Name:S.Varun
Roll.no:2403A53L02

Batch - 46

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: B. Tech		**Assignment Type: Lab**	**Academic Year:**2025-2026
Course Coordinator Name	Dr. Rishabh Mittal		
Instructor(s) Name	Mr. S Naresh Kumar Ms. B. Swathi Dr. Sasanko Shekhar Gantayat Mr. Md Sallauddin Dr. Mathivanan Mr. Y Srikanth Ms. N Shilpa Dr. Rishabh Mittal (Coordinator) Dr. R. Prashant Kumar Mr. Ankushavali MD Mr. B Viswanath Ms. Sujitha Reddy Ms. A. Anitha Ms. M.Madhuri Ms. Katherashala Swetha Ms. Velpula sumalatha Mr. Bingi Raju		
Course Code	23CS002PC304	**Course Title**	AI Assisted Coding
Year/Sem	III/I	**Regulation**	R23
Date and Day of Assignment	Week 2 - Wednesday	**Time(s)**	23CSBTB01 To 23CSBTB52
Duration	2 Hours	**Applicableto Batches**	All batches
Assignment Number: 3.3(Present assignment number)/**24**(Total number of assignments)			

Q.No.	Question	*Expected Time 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 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	Week2 - Wednesday

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 different prompting strategies

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

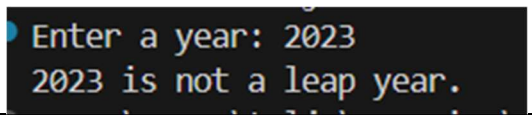
- AI-generated leap year checking function
- Correct logical conditions
- Sample input and output
- Screenshot of AI-generated response (if required)

PROMPT:
Generate a Python function that accepts a year as input and determines whether it is a leap year or not.
Examples:
Input: 2020 → Output: True
Input: 1900 → Output: False
Input: 2000 → Output: True

CODE:

```
def is_leap_year(year):
    return year % 4 == 0 and (year % 100 != 0 or year % 400 == 0)
#Example usage:
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.")
```

OUTPUT:



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

PROMPT:
Generate a Python function that converts centimeters to inches using the correct mathematical formula.
Example:
Input: 10 cm
Output: 3.94 inches

CODE:

```
def cm_to_inches(cm):
    inches = cm / 2.54
    return round(inches, 2)
#Example usage:
cm = float(input("Enter length in centimeters: "))
inches = cm_to_inches(cm)
print(f"{cm} cm is equal to {inches} inches.")
```

OUTPUT:

```
Enter length in centimeters: 10
10.0 cm is equal to 3.94 inches.
```

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

PROMPT:

Create a Python function that accepts a full name as input and formats it in the form "Last, First".
Examples:
"John Smith" → "Smith, John"
"Anita Rao" → "Rao, Anita"
"Suresh Kumar" → "Kumar, Suresh"

CODE:

```
def format_name(full_name):
    parts = full_name.split()
    if len(parts) >= 2:
        first_name = parts[0]
        last_name = parts[-1]
        return f"{last_name}, {first_name}"
    else:
        return full_name
#Example usage:
name = input("Enter full name: ")
formatted_name = format_name(name)
print(f"Formatted name: {formatted_name}")
```

OUTPUT:

```
Enter full name: allu arjun
Formatted name: arjun, allu
```

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:
    - Accuracy
    - Readability
    - Logical clarity

Expected Output

- Two vowel-counting functions
- Comparison table or short reflection paragraph
- Conclusion on prompt effectiveness

PROMPT:

Generate a Python function that counts the number of vowels present in a given string.Provide sample input and output.
Examples:
Input: "hello" → Output: 2
Input: "artificial intelligence" → Output: 9

CODE:

```
def count_vowels(input_string):
    vowels = "aeiouAEIOU"
    count = sum(1 for char in input_string if char in vowels)
    return count
#Example usage:
input_string = input("Enter a string: ")
vowel_count = count_vowels(input_string)
print(f"Number of vowels in the string: {vowel_count}")
```

OUTPUT:

```
Enter a string: hello
Number of vowels in the string: 2
```

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:

Generate a Python function that reads a .txt file and counts the number of lines present in the file.

Examples:

file containing 4 lines → Output: 4

A file containing 12 lines → Output: 12

CODE:

```
def count_lines_in_file(file_path):
    with open(file_path, 'r') as file:
        lines = file.readlines()
        return len(lines)
#Example usage:
file_path = input("Enter the path to the .txt file: ")
line_count = count_lines_in_file(file_path)
print(f"Number of lines in the file: {line_count}")
```

OUTPUT:

```
Enter the path to the .txt file: C:\Users\telje\OneDrive\Desktop\AI\example.txt
Number of lines in the file: 4
```