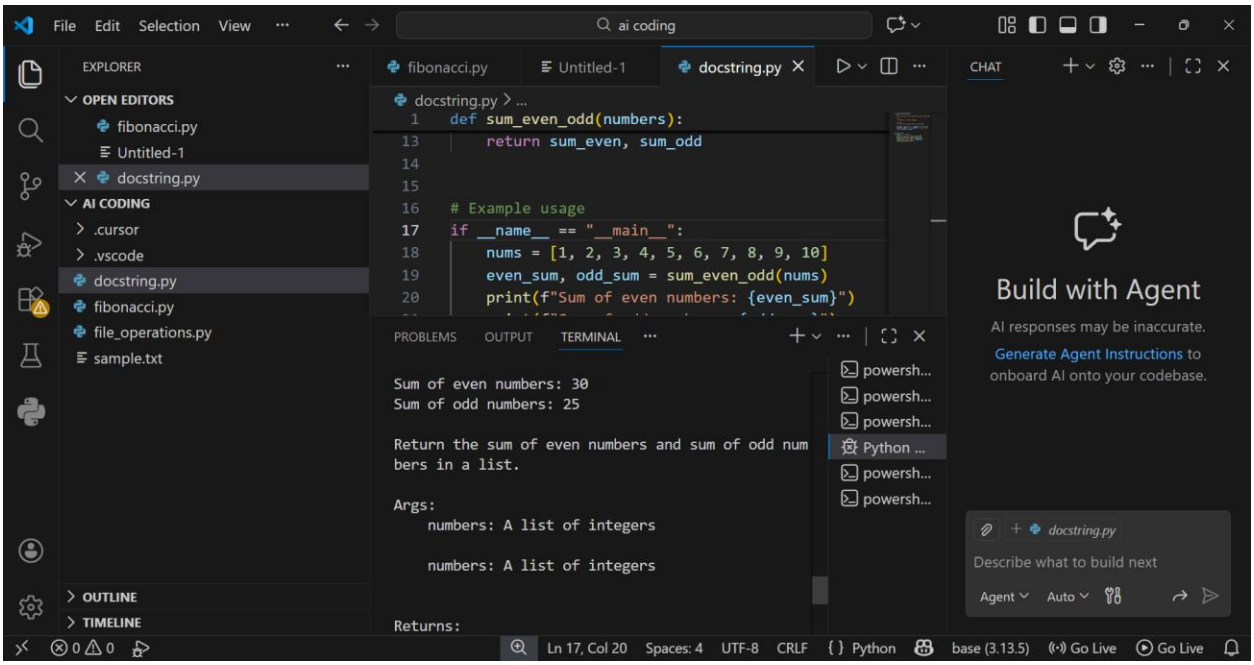
• Improved understanding of AI-generated function-level documentation.

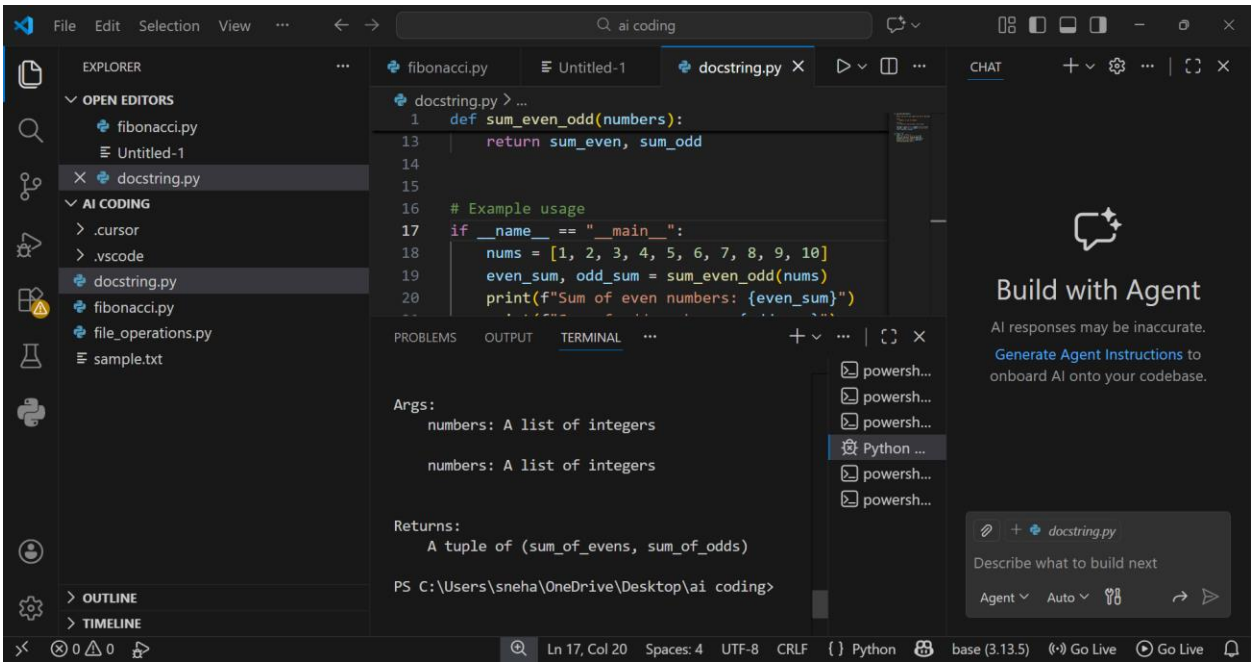
Code:


File Edit Selection View
ai coding
EXPLORER
OPEN EDITORS
fibonacci.py
Untitled-1
docstring.py
AI CODING
.cursor
.vscode
docstring.py
fibonacci.py
file_operations.py
sample.txt
OUTLINE
TIMELINE
fibonacci.py
Untitled-1
docstring.py
docstring.py > ...
def sum_even_odd(numbers):
"""
Return the sum of even numbers and sum of odc
Args:
numbers: A list of integers
Returns:
A tuple of (sum_of_evens, sum_of_odds)
"""
sum_even = sum(n for n in numbers if n % 2 ==
sum_odd = sum(n for n in numbers if n % 2 !=
return sum_even, sum_odd
# Example usage
if __name__ == "__main__":
nums = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
PROBLEMS OUTPUT TERMINAL
powersh...
powersh...
powersh...
A tuple of (sum_of_evens, sum_of_odds)
PS C:\Users\sneha\OneDrive\Desktop\ai coding>
CHAT
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
docstring.py
Describe what to build next
Agent
Auto
Ln 17, Col 20
Spaces: 4
UTF-8
CRLF
Python
base (3.13.5)
Go Live
Go Live


