SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE	**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING**

ProgramName:B. Tech	**Assignment Type: Lab**	**AcademicYear:**2025-2026

CourseCoordinatorName	Venkataramana Veeramsetty		
Instructor(s)Name	Dr. V. Venkataramana (Co-ordinator) Dr. T. Sampath Kumar Dr. Pramoda Patro Dr. Brij Kishor Tiwari Dr.J.Ravichander Dr. Mohammand Ali Shaik Dr. Anirodh Kumar Mr. S.Naresh Kumar Dr. RAJESH VELPULA Mr. Kundhan Kumar Ms. Ch.Rajitha Mr. M Prakash Mr. B.Raju Intern 1 (Dharma teja) Intern 2 (Sai Prasad) Intern 3 (Sowmya) NS_2 (Mounika)		
CourseCode	24CS002PC215	**CourseTitle**	AI Assisted Coding
Year/Sem	II/I	**Regulation**	R24
Date and Day of Assignment	Week2 - Wednesday	**Time(s)**	
Duration	2 Hours	**Applicableto Batches**	

AssignmentNumber:4.3(Present assignment number)/**24**(Total number of assignments)

Q.No.	**Question**	***ExpectedTime to complete***
1	Lab 4: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques **Lab Objectives:** ● To explore and apply different levels of prompt examples in AI-assisted code	Week2 - Wednesday

generation.

- To understand how zero-shot, one-shot, and few-shot prompting affect AI output quality.
- To evaluate the impact of context richness and example quantity on AI performance.
- To build awareness of prompt strategy effectiveness for different problem types.

Lab Outcomes (LOs):
After completing this lab, students will be able to:

- Use zero-shot prompting to instruct AI with minimal context.
- Use one-shot prompting with a single example to guide AI code generation.
- Apply few-shot prompting using multiple examples to improve AI responses.
- Compare AI outputs across the three prompting strategies.

Task Description#1

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

Expected Output#1

- AI-generated function with no examples provided

Task Description#2

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

Expected Output#2

- Function with correct conversion logic

Task Description#3

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

Expected Output#3

- Well-structured function respecting the examples

Task Description#4

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

Expected Output#4

- Functional output and comparative reflection

Task Description#5

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

Expected Output#5

- Working file-processing function with AI-guided logic

Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots

Evaluation Criteria:

Criteria	Max Marks

	Zero Shot (Task #1)	0.5		
	One Shot (Task#2)	0.5		
	Few Shot (Task#3 & Task #5)	1.0		
	Comparison (Task#4)	0.5		
	Total	**2.5 Marks**		

Roll no: 2403A510F4
Name: KOLA SNEHA

VS code with github copilot

Task Description#1

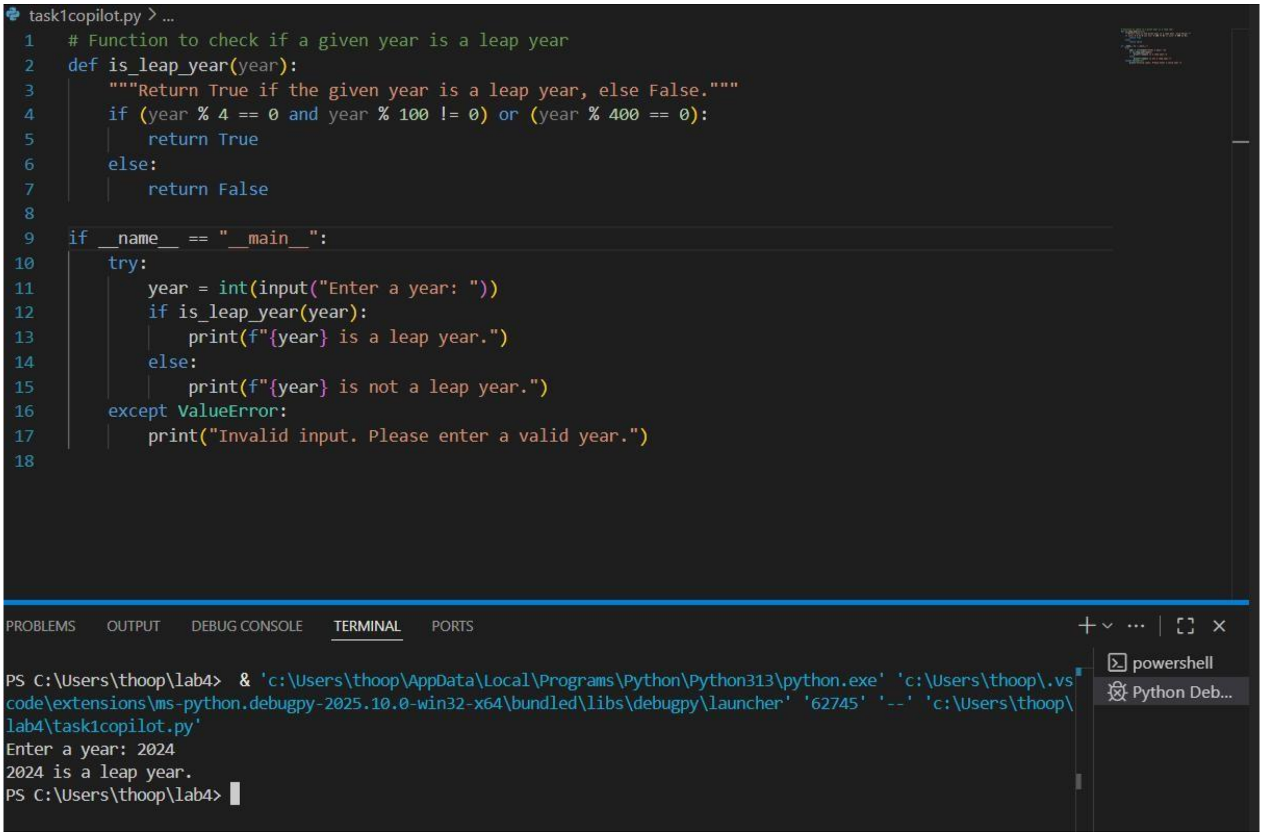
- Zero-shot: Prompt AI to write a function that checks whether a given year is a leap year.

