

Assignment 4.3

2303A51172

T.Rakshitha

Batch – 18

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:

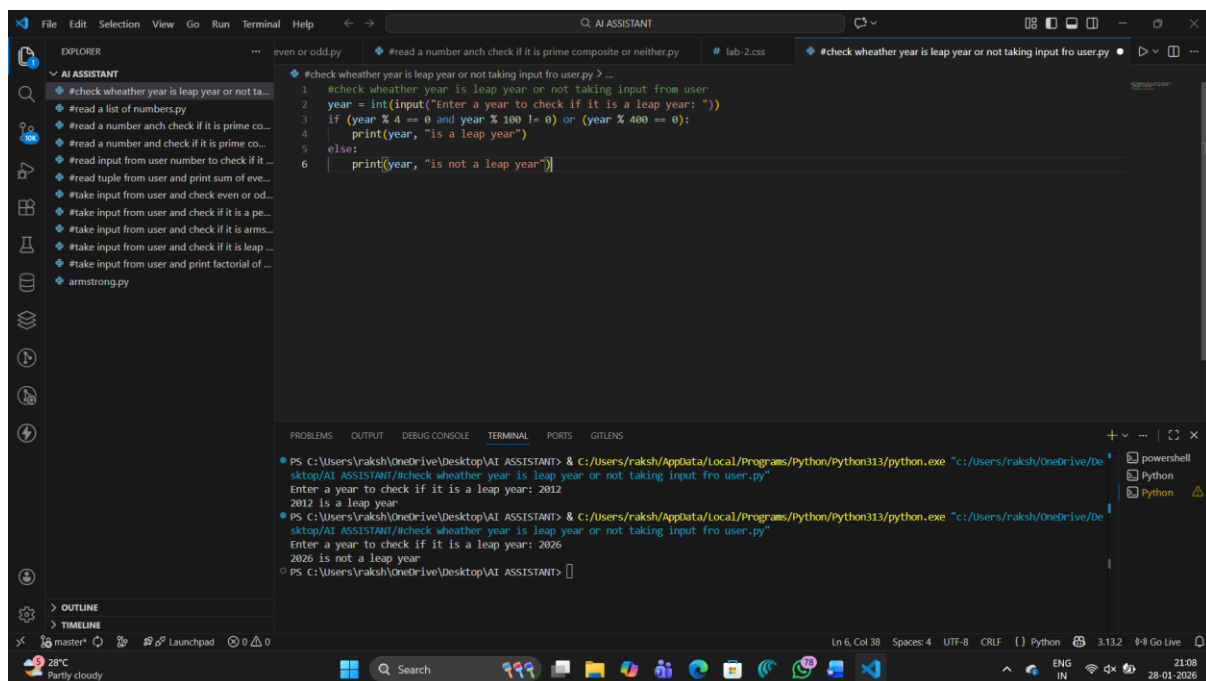
- Accepts a year as input
- Checks whether the given year is a leap year
- Returns an appropriate result

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

Prompt :

#check wheather year is leap year or not taking input from user

Code :



The screenshot shows a Visual Studio Code editor with a file named `check wheather year is leap year or not taking input fro user.py`. The code in the editor is as follows:

```
1 #check wheather year is leap year or not taking input from user
2 year = int(input("Enter a year to check if it is a leap year: "))
3 if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
4     print(year, "is a leap year")
5 else:
6     print(year, "is not a leap year")
```

The terminal at the bottom shows the execution of the script. It runs the command `python.exe "c:/Users/raksh/OneDrive/Desktop/AI ASSISTANT/check wheather year is leap year or not taking input fro user.py"` and shows the output for the input year 2012, which is a leap year, and 2026, which is not a leap year.

Analysis :

AI correctly understood the problem without examples.

Logical conditions were accurate (divisible by 4, 100, 400 rules).

Function structure was simple and efficient.

