Assignment – 2.5

Name: M.Sprusheeth

Roll Number: 2303A51206

Batch - 04

AI Assisted Coding

16-01-2026

Task 1: Refactoring Odd/Even Logic (List Version)

❖ Scenario:

You are improving legacy code.

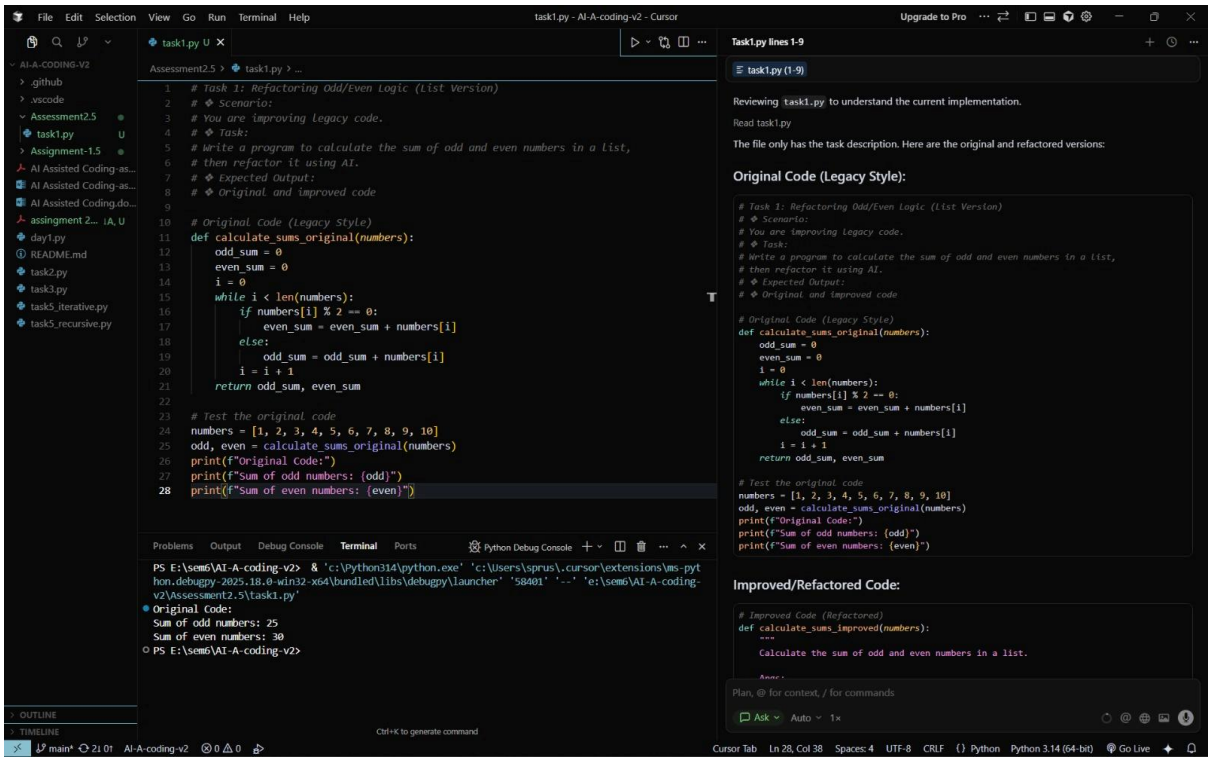
❖ Task:

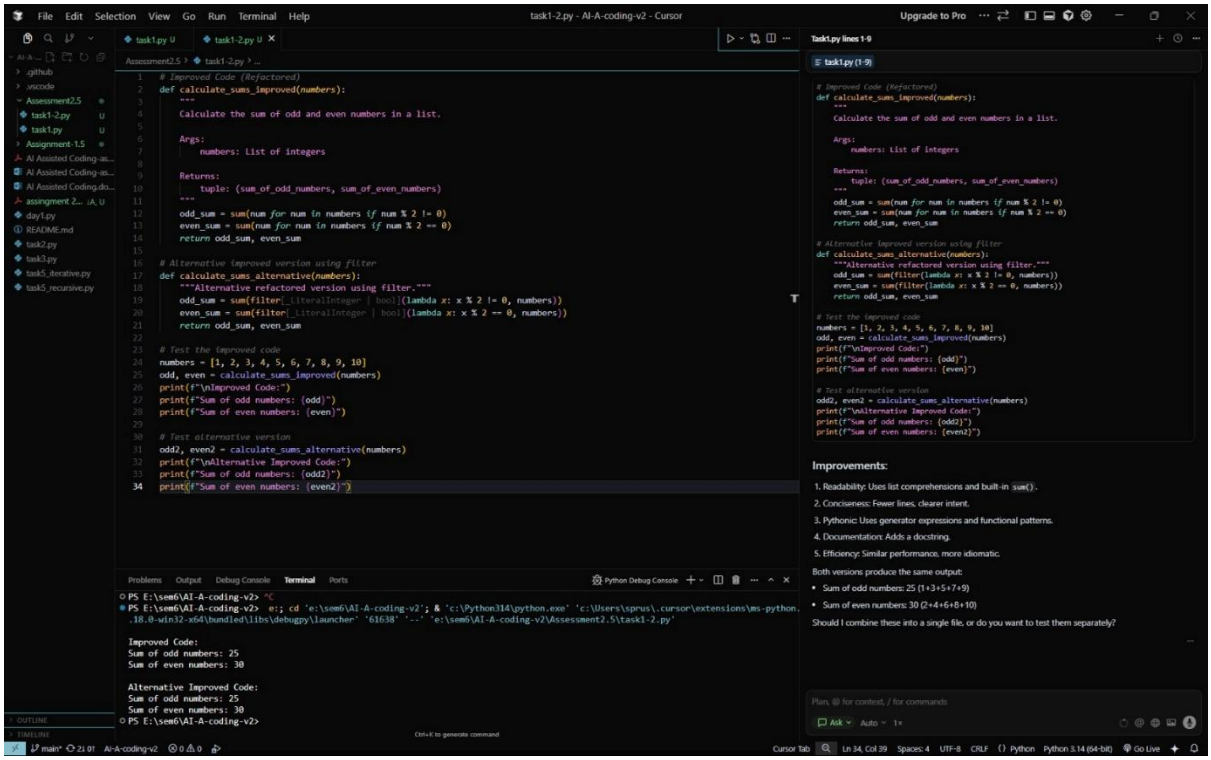
Write a program to calculate the sum of odd and even numbers in a list,

then refactor it using AI.

