AI ASSISTED CODING

ASSIGNMENT-4.3

2303A52315

M.AKHIL REDDY

BATCH-45

ASSIGNMENT-4.3

Task 1: Zero-Shot Prompting – Leap Year Check

Scenario

Zero-shot prompting involves giving instructions without providing examples.

Task Description

Use zero-shot prompting to instruct an AI tool to generate a Python function that:

• Accepts a year as input

• Checks whether the given year is a leap year

• Returns an appropriate result

Note: No input-output examples should be provided in the prompt.

Expected Output

• AI-generated leap year checking function

• Correct logical conditions

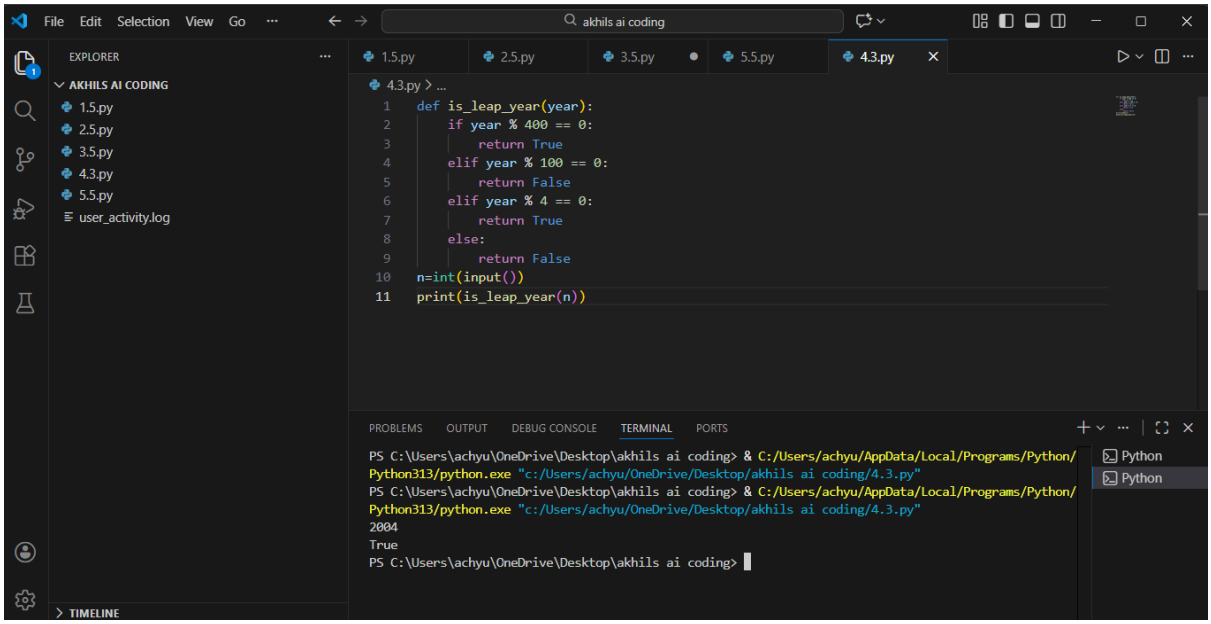
• Sample input and output

• Screenshot of AI-generated response (if required)

PROMPT:

Write a Python function that takes an integer representing a year as input and determines if it is a leap year. The function should return True if the year is a leap year and False otherwise. Use the standard rules for leap years: a year is a leap year if it is divisible by 4, but not by 100, unless it is also divisible by 400. Do not include any examples or test cases in the function code.

EXPECTED OUTPUT:



Task 2: One-Shot Prompting – Centimeters to Inches Conversion

Scenario

One-shot prompting guides AI using a single example.

