SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE	DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
ProgramName: B. Tech	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:4.3(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:** ● To explore and apply different levels of prompt examples in AI-assisted code	Week2 - Wednesday

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.

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

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

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

Expected Output#2
- Function with correct conversion logic

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

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

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

Expected Output#4
- Functional output and comparative reflection

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

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**		

Roll no: 2403A510D1
Name: Likhitha Pothunuri

VS code with github copilot

Task Description#1

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

Expected Output#1

- AI-generated function with no examples provided

Prompt:

write code to check whether given year is leap or not.

```
task1copilot.py > ...
1    # Function to check if a given year is a leap year
2    def is_leap_year(year):
3        """Return True if the given year is a leap year, else False."""
4        if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
5            return True
6        else:
7            return False
8
9    if __name__ == "__main__":
10       try:
11           year = int(input("Enter a year: "))
12           if is_leap_year(year):
13               print(f"{year} is a leap year.")
14           else:
15               print(f"{year} is not a leap year.")
16       except ValueError:
17           print("Invalid input. Please enter a valid year.")
18
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   **TERMINAL**   PORTS

PS C:\Users\thoop\lab4> & 'c:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\thoop\.vs
code\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '62745' '--' 'c:\Users\thoop\
lab4\task1copilot.py'
Enter a year: 2024
2024 is a leap year.
PS C:\Users\thoop\lab4>

Task Description#2

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

Expected Output#2

● Function with correct conversion logic

Prompt:

Write a function to convert centimeters to inches.

Example:10.0 centimeters is equal to 3.912345 inches.

read input from the user.

```
task2copilot.py > ...
1    def cm_to_inches(cm):
2        return cm / 2.54
3
4    cm = float(input("Enter length in centimeters: "))
5    inches = cm_to_inches(cm)
6    print(f"{cm} centimeters is equal to {inches} inches.")


PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS C:\Users\thoop\lab4>  & 'c:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\thoop\.vs
code\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '62927' '--' 'c:\Users\thoop\
lab4\task2copilot.py'
Enter length in centimeters: 100
100.0 centimeters is equal to 39.37007874015748 inches.
PS C:\Users\thoop\lab4> █
```

Task Description#3

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

Expected Output#3

- Well-structured function respecting the examples

Prompt:

Write a python function that generates formats full names as "last,first".
Example: peter parker =parker peter.
          Harry potter=Potter Harry

Read full name from the user.

Code:

```
 task3copilot.py > ...
  1
  2    # Function to format full name as "Last First"
  3    def format_full_name(full_name):
  4        """Return the full name formatted as 'Last First' with proper capitalization."""
  5        parts = full_name.strip().split()
  6        if len(parts) < 2:
  7            return "Invalid input. Please enter both first and last name."
  8        first = parts[0].capitalize()
  9        last = parts[-1].capitalize()
 10        return f"{last} {first}"
 11
 12    if __name__ == "__main__":
 13        full_name = input("Enter full name (first last): ")
 14        formatted = format_full_name(full_name)
 15        print(f"Formatted name: {formatted}")
 16

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS                                                     +

PS C:\Users\thoop\lab4> & C:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe c:/Users/thoop/lab4/ta
sk3copilot.py
Enter full name (first last): harichandana thoopukari
Formatted name: Thoopukari Harichandana
PS C:\Users\thoop\lab4>
```

Task Description#4

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

Expected Output#4

- Functional output and comparative reflection

Prompt:

1.Zero shot:

Write a python program to count number of vowels in a string. read input from the user.

2.few-shot:

Write a python program to count number of vowels in a given string.
Example: sruniversity
Vowel count=4.
Read input from the user.

Zero shot   few-shot

Task Description#5

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

Expected Output#5

- Working file-processing function with AI-guided logic

Prompt:
 Generate a function that reads the given a.txt file and returns the number of lines
 Example:hello sr university.hanmakonda warangal.
            Top engineering college.
             Offers wide variety of courses.
No.of lines in a.txt=3.

