

NAME :P.Pranay Kumar

H.NO:2303A51829

BATCH:26

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: B. Tech		Assignment Type: Lab	Academic Year:2025-2026
Course Coordinator Name		Dr. Rishabh Mittal	
Instructor(s) Name		Mr. S Naresh Kumar	
		Ms. B. Swathi	
		Dr. Sasanko Shekhar Gantayat	
		Mr. Md Sallauddin	
		Dr. Mathivanan	
		Mr. Y Srikanth	
		Ms. N Shilpa	
		Dr. Rishabh Mittal (Coordinator)	
		Dr. R. Prashant Kumar	
		Mr. Ankushavali MD	
		Mr. B Viswanath	
		Ms. Sujitha Reddy	
		Ms. A. Anitha	
		Ms. M.Madhuri	
		Ms. Katherashala Swetha	
		Ms. Velpula sumalatha	
		Mr. Bingi Raju	
Course Code	23CS002PC304	Course Title	AI Assisted Coding
Year/Sem	III/I	Regulation	R23
Date and Day of Assignment	Week 2 - Wednesday	Time(s)	23CSBTB01 To 23CSBTB52
Duration	2 Hours	Applicable to Batches	All batches
Assignment Number: 3.3(Present assignment number)/24(Total number of assignments)			
Q.No.	Question	Expected Time to complete	
1	<p><b>Lab 4: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques</b></p> <p><b>Lab Objectives</b></p> <ul style="list-style-type: none"> <li>To explore and apply different levels of prompt examples in AI-assisted code generation</li> <li>To understand how zero-shot, one-shot, and few-shot prompting affect AI output quality</li> <li>To evaluate the impact of context richness and example quantity on AI performance</li> <li>To build awareness of prompt strategy effectiveness for different problem types</li> </ul>	Week2 - Wednesday	

**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**

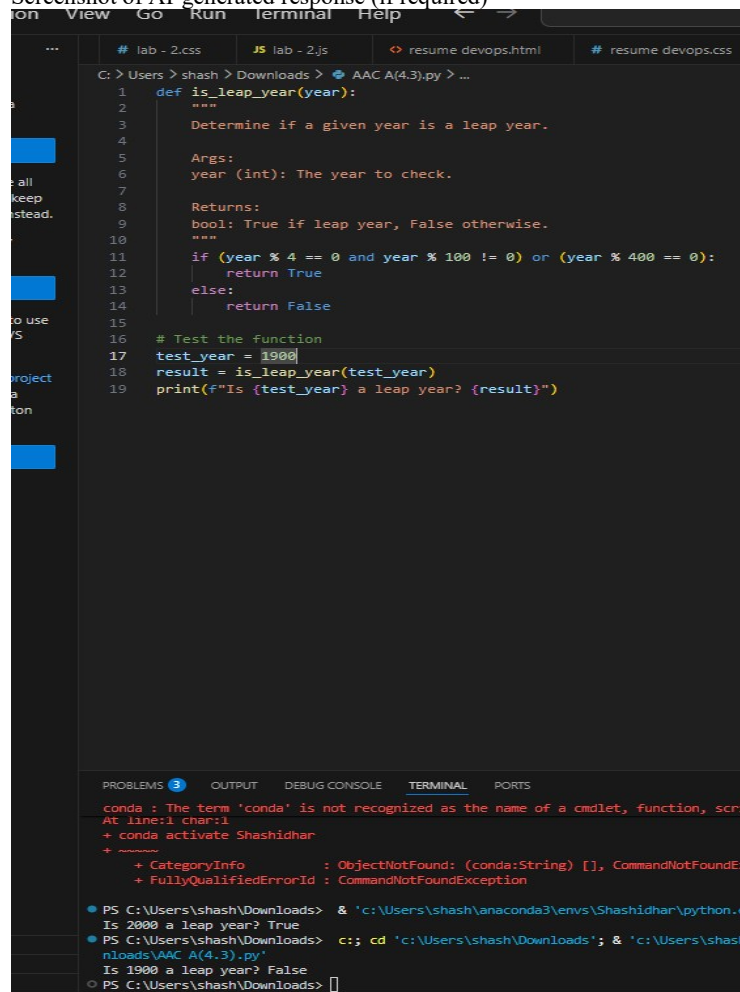
Use zero-shot prompting to instruct an AI tool to generate a Python function that:

- 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**

- AI-generated leap year checking function
- Correct logical conditions
- Sample input and output
- Screenshot of AI-generated response (if required)



```
C: > Users > shash > Downloads > AAC A(4.3).py > ...
1 def is_leap_year(year):
2     """
3     Determine if a given year is a leap year.
4
5     Args:
6     year (int): The year to check.
7
8     Returns:
9     bool: True if leap year, False otherwise.
10    """
11    if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
12        return True
13    else:
14        return False
15
16    # Test the function
17    test_year = 1900
18    result = is_leap_year(test_year)
19    print(f"Is {test_year} a leap year? {result}")

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS
conda : The term 'conda' is not recognized as the name of a cmdlet, function, scrip
AT line:1 char:1
+ conda activate Shashidhar
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (conda:String) [], CommandNotFoundEx
+ FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\shash\Downloads> & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.e
Is 2000 a leap year? True
PS C:\Users\shash\Downloads> c:; cd 'c:\Users\shash\Downloads'; & 'c:\Users\shash
nloads\AAC A(4.3).py'
Is 1900 a leap year? False
PS C:\Users\shash\Downloads> 
```

**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:

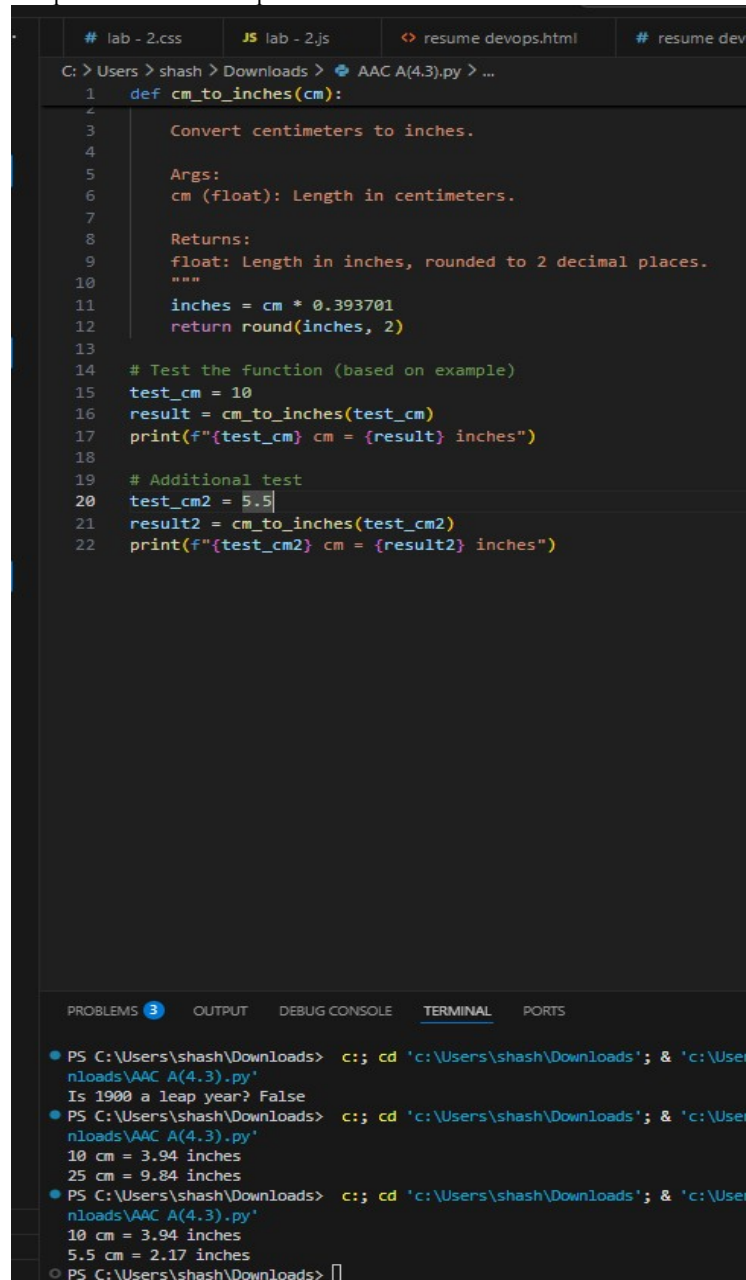
- Converts centimeters to inches
- Uses the correct mathematical formula