Expected Output#1

- AI-generated function with no examples provided

Prompt:

write code to check whether given year is leap or not.



```python
# Function to check if a given year is a leap year
def is_leap_year(year):
    """Return True if the given year is a leap year, else False."""
    if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
        return True
    else:
        return False

if __name__ == "__main__":
    try:
        year = int(input("Enter a year: "))
        if is_leap_year(year):
            print(f"{year} is a leap year.")
        else:
            print(f"{year} is not a leap year.")
    except ValueError:
        print("Invalid input. Please enter a valid year.")
```

```
PS C:\Users\thoop\lab4>  & 'c:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\thoop\.vs
code\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '62745' '--' 'c:\Users\thoop\
lab4\task1copilot.py'
Enter a year: 2024
2024 is a leap year.
PS C:\Users\thoop\lab4> 
```

Task Description#2

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

Expected Output#2

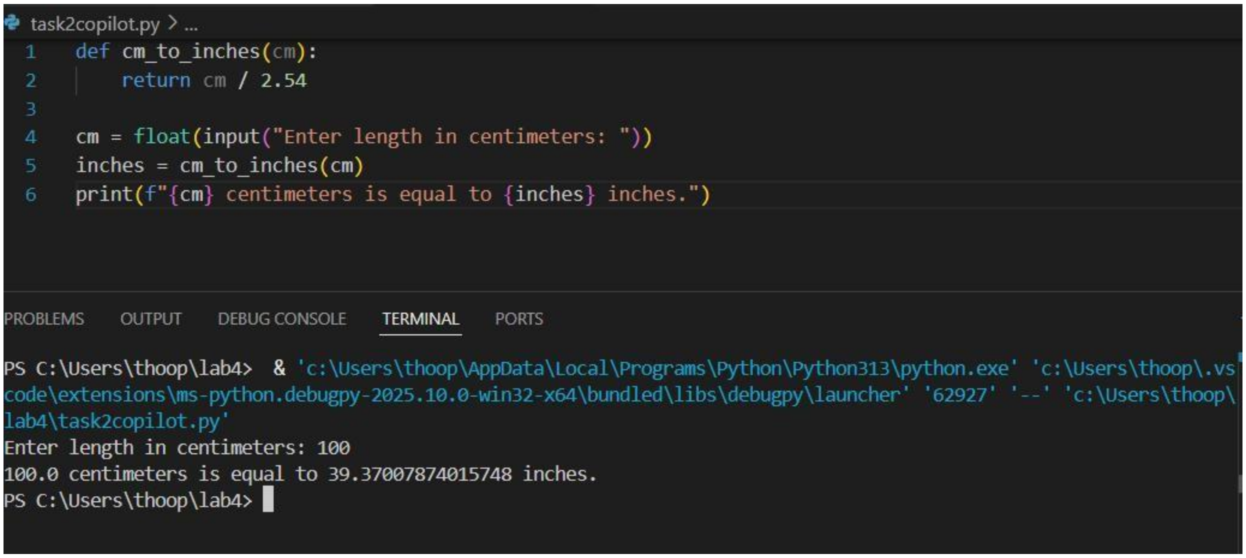
- Function with correct conversion logic

Prompt:

Write a function to convert centimeters to inches.

Example:10.0 centimeters is equal to 3.912345 inches.

read input from the user.



```python
def cm_to_inches(cm):
    return cm / 2.54

cm = float(input("Enter length in centimeters: "))
inches = cm_to_inches(cm)
print(f"{cm} centimeters is equal to {inches} inches.")
```

```
PS C:\Users\thoop\lab4>  & 'c:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\thoop\.vs
code\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '62927' '--' 'c:\Users\thoop\
lab4\task2copilot.py'
Enter length in centimeters: 100
100.0 centimeters is equal to 39.37007874015748 inches.
PS C:\Users\thoop\lab4> 
```

Task Description#3

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

Expected Output#3

- Well-structured function respecting the examples

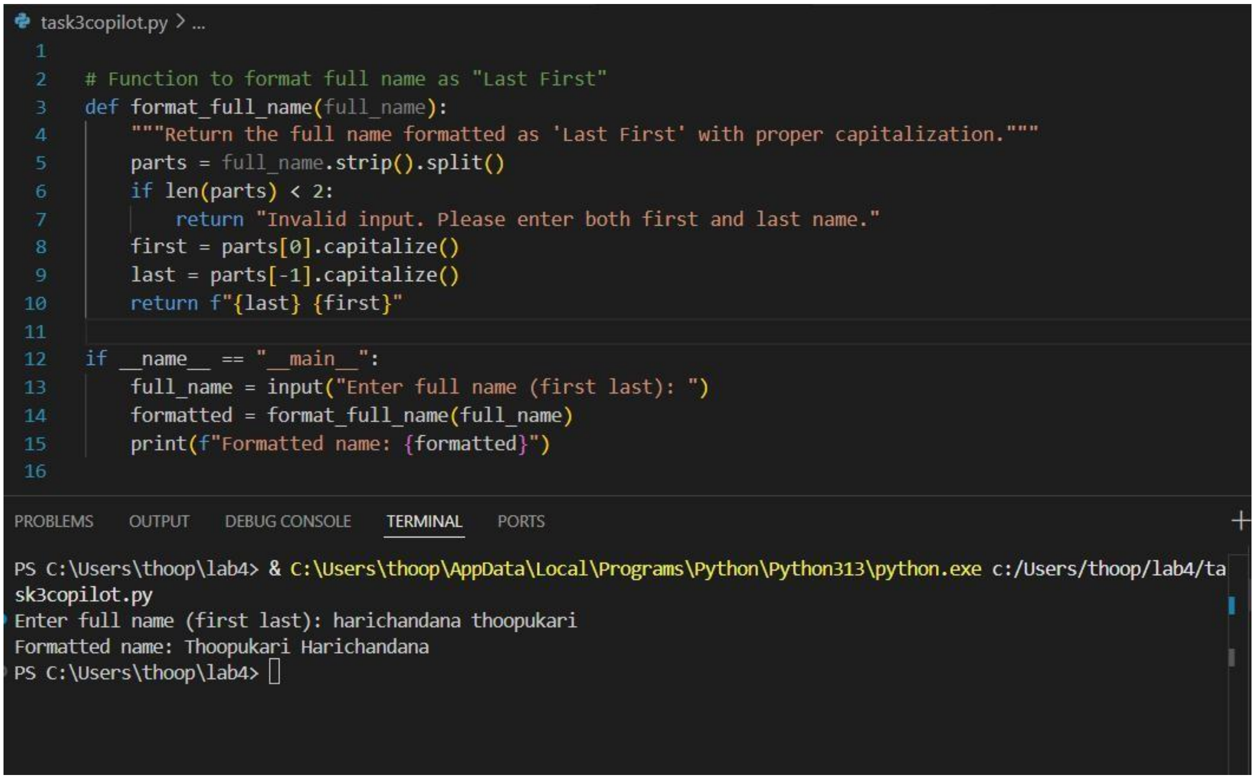
Prompt:

Write a python function that generates formats full names as “last,first”.

Example: peter parker =parker peter.
Harry potter=Potter Harry

Read full name from the user.

Code:



```python

# Function to format full name as "Last First"
def format_full_name(full_name):
    """Return the full name formatted as 'Last First' with proper capitalization."""
    parts = full_name.strip().split()
    if len(parts) < 2:
        return "Invalid input. Please enter both first and last name."
    first = parts[0].capitalize()
    last = parts[-1].capitalize()
    return f"{last} {first}"

if __name__ == "__main__":
    full_name = input("Enter full name (first last): ")
    formatted = format_full_name(full_name)
    print(f"Formatted name: {formatted}")

```

```
PS C:\Users\thoop\lab4> & C:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe c:/Users/thoop/lab4/ta
sk3copilot.py
Enter full name (first last): harichandana thoopukari
Formatted name: Thoopukari Harichandana
PS C:\Users\thoop\lab4> 
```

Task Description#4

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

Expected Output#4

- Functional output and comparative reflection

Prompt:

1.Zero shot:

Write a python program to count number of vowels in a string. read input from the user.

2.few-shot:

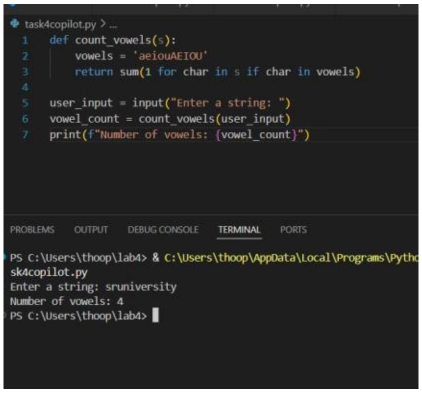
Write a python program to count number of vowels in a given string.

Example: sruniversity

Vowel count=4.

Read input from the user.

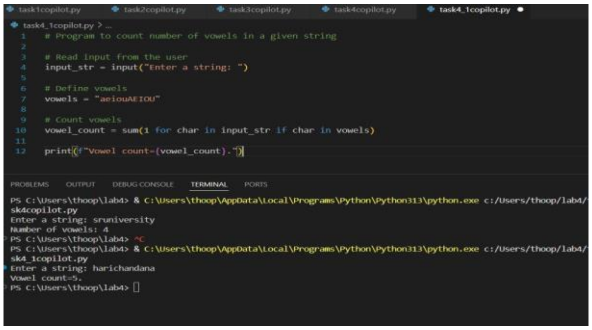
Zero shot



```python
def count_vowels(s):
    vowels = 'aeiouAEIOU'
    return sum(1 for char in s if char in vowels)

user_input = input("Enter a string: ")
vowel_count = count_vowels(user_input)
print(f"Number of vowels: {vowel_count}")
```

```
PS C:\Users\thoop\lab4> & C:\Users\thoop\AppData\Local\Programs\Pytho
sk4copilot.py
Enter a string: sruniversity
Number of vowels: 4
PS C:\Users\thoop\lab4> █
```

few-shot



```python
# Program to count number of vowels in a given string

# Read input from the user
input_str = input("Enter a string: ")

# Define vowels
vowels = "aeiouAEIOU"

# Count vowels
vowel_count = sum(1 for char in input_str if char in vowels)

print(f"Vowel count={vowel_count}.")
```

```
PS C:\Users\thoop\lab4> & C:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe c:/Users/thoop/lab4/
sk4copilot.py
Enter a string: sruniversity
Number of vowels: 4
PS C:\Users\thoop\lab4> ^C
PS C:\Users\thoop\lab4> & C:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe c:/Users/thoop/lab4/
sk4_1copilot.py
Enter a string: harichandana
Vowel count=5.
PS C:\Users\thoop\lab4> 
```

Task Description#5

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

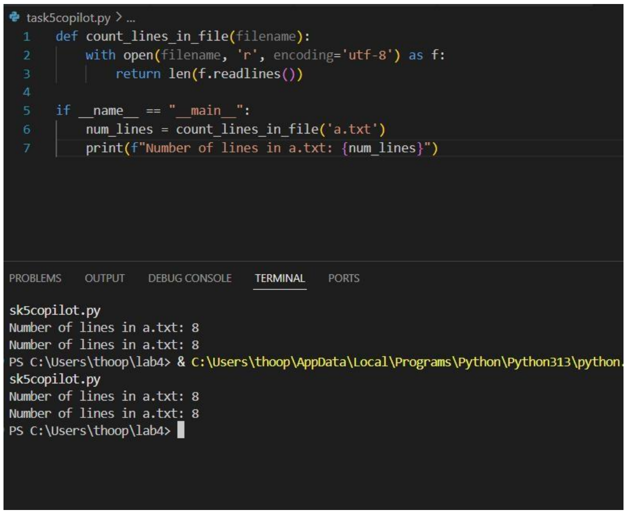
Expected Output#5

- Working file-processing function with AI-guided logic

Prompt:

Generate a function that reads the given a.txt file and returns the number of lines

Example:hello sr university.hanmakonda warangal.
Top engineering college.
Offers wide variety of courses.
No.of lines in a.txt=3.



```python
def count_lines_in_file(filename):
    with open(filename, 'r', encoding='utf-8') as f:
        return len(f.readlines())

if __name__ == "__main__":
    num_lines = count_lines_in_file('a.txt')
    print(f"Number of lines in a.txt: {num_lines}")
```

```
sk5copilot.py
Number of lines in a.txt: 8
Number of lines in a.txt: 8
PS C:\Users\thoop\lab4> & C:\Users\thoop\AppData\Local\Programs\Python\Python313\python.
sk5copilot.py
Number of lines in a.txt: 8
Number of lines in a.txt: 8
PS C:\Users\thoop\lab4>
```

Cursor AI

Task Description#1

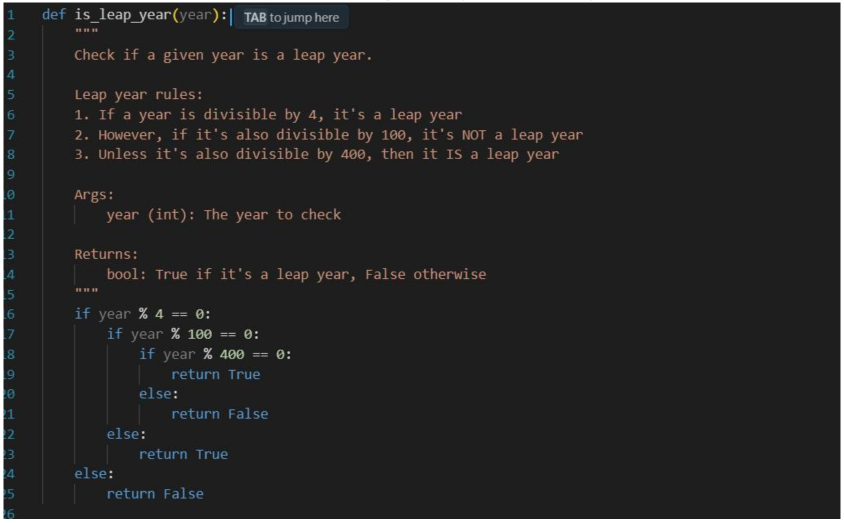
- Zero-shot: Prompt AI to write a function that checks whether a given year is a leap year.

Expected Output#1

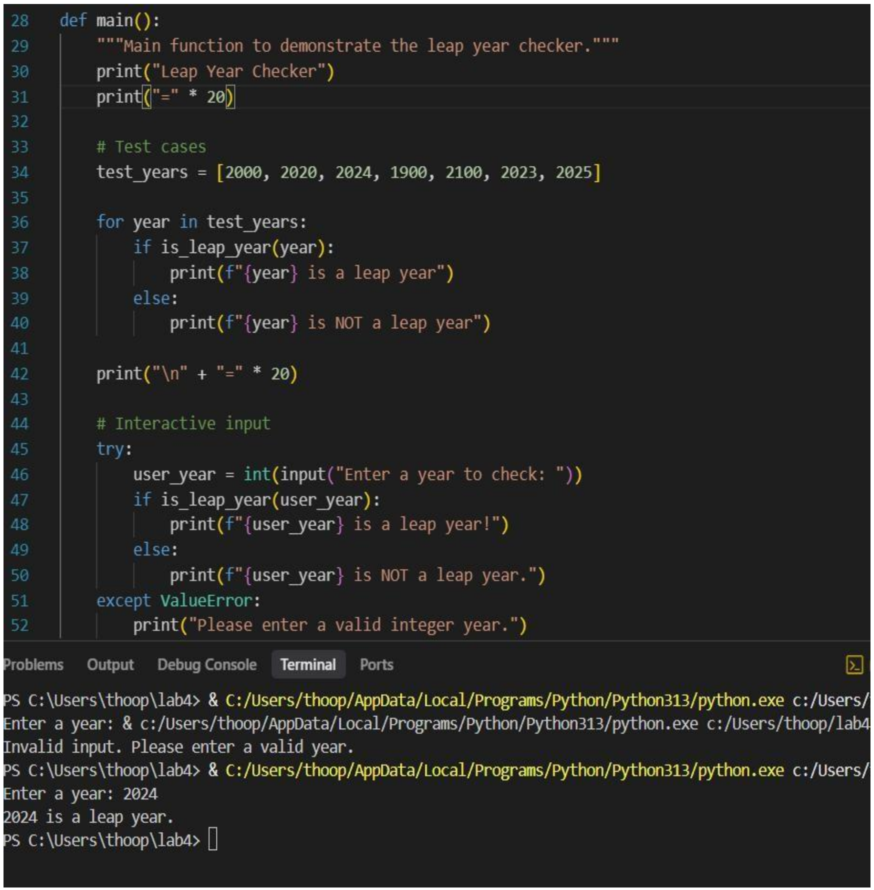
- AI-generated function with no examples provided

