

Task Description#1

- Zero-shot: Prompt AI to write a function that checks whether a given year is a leap year.
-

The image consists of two screenshots of a VS Code editor window, showing the development and testing of a Python function to check for leap years.

Top Screenshot: The editor shows a file named `is_leap_year.py` with the following code:

```
1 def is_leap_year(year):
2     """Check if a given year is a leap year.
3     A year is a leap year if:
4     1. It is divisible by 4
5     2. If it is divisible by 100, it must also be divisible by 400
6     Args:
7         year (int): The year to check
8     Returns:
9         bool: True if the year is a leap year, False otherwise
10    """
11    if year % 4 != 0:
12        return False
13    elif year % 100 != 0:
14        return True
15    else:
16        return year % 400 == 0
17    # Get user input and check if it's a leap year
18    try:
19        user_year = int(input("Enter a year to check if it's a leap year: "))
20        result = is_leap_year(user_year)
21        print(f"{user_year} is {'a leap year' if result else 'not a leap year'}")
22    except ValueError:
23        print("Please enter a valid integer year.")
```

The right sidebar shows the "Generate" panel with the task description and a chat window with the AI agent `gh1`.

Bottom Screenshot: The editor shows the same file, but the code is now only the last line of the `try` block:

```
23 print("Please enter a valid integer year.")
```

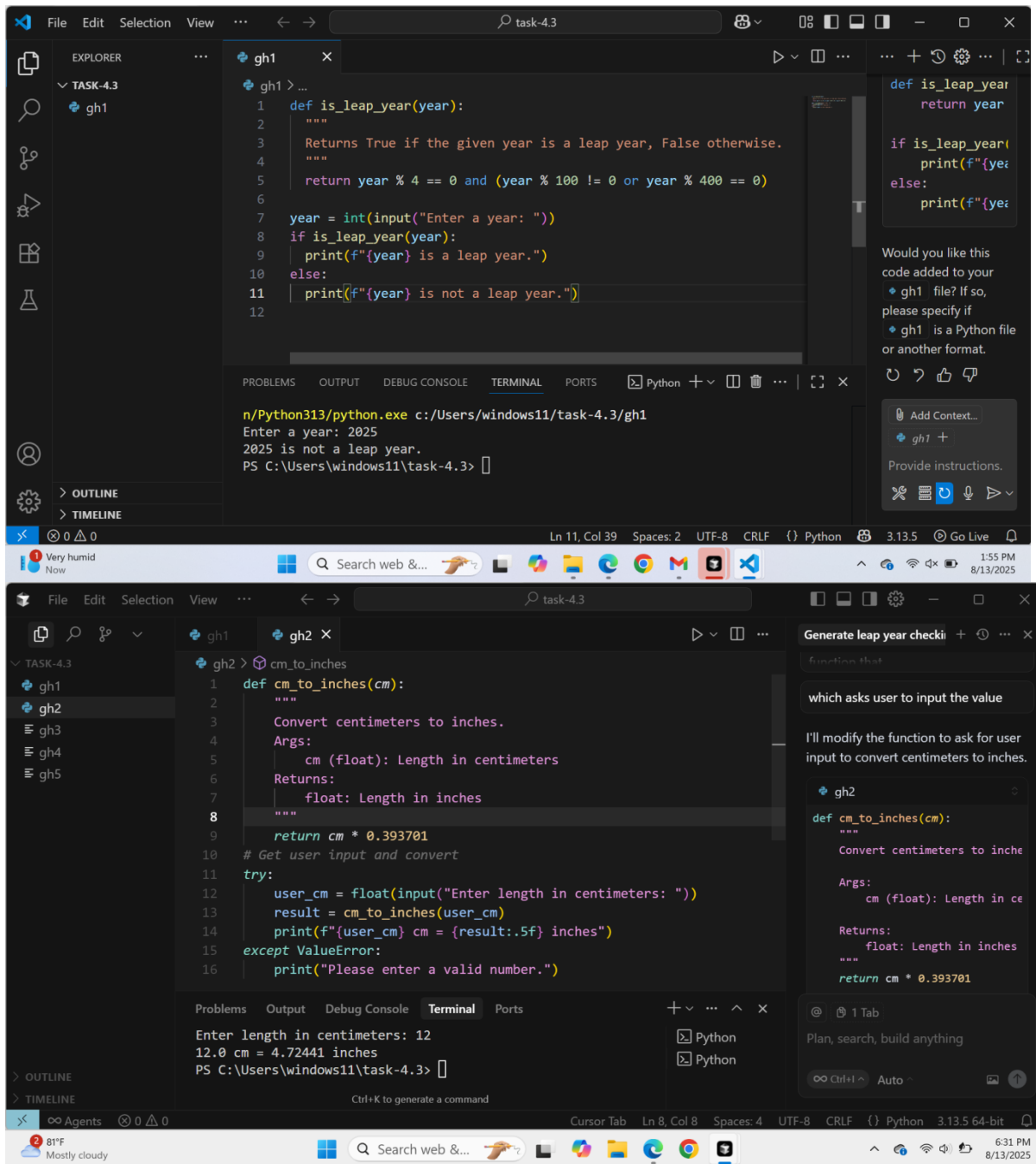
The bottom panel shows the "Terminal" with the following output:

```
PS C:\Users\windows11\task-4.3> & C:/Users/windows11/AppData/Local/Programs/Python/Python313/python.exe c:/Users/windows11/task-4.3/gh1
Enter a year to check if it's a leap year: 2025
2025 is not a leap year
PS C:\Users\windows11\task-4.3> & C:/Users/windows11/AppData/Local/Programs/Python/Python313/python.exe c:/Users/windows11/task-4.3/gh1
Enter a year to check if it's a leap year: 2024
2024 is a leap year
PS C:\Users\windows11\task-4.3>
```