Task Description

Use one-shot prompting by providing one input-output example to generate a Python

function that:

• Converts centimeters to inches

• Uses the correct mathematical formula

Example provided in prompt:

Input: 10 cm → Output: 3.94 inches

Expected Output

• Python function with correct conversion logic

• Accurate calculation

• Sample test cases and outputs

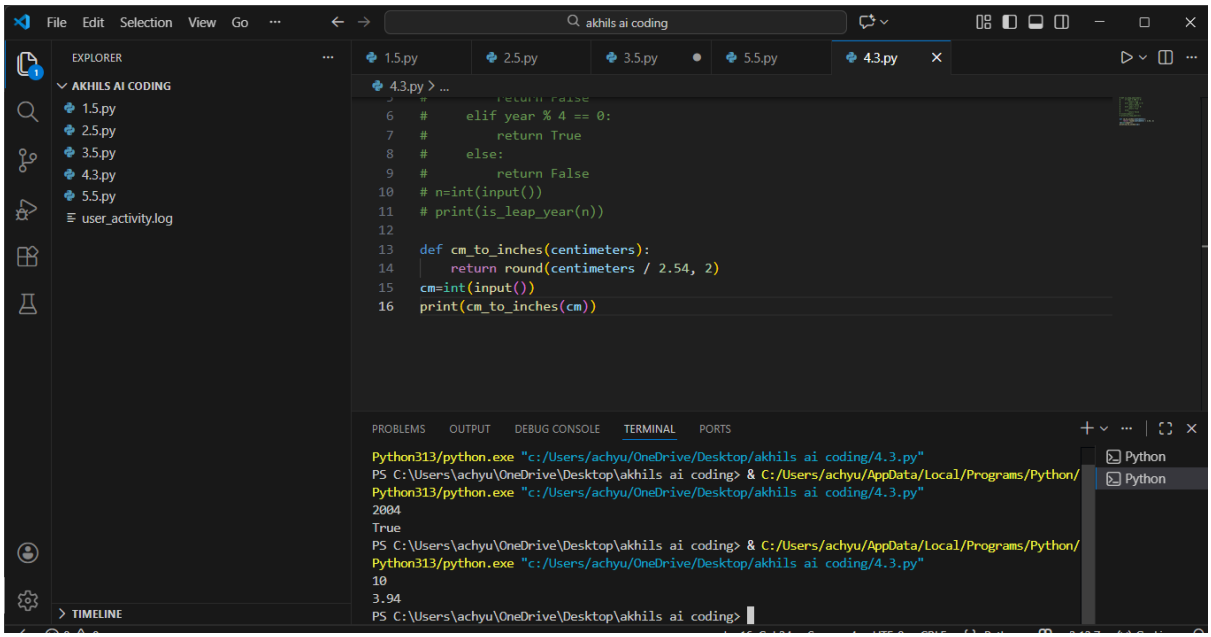
PROMPT:

Write a Python function that takes a float representing centimeters as input and converts it to inches. The function should return the value in inches rounded to two decimal places. Use the conversion formula: inches = centimeters / 2.54.

Example: Input: 10 Output: 3.94

Do not include any additional examples or test cases in the function code.

EXPECTED OUTPUT:



Task 3: Few-Shot Prompting – Name Formatting

Scenario

Few-shot prompting improves accuracy by providing multiple examples.

Task Description

Use few-shot prompting with 2–3 examples to generate a Python function that:

• Accepts a full name as input

• Formats it as “Last, First”

Example formats:

• "John Smith" → "Smith, John"

• "Anita Rao" → "Rao, Anita"

Expected Output

• Well-structured Python function

• Output strictly following example patterns

• Correct handling of names

• Sample inputs and outputs

PROMPT:

Write a Python function that takes a string representing a full name as input and formats it as "Last, First". The function should return the formatted string. Assume the input consists of exactly two names: first and last.