Prompt:

write code to check whether given year is leap or not.



```python
def is_leap_year(year):
    """
    Check if a given year is a leap year.

    Leap year rules:
    1. If a year is divisible by 4, it's a leap year
    2. However, if it's also divisible by 100, it's NOT a leap year
    3. Unless it's also divisible by 400, then it IS a leap year

    Args:
        year (int): The year to check

    Returns:
        bool: True if it's a leap year, False otherwise
    """
    if year % 4 == 0:
        if year % 100 == 0:
            if year % 400 == 0:
                return True
            else:
                return False
        else:
            return True
    else:
        return False
```



```python
def main():
    """Main function to demonstrate the leap year checker."""
    print("Leap Year Checker")
    print("=" * 20)

    # Test cases
    test_years = [2000, 2020, 2024, 1900, 2100, 2023, 2025]

    for year in test_years:
        if is_leap_year(year):
            print(f"{year} is a leap year")
        else:
            print(f"{year} is NOT a leap year")

    print("\n" + "=" * 20)

    # Interactive input
    try:
        user_year = int(input("Enter a year to check: "))
        if is_leap_year(user_year):
            print(f"{user_year} is a leap year!")
        else:
            print(f"{user_year} is NOT a leap year.")
    except ValueError:
        print("Please enter a valid integer year.")
```

```
PS C:\Users\thoop\lab4> & C:/Users/thoop/AppData/Local/Programs/Python/Python313/python.exe c:/Users/
Enter a year: & c:/Users/thoop/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thoop/lab4
Invalid input. Please enter a valid year.
PS C:\Users\thoop\lab4> & C:/Users/thoop/AppData/Local/Programs/Python/Python313/python.exe c:/Users/
Enter a year: 2024
2024 is a leap year.
PS C:\Users\thoop\lab4>
```

Task Description#2

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

Expected Output#2

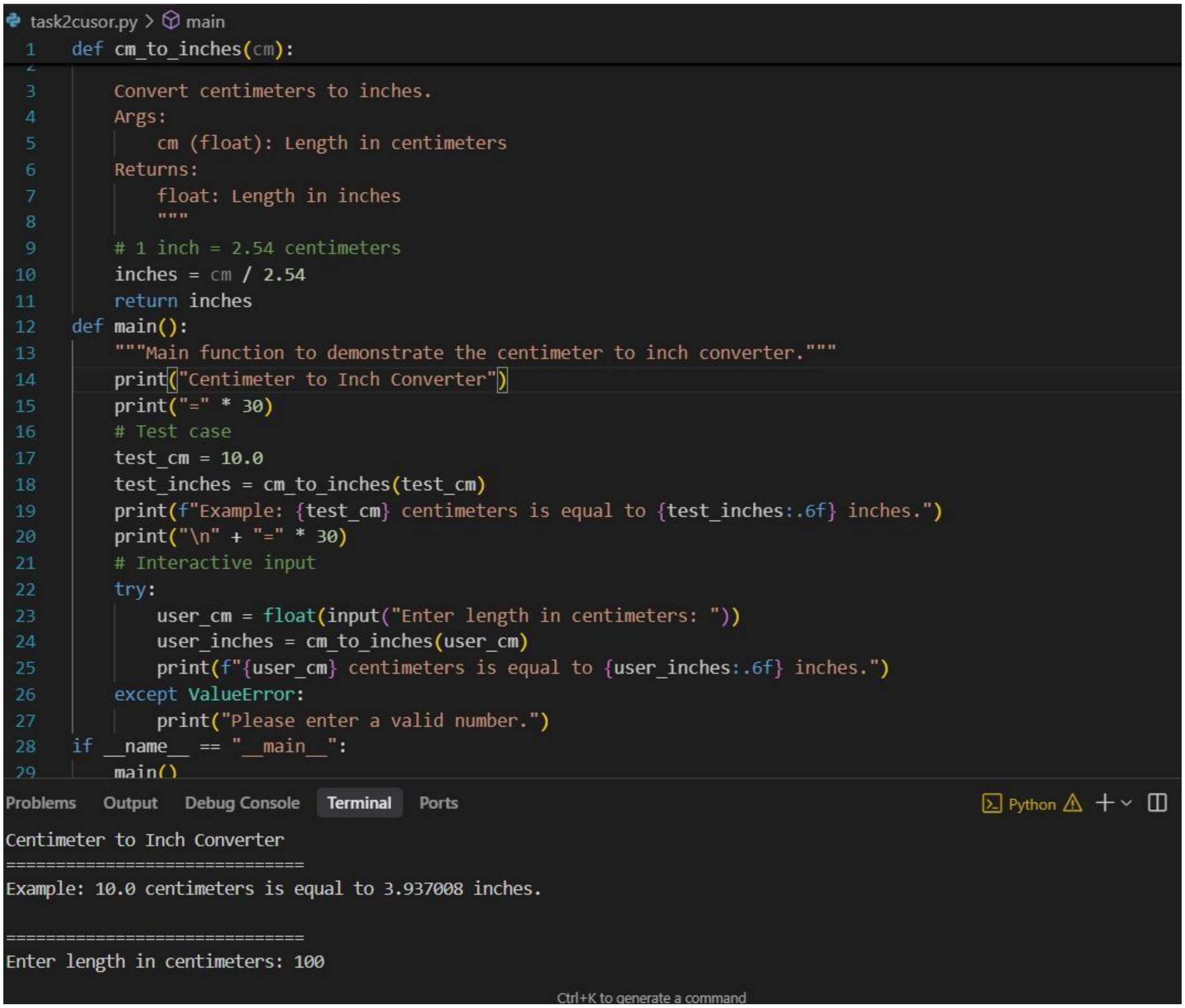
- Function with correct conversion logic

Prompt:

Write a function to convert centimeters to inches.

Example:10.0 centimeters is equal to 3.912345 inches.

read input from the user.



```python
def cm_to_inches(cm):
    Convert centimeters to inches.
    Args:
        cm (float): Length in centimeters
    Returns:
        float: Length in inches
        """
    # 1 inch = 2.54 centimeters
    inches = cm / 2.54
    return inches
def main():
    """Main function to demonstrate the centimeter to inch converter."""
    print("Centimeter to Inch Converter")
    print("=" * 30)
    # Test case
    test_cm = 10.0
    test_inches = cm_to_inches(test_cm)
    print(f"Example: {test_cm} centimeters is equal to {test_inches:.6f} inches.")
    print("\n" + "=" * 30)
    # Interactive input
    try:
        user_cm = float(input("Enter length in centimeters: "))
        user_inches = cm_to_inches(user_cm)
        print(f"{user_cm} centimeters is equal to {user_inches:.6f} inches.")
    except ValueError:
        print("Please enter a valid number.")
if __name__ == "__main__":
    main()
```

```
Centimeter to Inch Converter
==============================
Example: 10.0 centimeters is equal to 3.937008 inches.

==============================
Enter length in centimeters: 100
```