The right sidebar shows the "Generate" panel with the task description and a chat window with the AI agent `gh1`.

Task Description#2

- One-shot: Give one input-output example to guide AI in writing a function that converts centimeters to inches.



Task Description#3

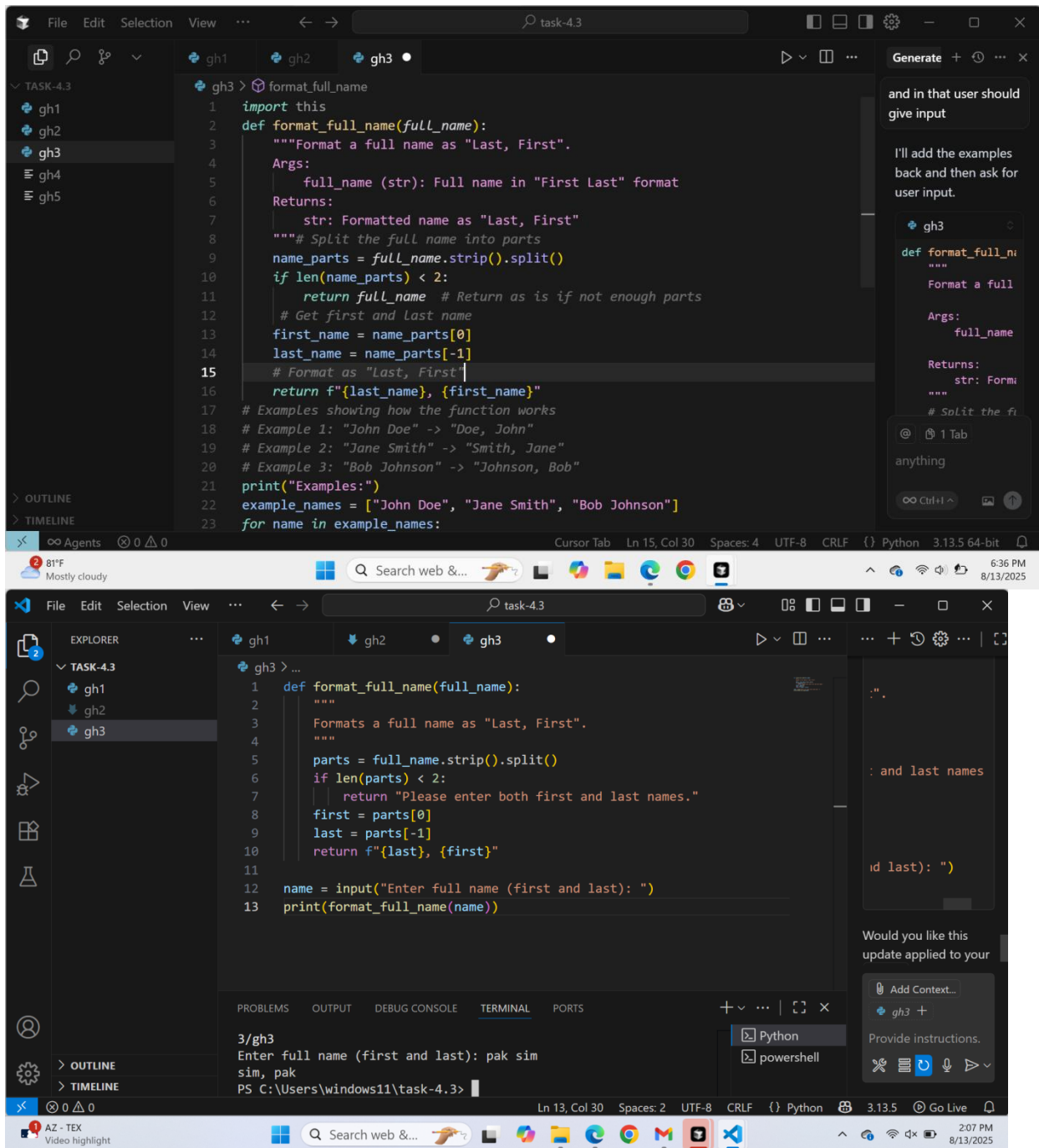
Few-shot: Provide 2–3 examples to generate a function that formats full names as “Last, First

