2303A51504

Batch-25

Lab 4.3

Task 1: Zero-Shot Prompting – Leap Year Check

Scenario

Zero-shot prompting involves giving instructions without providing examples.

Task Description

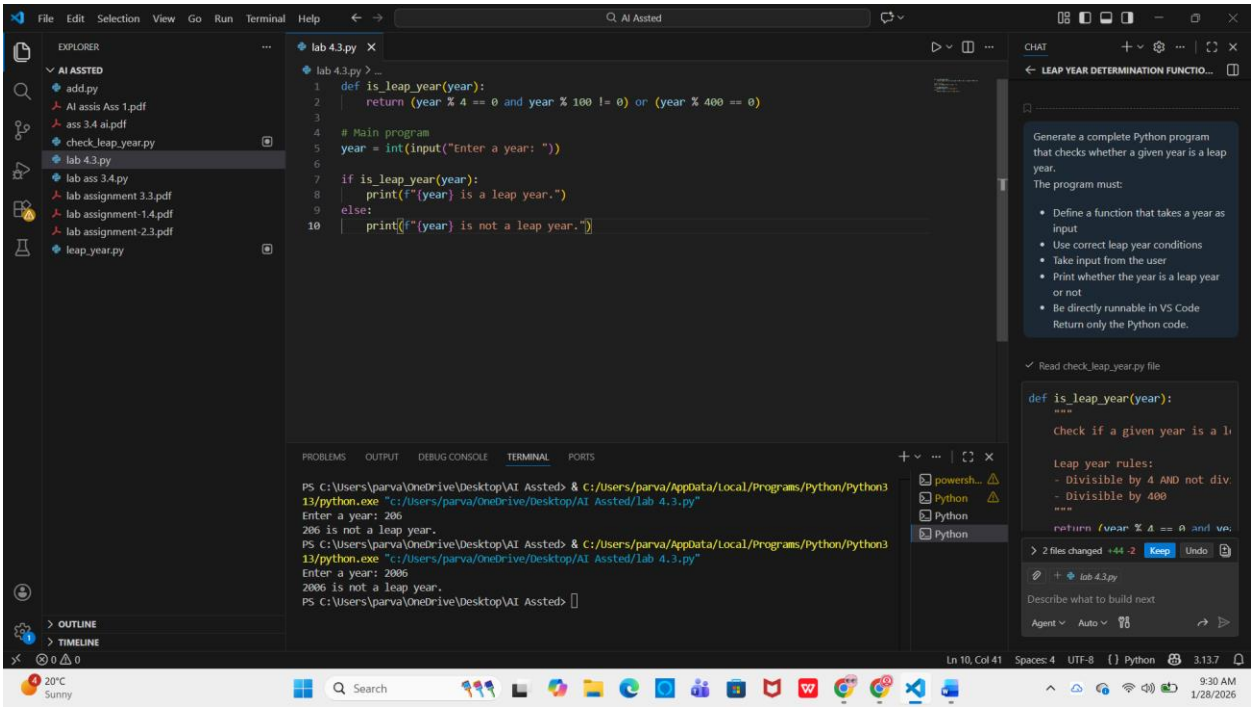
Use zero-shot prompting to instruct an AI tool to generate a Python function that:

- 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)



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:

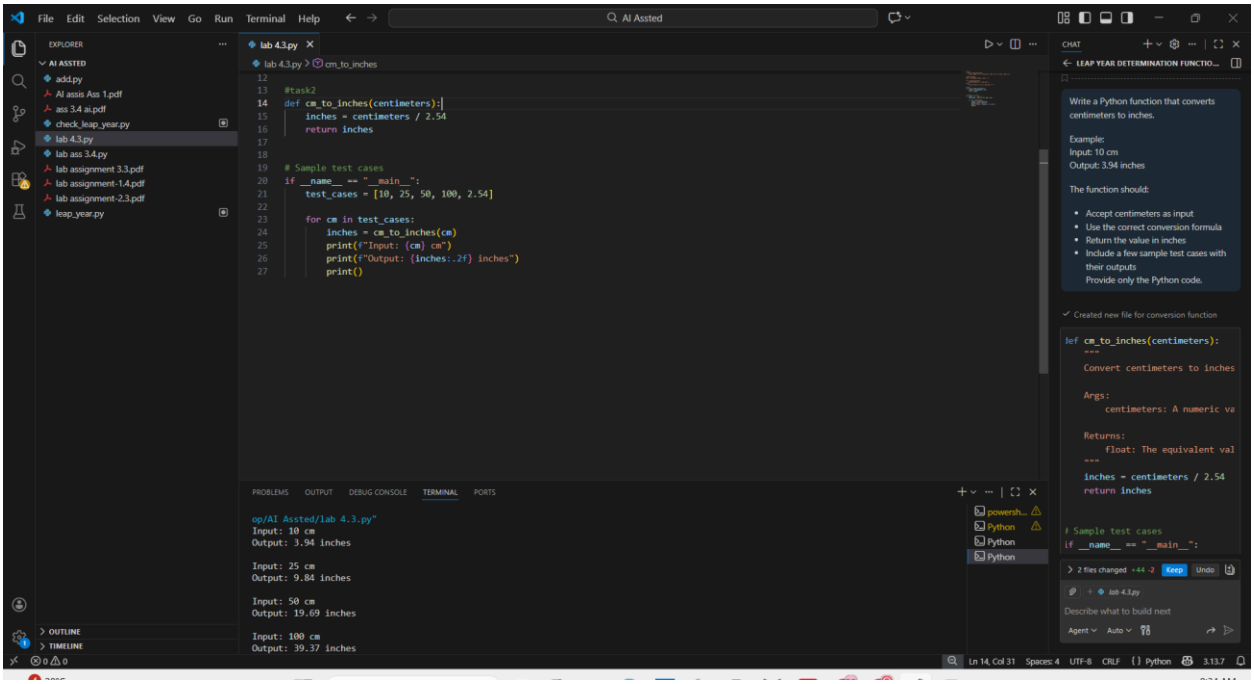
- Converts centimeters to inches
- Uses the correct mathematical formula

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



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"

