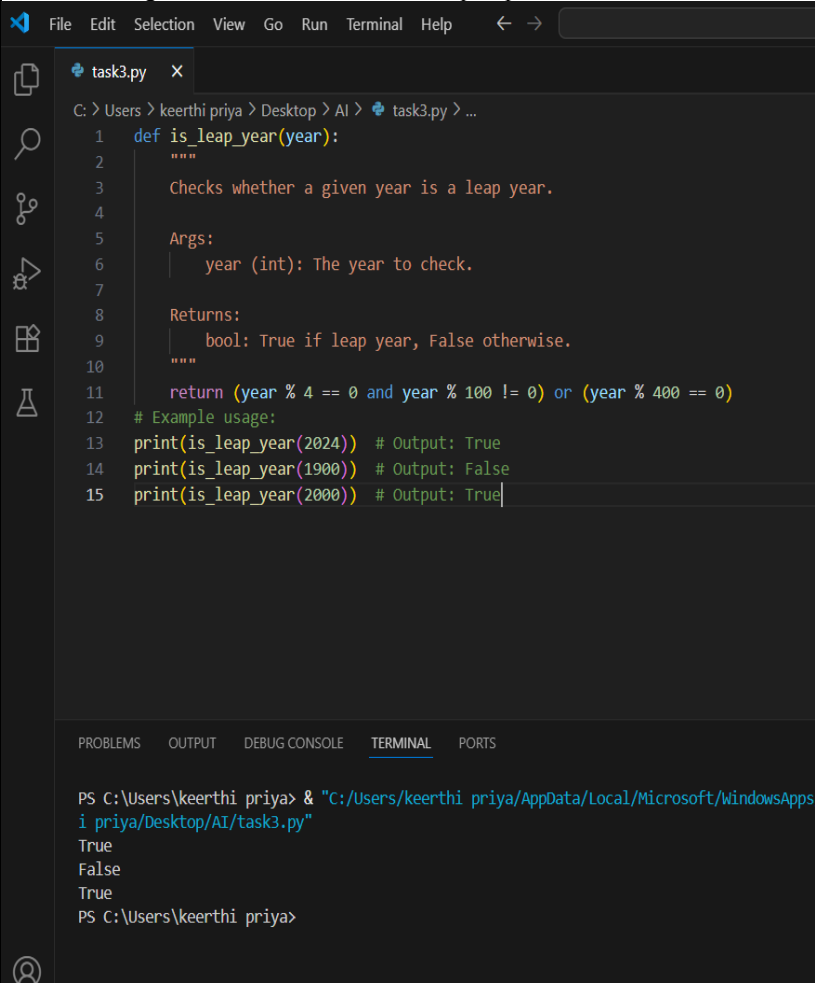



	<div>AI ASSISTED CODING</div> <div>NAME : K. Shiva Shankar</div> <div>ROLL NO:2403A510E5</div> <div>ASSIGNMENT:4.3</div>	
	<div><div>Task Description#1</div><div><ul style="list-style-type: none">Zero-shot: Prompt AI to write a function that checks whether a given year is a leap year.</div><div><div>Expected Output#1</div><div><ul style="list-style-type: none">AI-generated function with no examples provided</div></div><div><p>The screenshot shows a VS Code editor window with a file named 'task3.py'. The code defines a function 'is_leap_year(year)' that checks if a year is a leap year. The function includes a docstring with a description, arguments, and return values. The logic uses the formula: (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0). Below the function, there are example usage lines: 'print(is_leap_year(2024))', 'print(is_leap_year(1900))', and 'print(is_leap_year(2000))'. The terminal at the bottom shows the command 'PS C:\Users\keerthi priya> & "C:/Users/keerthi priya/AppData/Local/Microsoft/WindowsApps/i priya/Desktop/AI/task3.py"' and the output: 'True', 'False', 'True'.</p></div><div><div>Task Description#2</div><div><ul style="list-style-type: none">One-shot: Give one input-output example to guide AI in writing a function that converts centimeters to inches.</div><div><div>Expected Output#2</div><div><ul style="list-style-type: none">Function with correct conversion logic</div></div></div></div>	

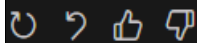


writing a function that converts centimeters to inches. ONLY THIS CODE

 task3.py

> Used 1 reference

```
def cm_to_inches(cm):  
    """  
    Converts centimeters to inches.  
  
    Args:  
        cm (float): Length in centimeters.  
  
    Returns:  
        float: Length in inches.  
    """  
    return cm / 2.54
```



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



Restore Checkpoint

generate a PYTHON function that formats full names as "Last, First".

 task3.py

> Used 1 reference

```
def format_full_name(first_name, last_name):  
    """  
    Formats full names as 'Last, First'.  
  
    Args:  
        first_name (str): First name.  
        last_name (str): Last name.  
  
    Returns:  
        str: Formatted name.  
    """  
    return f"{last_name}, {first_name}"
```

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



Restore Checkpoint

write a function that counts the number of vowels in a string.

 task3.py

> Used 1 reference

```
def count_vowels(s):  
    """  
    Counts the number of vowels in a string.  
  
    Args:  
        s (str): Input string.  
  
    Returns:  
        int: Number of vowels in the string.  
    """  
    vowels = 'aeiouAEIOU'  
    return sum(1 for char in s if char in vowels)
```

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



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

 task3.py

> Used 1 reference

```
def count_lines_in_file(filename):  
    """  
    Reads a .txt file and returns the number of lines  
  
    Args:  
        filename (str): Path to the .txt file.  
  
    Returns:  
        int: Number of lines in the file.  
    """  
    with open(filename, 'r', encoding='utf-8') as fil  
        return sum(1 for _ in file)
```

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