Task Description#3

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

Expected Output#3

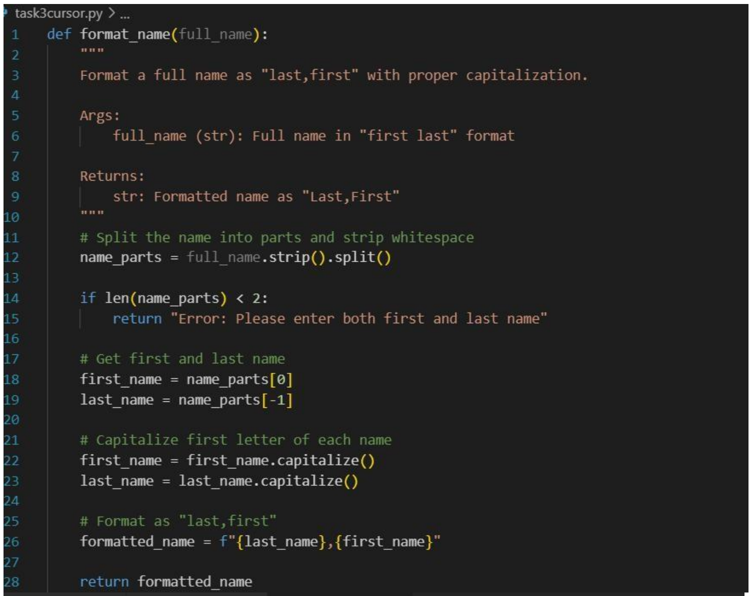
- Well-structured function respecting the examples

Prompt:

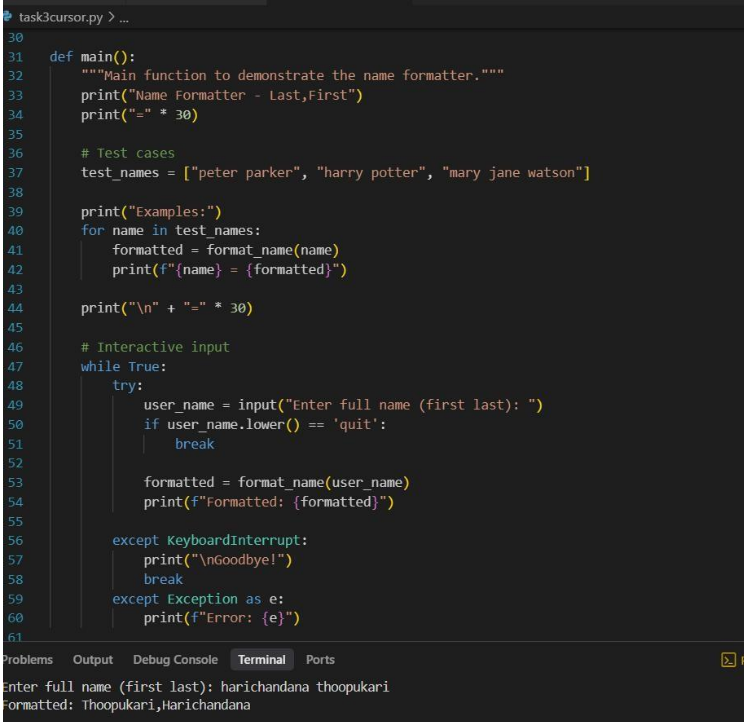
Write a python function that generates formats full names as “last,first”.

Example: peter parker =parker peter.
Harry potter=Potter Harry

Read full name from the user.



```python
def format_name(full_name):
    """
    Format a full name as "last,first" with proper capitalization.

    Args:
        full_name (str): Full name in "first last" format

    Returns:
        str: Formatted name as "Last,First"
    """
    # Split the name into parts and strip whitespace
    name_parts = full_name.strip().split()

    if len(name_parts) < 2:
        return "Error: Please enter both first and last name"

    # Get first and last name
    first_name = name_parts[0]
    last_name = name_parts[-1]

    # Capitalize first letter of each name
    first_name = first_name.capitalize()
    last_name = last_name.capitalize()

    # Format as "last,first"
    formatted_name = f"{last_name},{first_name}"

    return formatted_name
```



```python

def main():
    """Main function to demonstrate the name formatter."""
    print("Name Formatter - Last,First")
    print("=" * 30)

    # Test cases
    test_names = ["peter parker", "harry potter", "mary jane watson"]

    print("Examples:")
    for name in test_names:
        formatted = format_name(name)
        print(f"{name} = {formatted}")

    print("\n" + "=" * 30)

    # Interactive input
    while True:
        try:
            user_name = input("Enter full name (first last): ")
            if user_name.lower() == 'quit':
                break

            formatted = format_name(user_name)
            print(f"Formatted: {formatted}")

        except KeyboardInterrupt:
            print("\nGoodbye!")
            break
        except Exception as e:
            print(f"Error: {e}")

```

```
nter full name (first last): harichandana thoopukari
ormatted: Thoopukari,Harichandana
```

Task Description#4

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

Expected Output#4

- Functional output and comparative reflection

Prompt:

1. Zero shot:

Write a python program to count number of vowels in a string. read input from the user.

2. few-shot:

Write a python program to count number of vowels in a given string.