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

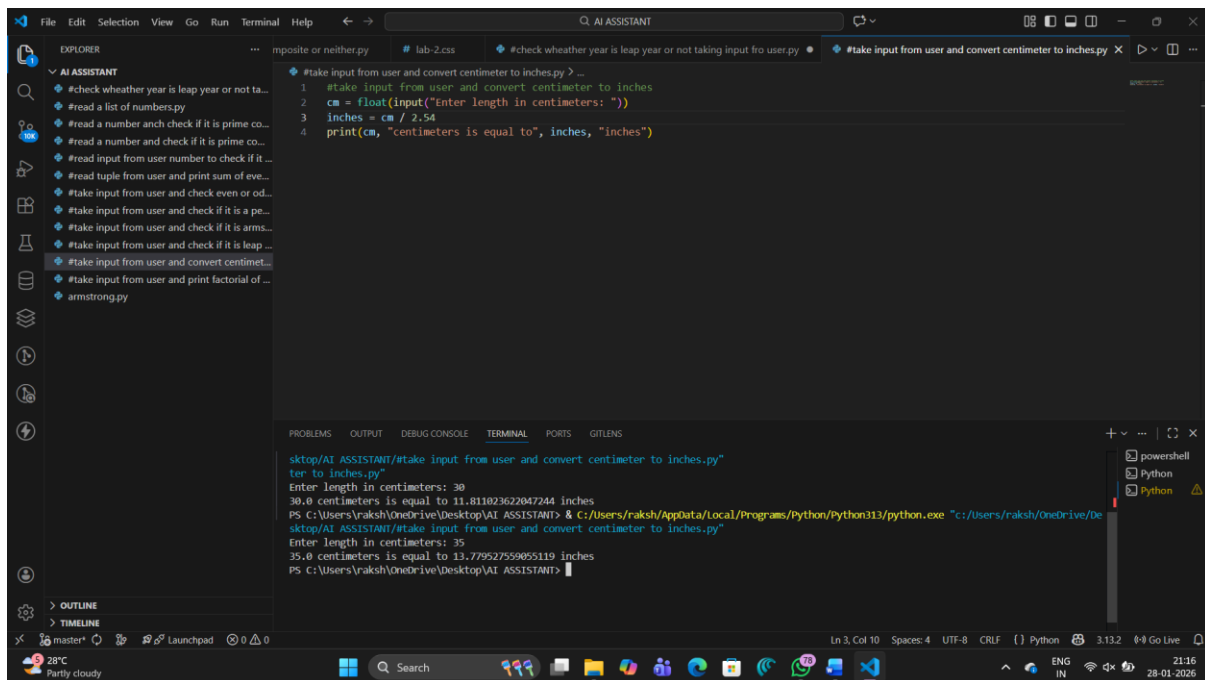
Example provided in prompt:

Input: 10 cm → Output: 3.94 inches

Prompt :

#take input from user and convert centimeter to inches

Code:



```
#take input from user and convert centimeter to inches.py
1 #take input from user and convert centimeter to inches
2 cm = float(input("Enter length in centimeters: "))
3 inches = cm / 2.54
4 print(cm, "centimeters is equal to", inches, "inches")

sktop/AI ASSISTANT/#take input from user and convert centimeter to inches.py"
Enter length in centimeters: 30
30.0 centimeters is equal to 11.811023622047244 inches
PS C:\Users\raksh\OneDrive\Desktop\AI ASSISTANT> & C:/Users/raksh/AppData/Local/Programs/Python/Python313/python.exe "C:/Users/raksh/OneDrive/De
sktop/AI ASSISTANT/#take input from user and convert centimeter to inches.py"
Enter length in centimeters: 35
35.0 centimeters is equal to 13.779527559055119 inches
PS C:\Users\raksh\OneDrive\Desktop\AI ASSISTANT>
```

Analysis

Single example helped AI understand the formula requirement.

Conversion logic ($\text{cm} \div 2.54$) was correctly applied.

Output matched expected precision.