The image shows a VS Code editor window with a file explorer on the left and a terminal at the bottom. The main editor area displays a Python script in a file named `gh2`. The script is a function `convert_cm_to_inches` that takes a centimeter value and returns the equivalent in inches. The script includes a docstring, a type hint, and a test case. The terminal shows the output of the script, which is `3.937`. The status bar at the bottom indicates the file is encoded in UTF-8 with CRLF line endings.

```
1  ## Example: Convert Centimeters to Inches
2
3  **Input:**
4  `convert_cm_to_inches(10)`
5
6  **Output:**
7  `3.937`
8
9  ```python
10 def convert_cm_to_inches(cm):
11     """Convert centimeters to inches."""
12     return cm / 2.54
13 ```
```

The terminal output shows the function being called with the argument `10` and returning `3.937`.

```
Enter a full name (First Last): sree varsh
'sree varsh' -> 'varsh, sree'
PS C:\Users\windows11\task-4.3>
```



Task Description#4

- Compare zero-shot and few-shot prompts for writing a function that counts the number of vowels in a string.

File Edit Selection View ... task-4.3

Problems Output Debug Console **Terminal** Ports

Enter a string to count vowels: sreevarshinnee

Test string: 'sreevarshinnee'
Zero-shot result: 6 vowels
Few-shot result: 6 vowels

Comparative Reflection:
1. Both functions produce identical results - the examples don't change the logic
2. Few-shot provides better documentation and understanding
3. Zero-shot is more direct but may lack context
4. Both approaches work for simple, well-defined tasks like counting vowels
PS C:\Users\windows11\task-4.3>

