

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 :

```
even or odd.py #read a number and check if it is prime composite or neither.py
#check wheather year is leap year or not taking input fro user.py
#read a list of numbers
#read a number and check if it is prime co...
#read a number and check if it is prime co...
#read input from user to check if it ...
#read tuple from user and print sum of eve...
#take input from user and check even or od...
#take input from user and check if it is a pe...
#take input from user and check if it is arms...
#take input from user and check if it is leap ...
#take input from user and print factorial of ...
armstrong.py

#check wheather year is leap year or not taking input fro user.py
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")

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/#check wheather year is leap year or not taking input fro user.py"
Enter a year to check if it is a leap year: 2012
2012 is a leap year
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/#check wheather year is leap year or not taking input fro user.py"
Enter a year to check if it is a leap year: 2026
2026 is not a leap year
PS C:\Users\raksh\OneDrive\Desktop\AI ASSISTANT>
```

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:

The screenshot shows the Visual Studio Code interface. In the Explorer sidebar, there's a folder named 'AI ASSISTANT' containing several Python files like 'composite or neither.py', 'lab-2.css', etc. The main editor area has a snippet of Python code for converting centimeters to inches. The terminal at the bottom shows the execution of this script and its output.

```
#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")
```

Terminal Output:

```
sktop/AI ASSISTANT/#take input from user and convert centimeter to inches.py"
Enter length in centimeters: 30
30.0 centimeters is equal to 11.81023622047244 inches
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 and convert centimeter to inches.py"
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 ($cm \div 2.54$) 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 the Visual Studio Code interface with the AI Assistant extension active. The Explorer sidebar on the left lists various Python files, including 'armstrong.py'. The main editor area displays a snippet of code for reversing a name:

```
#take input from user as string and print word reversed.py
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:

```
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"
sktop\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"
sktop\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

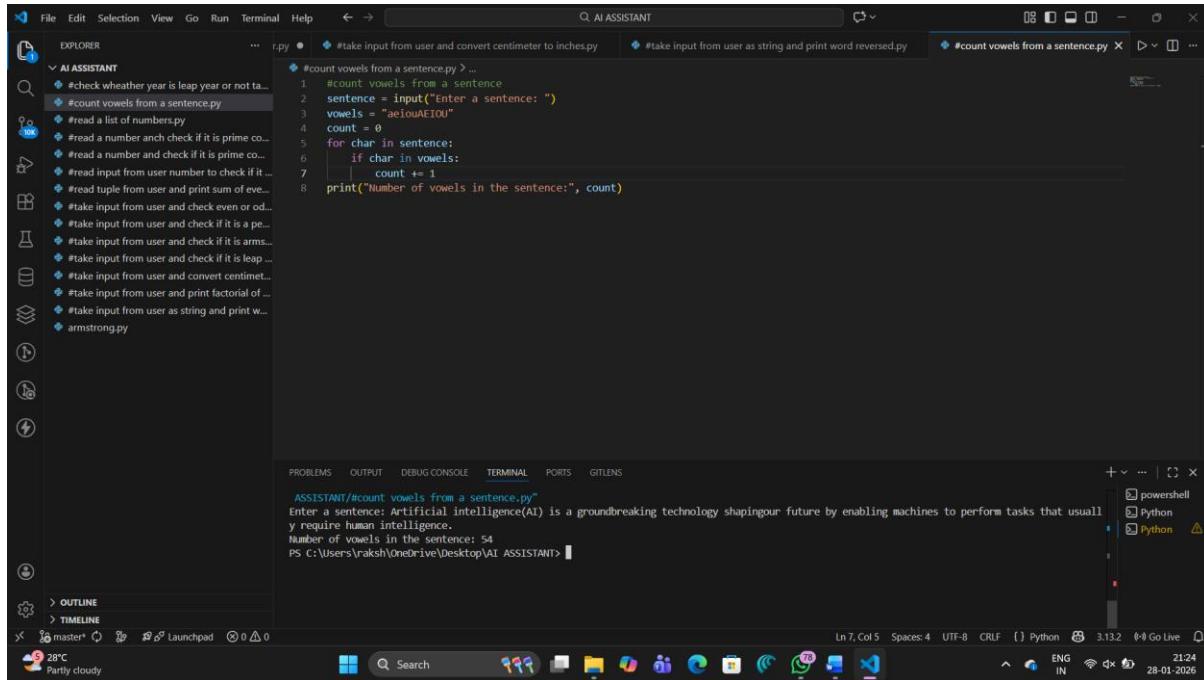
- Use zero-shot prompting to generate a function that counts vowels in a string
- Use few-shot prompting for the same problem
- Compare both outputs based on:
 - o Accuracy
 - o Readability

o Logical clarity

prompt

#count vowels from a sentence

Code:



The screenshot shows the Visual Studio Code interface with the AI Assistant extension open. The Explorer sidebar on the left lists several Python files, including 'count vowels from a sentence.py'. The main code editor window displays the following Python code:

```
1 #count vowels from a sentence.py ...
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)
```

The AI Assistant panel on the right shows a list of generated prompts for this code, such as 'check whether year is leap year or not' and 'read a list of numbers'. The terminal at the bottom shows the output of running the script: 'Enter a sentence: Artificial Intelligence(AI) is a groundbreaking technology shaping our future by enabling machines to perform tasks that usually require human intelligence.' followed by 'Number of vowels in the sentence: 54'.

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 the Visual Studio Code interface with the following details:

- Editor:** The main editor window contains Python code to read a file and count its lines. The code is as follows:

```
lab.txt >_
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.")
```

- Terminal:** The terminal shows the command being run: `python.exe "c:/Users/raksh/OneDrive/Desktop/AI ASSISTANT/lab.txt"`. The output indicates the file 'lab.txt' has 9 lines.
- Bottom Status Bar:** Shows the file path as `C:\Users\raksh\OneDrive\Desktop\AI ASSISTANT\lab.txt`, the line and column numbers (Ln 9, Col 53), and the date and time (28-01-2026).

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.