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:

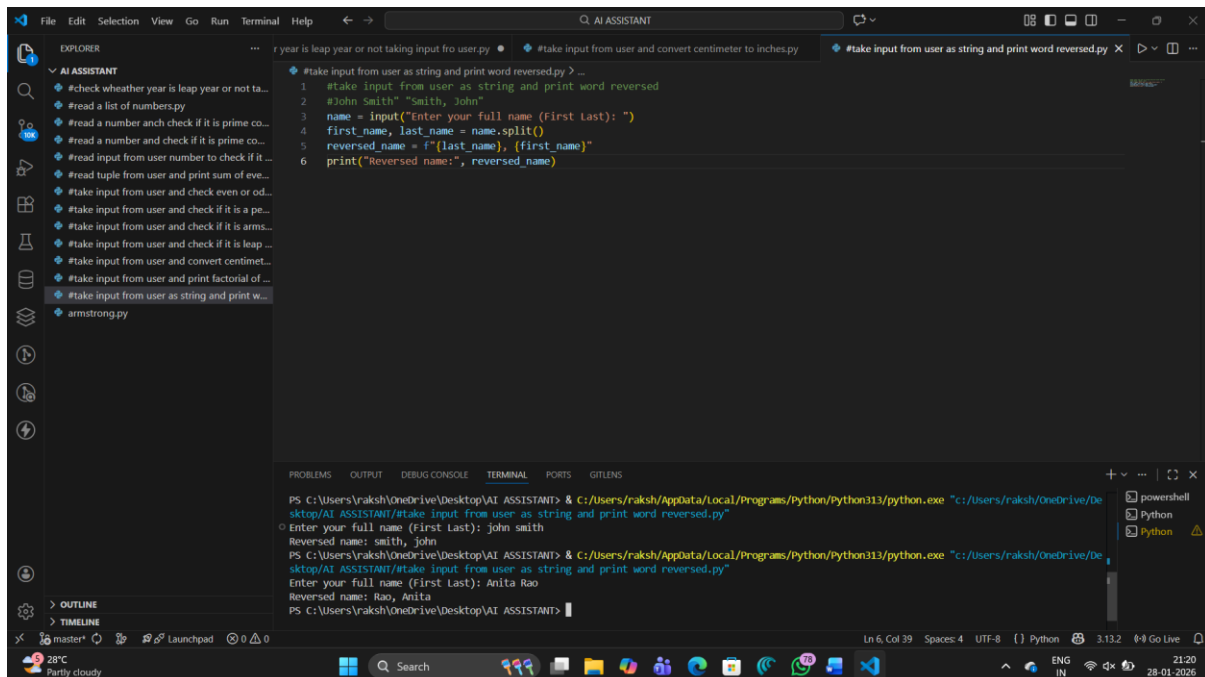
- "John Smith" → "Smith, John"
- "Anita Rao" → "Rao, Anita"

Prompt

#take input from user as string and print word reversed

#John Smith" → "Smith, John"

Code:



The screenshot shows a Visual Studio Code editor window with a Python file named `#take input from user as string and print word reversed.py`. The code in the editor is as follows:

```
1 #take input from user as string and print word reversed
2 #John Smith" "Smith, John"
3 name = input("Enter your full name (First Last): ")
4 first_name, last_name = name.split()
5 reversed_name = f"{last_name}, {first_name}"
6 print("Reversed name:", reversed_name)
```

The terminal at the bottom shows the execution of the script. It prompts the user to enter their full name, and the output shows the name reversed according to the "Last, First" pattern.

```
PS C:\Users\raksh\OneDrive\Desktop\AI ASSISTANT> & C:/Users/raksh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/raksh/OneDrive/Desktop/AI ASSISTANT/#take input from user as string and print word reversed.py"
Enter your full name (First Last): john smith
Reversed name: smith, john
PS C:\Users\raksh\OneDrive\Desktop\AI ASSISTANT> & C:/Users/raksh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/raksh/OneDrive/Desktop/AI ASSISTANT/#take input from user as string and print word reversed.py"
Enter your full name (First Last): Anita Rao
Reversed name: Rao, Anita
PS C:\Users\raksh\OneDrive\Desktop\AI ASSISTANT>
```

Analysis

Multiple examples improved format understanding.

Output strictly followed “Last, First” pattern.

Function handled input consistently.

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

Scenario

Different prompt strategies may produce different code quality.