File Edit Selection View
ai coding
EXPLORER
OPEN EDITORS
fibonacci.py
Untitled-1
docstring.py
AI CODING
.cursor
.vscode
docstring.py
fibonacci.py
file_operations.py
sample.txt
OUTLINE
TIMELINE
fibonacci.py
Untitled-1
docstring.py
docstring.py > ...
def sum_even_odd(numbers):
return sum_even, sum_odd
# Example usage
if __name__ == "__main__":
nums = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
even_sum, odd_sum = sum_even_odd(nums)
print(f"Sum of even numbers: {even_sum}")
print(f"Sum of odd numbers: {odd_sum}")
print(sum_even_odd.__doc__)
PROBLEMS OUTPUT TERMINAL
powersh...
powersh...
powersh...
A tuple of (sum_of_evens, sum_of_odds)
PS C:\Users\sneha\OneDrive\Desktop\ai coding>
CHAT
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
docstring.py
Describe what to build next
Agent
Auto
Ln 17, Col 20
Spaces: 4
UTF-8
CRLF
Python
base (3.13.5)
Go Live
Go Live

Output:





Explanation:

-->The manual docstring is more detailed and follows proper Google style. It clearly explains the input type, return values, and includes an example, making it easier to understand.

-->The AI-generated docstring is correct but shorter. It provides basic information but lacks detailed explanation and an example.

Conclusion: AI saves time, but manual improvement makes documentation clearer and more complete.

Task 2: Automatic Inline Comments

Scenario

You are developing a student management module that must be easy to understand for new

developers.

Requirements

• Write a Python program for an sru_student class with the following:

– Attributes: name, roll_no, hostel_status

– Methods: fee_update() and display_details()

• Manually write inline comments for each line or logical block

• Use an AI-assisted tool to automatically add inline comments

• Compare manual comments with AI-generated comments

• Identify missing, redundant, or incorrect AI comments

Expected Output

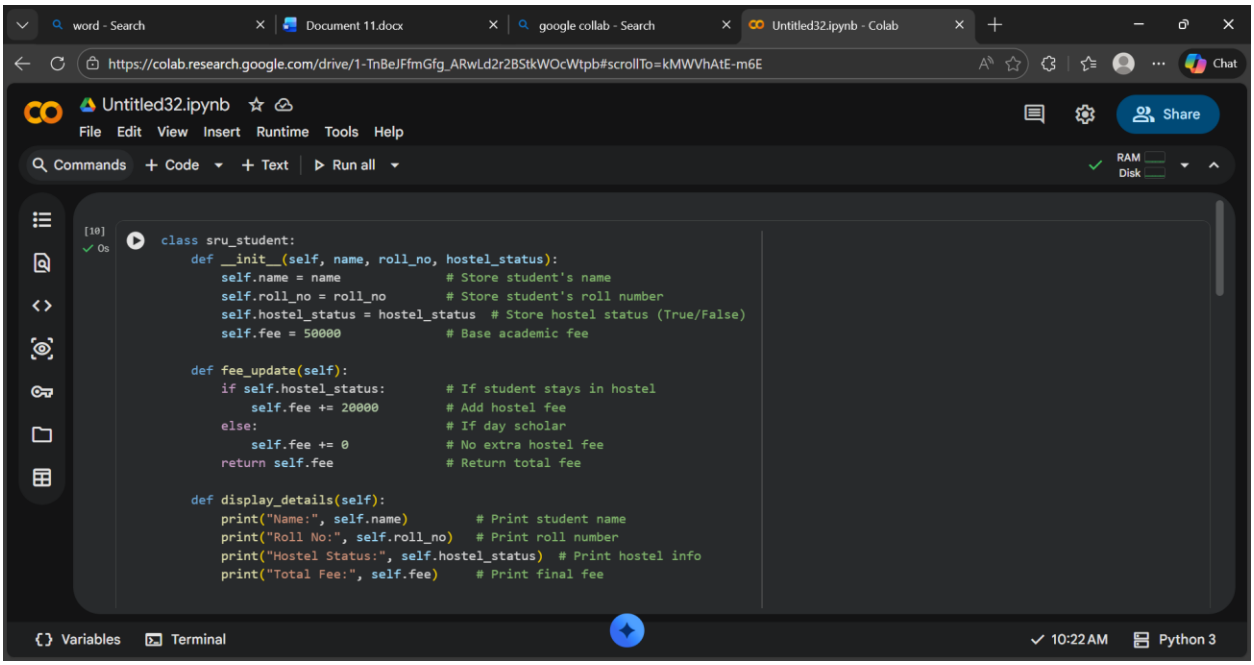
• Python class with manually written inline comments