❖ Expected Output:

❖ Original and improved code





Task 2: Area Calculation Explanation

❖ Scenario:

You are onboarding a junior developer.

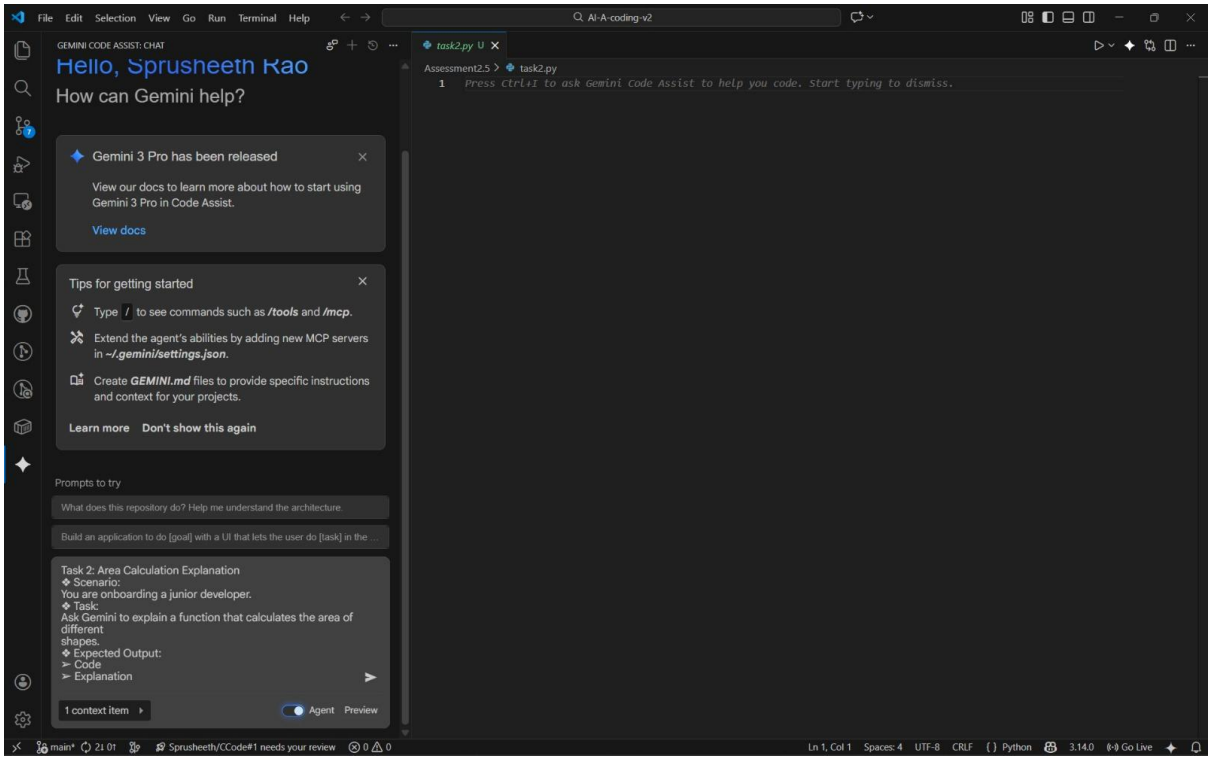
❖ Task:

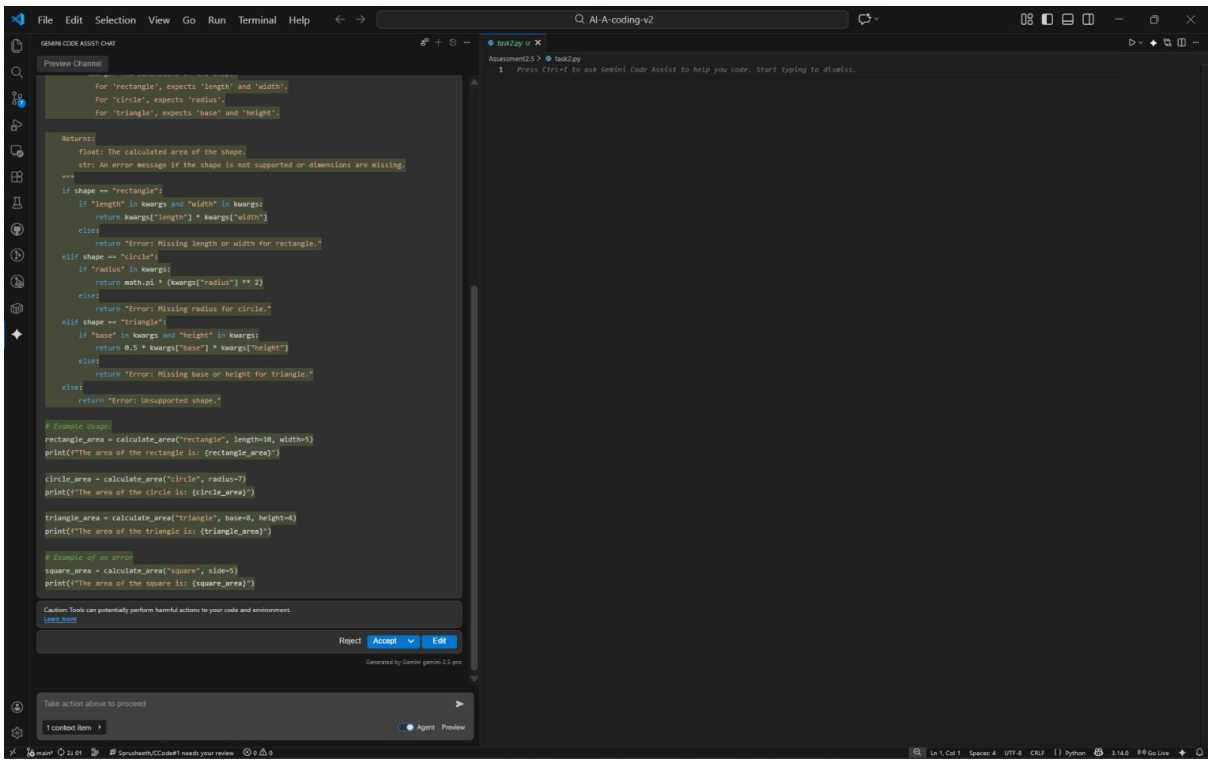
Ask Gemini to explain a function that calculates the area of different shapes.