Task Description

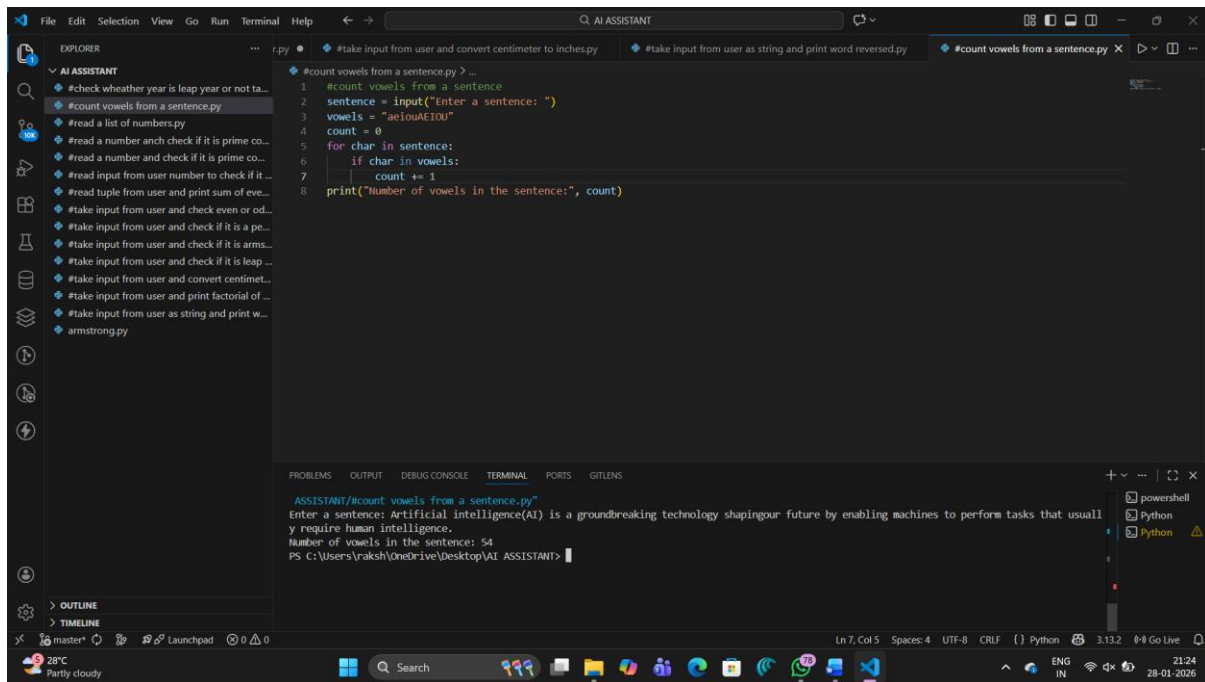
- Use zero-shot prompting to generate a function that counts vowels in a string
- Use few-shot prompting for the same problem
- Compare both outputs based on:
 - o Accuracy
 - o Readability

o Logical clarity

prompt

#count vowels from a sentence

Code:



```
#count vowels from a sentence.py
1 #count vowels from a sentence
2 sentence = input("Enter a sentence: ")
3 vowels = "aeiouAEIOU"
4 count = 0
5 for char in sentence:
6     if char in vowels:
7         count += 1
8 print("Number of vowels in the sentence:", count)
```

ASSISTANT/#count vowels from a sentence.py
Enter a sentence: Artificial intelligence(AI) is a groundbreaking technology shaping our future by enabling machines to perform tasks that usually require human intelligence.
Number of vowels in the sentence: 54
PS C:\Users\raksh\OneDrive\Desktop\AI ASSISTANT>

Analysis

Zero-shot version was logically correct but longer.

Few-shot version was more concise and readable.

Few-shot prompting improved code quality.