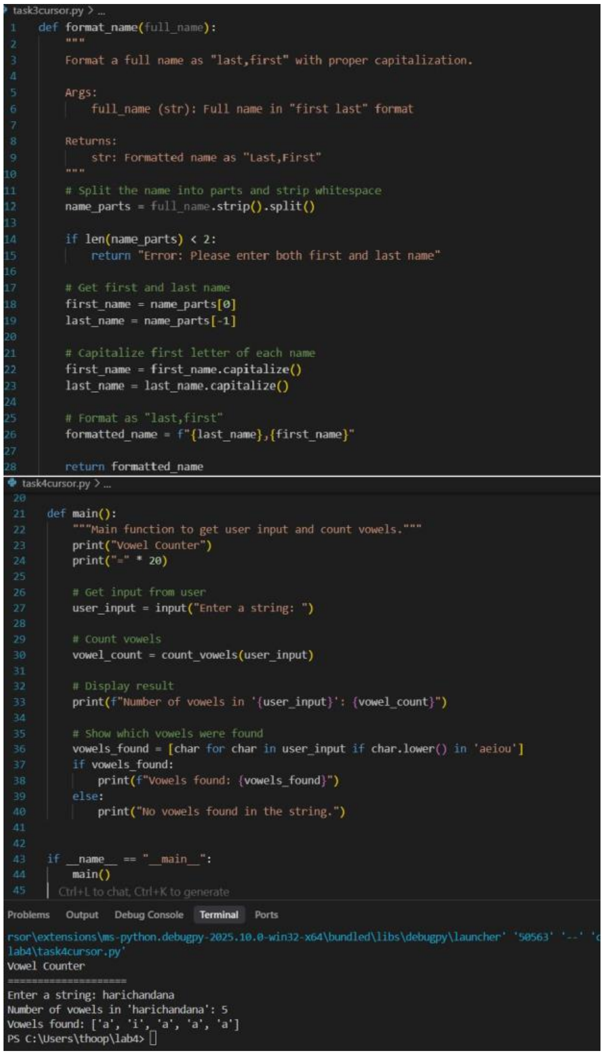
Example: sruniversity

Vowel count=4.

Read input from the user.

Zero-shot

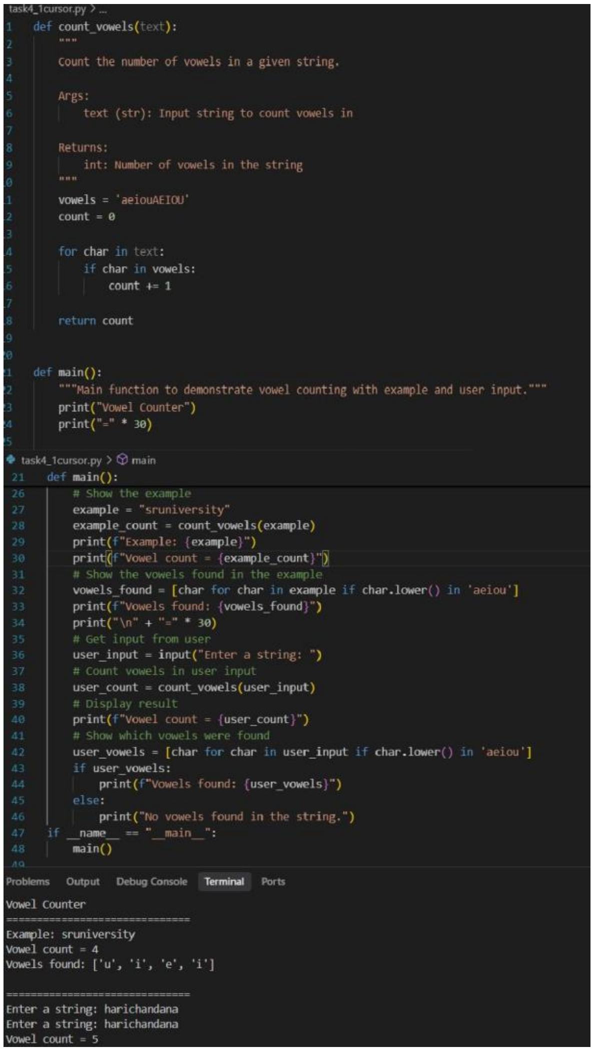
few-shot



```python
def format_name(full_name):
    """
    Format a full name as "last,first" with proper capitalization.

    Args:
        full_name (str): Full name in "first last" format

    Returns:
        str: Formatted name as "Last,First"
    """
    # Split the name into parts and strip whitespace
    name_parts = full_name.strip().split()

    if len(name_parts) < 2:
        return "Error: Please enter both first and last name"

    # Get first and last name
    first_name = name_parts[0]
    last_name = name_parts[-1]

    # Capitalize first letter of each name
    first_name = first_name.capitalize()
    last_name = last_name.capitalize()

    # Format as "last,first"
    formatted_name = f"{last_name},{first_name}"

    return formatted_name
```

```python

def main():
    """Main function to get user input and count vowels."""
    print("Vowel Counter")
    print("=" * 20)

    # Get input from user
    user_input = input("Enter a string: ")

    # Count vowels
    vowel_count = count_vowels(user_input)

    # Display result
    print(f"Number of vowels in '{user_input}': {vowel_count}")

    # Show which vowels were found
    vowels_found = [char for char in user_input if char.lower() in 'aeiou']
    if vowels_found:
        print(f"Vowels found: {vowels_found}")
    else:
        print("No vowels found in the string.")


if __name__ == "__main__":
    main()
```

```
rsor\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '50563' '--' 'c
lab4\task4cursor.py'
Vowel Counter
====================
Enter a string: harichandana
Number of vowels in 'harichandana': 5
Vowels found: ['a', 'i', 'a', 'a', 'a']
PS C:\Users\thoop\lab4> 
```



```python
def count_vowels(text):
    """
    Count the number of vowels in a given string.

    Args:
        text (str): Input string to count vowels in

    Returns:
        int: Number of vowels in the string
    """
    vowels = 'aeiouAEIOU'
    count = 0

    for char in text:
        if char in vowels:
            count += 1

    return count


def main():
    """Main function to demonstrate vowel counting with example and user input."""
    print("Vowel Counter")
    print("=" * 30)