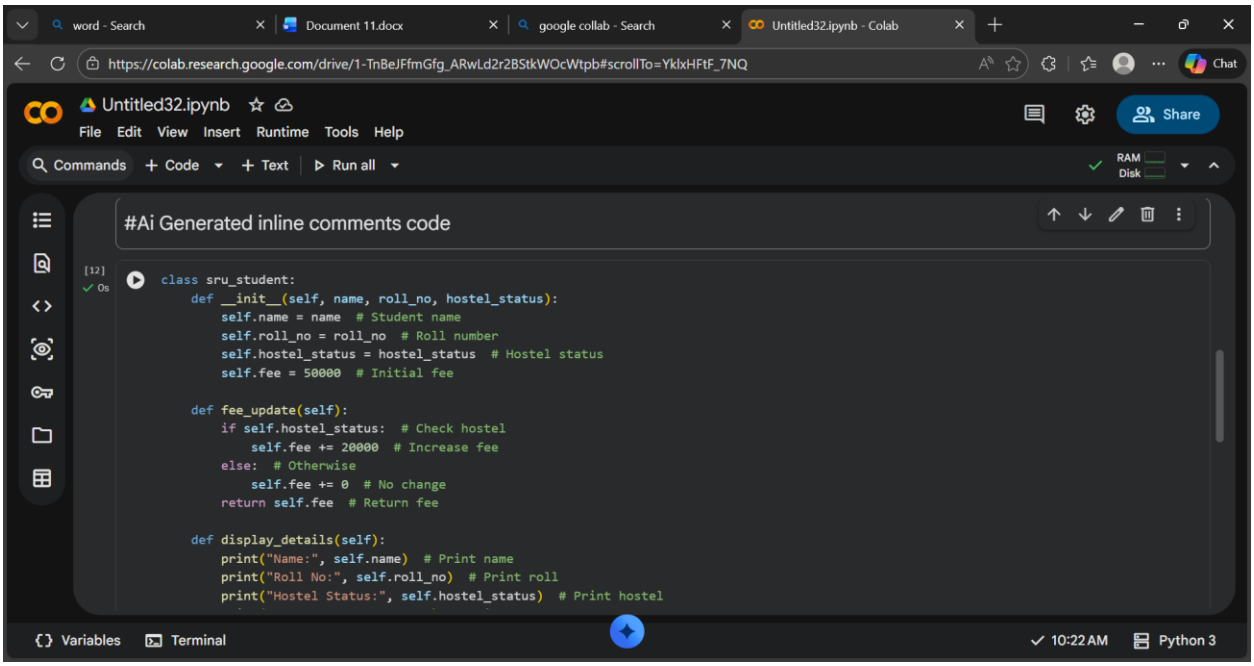
• AI-generated inline comments added to the same code

• Comparative analysis of manual vs AI comments

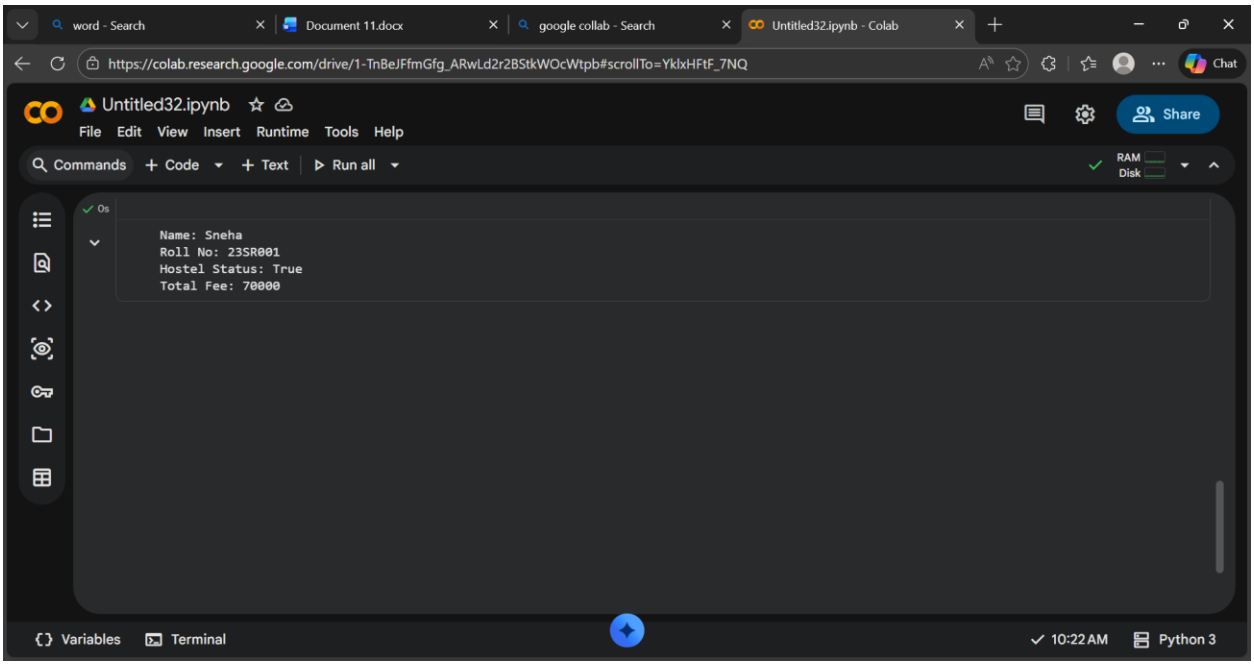
• Critical discussion on strengths and limitations of AI-generated comments

Code:


word - Search
Document 11.docx
google collab - Search
Untitled32.ipynb - Colab
https://colab.research.google.com/drive/1-TnBeJFfmGfg_ARwLd2r2BStkWOcWtpb#scrollTo=kMWVhAtE-m6E
Untitled32.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM Disk
[10] 0s
class sru_student:
def __init__(self, name, roll_no, hostel_status):
self.name = name # Store student's name
self.roll_no = roll_no # Store student's roll number
self.hostel_status = hostel_status # Store hostel status (True/False)
self.fee = 50000 # Base academic fee
def fee_update(self):
if self.hostel_status: # If student stays in hostel
self.fee += 20000 # Add hostel fee
else:
self.fee += 0 # If day scholar
return self.fee # Return total fee
def display_details(self):
print("Name:", self.name) # Print student name
print("Roll No:", self.roll_no) # Print roll number
print("Hostel Status:", self.hostel_status) # Print hostel info
print("Total Fee:", self.fee) # Print final fee
Variables Terminal
10:22 AM Python 3


word - Search
Document 11.docx
google collab - Search
Untitled32.ipynb - Colab
https://colab.research.google.com/drive/1-TnBeJFfmGfg_ARwLd2r2BStkWOcWtpb#scrollTo=YklxHFtF_7NQ
Untitled32.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM Disk
#Ai Generated inline comments code
[12] 0s
class sru_student:
def __init__(self, name, roll_no, hostel_status):
self.name = name # Student name
self.roll_no = roll_no # Roll number
self.hostel_status = hostel_status # Hostel status
self.fee = 50000 # Initial fee
def fee_update(self):
if self.hostel_status: # Check hostel
self.fee += 20000 # Increase fee
else: # Otherwise
self.fee += 0 # No change
return self.fee # Return fee
def display_details(self):
print("Name:", self.name) # Print name
print("Roll No:", self.roll_no) # Print roll
print("Hostel Status:", self.hostel_status) # Print hostel
Variables Terminal
10:22 AM Python 3

Output:



Explanation:

Comparison: Manual vs AI Comments

Feature	Manual Comments	AI Comments	Analysis
Detail Level	More descriptive	Short/basic	AI lacks detailed explanation
Clarity	Explains purpose clearly	Basic meaning only	Manual helps beginners more
Redundancy	Minimal	Some redundant (e.g., “Print name”)	AI often states obvious
Logic Explanation	Explains fee logic clearly	Just “increase fee”	AI misses context
Completeness	Covers all important logic	Covers lines but shallow	AI comments are surface-level

Task 3:

Module-Level and Function-Level Documentation

Scenario

You are building a small calculator module that will be shared across multiple projects and

requires structured documentation.

