

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		<table border="1"> <tr><td>Dr. V. Venkataramana (Co-ordinator)</td></tr> <tr><td>Dr. T. Sampath Kumar</td></tr> <tr><td>Dr. Pramoda Patro</td></tr> <tr><td>Dr. Brij Kishor Tiwari</td></tr> <tr><td>Dr.J.Ravichander</td></tr> <tr><td>Dr. Mohammand Ali Shaik</td></tr> <tr><td>Dr. Anirodh Kumar</td></tr> <tr><td>Mr. S.Naresh Kumar</td></tr> <tr><td>Dr. RAJESH VELPULA</td></tr> <tr><td>Mr. Kundhan Kumar</td></tr> <tr><td>Ms. Ch.Rajitha</td></tr> <tr><td>Mr. M Prakash</td></tr> <tr><td>Mr. B.Raju</td></tr> <tr><td>Intern 1 (Dharma teja)</td></tr> <tr><td>Intern 2 (Sai Prasad)</td></tr> <tr><td>Intern 3 (Sowmya)</td></tr> <tr><td>NS_2 ( Mounika)</td></tr> </table>		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)
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 <b>Lab Objectives:</b> <ul style="list-style-type: none"> <li>To explore and apply different levels of prompt examples in AI-assisted code</li> </ul>	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:

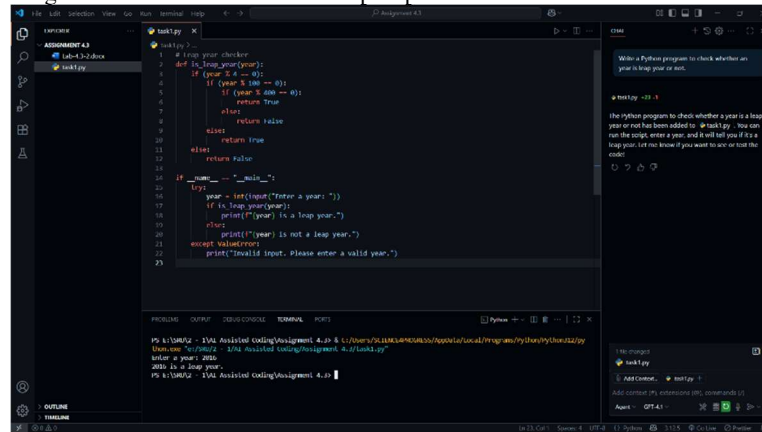
- Use zero-shot prompting to instruct AI with minimal context.
- Use one-shot prompting with a single example to guide AI code generation.
- Apply few-shot prompting using multiple examples to improve AI responses.
- Compare AI outputs across the three prompting strategies.

### 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



```
task1.py
1 # leap year checker
2 def is_leap_year(year):
3     if (year % 4 == 0):
4         if (year % 100 == 0):
5             if (year % 400 == 0):
6                 return True
7             else:
8                 return False
9         else:
10            return True
11     else:
12        return False
13
14 if __name__ == "__main__":
15     year = int(input("Enter a year: "))
16     if is_leap_year(year):
17         print("Year is a leap year.")
18     else:
19         print("Year is not a leap year.")
20 except ValueError:
21     print("Invalid input. Please enter a valid year.")
22
```

Write a Python program to check whether a year is leap year or not.

task1.py - 11

the python program to check whether a year is a leap year or not has been added to task1.py. You can run the script, enter a year, and it will tell you if it's a leap year. Let me know if you want to see or test the code.

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS E:\SRU2 - 1\AI Assisted Coding\Assignment 4.3> python task1.py

Enter a year: 2020

2020 is a leap year.