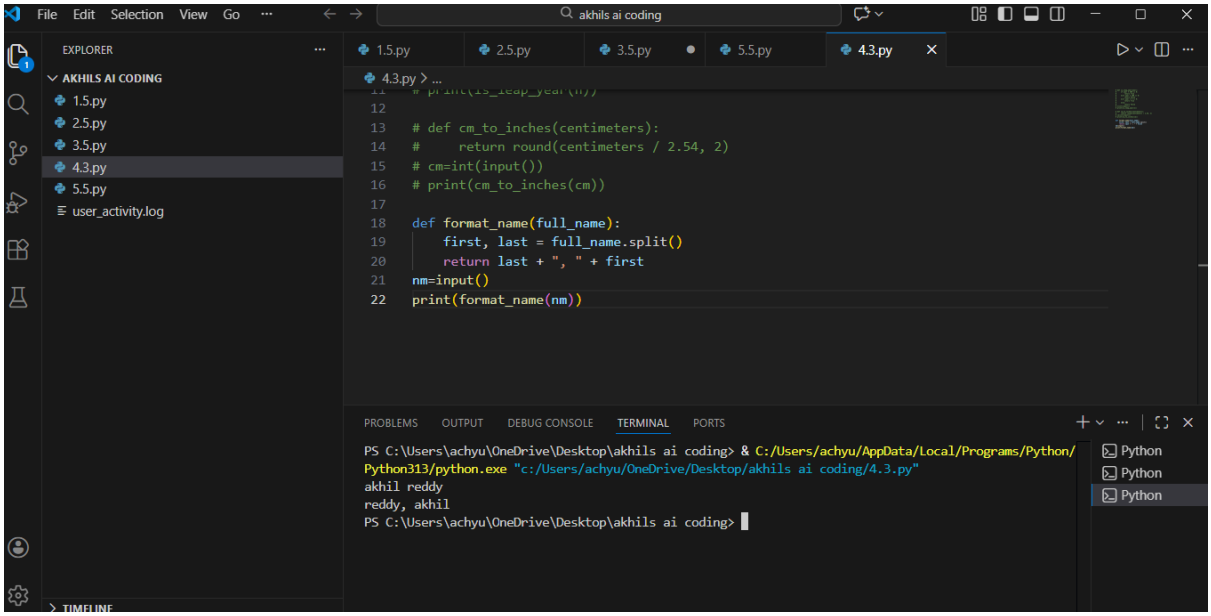
Examples: Input: "John Smith" Output: "Smith, John"

Input: "Anita Rao" Output: "Rao, Anita"

Input: "Michael Jordan" Output: "Jordan, Michael"

Do not include any additional examples or test cases in the function code.

EXPECTED OUTPUT:



Task 4: Comparative Analysis – Zero-Shot vs Few-Shot

Scenario

Different prompt strategies may produce different code quality.

Task Description

• Use zero-shot prompting to generate a function that counts vowels in a string

• Use few-shot prompting for the same problem

• Compare both outputs based on:

o Accuracy

o Readability

o Logical clarity

Expected Output

• Two vowel-counting functions

• Comparison table or short reflection paragraph

• Conclusion on prompt effectiveness

PROMPT:

Write a Python function that takes a string as input and returns the count of vowels in it. Vowels are a, e, i, o, u, considering both lowercase and uppercase. Do not include any examples or test cases in the function code.

Write a Python function that takes a string as input and returns the count of vowels in it. Vowels are a, e, i, o, u, considering both lowercase and uppercase.