**Example provided in prompt:**

Input: 10 cm → Output: 3.94 inches

**Expected Output**

- Python function with correct conversion logic
- Accurate calculation
- Sample test cases and outputs



```
# lab - 2.css JS lab - 2.js resume devops.html # resume devops.html
C: > Users > shash > Downloads > AAC A(4.3).py > ...
1 def cm_to_inches(cm):
2
3     Convert centimeters to inches.
4
5     Args:
6     cm (float): Length in centimeters.
7
8     Returns:
9     float: Length in inches, rounded to 2 decimal places.
10    """
11    inches = cm * 0.393701
12    return round(inches, 2)
13
14    # Test the function (based on example)
15    test_cm = 10
16    result = cm_to_inches(test_cm)
17    print(f"{test_cm} cm = {result} inches")
18
19    # Additional test
20    test_cm2 = 5.5
21    result2 = cm_to_inches(test_cm2)
22    print(f"{test_cm2} cm = {result2} inches")

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS
• PS C:\Users\shash\Downloads> c::; cd 'c:\Users\shash\Downloads'; & 'c:\Users\shash\Downloads\AAC A(4.3).py'
Is 1900 a leap year? False
• PS C:\Users\shash\Downloads> c::; cd 'c:\Users\shash\Downloads'; & 'c:\Users\shash\Downloads\AAC A(4.3).py'
10 cm = 3.94 inches
25 cm = 9.84 inches
• PS C:\Users\shash\Downloads> c::; cd 'c:\Users\shash\Downloads'; & 'c:\Users\shash\Downloads\AAC A(4.3).py'
10 cm = 3.94 inches
5.5 cm = 2.17 inches
• PS C:\Users\shash\Downloads> 
```

**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:

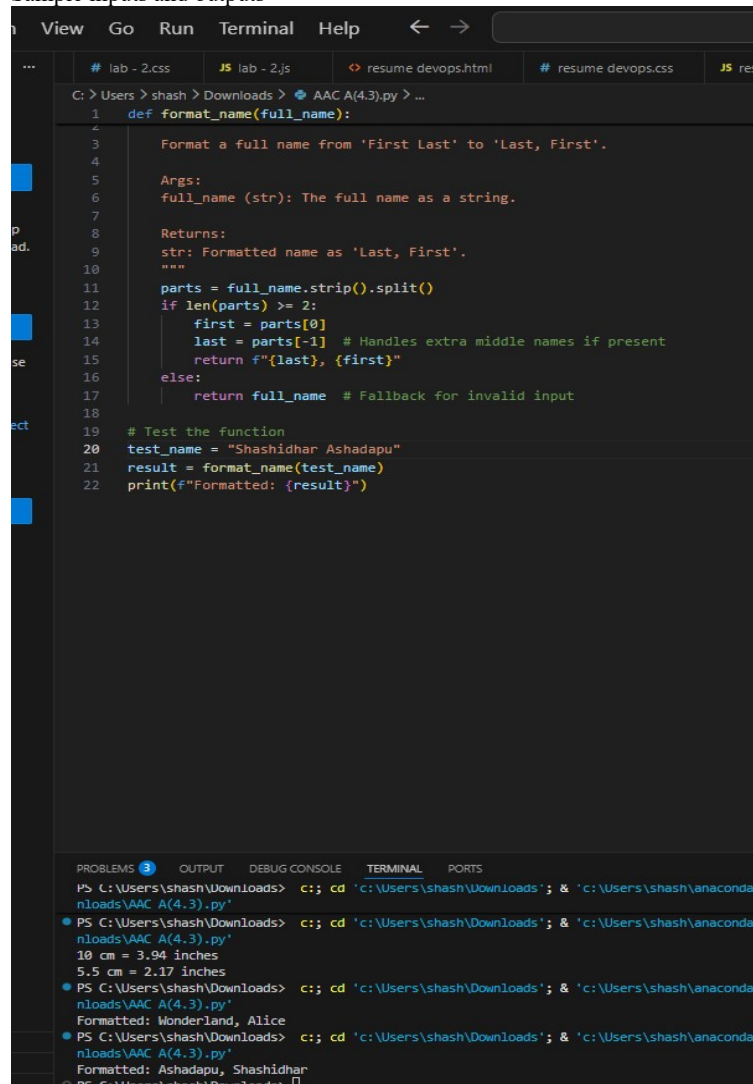
- Accepts a full name as input
- Formats it as "Last, First"

**Example formats:**

- "John Smith" → "Smith, John"
- "Anita Rao" → "Rao, Anita"