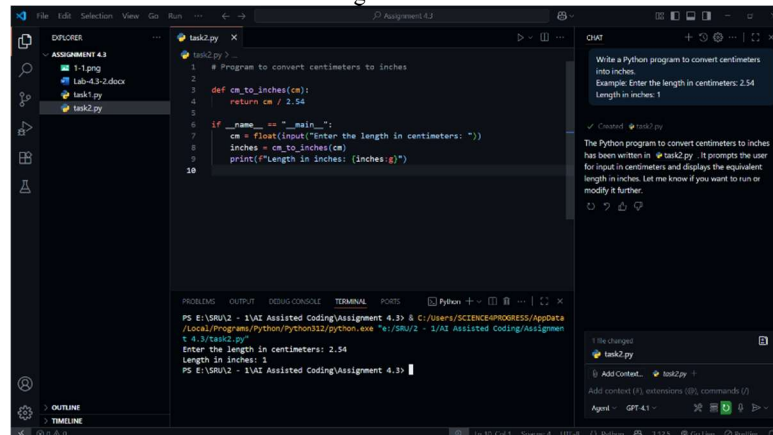
PS E:\SRU2 - 1\AI Assisted Coding\Assignment 4.3>

### 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



```
task2.py
1 # Program to convert centimeters to inches
2 def cm_to_inches(cm):
3     return cm / 2.54
4
5
6 if __name__ == "__main__":
7     cm = float(input("Enter the length in centimeters: "))
8     inches = cm_to_inches(cm)
9     print("Length in inches: (inches:g)")
10
```

Write a Python program to convert centimeters into inches.

Example: Enter the length in centimeters: 2.54

Length in inches: 1

✓ Created task2.py

The Python program to convert centimeters to inches has been written in task2.py. It prompts the user for input in centimeters and displays the equivalent length in inches. Let me know if you want to run or modify it further.

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS E:\SRU2 - 1\AI Assisted Coding\Assignment 4.3> python task2.py

Enter the length in centimeters: 2.54

Length in inches: 1

PS E:\SRU2 - 1\AI Assisted Coding\Assignment 4.3>

### 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

```

1 # Program to format full names as First Name and Last Name
2 full_name = input("Enter Your Full Name: ").strip()
3
4 # Split the name into parts
5 name_parts = full_name.split()
6
7 if len(name_parts) == 2:
8     first_name = name_parts[0]
9     last_name = name_parts[1]
10    print(f"First Name: {first_name}")
11    print(f"Last Name: {last_name}")
12 else:
13    print("Please enter both first and last name.")
14

```

Terminal Output:

```

PS E:\SRUJ2 - 1\AI2 Assisted Coding\Assignment 4.3> python.exe "e:\SRUJ2 - 1\AI2 Assisted Coding\Assignment 4.3\task3.py"
Enter Your Full Name: Bhukya Karthik
First Name: Bhukya
Last Name: Karthik

```

### 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

Initial Prompt:

```

Write a Python program to count the number of vowels in a string.

```

Initial Code:

```

1 def count_vowels(s):
2     vowels = "aeiouAEIOU"
3     count = 0
4     for char in s:
5         if char in vowels:
6             count += 1
7     return count
8
9 # Example usage
10 input_str = input("Enter a string: ")
11 print("Number of vowels:", count_vowels(input_str))

```

Terminal Output (Initial):

```

PS E:\SRUJ2 - 1\AI2 Assisted Coding\Assignment 4.3> python.exe "e:\SRUJ2 - 1\AI2 Assisted Coding\Assignment 4.3\task4.py"
Enter a string: Mississippi
Number of vowels: 4

```

Updated Prompt (Few-shot):

```

Write a Python program to count the number of vowels in a string.
Example 1: Enter a string: Karthik
Number of vowels: 2
Example 2: Enter a string: Lugman
Number of vowels: 3
Example 3: Enter a string: Shaheed
Number of vowels: 3

```

Updated Code:

```

1 def count_vowels(s):
2     vowels = "aeiouAEIOU"
3     count = 0
4     for char in s:
5         if char in vowels:
6             count += 1
7     return count
8
9 if __name__ == "__main__":
10    input_str = input("Enter a string: ")
11    print("Number of vowels:", count_vowels(input_str))