```
task5copilot.py > ...
1    def count_lines_in_file(filename):
2        with open(filename, 'r', encoding='utf-8') as f:
3            return len(f.readlines())
4
5    if __name__ == "__main__":
6        num_lines = count_lines_in_file('a.txt')
7        print(f"Number of lines in a.txt: {num_lines}")
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

sk5copilot.py
Number of lines in a.txt: 8
Number of lines in a.txt: 8
PS C:\Users\thoop\lab4> & C:\Users\thoop\AppData\Local\Programs\Python\Python313\python.
sk5copilot.py
Number of lines in a.txt: 8
Number of lines in a.txt: 8
PS C:\Users\thoop\lab4>
```

Cursor AI

Task Description#1

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

Expected Output#1

- AI-generated function with no examples provided

Prompt:

write code to check whether given year is leap or not.

```
1    def is_leap_year(year): | TAB to jump here
2        """
3        Check if a given year is a leap year.
4
5        Leap year rules:
6        1. If a year is divisible by 4, it's a leap year
7        2. However, if it's also divisible by 100, it's NOT a leap year
8        3. Unless it's also divisible by 400, then it IS a leap year
9
10       Args:
11           year (int): The year to check
12
13       Returns:
14           bool: True if it's a leap year, False otherwise
15       """
16       if year % 4 == 0:
17           if year % 100 == 0:
18               if year % 400 == 0:
19                   return True
20               else:
21                   return False
22           else:
23               return True
24       else:
25           return False
26
```

```
28    def main():
29        """Main function to demonstrate the leap year checker."""
30        print("Leap Year Checker")
31        print("=" * 20)
32
33        # Test cases
34        test_years = [2000, 2020, 2024, 1900, 2100, 2023, 2025]
35
36        for year in test_years:
37            if is_leap_year(year):
38                print(f"{year} is a leap year")
39            else:
40                print(f"{year} is NOT a leap year")
41
42        print("\n" + "=" * 20)
43
44        # Interactive input
45        try:
46            user_year = int(input("Enter a year to check: "))
47            if is_leap_year(user_year):
48                print(f"{user_year} is a leap year!")
49            else:
50                print(f"{user_year} is NOT a leap year.")
51        except ValueError:
52            print("Please enter a valid integer year.")
```

Problems   Output   Debug Console   **Terminal**   Ports

```
PS C:\Users\thoop\lab4> & C:/Users/thoop/AppData/Local/Programs/Python/Python313/python.exe c:/Users/
Enter a year: & c:/Users/thoop/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thoop/lab4
Invalid input. Please enter a valid year.
PS C:\Users\thoop\lab4> & C:/Users/thoop/AppData/Local/Programs/Python/Python313/python.exe c:/Users/
Enter a year: 2024
2024 is a leap year.
PS C:\Users\thoop\lab4>
```

Task Description#2

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

Expected Output#2

- Function with correct conversion logic

Prompt:

  Write a function to convert centimeters to inches.

Example:10.0 centimeters is equal to 3.912345 inches.

read input from the user.

```
task2cusor.py > main
1    def cm_to_inches(cm):
2
3        Convert centimeters to inches.
4        Args:
5            cm (float): Length in centimeters
6        Returns:
7            float: Length in inches
8        """
9        # 1 inch = 2.54 centimeters
10       inches = cm / 2.54
11       return inches
12   def main():
13       """Main function to demonstrate the centimeter to inch converter."""
14       print("Centimeter to Inch Converter")
15       print("=" * 30)
16       # Test case
17       test_cm = 10.0
18       test_inches = cm_to_inches(test_cm)
19       print(f"Example: {test_cm} centimeters is equal to {test_inches:.6f} inches.")
20       print("\n" + "=" * 30)
21       # Interactive input
22       try:
23           user_cm = float(input("Enter length in centimeters: "))
24           user_inches = cm_to_inches(user_cm)
25           print(f"{user_cm} centimeters is equal to {user_inches:.6f} inches.")
26       except ValueError:
27           print("Please enter a valid number.")
28   if __name__ == "__main__":
29       main()
```

Problems   Output   Debug Console   **Terminal**   Ports   Python ⚠ + ∨ ⊡

```
Centimeter to Inch Converter
==============================
Example: 10.0 centimeters is equal to 3.937008 inches.