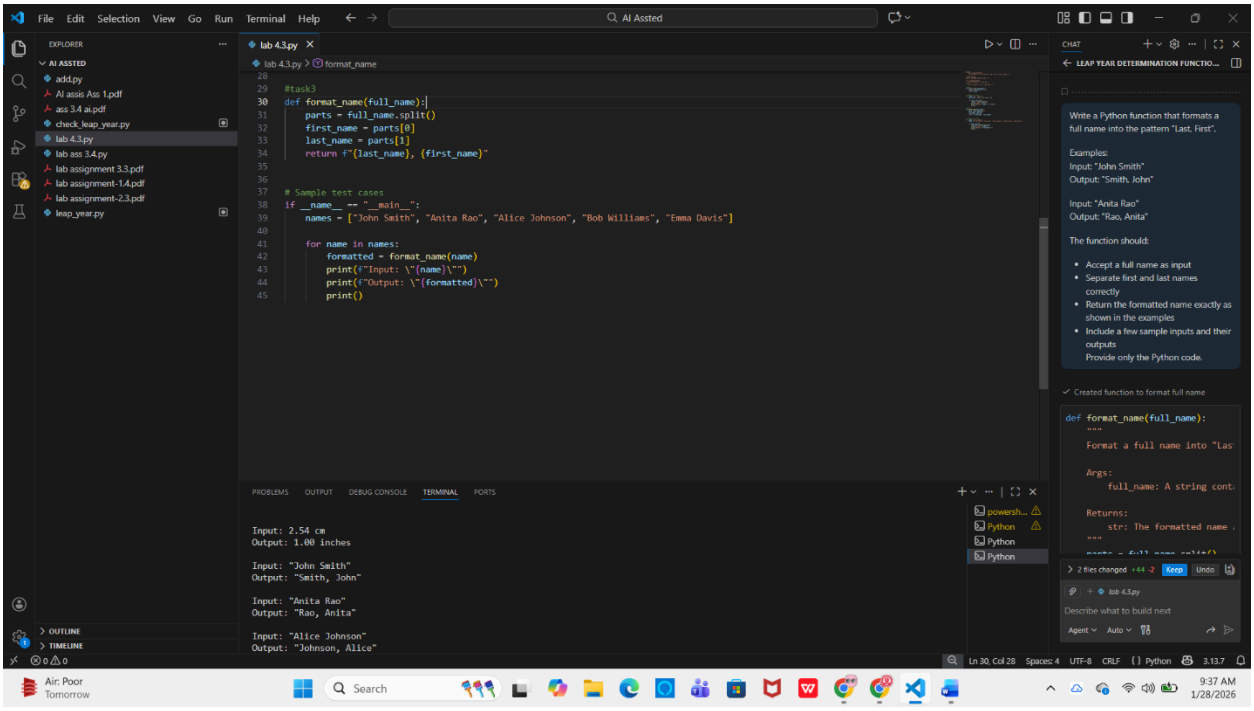
Expected Output

• Well-structured Python function

• Output strictly following example patterns

• Correct handling of names

• Sample inputs and outputs



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

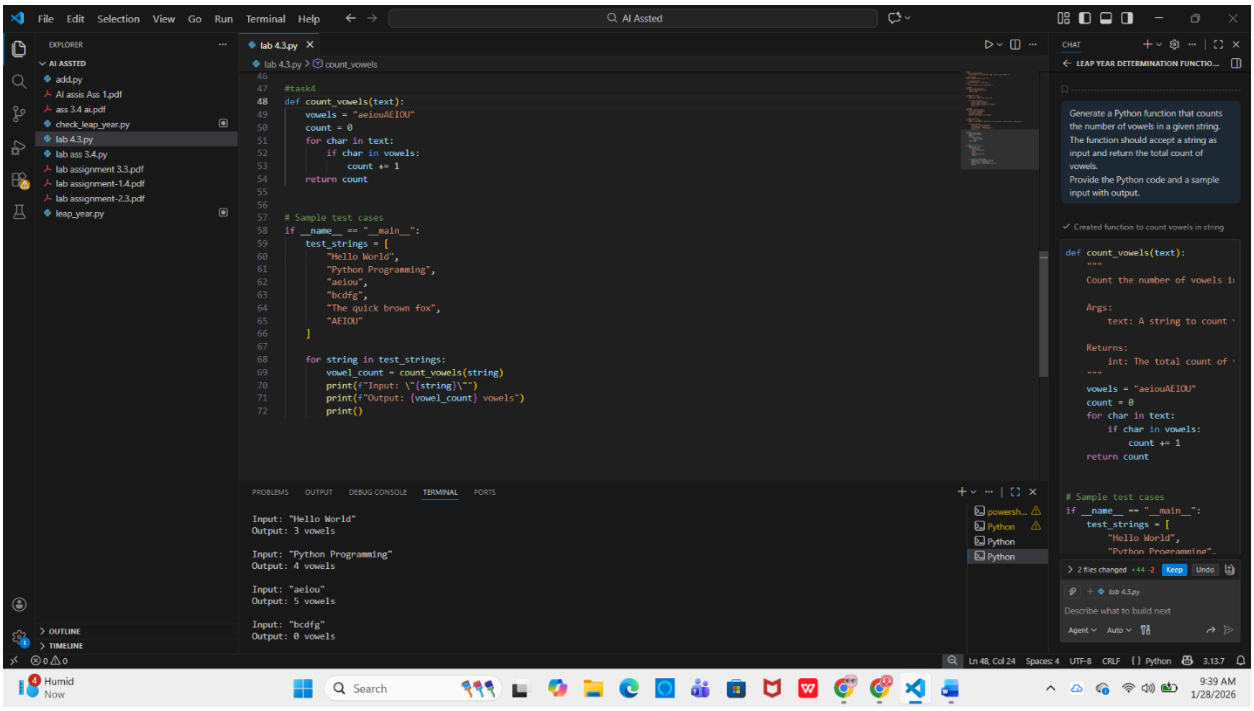
o Logical clarity

Expected Output

• Two vowel-counting functions

• Comparison table or short reflection paragraph

• Conclusion on prompt effectiveness



Task 5: Few-Shot Prompting – File Handling

Scenario

File processing requires clear logical understanding.