Python Python

TASK-4.3
gh1
gh2
gh3
gh4
gh5

OUTLINE
TIMELINE

Ctrl+K to generate a command

Agents 0 0 0

Cursor Tab Ln 10, Col 23 Spaces: 2 UTF-8 CRLF Python 3.13.5 64-bit

File Edit Selection View ... task-4.3

gh1 gh2 gh3 gh4

TASK-4.3
gh1
gh2
gh3
gh4
gh5

```
13 gh4 > count_vowels_zero_shot
14
15 def count_vowels_few_shot(text):
16     vowels = 'aeiouAEIOU'
17     count = 0
18     for char in text:
19         if char in vowels:
20             count += 1
21     return count
22
23 # Get user input for testing
24 user_string = input("Enter a string to count vowels: ")
25
26 # Test both functions with user input (variable) user_string: str
27 result_zero = count_vowels_zero_shot(user_string)
28 result_few = count_vowels_few_shot(user_string)
29 print(f"\nTest string: '{user_string}'")
30 print(f"Zero-shot result: {result_zero} vowels")
31 print(f"Few-shot result: {result_few} vowels")
32
33 # Comparative reflection
34 print("\nComparative Reflection:")
35 print("1. Both functions produce identical results - the examples don't change the logic")
36 print("2. Few-shot provides better documentation and understanding")
37 print("3. Zero-shot is more direct but may lack context")
38 print("4. Both approaches work for simple, well-defined tasks like counting vowels")
39
40
```

OUTLINE
TIMELINE

Agents 0 0 0

Cursor Tab Ln 10, Col 23 Spaces: 2 UTF-8 CRLF Python 3.13.5 64-bit

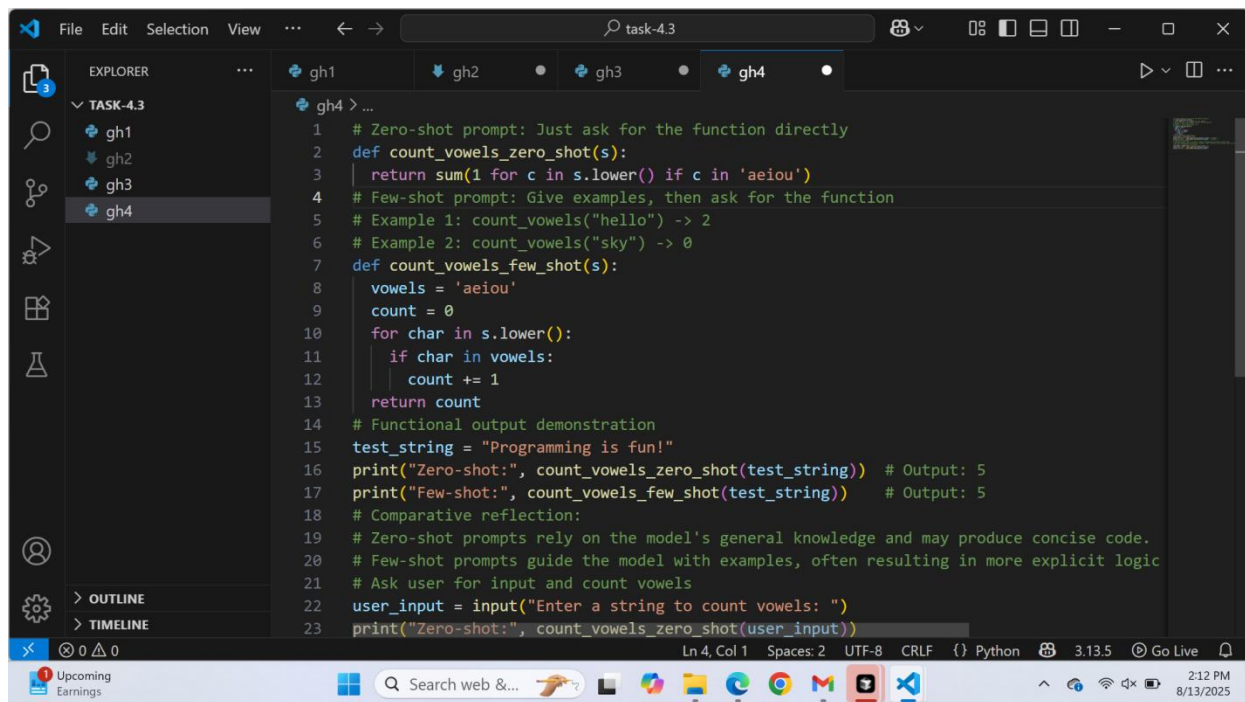
```
File Edit Selection View ... task-4.3
gh1 gh2 gh3 gh4
TASK-4.3
gh1
gh2
gh3
gh4
gh5
OUTLINE
TIMELINE
Agents 0 0 0
CursorTab Ln 10, Col 23 Spaces: 2 UTF-8 CRLF Python 3.13.5 64-bit
81°F Mostly cloudy 6:42 PM 8/13/2025
```

```
gh4 > count_vowels_zero_shot
1 # Zero-shot prompt: Just ask for the function directly
2 def count_vowels_zero_shot(text):
3     """
4     Count the number of vowels in a string.
5     """
6     vowels = 'aeiouAEIOU'
7     count = 0
8     for char in text:
9         if char in vowels:
10            count += 1
11    return count
12 # Few-shot prompt: Provide examples to guide the function
13 def count_vowels_few_shot(text):
14     """
15     Count the number of vowels in a string.
16     Examples:
17     - "hello" -> 2 vowels (e, o)
18     - "world" -> 1 vowel (o)
19     - "python" -> 1 vowel (o)
20     """
21     vowels = 'aeiouAEIOU'
22     count = 0
23     for char in text:
```

```
File Edit Selection View ... task-4.3
EXPLORER gh1 gh2 gh3 gh4
TASK-4.3
gh1
gh2
gh3
gh4
OUTLINE
TIMELINE
Agents 0 0 0
CursorTab Ln 4, Col 1 Spaces: 2 UTF-8 CRLF Python 3.13.5 Go Live
Trending videos Avatar: Fire and... 2:13 PM 8/13/2025
```

```
gh4 > ...
4 # Few-shot prompt: Give examples, then ask for the function
5 # Example 1: count_vowels("hello") -> 2
6 # Example 2: count_vowels("sky") -> 0
7 def count_vowels_few_shot(s):
8     vowels = 'aeiou'
9     count = 0
10    for char in s.lower():
11        if char in vowels:
12            count += 1
13    return count
14 # Functional output demonstration
15 test_string = "Programming is fun!"
16 print("Zero-shot:", count_vowels_zero_shot(test_string)) # Output: 5
17 print("Few-shot:", count_vowels_few_shot(test_string)) # Output: 5
18 # Comparative reflection:
19 # Zero-shot prompts rely on the model's general knowledge and may produce concise code.
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Few-shot: 5
Enter a string to count vowels: weee
Zero-shot: 3
Few-shot: 3
PS C:\Users\windows11\task-4.3>
```