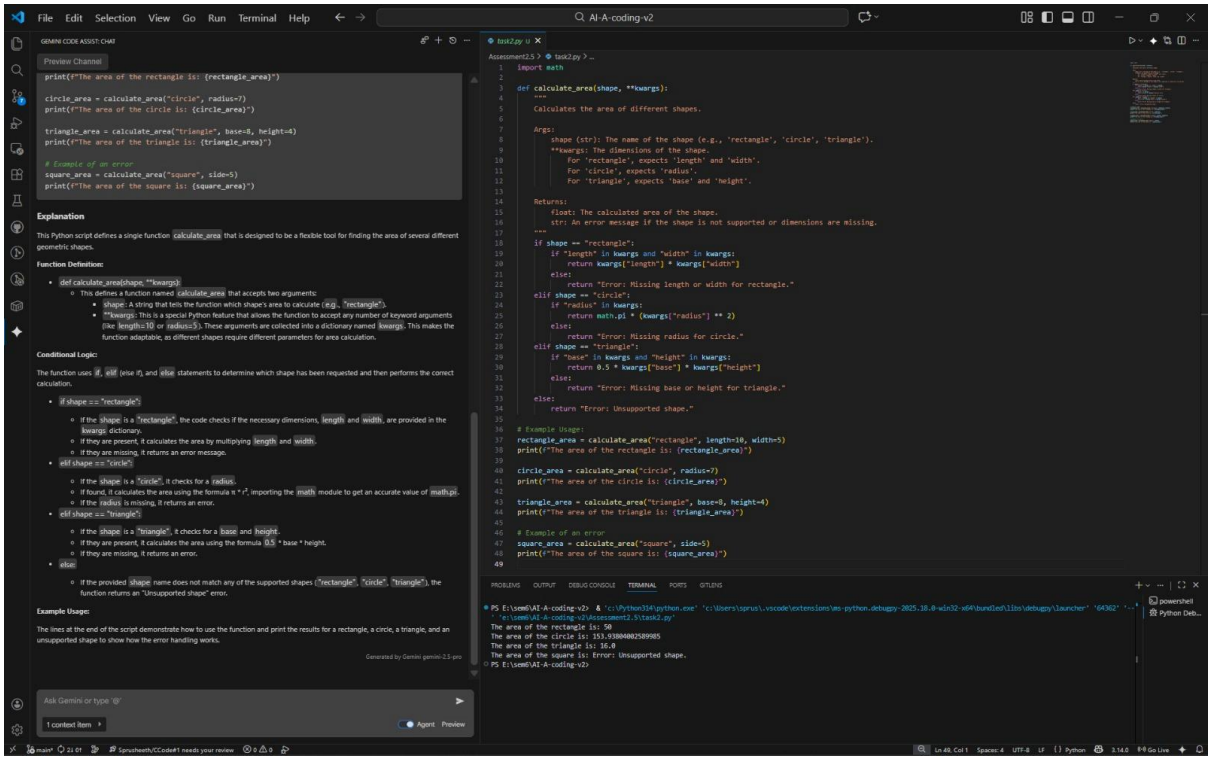
❖ Expected Output:

➢ Code

➢ Explanation


File Edit Selection View Go Run Terminal Help
AI-A-coding-v2
GEMINI CODE ASSIST: CHAT
Hello, Sprusheeth Rao
How can Gemini help?
Gemini 3 Pro has been released
View our docs to learn more about how to start using Gemini 3 Pro in Code Assist.
View docs
Tips for getting started
Type / to see commands such as /tools and /mcp.
Extend the agent's abilities by adding new MCP servers in ~/.gemini/settings.json.
Create GEMINI.md files to provide specific instructions and context for your projects.
Learn more
Don't show this again
Prompts to try
What does this repository do? Help me understand the architecture.
Build an application to do [goal] with a UI that lets the user do [task] in the ...
Task 2: Area Calculation Explanation
❖ Scenario:
You are onboarding a junior developer.
❖ Task:
Ask Gemini to explain a function that calculates the area of different
shapes.
❖ Expected Output:
➢ Code
➢ Explanation
1 context item
Agent
Preview
task2.py U
Assessment2.5 > task2.py
1 Press Ctrl+I to ask Gemini Code Assist to help you code. Start typing to dismiss.
main* 21 01
Sprusheeth/CCode#1 needs your review
Ln 1, Col 1
Spaces: 4
UTF-8
CRLF
Python
3.14.0
Go Live


File Edit Selection View Go Run Terminal Help
AI-A-coding-v2
GEMINI CODE ASSIST: CHAT
Preview Channel
For 'rectangle', expects 'length' and 'width'.
For 'circle', expects 'radius'.
For 'triangle', expects 'base' and 'height'.
Returns:
float: The calculated area of the shape.
str: An error message if the shape is not supported or dimensions are missing.
"""
if shape == "rectangle":
if "length" in kwargs and "width" in kwargs:
return kwargs["length"] * kwargs["width"]
else:
return "Error: Missing length or width for rectangle."
elif shape == "circle":
if "radius" in kwargs:
return math.pi * (kwargs["radius"] ** 2)
else:
return "Error: Missing radius for circle."
elif shape == "triangle":
if "base" in kwargs and "height" in kwargs:
return 0.5 * kwargs["base"] * kwargs["height"]
else:
return "Error: Missing base or height for triangle."
else:
return "Error: Unsupported shape."
# Example Usage:
rectangle_area = calculate_area("rectangle", length=10, width=5)
print(f"The area of the rectangle is: {rectangle_area}")
circle_area = calculate_area("circle", radius=7)
print(f"The area of the circle is: {circle_area}")
triangle_area = calculate_area("triangle", base=8, height=4)
print(f"The area of the triangle is: {triangle_area}")
# Example of an error
square_area = calculate_area("square", side=5)
print(f"The area of the square is: {square_area}")
Caution: Tools can potentially perform harmful actions to your code and environment.
Learn more
Reject
Accept
Edit
Generated by Gemini gemini-2.5-pro
Take action above to proceed
1 context item
Agent
Preview
task2.py U
Assessment2.5 > task2.py
1 Press Ctrl+I to ask Gemini Code Assist to help you code. Start typing to dismiss.
main* 21 01
Sprusheeth/CCode#1 needs your review
Ln 1, Col 1
Spaces: 4
UTF-8
CRLF
Python
3.14.0
Go Live



Explanation

This Python script defines a single function calculate_area that is designed to be a flexible tool for finding the area of several different geometric shapes.