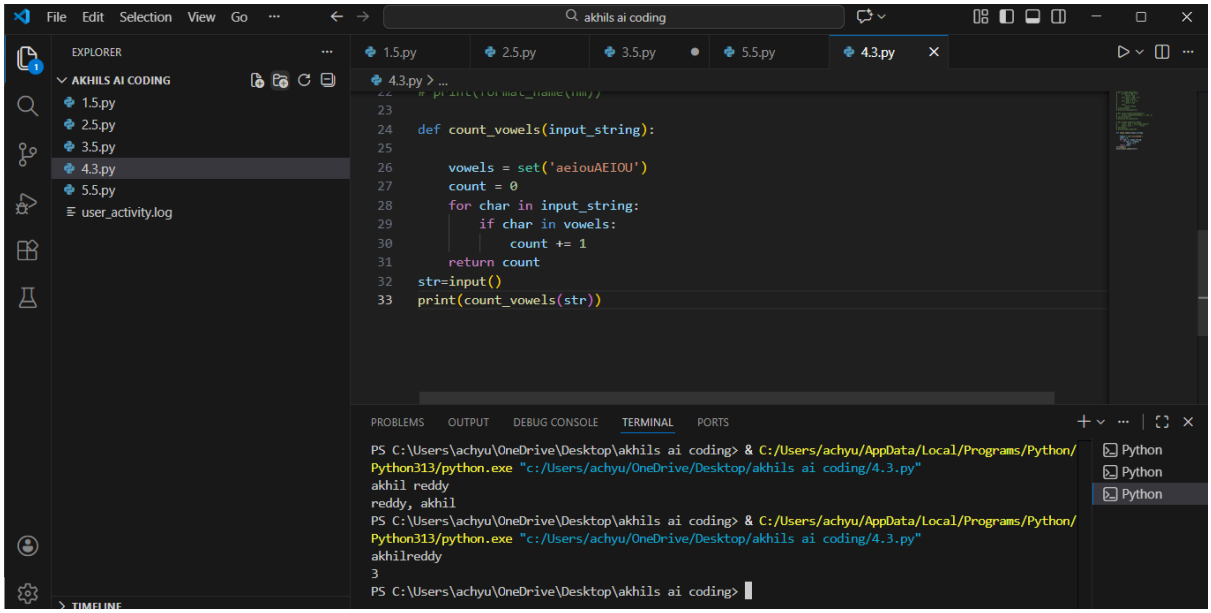
Examples: Input: "hello" Output: 2

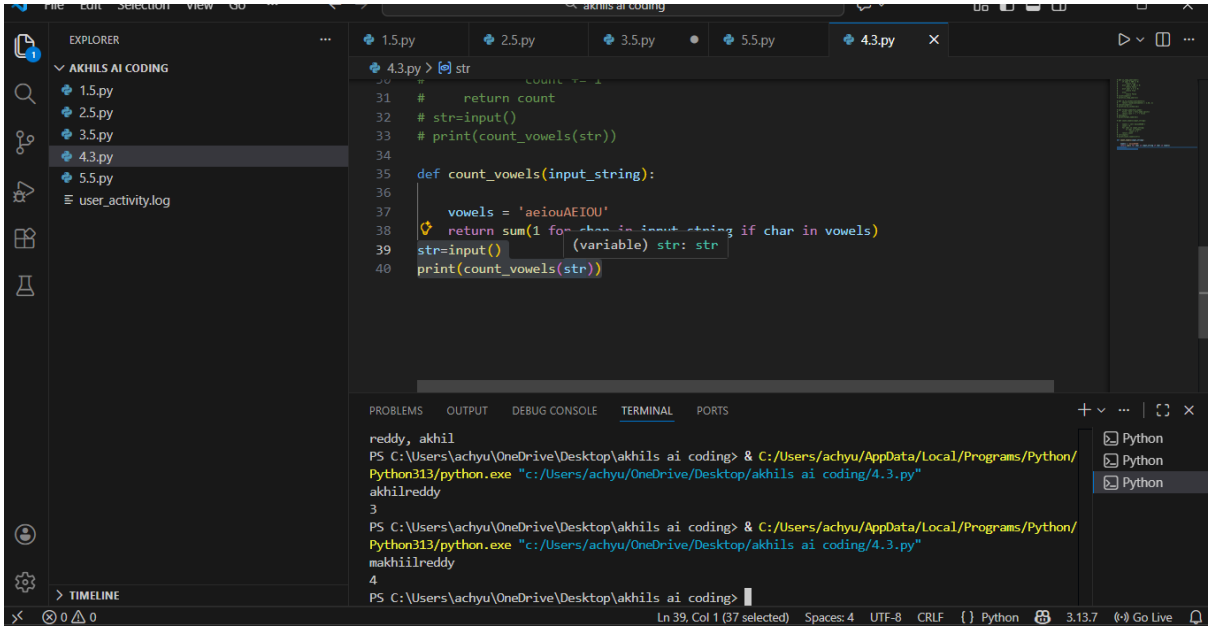
Input: "Python" Output: 1

Input: "AEIOU" Output: 5

Do not include any additional examples or test cases in the function code.

EXPECTED OUTPUT:





Comparison

Aspect	Zero-Shot Function	Few-Shot Function
Accuracy	Fully accurate; correctly handles case insensitivity and counts only vowels.	Fully accurate; identical logic ensures correct vowel counting with case insensitivity.
Readability	Readable with explicit loop and counter, but slightly verbose.	Highly readable; uses concise generator expression for counting, which is more Pythonic.
Logical Clarity	Clear step-by-step logic: initialize counter, iterate, check membership, increment.	Clear and efficient; leverages built-in sum with comprehension for straightforward logic.

Conclusion on Prompt Effectiveness

Both prompting strategies produced accurate and functional code, but few-shot prompting led to a more concise and idiomatic implementation, likely because the examples guided the AI toward optimized patterns. Zero-shot was effective for basic tasks but may require more refinement for elegance, while few-shot enhances reliability in capturing nuanced best practices. Overall, few-shot appears more effective for improving code quality in this scenario.