The screenshot shows a Visual Studio Code editor window with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'TASK-4.3' with four files: 'gh1', 'gh2', 'gh3', and 'gh4'. The 'gh4' file is selected and its content is displayed in the code editor. The code is a Python script that demonstrates zero-shot and few-shot prompting for a function that counts vowels. The script includes comments explaining the prompts and the results of the function calls. The status bar at the bottom shows the current line and column (Ln 4, Col 1), the file encoding (UTF-8), the line ending (CRLF), the language (Python), the version (3.13.5), and the 'Go Live' button. The system tray at the bottom right shows the time (2:12 PM) and the date (8/13/2025).

```
1 # Zero-shot prompt: Just ask for the function directly
2 def count_vowels_zero_shot(s):
3     return sum(1 for c in s.lower() if c in 'aeiou')
4 # Few-shot prompt: Give examples, then ask for the function
5 # Example 1: count_vowels("hello") -> 2
6 # Example 2: count_vowels("sky") -> 0
7 def count_vowels_few_shot(s):
8     vowels = 'aeiou'
9     count = 0
10    for char in s.lower():
11        if char in vowels:
12            count += 1
13    return count
14 # Functional output demonstration
15 test_string = "Programming is fun!"
16 print("Zero-shot:", count_vowels_zero_shot(test_string)) # Output: 5
17 print("Few-shot:", count_vowels_few_shot(test_string)) # Output: 5
18 # Comparative reflection:
19 # Zero-shot prompts rely on the model's general knowledge and may produce concise code.
20 # Few-shot prompts guide the model with examples, often resulting in more explicit logic
21 # Ask user for input and count vowels
22 user_input = input("Enter a string to count vowels: ")
23 print("Zero-shot:", count_vowels_zero_shot(user_input))
```

Task Description#5

- Use few-shot prompting to generate a function that reads a .txt file and returns the number of lines.