Requirements

• Write a Python script containing 3–4 functions (e.g., add, subtract, multiply, divide)

• Manually write NumPy Style docstrings for each function

• Use AI assistance to generate:

– A module-level docstring

– Individual function-level docstrings

• Compare AI-generated docstrings with manually written ones

• Evaluate documentation structure, accuracy, and readability

Expected Output

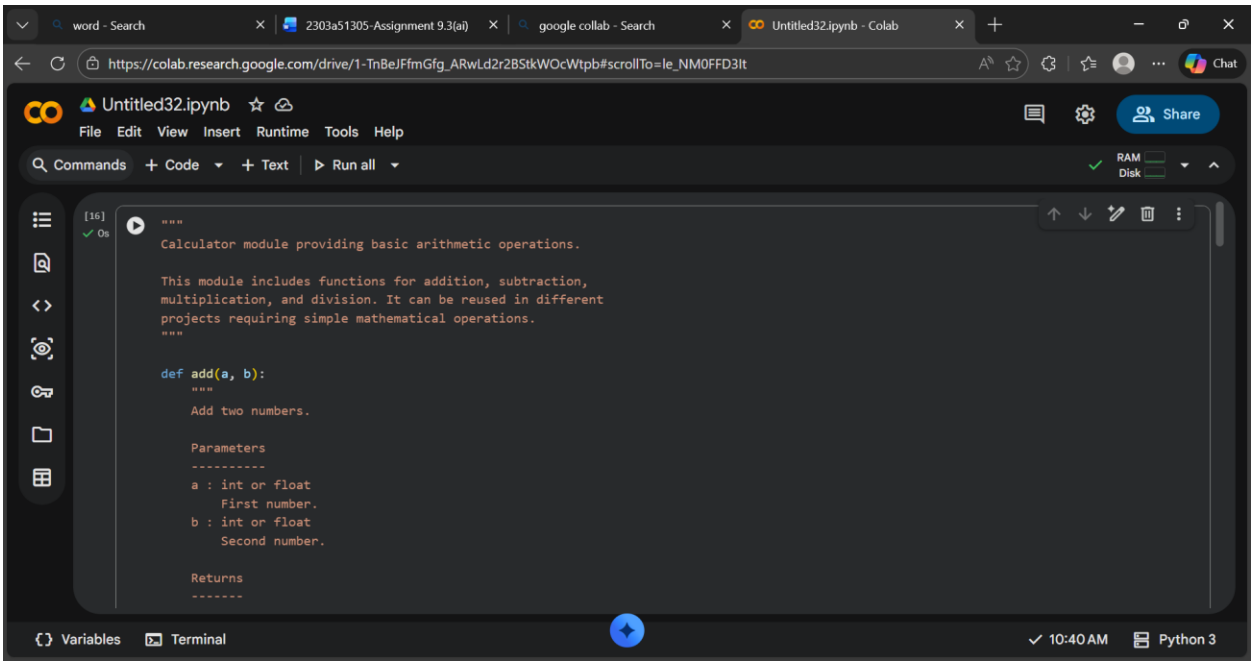
• Python script with manual NumPy-style docstrings

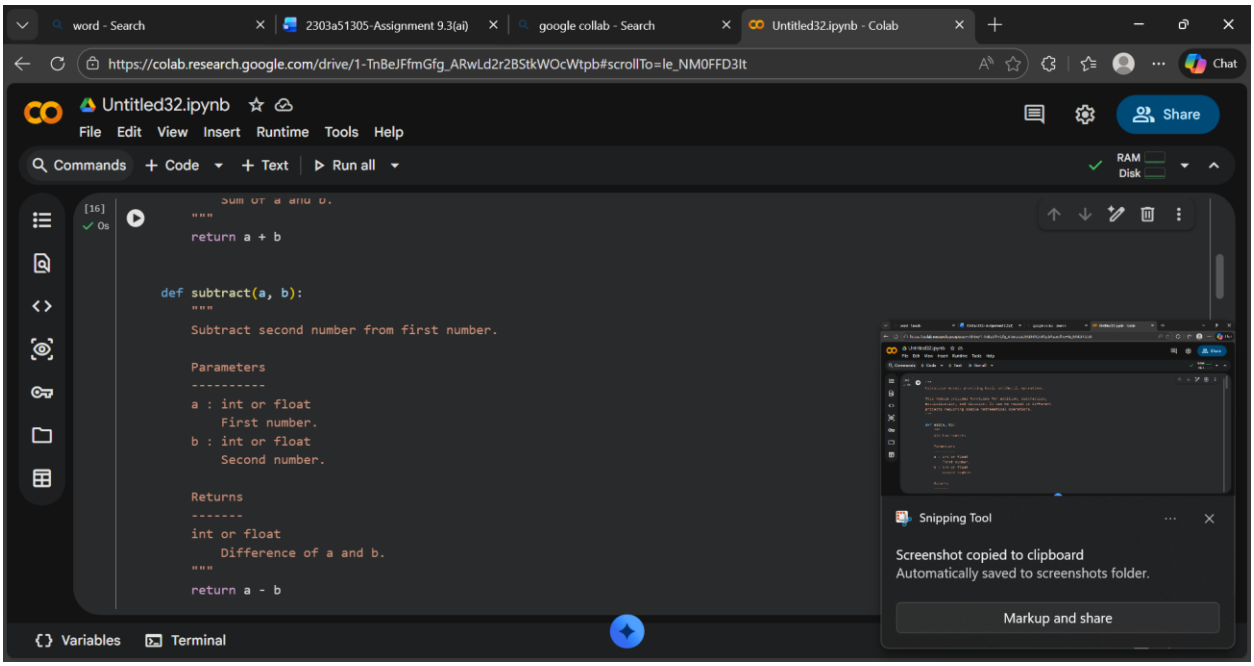
• AI-generated module-level and function-level documentation

