

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	Applicable to Batches	
AssignmentNumber: 4.3(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:	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.

Task Description#1

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

Expected Output#1

Prompt: Create a Python code that determines the year is a leap year or not and determine it as a leap year if it is one or determine as a not a leap year

The screenshot shows a Visual Studio Code editor window with the 'AI. ASSISTED CODING' sidebar open. The main editor displays a Python file named '99.py' with the following code:

```

1 def is_leap_year(year):
2     if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
3         return True
4     else:
5         return False
6
7 year = int(input("Enter a year: "))
8 if is_leap_year(year):
9     print(f"{year} is a leap year.")
10 else:
11     print(f"{year} is not a leap year.")

```

The AI sidebar shows the prompt: "create a python code that determines the year is a leap year or not and determine it as a leap year if it is one or determine as a not a leap year". Below the prompt, there is an 'Ask a question' input field and a 'GPT 4.1' model selector. The code generated by the AI is displayed in the editor.

Task Description#2

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

Expected Output#2

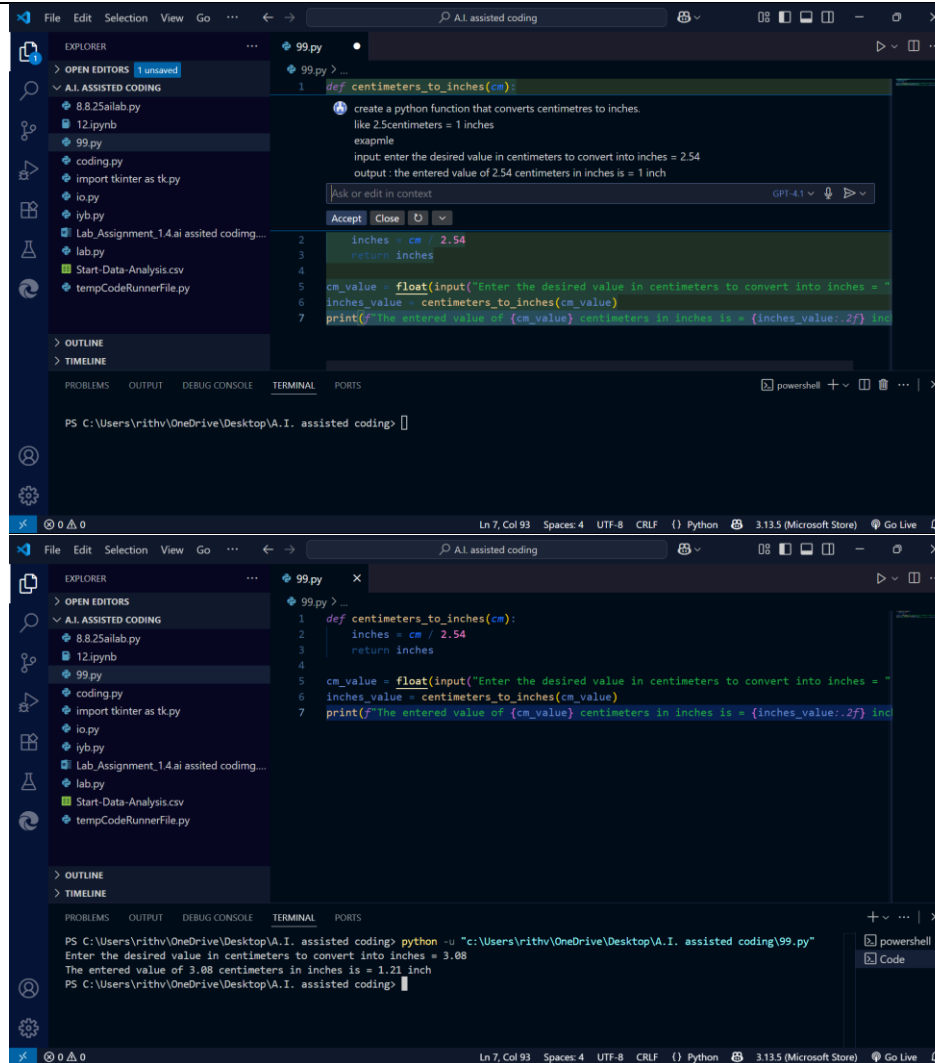
Prompt: create a python function that converts centimetres to inches.

like 2.5centimeters = 1 inches

exapmle

input: enter the desired value in centimeters to convert into inches = 2.54

output : the entered value of 2.54 centimeters in inches is = 1 inch



Task Description#3

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

Expected Output#3

Prompt: generate a function that formats full names as "Last, First".