File Edit Selection View ... task-4.3

gh1 gh2 gh3 gh4 gh5

gh5 > ...

```
1 def count_lines_in_file(filename):
23     print(f"Error: File '{filename}' not found in current directory.")
24     print("Current directory contents:")
25     import os
26     for file in os.listdir('.'):
27         if file.endswith('.txt'):
28             print(f" - {file}")
29     return -1
30 except PermissionError:
31     print(f"Error: Permission denied to read file '{filename}'")
32     return -1
33 except Exception as e:
34     print(f"Unexpected error reading file: {e}")
35     return -1
36 # Get user input for filename
37 filename = input("Enter the name of the .txt file to count lines: ")
38 # Count lines in the file
39 line_count = count_lines_in_file(filename)
40 if line_count >= 0:
41     print(f"The file '{filename}' contains {line_count} lines.")
42 else:
43     print("Could not count lines due to an error.")
```

not getting

check what's in the current file and then run it to see what's happening.

Read gh5

I see the issue! The code is missing the part that creates the sample file. Let me add that back and also add some debugging to help you see what's happening:

@ 1 Tab

Plan, search, build

Ctrl+I

80°F Cloudy

Search web &...

Cursor Tab Ln 36, Col 1 Spaces: 4 UTF-8 CRLF Python 3.13.5 64-bit

6:54 PM 8/13/2025

File Edit Selection View ... task-4.3

gh1 gh2 gh3 gh4 gh5

gh5 > count_lines_in_file

```
1 def count_lines_in_file(filename): TAB to jump here
2     """
3     Read a .txt file and return the number of lines.
4
5     Examples:
6     - "sample.txt" with 5 lines -> returns 5
7     - "empty.txt" with 0 lines -> returns 0
8     - "long_text.txt" with 100 lines -> returns 100
9
10    Args:
11        filename (str): Name of the .txt file to read
12
13    Returns:
14        int: Number of lines in the file
15    """
16    try:
17        print(f"Attempting to open file: {filename}")
18        with open(filename, 'r', encoding='utf-8') as file:
19            line_count = sum(1 for line in file)
20        print(f"Successfully read file: {filename}")
21        return line_count
22    except FileNotFoundError:
23        print(f"Error: File '{filename}' not found in current directory.")
```

not getting

check what's in the current file and then run it to see what's happening.

Read gh5

I see the issue! The code is missing the part that creates the sample file. Let me add that back and also add some debugging to help you see what's happening:

@ 1 Tab

Plan, search, build

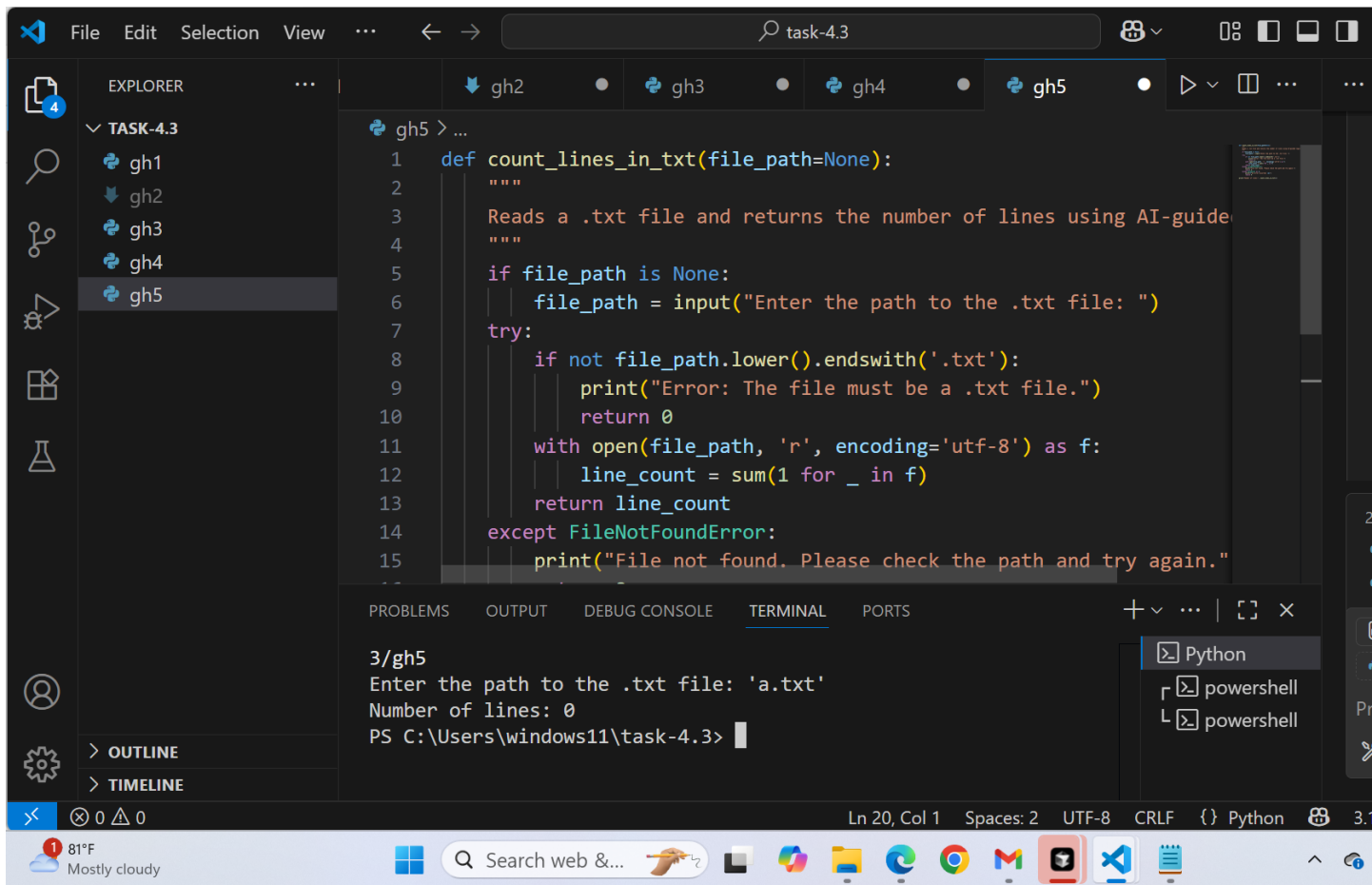
Ctrl+I

80°F Cloudy

Search web &...

Cursor Tab Ln 14, Col 41 Spaces: 4 UTF-8 CRLF Python 3.13.5 64-bit

6:53 PM 8/13/2025



File Edit Selection View ... task-4.3

EXPLORER

- TASK-4.3
 - gh1
 - gh2
 - gh3
 - gh4
 - gh5

gh5 > ...

```
1 def count_lines_in_txt(file_path=None):
2     """
3     Reads a .txt file and returns the number of lines using AI-guide
4     """
5     if file_path is None:
6         file_path = input("Enter the path to the .txt file: ")
7     try:
8         if not file_path.lower().endswith('.txt'):
9             print("Error: The file must be a .txt file.")
10            return 0
11            with open(file_path, 'r', encoding='utf-8') as f:
12                line_count = sum(1 for _ in f)
13            return line_count
14    except FileNotFoundError:
15        print("File not found. Please check the path and try again.")
16        return 0
17    except Exception as e:
18        print(f"An error occurred: {e}")
19        return 0
20
21 print("Number of lines:", count_lines_in_txt())
```

2 files changed

- gh4
- gh5

Add Context...

gh5 +

Provide instructions.

Ln 20, Col 1 Spaces: 2 UTF-8 CRLF {} Python 3.13.5 Go Live

81°F Mostly cloudy

Search web &...

2:29 PM 8/13/2025