Task 5: Few-Shot Prompting – File Handling

Scenario

File processing requires clear logical understanding.

Task Description

Use few-shot prompting to generate a Python function that:

• Reads a .txt file

• Counts the number of lines in the file

• Returns the line count

Expected Output

• Working Python file-processing function

• Correct line count

• Sample .txt input and output

• AI-assisted logic explanation

PROMPT:

Write a Python function that takes a string representing the filename of a .txt file as input, reads the file, counts the number of lines in it, and returns the integer line count.

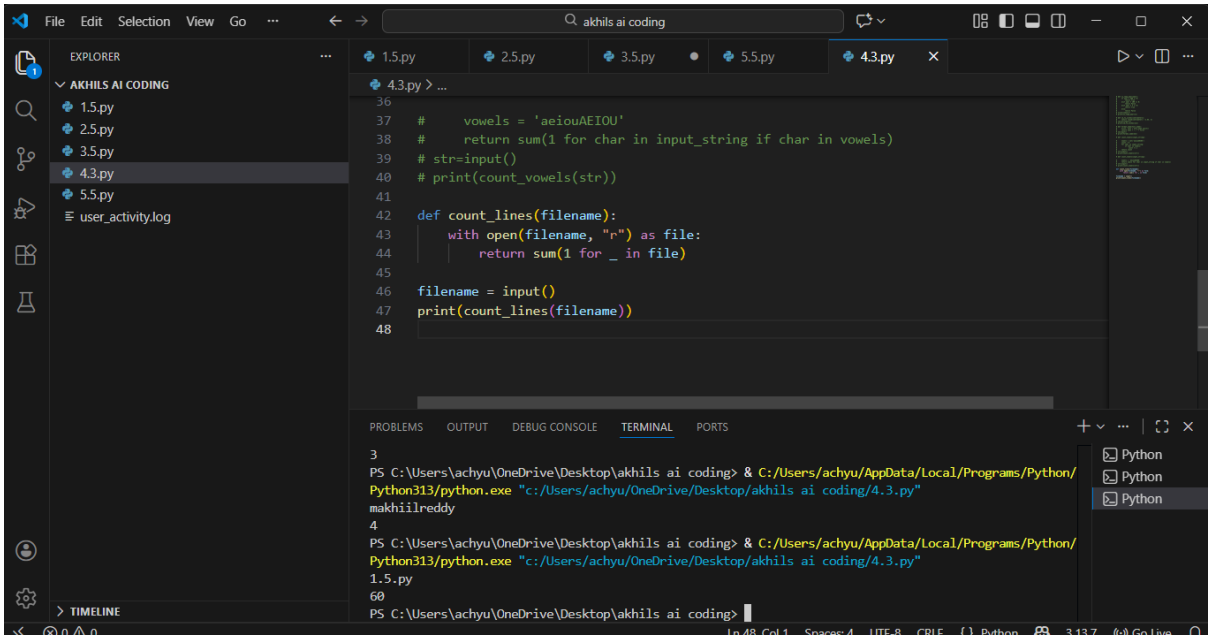
Examples: Input: "example1.txt" (assuming content with 3 lines: "Hello\nWorld\n!") Output: 3

Input: "example2.txt" (assuming content with 1 line: "Single line") Output: 1

Input: "example3.txt" (assuming empty file) Output: 0

Do not include any additional examples or test cases in the function code.

EXPECTED OUTPUT:


File Edit Selection View Go
akhils ai coding
EXPLORER
AKHILS AI CODING
1.5.py
2.5.py
3.5.py
4.3.py
5.5.py
user_activity.log
TIMELINE
1.5.py
2.5.py
3.5.py
5.5.py
4.3.py
4.3.py > ...
36
37 # vowels = 'aeiouAEIOU'
38 # return sum(1 for char in input_string if char in vowels)
39 # str=input()
40 # print(count_vowels(str))
41
42 def count_lines(filename):
43 with open(filename, "r") as file:
44 return sum(1 for _ in file)
45
46 filename = input()
47 print(count_lines(filename))
48
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
3
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding> & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/4.3.py"
makhiilreddy
4
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding> & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/4.3.py"
1.5.py
60
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding>
Python
Python
Python