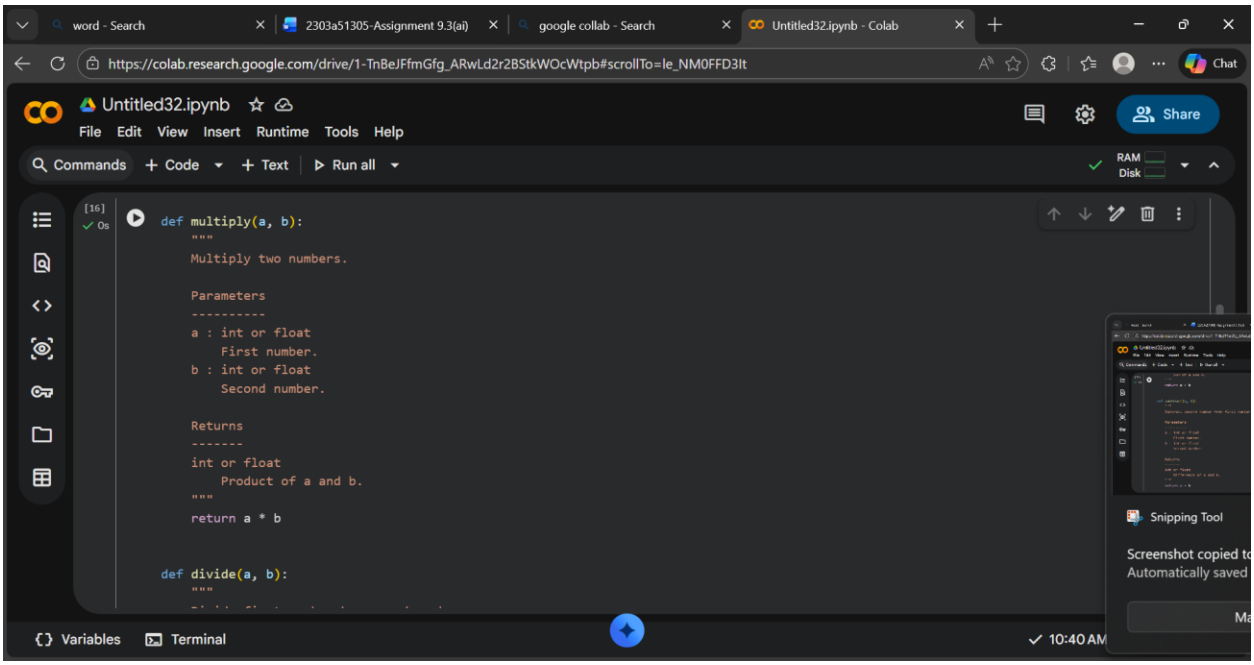
• Comparison between AI-generated and manual documentation

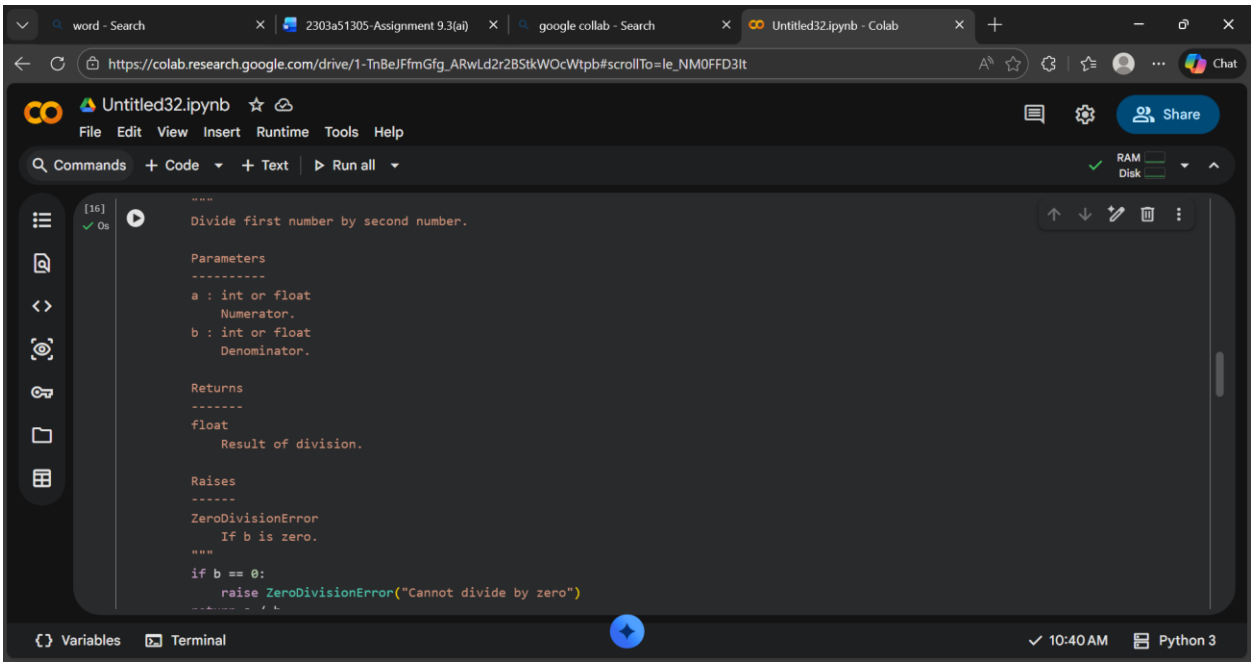
• Clear understanding of structured documentation for multi-function scripts.