==============================
Enter length in centimeters: 100
```
Ctrl+K to generate a command

Task Description#3

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

Expected Output#3

- Well-structured function respecting the examples

Prompt:

Write a python function that generates formats full names as "last,first".
Example: peter parker =parker peter.
          Harry potter=Potter Harry

Read full name from the user.

```python
def format_name(full_name):
    """
    Format a full name as "last,first" with proper capitalization.

    Args:
        full_name (str): Full name in "first last" format

    Returns:
        str: Formatted name as "Last,First"
    """
    # Split the name into parts and strip whitespace
    name_parts = full_name.strip().split()

    if len(name_parts) < 2:
        return "Error: Please enter both first and last name"

    # Get first and last name
    first_name = name_parts[0]
    last_name = name_parts[-1]

    # Capitalize first letter of each name
    first_name = first_name.capitalize()
    last_name = last_name.capitalize()

    # Format as "last,first"
    formatted_name = f"{last_name},{first_name}"

    return formatted_name
```

```python
def main():
    """Main function to demonstrate the name formatter."""
    print("Name Formatter - Last,First")
    print("=" * 30)

    # Test cases
    test_names = ["peter parker", "harry potter", "mary jane watson"]

    print("Examples:")
    for name in test_names:
        formatted = format_name(name)
        print(f"{name} = {formatted}")

    print("\n" + "=" * 30)

    # Interactive input
    while True:
        try:
            user_name = input("Enter full name (first last): ")
            if user_name.lower() == 'quit':
                break

            formatted = format_name(user_name)
            print(f"Formatted: {formatted}")

        except KeyboardInterrupt:
            print("\nGoodbye!")
            break
        except Exception as e:
            print(f"Error: {e}")
```

```
Problems   Output   Debug Console   Terminal   Ports

Enter full name (first last): harichandana thoopukari
Formatted: Thoopukari,Harichandana
```

Task Description#4

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

Expected Output#4

- Functional output and comparative reflection

Prompt:
1.Zero shot:

Write a python program to count number of vowels in a string. read   input from the user.
2.few-shot:
    Write a python program to count number of vowels in a given string.
Example: sruniversity
Vowel count=4.
Read input from the user.

Zero-shot   few-shot

```python
task3cursor.py > ...
1    def format_name(full_name):
2        """
3        Format a full name as "last,first" with proper capitalization.
4
5        Args:
6            full_name (str): Full name in "first last" format
7
8        Returns:
9            str: Formatted name as "Last,First"
10       """
11       # Split the name into parts and strip whitespace
12       name_parts = full_name.strip().split()
13
14       if len(name_parts) < 2:
15           return "Error: Please enter both first and last name"
16
17       # Get first and last name
18       first_name = name_parts[0]
19       last_name = name_parts[-1]
20
21       # Capitalize first letter of each name
22       first_name = first_name.capitalize()
23       last_name = last_name.capitalize()
24
25       # Format as "last,first"
26       formatted_name = f"{last_name},{first_name}"
27
28       return formatted_name
```

```python
task4cursor.py > ...
20
21   def main():
22       """Main function to get user input and count vowels."""
23       print("Vowel Counter")
24       print("=" * 20)
25
26       # Get input from user
27       user_input = input("Enter a string: ")
28
29       # Count vowels
30       vowel_count = count_vowels(user_input)
31
32       # Display result
33       print(f"Number of vowels in '{user_input}': {vowel_count}")
34
35       # Show which vowels were found
36       vowels_found = [char for char in user_input if char.lower() in 'aeiou']
37       if vowels_found:
38           print(f"Vowels found: {vowels_found}")
39       else:
40           print("No vowels found in the string.")
41
42
43   if __name__ == "__main__":
44       main()
45   | Ctrl+L to chat, Ctrl+K to generate
```

Problems   Output   Debug Console   Terminal   Ports

```
rsor\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '50563' '--' 'c
lab4\task4cursor.py'
Vowel Counter
====================
Enter a string: harichandana
Number of vowels in 'harichandana': 5
Vowels found: ['a', 'i', 'a', 'a', 'a']
PS C:\Users\thoop\lab4> []
```

