LAB ASSIGNMENT 4.3

Name: K.Abhinay

Hall No. : 2303A51899

Batch No. : 09.

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 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)

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

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

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

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

Code:

```python
# Task 1: Leap Year

# a simple function that checks if a year is leap or not .

def is_leap_year(year: int) -> bool:

    if year % 400 == 0:

        return True

    if year % 100 == 0:

        return False

    return year % 4 == 0




# Task 2: CM to Inches

# a simple function that converts CM TO INCHES

def cm_to_inches(cm: float) -> float:

    return cm / 2.54




# Task 3: Split Full Name

# a function which divides the first and last names by a space

def split_full_name(full_name: str) -> tuple[str, str]:

    parts = full_name.strip().split()
```

```
    if not parts:
        return "", ""
    if len(parts) == 1:
        return parts[0], ""
    return parts[0], " ".join(parts[1:])



# Task 4: Count Vowels
# a simple function that counts the number of vowels in a given string
def count_vowels(s: str) -> int:
    return sum(1 for ch in s.lower() if ch in "aeiou")



# Task 5: Count Lines in File
# a simple function that counts no.of lines in a given file path ....
def count_lines_in_file(file_path: str) -> int:
    with open(file_path, "r", encoding="utf-8", errors="replace") as f:
        return sum(1 for _ in f)
```

```python
# Main Testing


if __name__ == "__main__":


    # Test leap year
    examples = [2000, 1900, 2016, 2019, 2024]
    for y in examples:
        print(f"{y}: {'leap' if is_leap_year(y) else 'not leap'}")


    # Test cm to inches
    print("170 cm =", cm_to_inches(170), "inches")


    # Test name split
    first, last = split_full_name("Abhinay Kongonda")
    print("First:", first)
    print("Last:", last)


    # Test vowel count
    print("Vowels in 'Computer Science':", count_vowels("Computer Science"))
```

```
# Test file line count (give valid file path)

print("Lines:", count_lines_in_file("file.txt"))
```

Outputs:

2019: not leap

2024: leap

170 cm = 66.92913385826772 inches

First: Abhinay

Last: Kongonda

Vowels in 'Computer Science': 6

Lines: 2