example 1

input: enter a name: Rithvik Gona

output: last name = Gona

First name = Rithvik

example 2

input: enter a name: Shiva Nalumachu

output: Last name = Nalumachu

First name = Shiva

The top screenshot shows a VS Code editor with a file named '99.py'. The code in the editor is a Python function 'format_name' that takes a full name as input and returns the first and last names. The function is defined as follows:

```
def format_name(full_name):  
    parts = full_name.strip().split()  
    if len(parts) < 2:  
        print("Please enter both first and last names.")  
        return  
    first_name = parts[0]  
    last_name = parts[1]  
    print(f"Last name = {last_name}")  
    print(f"First name = {first_name}")  
  
# Example usage:  
name = input("Enter a name: ")  
format_name(name)
```

The bottom screenshot shows the same code being executed in a terminal. The output is as follows:

```
PS C:\Users\rithv\OneDrive\Desktop\A.I. assisted coding> python -u "c:\Users\rithv\OneDrive\Desktop\A.I. assisted coding\99.py"  
Enter a name: venkat sai  
Last name = sai  
First name = venkat  
PS C:\Users\rithv\OneDrive\Desktop\A.I. assisted coding>
```

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

Zero shot prompt : generate a python function to count the number of the vowels in the input string

```

1 def count_vowels(text):
2     vowels = 'aeiouAEIOU'
3     return sum(1 for char in text if char in vowels)
4
5 user_input = input("Enter some text: ")
6 vowel_count = count_vowels(user_input)
7 print(f"Number of vowels: {vowel_count}")

```

```

PS C:\Users\rithv\OneDrive\Desktop\A.I. assisted coding> python -u "c:\Users\rithv\OneDrive\Desktop\A.I. assisted coding\99.py"
Enter some text: rithvikigona
Number of vowels: 18
PS C:\Users\rithv\OneDrive\Desktop\A.I. assisted coding>

```

Few shot prompt : create a python function that counts the numbers of the vowels in the input given by the user and give back the vowels and the vowel count in the input

```

1 def count_vowels(user_input):
2     vowels = 'aeiouAEIOU'
3     found_vowels = [char for char in user_input if char in vowels]
4     vowel_count = len(found_vowels)
5     return found_vowels, vowel_count
6
7 if __name__ == "__main__":
8     text = input("Enter some text: ")
9     vowels, count = count_vowels(text)
10    print(f"Vowels found: {vowels}")
11    print(f"Number of vowels: {count}")

```

```

PS C:\Users\rithv\OneDrive\Desktop\A.I. assisted coding> python -u "c:\Users\rithv\OneDrive\Desktop\A.I. assisted coding\99.py"
Enter some text: rithvikigona
Vowels found: ['i', 'i', 'o', 'a']
Number of vowels: 4
PS C:\Users\rithv\OneDrive\Desktop\A.I. assisted coding>

```

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

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

Example 1

Input: A text file named example.txt with the following content:

Line 1

Line 2

Line 3

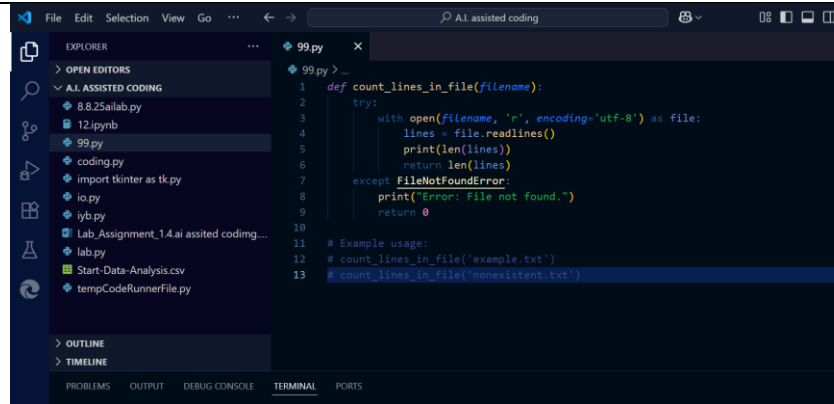
Expected Output:3

Example 2

Input: A non-existent text file named nonexistent.txt

Expected Output:

0 (with an appropriate error message indicating the file was not found)



```
1 def count_lines_in_file(filename):
2     try:
3         with open(filename, 'r', encoding='utf-8') as file:
4             lines = file.readlines()
5             print(len(lines))
6             return len(lines)
7     except FileNotFoundError:
8         print("Error: File not found.")
9     return 0
10
11 # Example usage:
12 # count_lines_in_file('example.txt')
13 # count_lines_in_file('nonexistent.txt')
```

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