Function Definition:

def calculate_area(shape, **kwargs):

This defines a function named calculate_area that accepts two arguments:

shape: A string that tells the function which shape's area to calculate (e.g., "rectangle").

**kwargs: This is a special Python feature that allows the function to accept any number of keyword arguments (like length=10 or radius=5). These arguments are collected into a dictionary named kwargs. This makes the function adaptable, as different shapes require different parameters for area calculation.

Conditional Logic:

The function uses if, elif (else if), and else statements to determine which shape has been requested and then performs the correct calculation.

if shape == "rectangle":

If the shape is a "rectangle", the code checks if the necessary dimensions, length and width, are provided in the kwargs dictionary.

If they are present, it calculates the area by multiplying length and width.

If they are missing, it returns an error message.

elif shape == "circle":

If the shape is a "circle", it checks for a radius.

If found, it calculates the area using the formula $\pi * r^2$, importing the math module to get an accurate value of math.pi.

If the radius is missing, it returns an error.

elif shape == "triangle":

If the shape is a "triangle", it checks for a base and height.

If they are present, it calculates the area using the formula 0.5 * base * height.

If they are missing, it returns an error.

else:

If the provided shape name does not match any of the supported shapes ("rectangle", "circle", "triangle"), the function returns an "Unsupported shape" error.

Example Usage:

The lines at the end of the script demonstrate how to use the function and print the results for a rectangle, a circle, a triangle, and an unsupported shape to show how the error handling works.

Task 3: Prompt Sensitivity Experiment

❖ Scenario:

You are testing how AI responds to different prompts.

❖ Task:

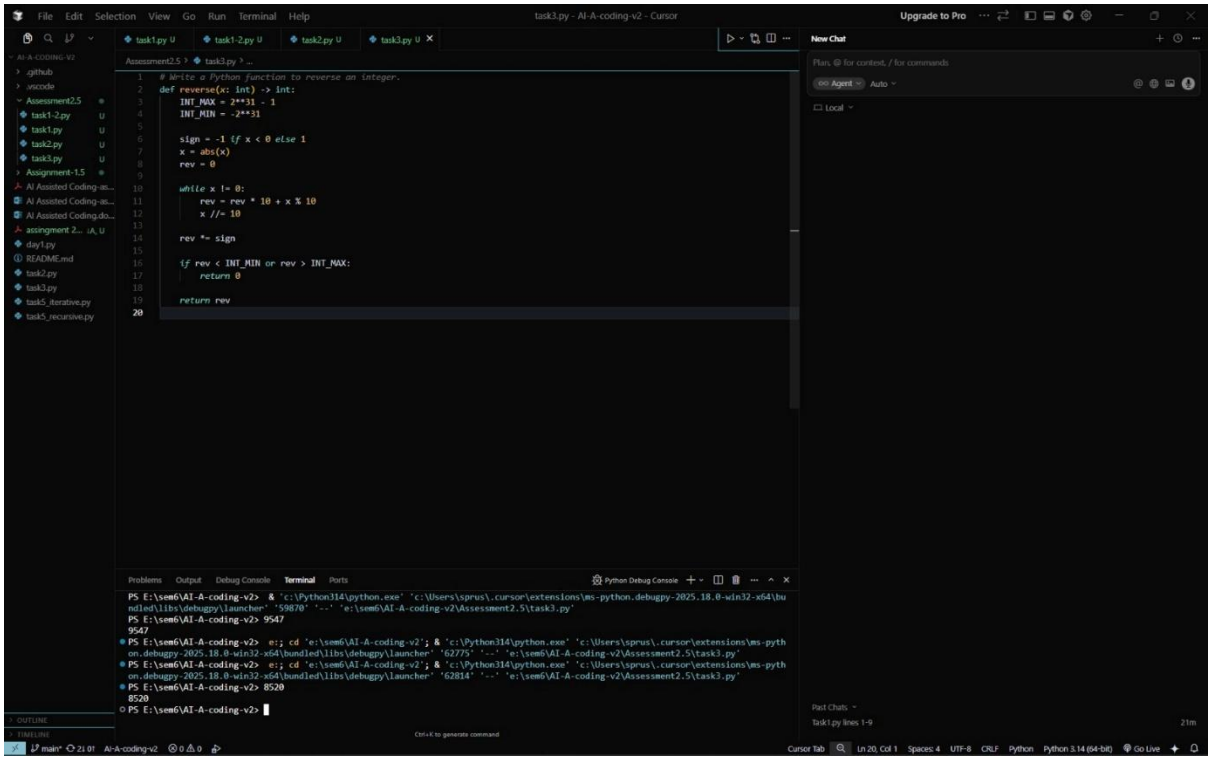
Use Cursor AI with different prompts for the same problem and observe