```

```python
def main():
    # Show the example
    example = "sruniversity"
    example_count = count_vowels(example)
    print(f"Example: {example}")
    print(f"Vowel count = {example_count}")
    # Show the vowels found in the example
    vowels_found = [char for char in example if char.lower() in 'aeiou']
    print(f"Vowels found: {vowels_found}")
    print("\n" + "=" * 30)
    # Get input from user
    user_input = input("Enter a string: ")
    # Count vowels in user input
    user_count = count_vowels(user_input)
    # Display result
    print(f"Vowel count = {user_count}")
    # Show which vowels were found
    user_vowels = [char for char in user_input if char.lower() in 'aeiou']
    if user_vowels:
        print(f"Vowels found: {user_vowels}")
    else:
        print("No vowels found in the string.")
if __name__ == "__main__":
    main()
```

```
Vowel Counter
==============================
Example: sruniversity
Vowel count = 4
Vowels found: ['u', 'i', 'e', 'i']

==============================
Enter a string: harichandana
Enter a string: harichandana
Vowel count = 5
```

Task Description#5

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

Expected Output#5

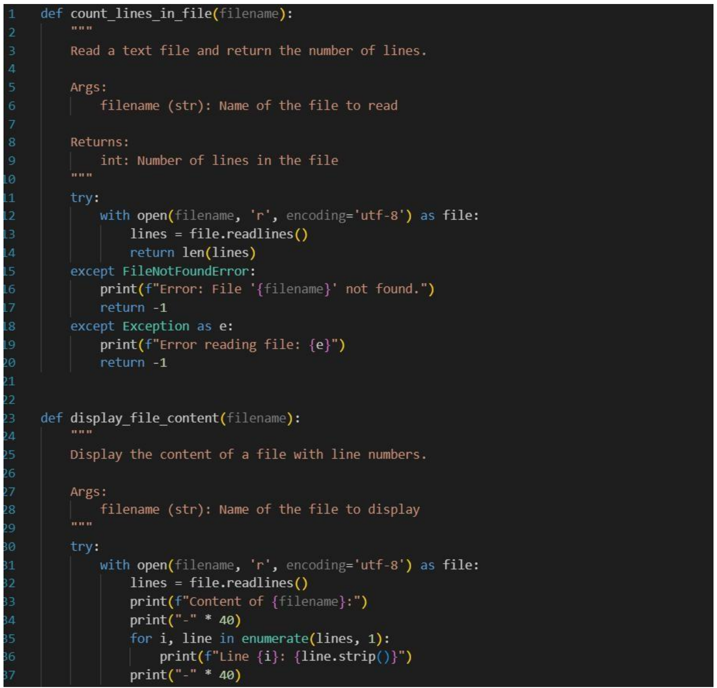
- Working file-processing function with AI-guided logic

Prompt:

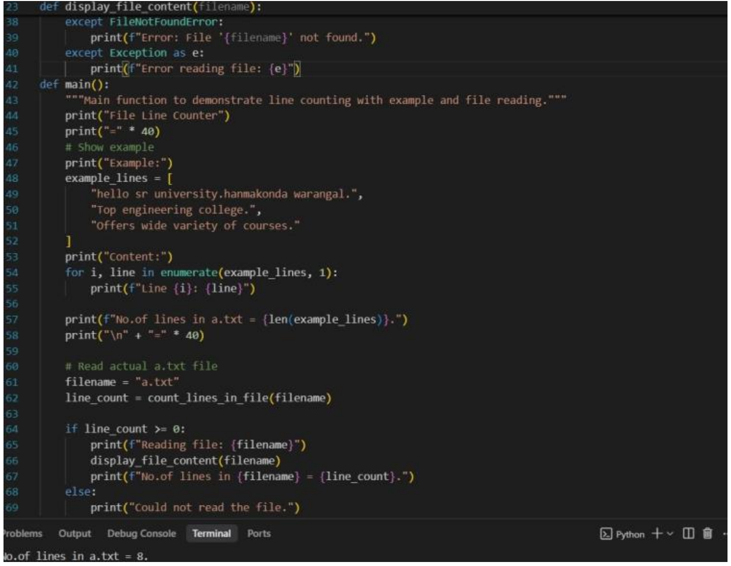
Generate a function that reads the given a.txt file and returns the number of lines

Example:hello sr university.hanmakonda warangal.
Top engineering college.
Offers wide variety of courses.

No.of lines in a.txt=3.



```python
def count_lines_in_file(filename):
    """
    Read a text file and return the number of lines.

    Args:
        filename (str): Name of the file to read

    Returns:
        int: Number of lines in the file
    """
    try:
        with open(filename, 'r', encoding='utf-8') as file:
            lines = file.readlines()
            return len(lines)
    except FileNotFoundError:
        print(f"Error: File '{filename}' not found.")
        return -1
    except Exception as e:
        print(f"Error reading file: {e}")
        return -1


def display_file_content(filename):
    """
    Display the content of a file with line numbers.

    Args:
        filename (str): Name of the file to display
    """
    try:
        with open(filename, 'r', encoding='utf-8') as file:
            lines = file.readlines()
            print(f"Content of {filename}:")
            print("-" * 40)
            for i, line in enumerate(lines, 1):
                print(f"Line {i}: {line.strip()}")
            print("-" * 40)
```



```python
def display_file_content(filename):
    except FileNotFoundError:
        print(f"Error: File '{filename}' not found.")
    except Exception as e:
        print(f"Error reading file: {e}")
def main():
    """Main function to demonstrate line counting with example and file reading."""
    print("File Line Counter")
    print("=" * 40)
    # Show example
    print("Example:")
    example_lines = [
        "hello sr university.hanmakonda warangal.",
        "Top engineering college.",
        "Offers wide variety of courses."
    ]
    print("Content:")
    for i, line in enumerate(example_lines, 1):
        print(f"Line {i}: {line}")

    print(f"No.of lines in a.txt = {len(example_lines)}.")
    print("\n" + "=" * 40)

    # Read actual a.txt file
    filename = "a.txt"
    line_count = count_lines_in_file(filename)

    if line_count >= 0:
        print(f"Reading file: {filename}")
        display_file_content(filename)
        print(f"No.of lines in {filename} = {line_count}.")
    else:
        print("Could not read the file.")
```

```
o.of lines in a.txt = 8.
```