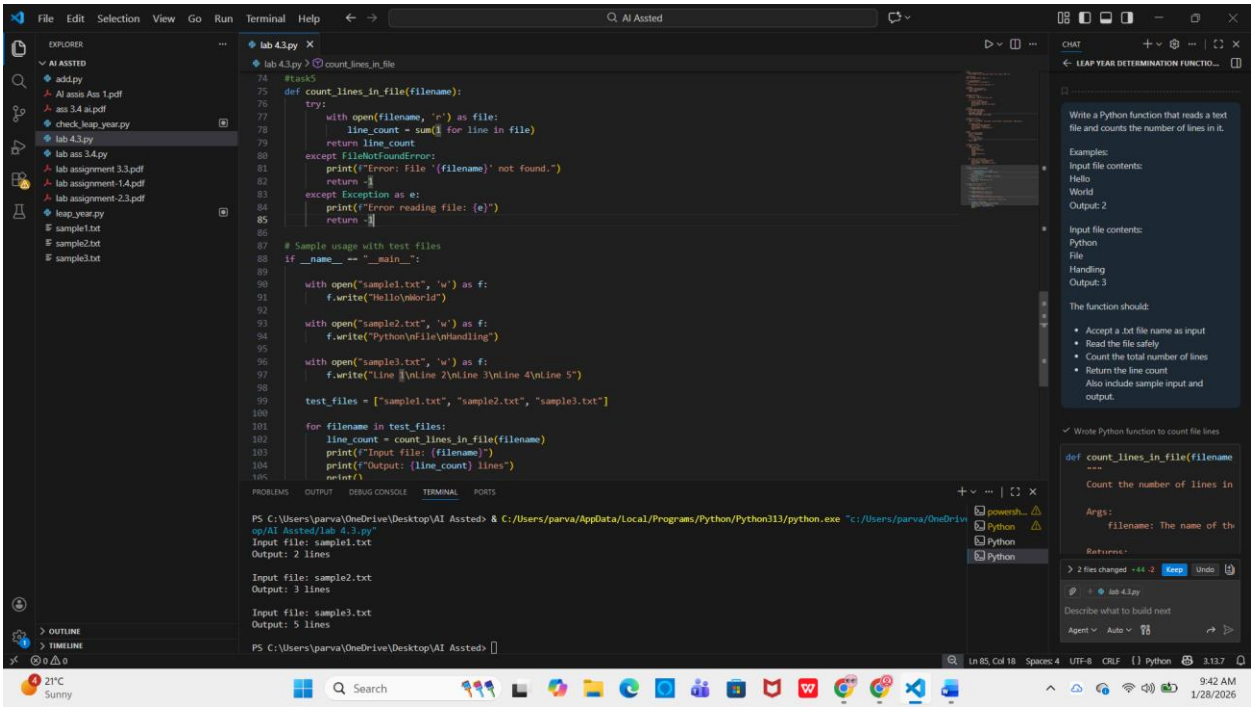
Task Description

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


File Edit Selection View Go Run Terminal Help
AI Assted
EXPLORER
AI ASSTED
add.py
AI assis Ass 1.pdf
ass 3.4 ai.pdf
check_leap_year.py
lab 4.3.py
lab ass 3.4.py
lab assignment 3.3.pdf
lab assignment-1.4.pdf
lab assignment-2.3.pdf
leap_year.py
sample1.txt
sample2.txt
sample3.txt
lab 4.3.py
lab 4.3.py > count_lines_in_file
#task5
def count_lines_in_file(filename):
try:
with open(filename, 'r') as file:
line_count = sum(1 for line in file)
return line_count
except FileNotFoundError:
print(f"Error: File '{filename}' not found.")
return -1
except Exception as e:
print(f"Error reading file: {e}")
return -1
# Sample usage with test files
if __name__ == "__main__":
with open("sample1.txt", 'w') as f:
f.write("Hello\nWorld")
with open("sample2.txt", 'w') as f:
f.write("Python\nFile\nHandling")
with open("sample3.txt", 'w') as f:
f.write("Line 1\nLine 2\nLine 3\nLine 4\nLine 5")
test_files = ["sample1.txt", "sample2.txt", "sample3.txt"]
for filename in test_files:
line_count = count_lines_in_file(filename)
print(f"Input file: {filename}")
print(f"Output: {line_count} lines")
print()
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\parva\OneDrive\Desktop\AI Assted> & C:/Users/parva/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/parva/OneDrive/Desktop/AI Assted/lab 4.3.py"
Input file: sample1.txt
Output: 2 lines
Input file: sample2.txt
Output: 3 lines
Input file: sample3.txt
Output: 5 lines
PS C:\Users\parva\OneDrive\Desktop\AI Assted>
powersh...
Python
Python
Python
OUTLINE
TIMELINE
Ln 85, Col 18 Spaces: 4 UTF-8 CRLF Python 3.13.7
CHAT
LEAP YEAR DETERMINATION FUNCTIO...
Write a Python function that reads a text file and counts the number of lines in it.
Examples:
Input file contents:
Hello
World
Output: 2
Input file contents:
Python
File
Handling
Output: 3
The function should:
Accept a .txt file name as input
Read the file safely
Count the total number of lines
Return the line count
Also include sample input and output.
Wrote Python function to count file lines
def count_lines_in_file(filename
Count the number of lines in
Args:
filename: The name of th
Returns:
2 files changed +64 -2 Keep Undo
lab 4.3.py
Describe what to build next
Agent Auto
21°C
Sunny
Search
9:42 AM
1/28/2026