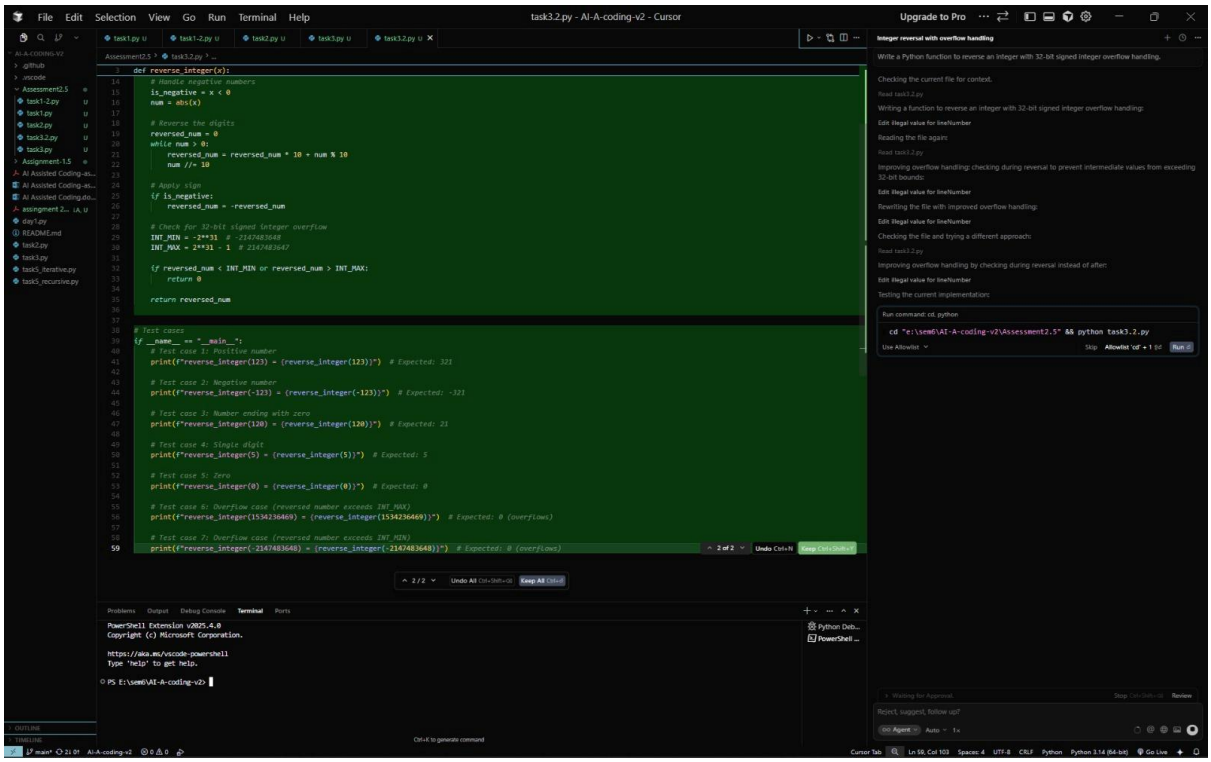
code changes.

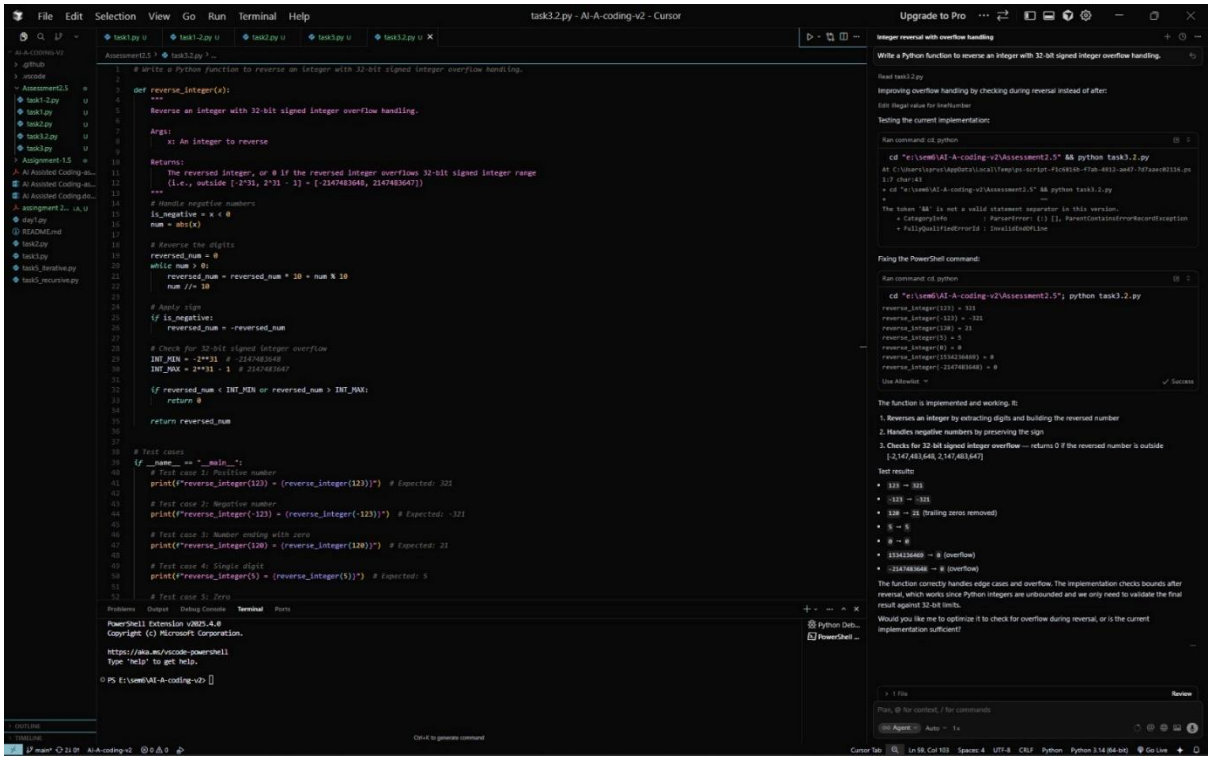
❖ Expected Output:

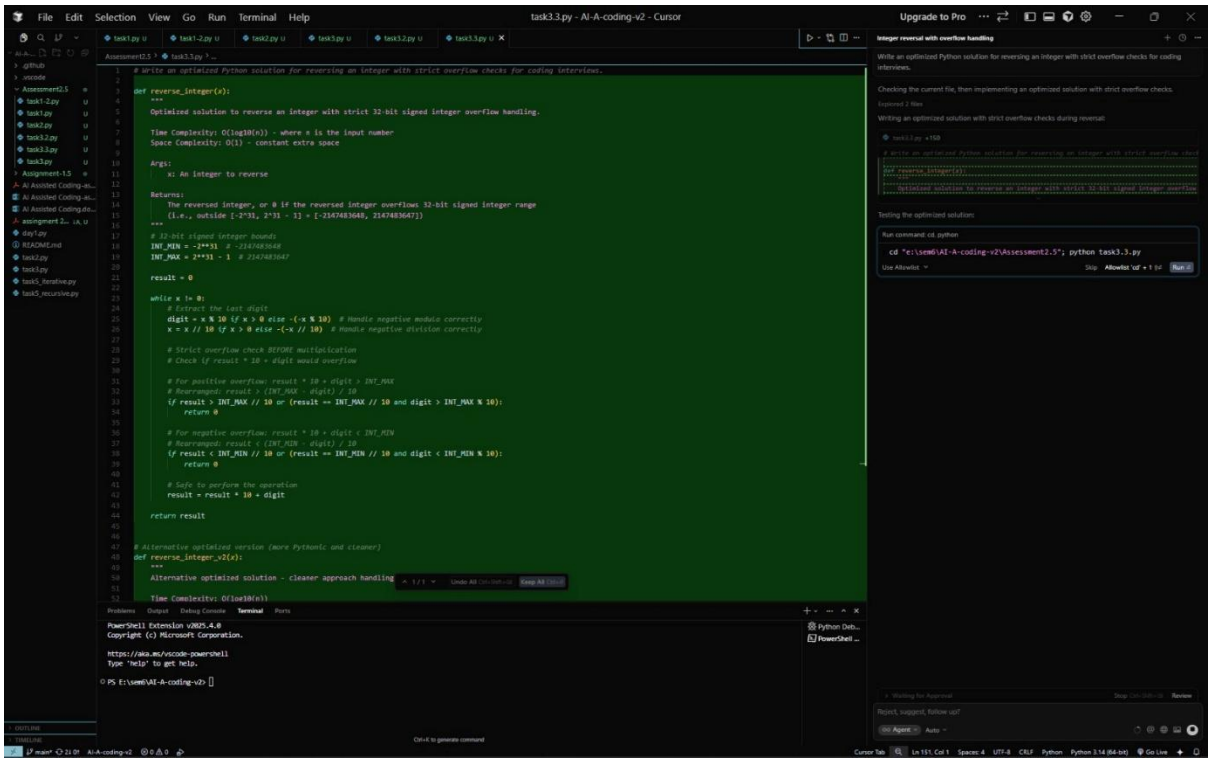
➢ Prompt list

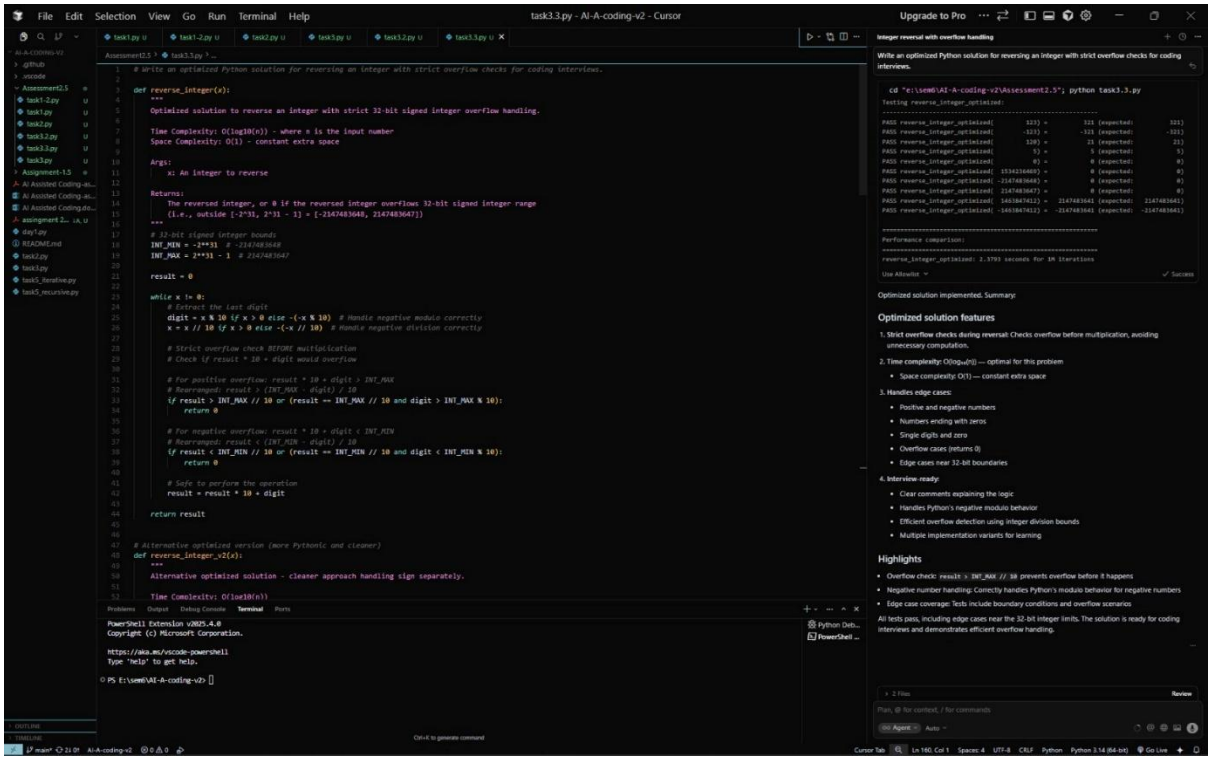
➢ Code variations


File Edit Selection View Go Run Terminal Help
task3.py - AI-A-coding-v2 - Cursor
Upgrade to Pro
task1.py U
task1-2.py U
task2.py U
task3.py U
Assessment2.5 > task3.py > ...
# Write a Python function to reverse an integer.
def reverse(x: int) -> int:
INT_MAX = 2**31 - 1
INT_MIN = -2**31
sign = -1 if x < 0 else 1
x = abs(x)
rev = 0
while x != 0:
rev = rev * 10 + x % 10
x //= 10
rev *= sign
if rev < INT_MIN or rev > INT_MAX:
return 0
return rev
New Chat
Plan, @ for context, / for commands
Agent
Auto
Local
Problems
Output
Debug Console
Terminal
Ports
Python Debug Console
PS E:\sem6\AI-A-coding-v2> & 'c:\Python314\python.exe' 'c:\Users\sprus\.cursor\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '59870' '--' 'e:\sem6\AI-A-coding-v2\Assessment2.5\task3.py'
PS E:\sem6\AI-A-coding-v2> 9547
9547
PS E:\sem6\AI-A-coding-v2> e:; cd 'e:\sem6\AI-A-coding-v2'; & 'c:\Python314\python.exe' 'c:\Users\sprus\.cursor\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '62775' '--' 'e:\sem6\AI-A-coding-v2\Assessment2.5\task3.py'
PS E:\sem6\AI-A-coding-v2> e:; cd 'e:\sem6\AI-A-coding-v2'; & 'c:\Python314\python.exe' 'c:\Users\sprus\.cursor\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '62814' '--' 'e:\sem6\AI-A-coding-v2\Assessment2.5\task3.py'
PS E:\sem6\AI-A-coding-v2> 8520
8520
PS E:\sem6\AI-A-coding-v2>
Past Chats
Task1.py lines 1-9
21m
Ctrl+K to generate command
main*
AI-A-coding-v2
Cursor Tab
Ln 20, Col 1
Spaces: 4
UTF-8
CRLF
Python
Python 3.14 (64-bit)
Go Live


