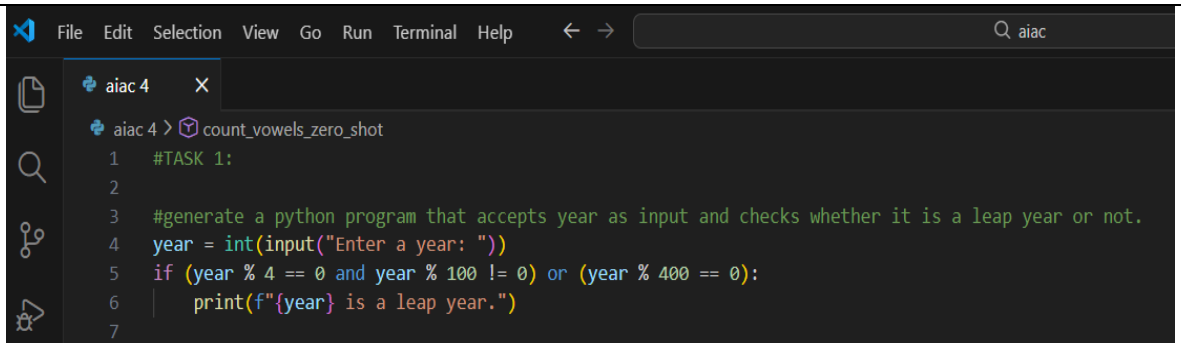
SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE			DEPARTMENT OF COMPUTER SCIENCE ENGINEERING
Program Name: B. Tech		**Assignment Type: Lab**	**Academic Year:**2025-2026
CourseCode	23CS002PC304	**Course Title**	AI Assisted Coding
Year/Sem	III/II	**Regulation**	R23
Date and Day of Assignment	**Week1 – Wednesday**	**Batch**	23CSBTB47B
Name	M.Yashaswini	**Hall Ticket No**	2303A54049
Assignment Number:4.1			

Q.No.	Question
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 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** • **Sample input and output** SyntaxError: (unicode error) 'unicodeescape' codec can't decode bytes in position 624-625: truncated \UXXXXXXXX escape PS C:\Users\srina\OneDrive\Desktop\aiac> & C:/Users/srina/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/srina/OneDrive/Desktop/aiac/aiac 4" Enter a year: 2004 2004 is a leap year. • **Screenshot of AI-generated response**



- **Explanation**

  There is a leap year every four years. However, years that end in 00 are unique, only years that are divisible by 400 are considered as the leap years so I used this logic a year which if it's divisible by 4, except century years.

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 :

```
#TASK 2:

#generate a python program that converts centimeters to inches like 10cm = 3.93701 inches
cm = float(input("Enter length in centimeters: "))
inches = cm / 2.54
print(f"{cm} cm = {inches} inches")
```

- **Accurate calculation**

```
inches = cm / 2.54
```

- **Sample test cases and outputs**

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

- **Explanation**

  This program is like a simple calculator

  When we type a length in centimeters. It divides by 2.54. And It prints the result in inches, rounded to two decimals.

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**

```
#TASK 3:

#write a python program that accepts a full name and formats it as "Last Name, First Name" like "John Smith" to "Smith, John" and "Anitha
full_name = input("Enter your full name: ")
first_name, last_name = full_name.split()
formatted_name = f"{last_name}, {first_name}"
print("Formatted name:", formatted_name)
```

- **Sample inputs and outputs**

```
Enter your full name: Smith John
John, Smith
Enter your full name: mittapally yashu
Formatted name: yashu, mittapally
```

- **Explanation**

  The program takes a full name then splits it into first and last names , and then prints it in the format Last, First by using split function.

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**

```
#TASK 4:
#generate a python code that reads a string and that code should contain a function that counts vowels in a string using zero-shot prompting,f
Qodo: Test this function
def count_vowels(input_string):
    vowels = "aeiouAEIOU"
    count = sum(1 for char in input_string if char in vowels)
    return count
input_string = input("Enter a string: ")
vowel_count = count_vowels(input_string)
print(f"The number of vowels in the string is: {vowel_count}")
```

```
Qodo: Test this function
def count_vowels_zero_shot(input_string):
    """
    Counts the number of vowels in a given string using zero-shot prompting.

    Args:
        input_string (str): The string to analyze

    Returns:
        int: The count of vowels in the string
    """
    vowels = "aeiouAEIOU"
    count = sum(1 for char in input_string if char in vowels)
    return count
count=count_vowels_zero_shot("Hello World")
print(f"Number of vowels (Zero-shot): {count}")
Qodo: Test this function
def count_vowels_few_shot(input_string):
    """
    Counts the number of vowels in a given string using few-shot prompting.

    Args:
        input_string (str): The string to analyze

    Returns:
        int: The count of vowels in the string
    """
    examples = [
        ("Hello", 2),
        ("World", 1),
        ("Python Programming", 4),
        ("OpenAI", 3)
    ]

    vowels = "aeiouAEIOU"
    count = sum(1 for char in input_string if char in vowels)
    return count
count=count_vowels_few_shot("Hello World")
print(f"Number of vowels (Few-shot): {count}")
# Comparison Table
comparison_table = """
| Criteria        | Zero-shot Prompting               | Few-shot Prompting                |
|-----------------|-----------------------------------|-----------------------------------|
| Accuracy        | High                              | High                              |
| Readability     | Clear and concise                 | Slightly more verbose             |
| Logical Clarity | Direct and straightforward        | Demonstrates examples for clarity |
"""
print(comparison_table)
```

- **Explanation**

The function count vowels counts the number of vowels in a given string. The zero-shot and few-shot prompting functions demonstrate different approaches to achieve the same result.

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**

The file is opened in **read mode**
Python reads the file **line by line**
A generator expression counts each line
sum() adds them together
The function returns the final count

- **Sample .txt input and output**

```
Enter the path of the text file: \Users\srina\Downloads\PATTERNS.txt
The number of lines in the file is: 260
```

- **AI-assisted logic explanation**

```python
#Task-5
#generate a python code that reads a text file and counts the number of lines in it and prints the count
Qodo: Test this function
def count_lines_in_file(file_path):
    try:
        with open(file_path, 'r') as file:
            lines = file.readlines()
            line_count = len(lines)
            return line_count
    except FileNotFoundError:
        return "File not found. Please check the file path."
file_path = input("Enter the path of the text file: ")
line_count = count_lines_in_file(file_path)
print(f"The number of lines in the file is: {line_count}")
#Logic : The function count_lines_in_file reads a text file and counts the number of lines in it, handling the case where the file may not be found.
```

- **Explanation**

The function count lines in file reads a text file and counts no of lines in it,handling the case where file may not be found.