Code:

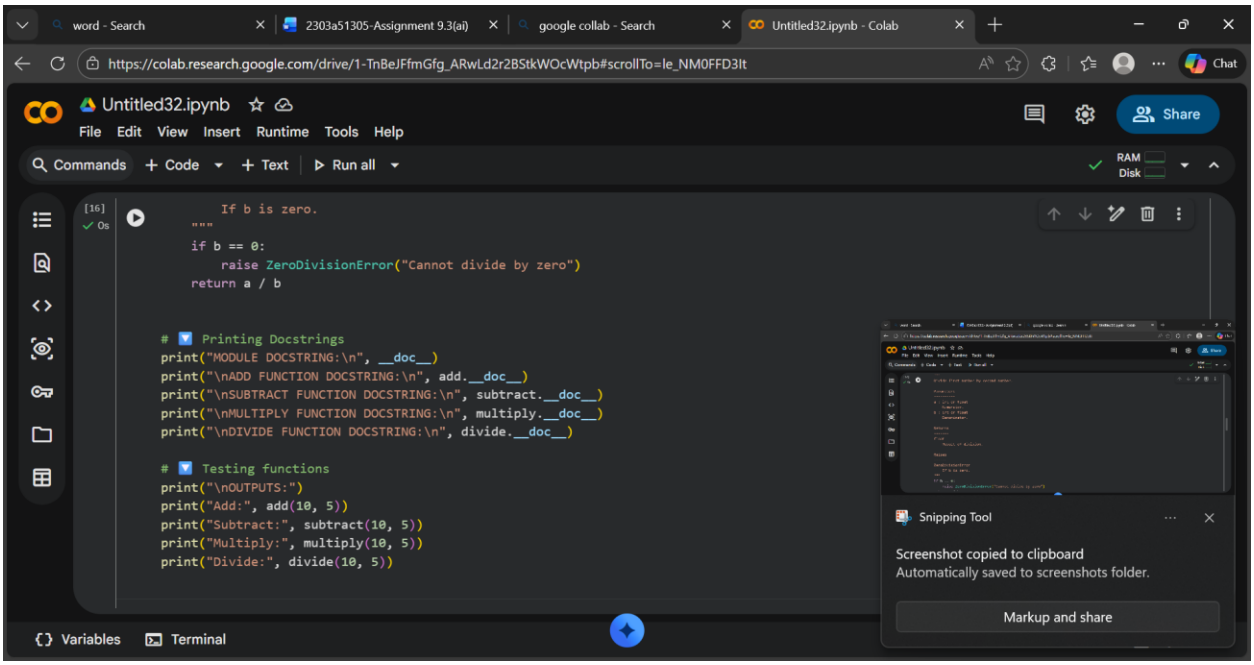

word - Search
2303a51305-Assignment 9.3(ai)
google collab - Search
Untitled32.ipynb - Colab
https://colab.research.google.com/drive/1-TnBeJFfmGfg_ARwLd2r2BStkWOcWtpb#scrollTo=le_NM0FFD3It
Chat
Untitled32.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM
Disk
[16]
0s
"""
Calculator module providing basic arithmetic operations.
This module includes functions for addition, subtraction,
multiplication, and division. It can be reused in different
projects requiring simple mathematical operations.
"""
def add(a, b):
"""
Add two numbers.
Parameters
a : int or float
First number.
b : int or float
Second number.
Returns
Variables
Terminal
10:40 AM
Python 3


word - Search
2303a51305-Assignment 9.3(ai)
google collab - Search
Untitled32.ipynb - Colab
https://colab.research.google.com/drive/1-TnBeJFfmGfg_ARwLd2r2BStkWOcWtpb#scrollTo=le_NM0FFD3It
Chat
Untitled32.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM
Disk
[16]
0s
Sum of a and b.
"""
return a + b
def subtract(a, b):
"""
Subtract second number from first number.
Parameters
a : int or float
First number.
b : int or float
Second number.
Returns
int or float
Difference of a and b.
"""
return a - b
Snipping Tool
Screenshot copied to clipboard
Automatically saved to screenshots folder.
Markup and share
Variables
Terminal

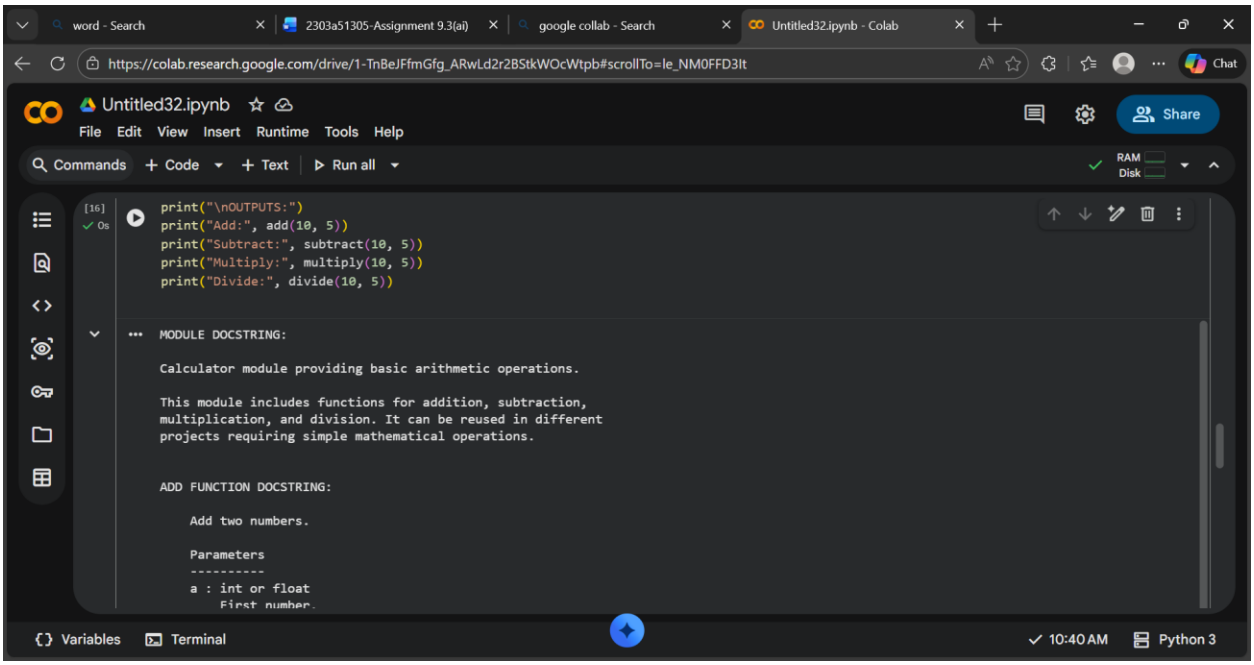

word - Search
2303a51305-Assignment 9.3(ai)
google collab - Search
Untitled32.ipynb - Colab
https://colab.research.google.com/drive/1-TnBeJFfmGfg_ARwLd2r2BStkWOcWtpb#scrollTo=le_NM0FFD3It
Chat
Untitled32.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM
Disk
[16]
0s
def multiply(a, b):
"""
Multiply two numbers.
Parameters
----
a : int or float
First number.
b : int or float
Second number.
Returns
----
int or float
Product of a and b.
"""
return a * b
def divide(a, b):
"""
Snipping Tool
Screenshot copied to
Automatically saved
Variables
Terminal
10:40 AM