```

Terminal Output (Updated):

```

PS E:\SRUJ2 - 1\AI2 Assisted Coding\Assignment 4.3> python.exe "e:\SRUJ2 - 1\AI2 Assisted Coding\Assignment 4.3\task4.py"
Enter a string: Srivastava
Number of vowels: 4

```

Compare these two prompts i.e. zero shot and few shot

Great — you've provided two screenshots that showcase two different types of AI prompts used in Visual Studio Code (with GitHub Copilot or ChatGPT extensions). Let's compare the Zero-shot and Few-shot prompt strategies as shown:

### First Screenshot: Zero-shot Prompt

**Prompt:**  
Write a Python program to count the number of vowels in a string.

**Response:**  
The AI is given only the task description, with no examples.

Ask anything

Attach Search Voice

ChatGPT can make mistakes. Check important info. See Cookie Preferences.

### Second Screenshot: Few-shot Prompt

**Prompt:**  
Write a Python program to count the number of vowels in a string.  
Example 1: Enter a string: Karthik — Number of vowels: 2  
Example 2: Enter a string: Lagan — Number of vowels: 2  
Example 3: Enter a string: Shalini — Number of vowels: 3

**Response:**

Ask anything

Attach Search Voice

ChatGPT can make mistakes. Check important info. See Cookie Preferences.

**Response:**

- Same core function (`count_vowels(s)`) is generated.
- The main difference AI was guided by examples, so it understands:
  - Expected input/output structure.
  - Case sensitivity.
  - Output formatting expectations.
- It added the `if __name__ == "__main__":` guard, showing better practice for modular code.
- Output shown: Correct vowel count for "Srivatsava".

**Pros:**

- More accurate in matching user expectations due to examples.
- Encourages better code structure.
- More robust when the task is ambiguous.

**Cons:**

- Requires more effort to write the prompt.
- Slightly slower response due to extra processing.

### Summary Table

Ask anything

Attach Search Voice

ChatGPT can make mistakes. Check important info. See Cookie Preferences.

Feature	Zero-shot Prompt	Few-shot Prompt
Prompt Type	Instruction only	Instruction + Examples
Code Accuracy	High (for simple tasks)	Higher (especially with edge cases)
Understanding of Format	Basic	Precise, guided by examples
Best For	Simple, well-defined tasks	Complex or subjective tasks
Code Structure	Functional but minimal	More complete (e.g., <code>__main__</code> )

**Verdict:**

- Use Zero-shot for quick prototyping or straightforward logic.
- Use Few-shot when the task has formatting requirements, edge cases, or expected outputs that aren't obvious from the instruction alone.

Ask anything

Attach Search Voice

ChatGPT can make mistakes. Check important info. See Cookie Preferences.

### 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

The screenshot shows a Jupyter Notebook interface with a file explorer on the left, a code editor in the center, and an output area at the bottom. The code editor contains a Python function `count_lines_in_file(filename)` that reads a file and returns the number of lines. The output area shows the function being called with `task5.txt` and the result `5`.

```
def count_lines_in_file(filename):
    with open(filename, 'r', encoding='utf-8') as file:
        lines = file.readlines()
        return len(lines)

if __name__ == "__main__":
    filename = input("Enter the path to the .txt file: ")
    try:
        num_lines = count_lines_in_file(filename)
        print(f"Number of lines in '{filename}': {num_lines}")
    except FileNotFoundError:
        print("File not found. Please check the file path.")
    except Exception as e:
        print(f"An error occurred: {e}")
```

PS E:\SRUJ - 1\AI Assisted Coding\Assignment 4.3> & C:\Users\SCIENCE\AppData\Local\Programs\Python\Python112\python.exe "e:\SRUJ - 1\AI Assisted Coding\Assignment 4.3\task5.py"

Enter the path to the .txt file: task5.txt  
Number of lines in 'task5.txt': 5  
PS E:\SRUJ - 1\AI Assisted Coding\Assignment 4.3>

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