SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE	DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

2403A53043 24BTCAICYB02	Assignment Type: Lab	AcademicYear:2025-2026
CourseCoordinatorName	Venkataramana Veeramsetty	
Instructor(s)Name	Dr. V. Venkataramana (Co-ordinator)	
	Dr. T. Sampath Kumar	
	Dr. Pramoda Patro	
	Dr. Brij Kishor Tiwari	
	Dr.J.Ravichander	
	Dr. Mohammand Ali Shaik	
	Dr. Anirodh Kumar	
	Mr. S.Naresh Kumar	
	Dr. RAJESH VELPULA	
	Mr. Kundhan Kumar	
	Ms. Ch.Rajitha	
	Mr. M Prakash	
	Mr. B.Raju	
	Intern 1 (Dharma teja)	
	Intern 2 (Sai Prasad)	
	Intern 3 (Sowmya)	
	NS_2 (Mounika)	
CourseCode	24CS002PC215	**CourseTitle**	AI Assisted Coding
Year/Sem	II/I	**Regulation**	R24
Date and Day of Assignment	Week2 - Wednesday	**Time(s)**	
Duration	2 Hours	**Applicableto Batches**	

AssignmentNumber:2.3(Present assignment number)/**24**(Total number of assignments)

Q.No.	Question	ExpectedTime to complete
1	Lab 4: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques **Lab Objectives:**	Week2 - Wednesday

- 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 the three prompting strategies.

PROMPT:write a function that checks whether a given year is a leap year.

Task Description#1
- Zero-shot: Prompt AI to write a function that checks whether a given year is a leap year.

```python
def is_leap_year(year):
  """Checks if a given year is a leap year."""
  if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
    return True
  else:
    return False
print(f"2020 is a leap year: {is_leap_year(2020)}")
print(f"1900 is a leap year: {is_leap_year(1900)}")
print(f"2000 is a leap year: {is_leap_year(2000)}")
print(f"2023 is a leap year: {is_leap_year(2023)}")
```

Expected Output#1
- AI-generated function with no examples provided

```
2020 is a leap year: True
1900 is a leap year: False
2000 is a leap year: True
2023 is a leap year: False
```

Explaination: The code I provided defines a function called is_leap_year that takes a year as input and returns True if it's a leap year and False otherwise.

Here's how it works:

- A year is a leap year if it is divisible by 4.
- However, if the year is also divisible by 100, it is not a leap year, unless...
- ...the year is also divisible by 400. In that case, it is a leap year.

The code implements these rules using conditional statements (if, else, and, or) and the modulo operator (%) to check for divisibility.

The output of the execution shows that:

- 2020 is a leap year (divisible by 4, not by 100)
- 1900 is not a leap year (divisible by 100, but not by 400)
- 2000 is a leap year (divisible by 400)
- 2023 is not a leap year (not divisible by 4)

Conclusion:

The function correctly identifies leap years based on the standard rules.

Prompt:Give one input-output example to guide AI in writing a function that converts centimeters to inches.

Task Description#2
- One-shot: Give one input-output example to guide AI in writing a function that converts centimeters to inches.

```python
def cm_to_inches(cm):
    inches = cm * 0.3937
    return inches
input_cm = 100
output_inches = cm_to_inches(input_cm)
print(f"{input_cm} centimeters is equal to {output_inches} in
```

Explanation:

- Key aspects of the code
- The conversion factor
- Potential applications of the conversion
- Something else

Conclusion:

- A summary of the key features of the conversion function
- Important considerations when performing unit conversions
- Specific details about the conversion factor or formula

Expected Output#2
- Function with correct conversion logic

```
100 centimeters is equal to 39.37 inches.
```

Prompt:
Generate a function that forats full names as "Last, First".

Task Description#3
- Few-shot: Provide 2–3 examples to generate a function that formats full names as "Last, First".

```python
def format_name(full_name):
  names = full_name.split()
  if len(names) > 1:
    first_name = names[0]
    last_name = names[-1]
    return f"{last_name}, {first_name}"
  else:
    return full_name
print(format_name("John Doe"))
print(format_name("Alice Wonderland"))
print(format_name("Peter Pan"))
print(format_name("Cher"))
```

Explaination:

- The function splits the input full name into separate parts based on spaces.

- It identifies the first name as the first part and the last name as the last   part of the split.

- It formats and returns the name as "Last, First" using the identified

- If the input doesn't contain at least two parts, it returns the originame unchanged.

Conclusion:

- The function reliably reformats names from "First Last" to "Last, First."

- It uses simple string splitting and indexing to extract name parts.

- It handles edge cases by returning the original input if formatting isn't possible.

- This approach ensures consistent and clear name presentation in applications.

Expected Output#3
- Well-structured function respecting the examples

```
Doe, John
Wonderland, Alice
Pan, Peter
Cher
```

Prompt: Write a function that counts the number of vowels in a given string

Task Description#4
- Compare zero-shot and few-shot prompts for writing a function that counts the number of vowels in a
  string.

```python
def count_vowels(text):
  vowels = "aeiou"
  count = 0
  for char in text:
    if char.lower() in vowels:
      count += 1
  return count
print(f"Number of vowels in 'hello': {count_vowels('hello')}")
print(f"Number of vowels in 'Programming': {count_vowels('Programming')")
print(f"Number of vowels in 'AEIOU': {count_vowels('AEIOU')}")
print(f"Number of vowels in 'Rhythm': {count_vowels('Rhythm')}")
```

Explanation

- :How the code handles case sensitivity
- The use of the for loop
- The in operator
- The variable initialization

Conclusion:

- Adding subheadings to the current notebook
- Organizing the explanations I've provided with subheadings
- Something else entirely

Expected Output#4

- Functional output and comparative reflection

```
Number of vowels in 'hello': 2
Number of vowels in 'Programming': 3
Number of vowels in 'AEIOU': 5
Number of vowels in 'Rhythm': 0
```

Prompt: Generate a function that reads a .txt file and returns the number of lines.

Task Description#5
- Use few-shot prompting to generate a function that reads a .txt file and returns the number of lines.

```python
import os

def count_lines_in_file(file_path):
  try:
    with open(file_path, 'r') as f:
      line_count = sum(1 for line in f)
    return line_count
  except FileNotFoundError:
    print(f"Error: File not found at {file_path}")
    return -1
  except Exception as e:
    print(f"An error occurred: {e}")
    return -1
```

Explanation:

- Breaking down the previous points into more detailed sub-points
- Creating a hierarchical structure for the information
- Something else related to organizing information or planning a task with sub-steps

Conclusion:

- We created a function to check if a year is a leap year, considering the special rules for years divisible by 100 and 400.
- We developed a function to convert measurements from centimeters to inches using the conversion factor 0.3937.
- We built a function to reformat full names into a "Last, First" format, handling names with multiple parts and single names.
- We implemented a function to count the number of lines in a text file, including error handling for missing files.

Expected Output#5
- Working file-processing function with AI-guided logic

Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots

Evaluation Criteria:

Criteria	Max Marks
Zero Shot (Task #1)	0.5
One Shot (Task#2)	0.5
Few Shot (Task#3 & Task #5)	1.0
Comparison (Task#4)	0.5
Total	**2.5 Marks**