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

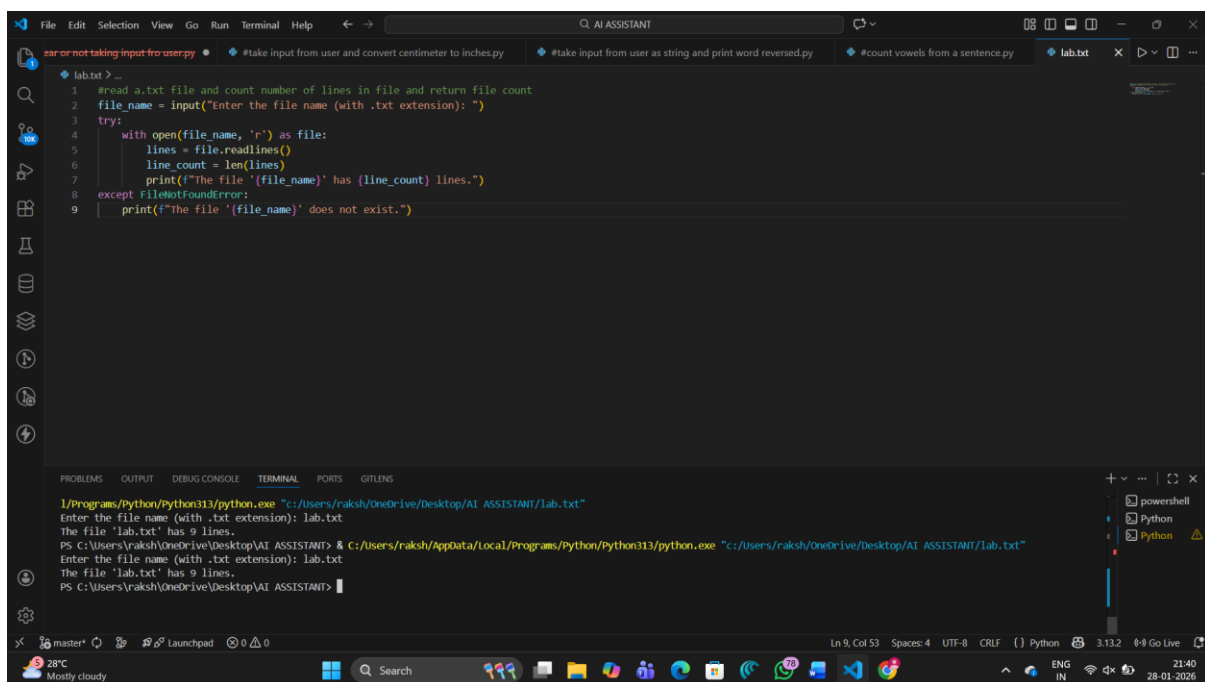
Expected Output

- Working Python file-processing function
- Correct line count
- Sample .txt input and output
- AI-assisted logic explanation

Prompt

#read a .txt file and count number of lines in file and return file count

Code:



The screenshot shows a Visual Studio Code editor with a Python script in a file named `lab.txt`. The script is designed to read a text file and count its lines. The code is as follows:

```
1 #read a.txt file and count number of lines in file and return file count
2 file_name = input("Enter the file name (with .txt extension): ")
3 try:
4     with open(file_name, 'r') as file:
5         lines = file.readlines()
6         line_count = len(lines)
7         print(f"The file '{file_name}' has {line_count} lines.")
8 except FileNotFoundError:
9     print(f"The file '{file_name}' does not exist.")
```

Below the editor, the terminal window shows the execution of the script. It prompts the user to enter a file name, and the user enters `lab.txt`. The script successfully reads the file and outputs: `The file 'lab.txt' has 9 lines.`

```
1/Programs/Python/Python313/python.exe "c:/Users/raksh/OneDrive/Desktop/AI ASSISTANT/lab.txt"
Enter the file name (with .txt extension): lab.txt
The file 'lab.txt' has 9 lines.
PS C:\Users\raksh\OneDrive\Desktop\AI ASSISTANT> & c:/Users/raksh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/raksh/OneDrive/Desktop/AI ASSISTANT/lab.txt"
Enter the file name (with .txt extension): lab.txt
The file 'lab.txt' has 9 lines.
PS C:\Users\raksh\OneDrive\Desktop\AI ASSISTANT>
```

Analysis

Few-shot prompting improved understanding of file operations.

Function correctly read file and counted lines.

Proper use of file handling methods.

Logic was clear and structured.