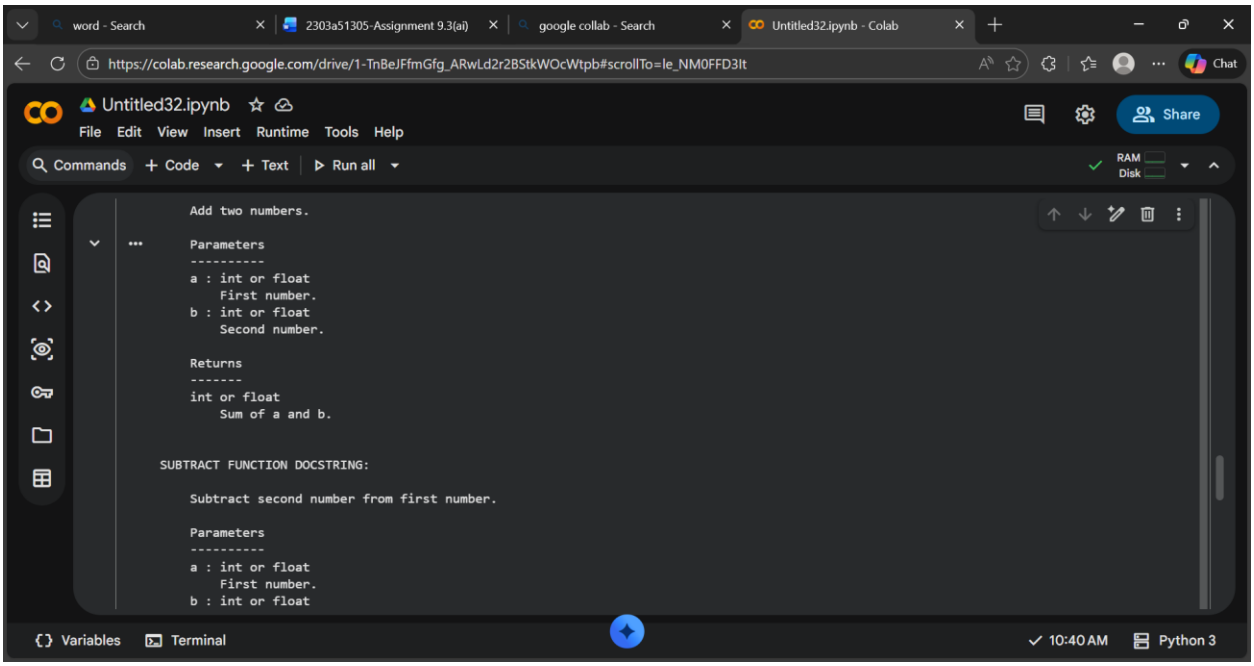

word - Search
2303a51305-Assignment 9.3(ai)
google collab - Search
Untitled32.ipynb - Colab
https://colab.research.google.com/drive/1-TnBeJFfmGfg_ARwLd2r2BStkWOcWtpb#scrollTo=le_NM0FFD3It
Chat
Untitled32.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM
Disk
[16]
0s
"""
Divide first number by second number.
Parameters
----
a : int or float
Numerator.
b : int or float
Denominator.
Returns
----
float
Result of division.
Raises

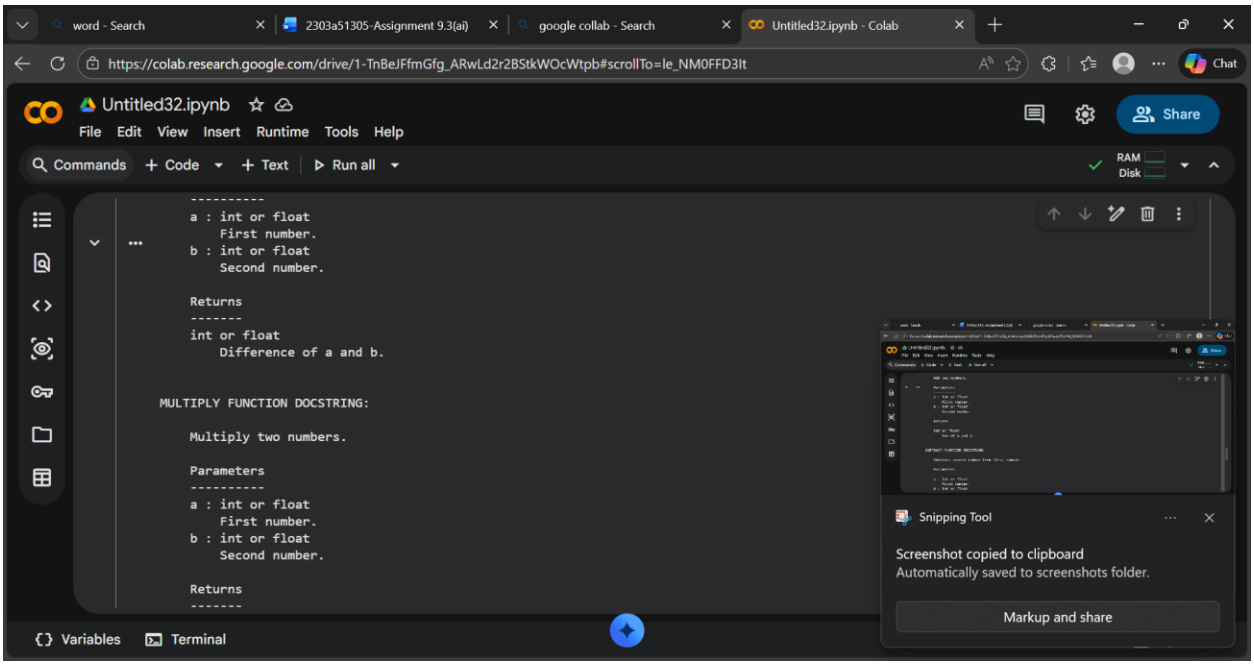
ZeroDivisionError
If b is zero.
"""
if b == 0:
raise ZeroDivisionError("Cannot divide by zero")
Variables
Terminal
10:40 AM
Python 3