```python
task4_1cursor.py > ...
1    def count_vowels(text):
2        """
3        Count the number of vowels in a given string.
4
5        Args:
6            text (str): Input string to count vowels in
7
8        Returns:
9            int: Number of vowels in the string
10       """
11       vowels = 'aeiouAEIOU'
12       count = 0
13
14       for char in text:
15           if char in vowels:
16               count += 1
17
18       return count
19
20
21   def main():
22       """Main function to demonstrate vowel counting with example and user input."""
23       print("Vowel Counter")
24       print("=" * 30)
```

```python
task4_1cursor.py > ⊙ main
21   def main():
26       # Show the example
27       example = "sruniversity"
28       example_count = count_vowels(example)
29       print(f"Example: {example}")
30       print(f"Vowel count = {example_count}")
31       # Show the vowels found in the example
32       vowels_found = [char for char in example if char.lower() in 'aeiou']
33       print(f"Vowels found: {vowels_found}")
34       print("\n" + "=" * 30)
35       # Get input from user
36       user_input = input("Enter a string: ")
37       # Count vowels in user input
38       user_count = count_vowels(user_input)
39       # Display result
40       print(f"Vowel count = {user_count}")
41       # Show which vowels were found
42       user_vowels = [char for char in user_input if char.lower() in 'aeiou']
43       if user_vowels:
44           print(f"Vowels found: {user_vowels}")
45       else:
46           print("No vowels found in the string.")
47   if __name__ == "__main__":
48       main()
49
```

Problems   Output   Debug Console   Terminal   Ports

```
Vowel Counter
====================
Example: sruniversity
Vowel count = 4
Vowels found: ['u', 'i', 'e', 'i']

====================
Enter a string: harichandana
Enter a string: harichandana
Vowel count = 5
```

Task Description#5

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

Expected Output#5

- Working file-processing function with AI-guided logic

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

Example:hello sr university.hanmakonda warangal.
Top engineering college.
Offers wide variety of courses.
No.of lines in a.txt=3.

```python
def count_lines_in_file(filename):
    """
    Read a text file and return the number of lines.

    Args:
        filename (str): Name of the file to read

    Returns:
        int: Number of lines in the file
    """
    try:
        with open(filename, 'r', encoding='utf-8') as file:
            lines = file.readlines()
            return len(lines)
    except FileNotFoundError:
        print(f"Error: File '{filename}' not found.")
        return -1
    except Exception as e:
        print(f"Error reading file: {e}")
        return -1


def display_file_content(filename):
    """
    Display the content of a file with line numbers.

    Args:
        filename (str): Name of the file to display
    """
    try:
        with open(filename, 'r', encoding='utf-8') as file:
            lines = file.readlines()
            print(f"Content of {filename}:")
            print("-" * 40)
            for i, line in enumerate(lines, 1):
                print(f"Line {i}: {line.strip()}")
            print("-" * 40)
    except FileNotFoundError:
        print(f"Error: File '{filename}' not found.")
    except Exception as e:
        print(f"Error reading file: {e}")

def main():
    """Main function to demonstrate line counting with example and file reading."""
    print("File Line Counter")
    print("=" * 40)
    # Show example
    print("Example:")
    example_lines = [
        "hello sr university.hanmakonda warangal.",
        "Top engineering college.",
        "Offers wide variety of courses."
    ]
    print("Content:")
    for i, line in enumerate(example_lines, 1):
        print(f"Line {i}: {line}")

    print(f"No.of lines in a.txt = {len(example_lines)}.")
    print("\n" + "=" * 40)


    # Read actual a.txt file
    filename = "a.txt"
    line_count = count_lines_in_file(filename)

    if line_count >= 0:
        print(f"Reading file: {filename}")
        display_file_content(filename)
        print(f"No.of lines in {filename} = {line_count}.")
    else:
        print("Could not read the file.")
```

Problems   Output   Debug Console   Terminal   Ports   Python + ∨ ⊓ 🗑

No.of lines in a.txt = 8.