File Edit Selection View Go Run Terminal Help
task3.2.py - AI-A-coding-v2 - Cursor
Upgrade to Pro
task1.py U
task1-2.py U
task2.py U
task3.py U
task3.2.py U
Assessment2.5 > task3.2.py > ...
def reverse_integer(x):
# Handle negative numbers
is_negative = x < 0
num = abs(x)
# Reverse the digits
reversed_num = 0
while num > 0:
reversed_num = reversed_num * 10 + num % 10
num //= 10
# Apply sign
if is_negative:
reversed_num = -reversed_num
# Check for 32-bit signed integer overflow
INT_MIN = -2**31 # -2147483648
INT_MAX = 2**31 - 1 # 2147483647
if reversed_num < INT_MIN or reversed_num > INT_MAX:
return 0
return reversed_num
# Test cases
if __name__ == "__main__":
# Test case 1: Positive number
print(f"reverse_integer(123) = {reverse_integer(123)}") # Expected: 321
Test case 2: Negative number
print(f"reverse_integer(-123) = {reverse_integer(-123)}") # Expected: -321
Test case 3: Number ending with zero
print(f"reverse_integer(120) = {reverse_integer(120)}") # Expected: 21
Test case 4: Single digit
print(f"reverse_integer(5) = {reverse_integer(5)}") # Expected: 5
Test case 5: Zero
print(f"reverse_integer(0) = {reverse_integer(0)}") # Expected: 0
# Test case 6: Overflow case (reversed number exceeds INT_MAX)
print(f"reverse_integer(1534236469) = {reverse_integer(1534236469)}") # Expected: 0 (overflows)
# Test case 7: Overflow case (reversed number exceeds INT_MIN)
print(f"reverse_integer(-2147483648) = {reverse_integer(-2147483648)}") # Expected: 0 (overflows)
2 of 2
Undo Ctrl+N
Keep Ctrl+Shift+↵
2 / 2
Undo All Ctrl+Shift+⌫
Keep All Ctrl+↵
Integer reversal with overflow handling
Write a Python function to reverse an integer with 32-bit signed integer overflow handling.
Checking the current file for context.
Read task3.2.py
Writing a function to reverse an integer with 32-bit signed integer overflow handling:
Edit illegal value for lineNumber
Reading the file again:
Read task3.2.py
Improving overflow handling: checking during reversal to prevent intermediate values from exceeding 32-bit bounds:
Edit illegal value for lineNumber
Rewriting the file with improved overflow handling:
Edit illegal value for lineNumber
Checking the file and trying a different approach:
Read task3.2.py
Improving overflow handling by checking during reversal instead of after:
Edit illegal value for lineNumber
Testing the current implementation:
Run command: cd, python
cd "e:\sem6\AI-A-coding-v2\Assessment2.5" && python task3.2.py
Use Allowlist
Skip
Allowlist 'cd' + 1 tid
Run ⇧
Problems
Output
Debug Console
Terminal
Ports
PowerShell Extension v2025.4.0
Copyright (c) Microsoft Corporation.
https://aka.ms/vscode-powershell
Type 'help' to get help.
PS E:\sem6\AI-A-coding-v2>
Python Deb...
PowerShell ...
Waiting for Approval
Skip Ctrl+Shift+⌫
Review
Reject, suggest, follow up?
Agent
Auto
1×
Ctrl+K to generate command
main*
AI-A-coding-v2
Cursor Tab
Ln 59, Col 103
Spaces: 4
UTF-8
CRLF
Python
Python 3.14 (64-bit)
Go Live


File Edit Selection View Go Run Terminal Help
task3.2.py - AI-A-coding-v2 - Cursor
Upgrade to Pro
task1.py U
task1-2.py U
task2.py U
task5.py U
task3.2.py U
Assessment2.5 › task3.2.py › ...
# Write a Python function to reverse an integer with 32-bit signed integer overflow handling.
def reverse_integer(x):
"""
Reverse an integer with 32-bit signed integer overflow handling.
Args:
x: An integer to reverse
Returns:
The reversed integer, or 0 if the reversed integer overflows 32-bit signed integer range
(i.e., outside [-2^31, 2^31 - 1] = [-2147483648, 2147483647])
"""
# Handle negative numbers
is_negative = x < 0
num = abs(x)
# Reverse the digits
reversed_num = 0
while num > 0:
reversed_num = reversed_num * 10 + num % 10
num //= 10
# Apply sign
if is_negative:
reversed_num = -reversed_num
# Check for 32-bit signed integer overflow
INT_MIN = -2**31 # -2147483648
INT_MAX = 2**31 - 1 # 2147483647
if reversed_num < INT_MIN or reversed_num > INT_MAX:
return 0
return reversed_num
# Test cases
if __name__ == "__main__":
# Test case 1: Positive number
print(f"reverse_integer(123) = {reverse_integer(123)}") # Expected: 321
Test case 2: Negative number
print(f"reverse_integer(-123) = {reverse_integer(-123)}") # Expected: -321
Test case 3: Number ending with zero
print(f"reverse_integer(120) = {reverse_integer(120)}") # Expected: 21
Test case 4: Single digit
print(f"reverse_integer(5) = {reverse_integer(5)}") # Expected: 5
Test case 5: Zero
Problems Output Debug Console Terminal Ports
PowerShell Extension v2025.4.0
Copyright (c) Microsoft Corporation.
https://aka.ms/vscode-powershell
Type 'help' to get help.
PS E:\sem6\AI-A-coding-v2>
Integer reversal with overflow handling
Write a Python function to reverse an integer with 32-bit signed integer overflow handling.
Read task3.2.py
Improving overflow handling by checking during reversal instead of after:
Edit Illegal value for lineNumber
Testing the current implementation:
Ran command: cd; python
cd "e:\sem6\AI-A-coding-v2\Assessment2.5" && python task3.2.py
At C:\Users\sprus\AppData\Local\Temp\ps-script-f1c481bb-4f2b-4912-ae47-7d7aaec02116.ps1:7 char:43
+ cd "e:\sem6\AI-A-coding-v2\Assessment2.5" && python task3.2.py
+ ~~
The token '&&' is not a valid statement separator in this version.
+ CategoryInfo : ParserError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : InvalidEndOfLine
Fixing the PowerShell command:
Ran command: cd; python
cd "e:\sem6\AI-A-coding-v2\Assessment2.5"; python task3.2.py
reverse_integer(123) = 321
reverse_integer(-123) = -321
reverse_integer(120) = 21
reverse_integer(5) = 5
reverse_integer(0) = 0
reverse_integer(1534236469) = 0
reverse_integer(-2147483648) = 0
Use Allowlist
Success
The function is implemented and working. It:
1. Reverses an integer by extracting digits and building the reversed number
2. Handles negative numbers by preserving the sign
3. Checks for 32-bit signed integer overflow — returns 0 if the reversed number is outside [-2,147,483,648, 2,147,483,647]
Test results:
123 → 321
-123 → -321
120 → 21 (trailing zeros removed)
5 → 5
0 → 0
1534236469 → 0 (overflow)
-2147483648 → 0 (overflow)
The function correctly handles edge cases and overflow. The implementation checks bounds after reversal, which works since Python integers are unbounded and we only need to validate the final result against 32-bit limits.
Would you like me to optimize it to check for overflow during reversal, or is the current implementation sufficient?
1 File
Review
Plan, @ for context, / for commands
Agent
Auto
Ctrl+K to generate command
Cursor Tab
Ln 59, Col 103
Spaces: 4
UTF-8
CRLF
Python
Python 3.14 (64-bit)
Go Live