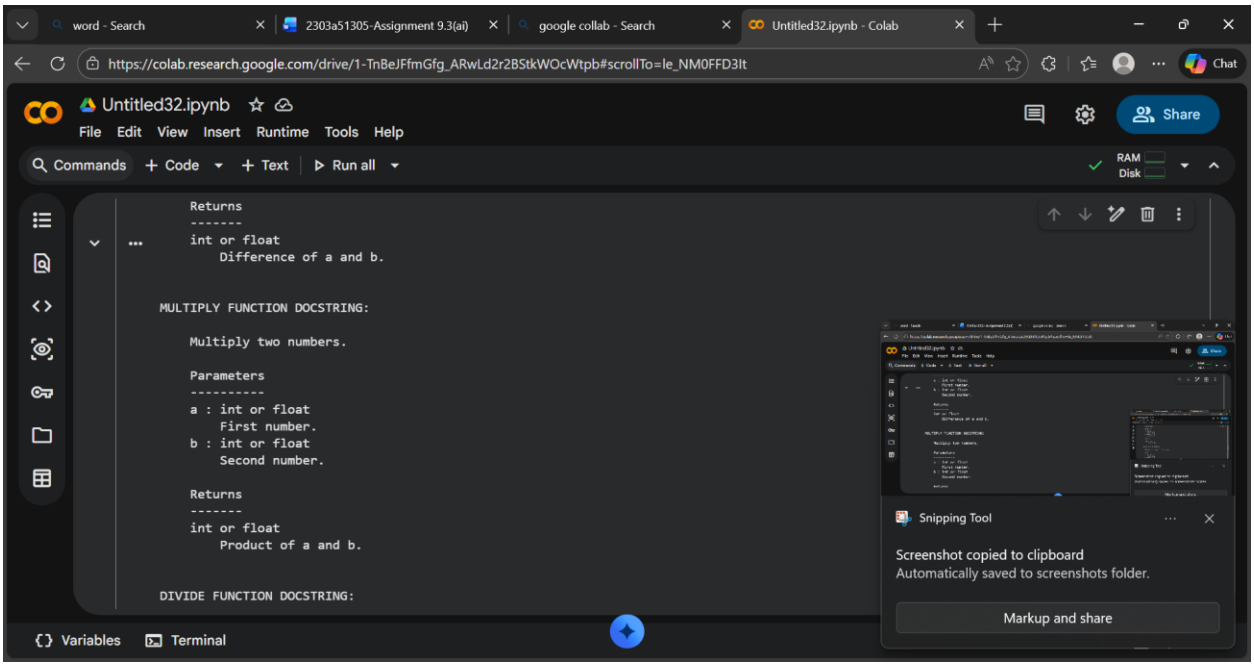

word - Search
2303a51305-Assignment 9.3(ai)
google collab - Search
Untitled32.ipynb - Colab
https://colab.research.google.com/drive/1-TnBeJFfmGfg_ARwLd2r2BStkWOcWtpb#scrollTo=le_NM0FFD3It
Chat
Untitled32.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM
Disk
[16]
0s
If b is zero.
"""
if b == 0:
raise ZeroDivisionError("Cannot divide by zero")
return a / b
# Printing Docstrings
print("MODULE DOCSTRING:\n", __doc__)
print("\nADD FUNCTION DOCSTRING:\n", add.__doc__)
print("\nSUBTRACT FUNCTION DOCSTRING:\n", subtract.__doc__)
print("\nMULTIPLY FUNCTION DOCSTRING:\n", multiply.__doc__)
print("\nDIVIDE FUNCTION DOCSTRING:\n", divide.__doc__)
# Testing functions
print("\nOUTPUTS:")
print("Add:", add(10, 5))
print("Subtract:", subtract(10, 5))
print("Multiply:", multiply(10, 5))
print("Divide:", divide(10, 5))
Snipping Tool
Screenshot copied to clipboard
Automatically saved to screenshots folder.
Markup and share
Variables
Terminal

Output:


word - Search
2303a51305-Assignment 9.3(ai)
google collab - Search
Untitled32.ipynb - Colab
https://colab.research.google.com/drive/1-TnBeJFfmGfg_ARwLd2r2BStkWOcWtpb#scrollTo=le_NM0FFD3It
Chat
Untitled32.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM Disk
[16] 0s
print("\nOUTPUTS:")
print("Add:", add(10, 5))
print("Subtract:", subtract(10, 5))
print("Multiply:", multiply(10, 5))
print("Divide:", divide(10, 5))
MODULE DOCSTRING:
Calculator module providing basic arithmetic operations.
This module includes functions for addition, subtraction,
multiplication, and division. It can be reused in different
projects requiring simple mathematical operations.
ADD FUNCTION DOCSTRING:
Add two numbers.
Parameters
a : int or float
First number.
Variables
Terminal
10:40 AM
Python 3


word - Search
2303a51305-Assignment 9.3(ai)
google collab - Search
Untitled32.ipynb - Colab
https://colab.research.google.com/drive/1-TnBeJFfmGfg_ARwLd2r2BStkWOcWtpb#scrollTo=le_NM0FFD3It
Chat
Untitled32.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM Disk
Add two numbers.
Parameters
a : int or float
First number.
b : int or float
Second number.
Returns
int or float
Sum of a and b.
SUBTRACT FUNCTION DOCSTRING:
Subtract second number from first number.
Parameters
a : int or float
First number.
b : int or float
Variables
Terminal
10:40 AM
Python 3





Explanation:

Aspect	Manual NumPy Docstrings	AI-Generated Docstrings	Analysis
Format	Proper NumPy style	Generic style	Manual follows structured standard
Detail	Clear sections	Short descriptions	AI is brief
Error Handling	Mentions ZeroDivisionError	Often missing	Manual more complete

Readability	Highly structured	Simple and readable	AI easier but less formal

Conclusion:

AI is helpful for generating initial documentation, but manual refinement is required for professional-quality module and function documentation.