File Edit Selection View Go Run Terminal Help
task3.3.py - AI-A-coding-v2 - Cursor
Upgrade to Pro
task1.py U
task1-2.py U
task2.py U
task5.py U
task3.2.py U
task3.3.py U
Assessment2.5 › task3.3.py › ...
# Write an optimized Python solution for reversing an integer with strict overflow checks for coding interviews.
def reverse_integer(x):
"""
Optimized solution to reverse an integer with strict 32-bit signed integer overflow handling.
Time Complexity: O(log10(n)) - where n is the input number
Space Complexity: O(1) - constant extra space
Args:
x: An integer to reverse
Returns:
The reversed integer, or 0 if the reversed integer overflows 32-bit signed integer range
(i.e., outside [-2^31, 2^31 - 1] = [-2147483648, 2147483647])
"""
# 32-bit signed integer bounds
INT_MIN = -2**31 # -2147483648
INT_MAX = 2**31 - 1 # 2147483647
result = 0
while x != 0:
# Extract the last digit
digit = x % 10 if x > 0 else -(-x % 10) # Handle negative modulo correctly
x = x // 10 if x > 0 else -(-x // 10) # Handle negative division correctly
# Strict overflow check BEFORE multiplication
# Check if result * 10 + digit would overflow
# For positive overflow: result * 10 + digit > INT_MAX
# Rearranged: result > (INT_MAX - digit) / 10
if result > INT_MAX // 10 or (result == INT_MAX // 10 and digit > INT_MAX % 10):
return 0
# For negative overflow: result * 10 + digit < INT_MIN
# Rearranged: result < (INT_MIN - digit) / 10
if result < INT_MIN // 10 or (result == INT_MIN // 10 and digit < INT_MIN % 10):
return 0
# Safe to perform the operation
result = result * 10 + digit
return result
# Alternative optimized version (more Pythonic and cleaner)
def reverse_integer_v2(x):
"""
Alternative optimized solution - cleaner approach handling
Time Complexity: O(log10(n))
Undo All
Keep All
Problems Output Debug Console Terminal Ports
PowerShell Extension v2025.4.0
Copyright (c) Microsoft Corporation.
https://aka.ms/vscode-powershell
Type 'help' to get help.
PS E:\sem6\AI-A-coding-v2>
Integer reversal with overflow handling
Write an optimized Python solution for reversing an integer with strict overflow checks for coding interviews.
Checking the current file, then implementing an optimized solution with strict overflow checks.
Explored 1 file
Writing an optimized solution with strict overflow checks during reversal:
task3.2.py +150
# Write an optimized Python solution for reversing an integer with strict overflow check
def reverse_integer(x):
"""
Optimized solution to reverse an integer with strict 32-bit signed integer overflow
Testing the optimized solution:
Run command: cd; python
cd "e:\sem6\AI-A-coding-v2\Assessment2.5"; python task3.3.py
Use Allowlist
Skip
Allowlist 'cd' + 1 bit
Run
Waiting for Approval
Skip
Review
Reject, suggest, follow up?
Agent
Auto
Ctrl+K to generate command
Cursor Tab
Ln 151, Col 1
Spaces: 4
UTF-8
CRLF
Python
Python 3.14 (64-bit)
Go Live



Task 4: Tool Comparison Reflection

❖ Scenario:

You must recommend an AI coding tool.

❖ Task:

Based on your work in this topic, compare Gemini, Copilot, and Cursor AI

for usability and code quality.

❖ Expected Output:

Short written reflection

Based on my experience using Gemini, GitHub Copilot, and Cursor AI during this topic, I observed clear differences in both usability and code quality.

Gemini is useful for understanding concepts and generating explanations, but it often produces generic code unless very strict constraints are provided. It is better suited for learning and problem understanding rather than competitive or production-level coding.

GitHub Copilot integrates smoothly with IDEs like VS Code and provides fast, context-aware code suggestions. However, its outputs sometimes assume the developer will

handle edge cases, so overflow handling and constraints may be missed unless explicitly guided.

Cursor AI provided the best balance of usability and code quality. It allows direct interaction with the codebase, understands existing files, and responds well to detailed prompts. When constraints are clearly mentioned, Cursor AI consistently generated correct, optimized, and readable code, making it ideal for real development and debugging tasks.

Conclusion:

For learning → Gemini

For quick coding assistance → Copilot

For serious development and prompt-based experimentation → Cursor AI