**Expected Output**

- Well-structured Python function
- Output strictly following example patterns
- Correct handling of names
- Sample inputs and outputs



```
View Go Run Terminal Help
# lab - 2.css JS lab - 2.js resume devops.html # resume devops.css JS res
C:\Users\shash\Downloads> AAC A(4.3).py > ...
1 def format_name(full_name):
2
3     Format a full name from 'First Last' to 'Last, First'.
4
5     Args:
6     full_name (str): The full name as a string.
7
8     Returns:
9     str: Formatted name as 'Last, First'.
10    """
11    parts = full_name.strip().split()
12    if len(parts) >= 2:
13        first = parts[0]
14        last = parts[-1] # Handles extra middle names if present
15        return f"{last}, {first}"
16    else:
17        return full_name # Fallback for invalid input
18
19 # Test the function
20 test_name = "Shashidhar Ashadapu"
21 result = format_name(test_name)
22 print(f"Formatted: {result}")

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\shash\Downloads> c:\> cd 'c:\Users\shash\Downloads'; & 'c:\Users\shash\anaconda
nloads\AAC A(4.3).py'
PS C:\Users\shash\Downloads> c:\> cd 'c:\Users\shash\Downloads'; & 'c:\Users\shash\anaconda
nloads\AAC A(4.3).py'
10 cm = 3.94 inches
5.5 cm = 2.17 inches
PS C:\Users\shash\Downloads> c:\> cd 'c:\Users\shash\Downloads'; & 'c:\Users\shash\anaconda
nloads\AAC A(4.3).py'
Formatted: Wonderland, Alice
PS C:\Users\shash\Downloads> c:\> cd 'c:\Users\shash\Downloads'; & 'c:\Users\shash\anaconda
nloads\AAC A(4.3).py'
Formatted: Ashadapu, Shashidhar
PS C:\Users\shash\Downloads> c:\> cd 'c:\Users\shash\Downloads'; & 'c:\Users\shash\anaconda
nloads\AAC A(4.3).py'
```

**Task 4: Comparative Analysis – Zero-Shot vs Few-Shot****Scenario**

Different prompt strategies may produce different code quality.

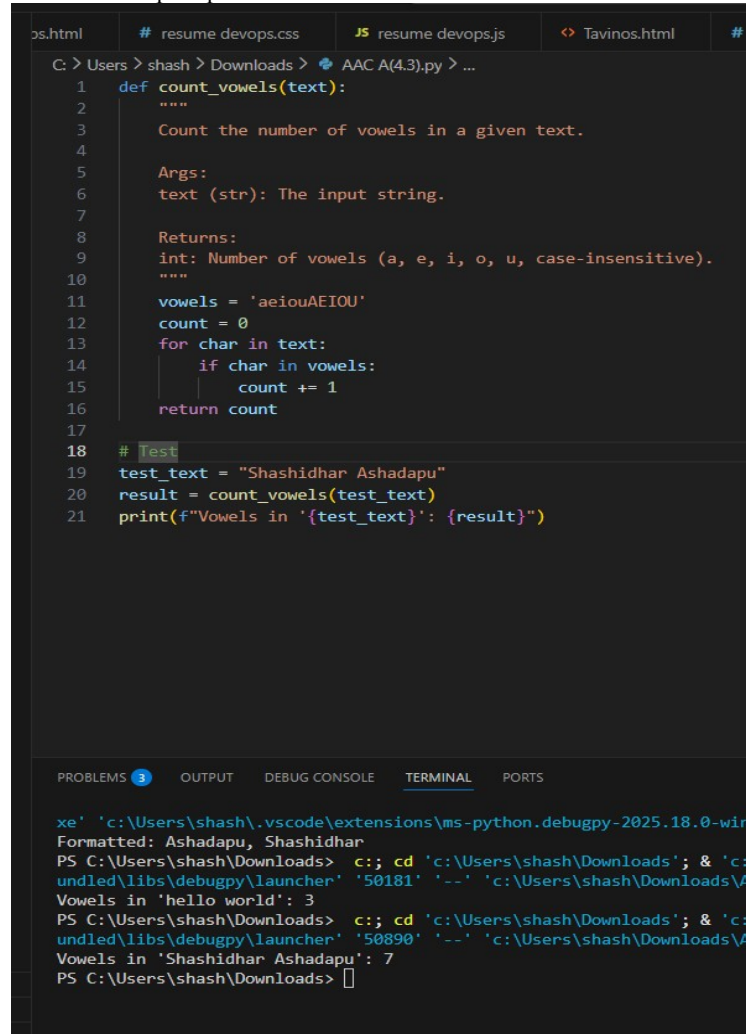
**Task Description**

- 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
- Comparison table or short reflection paragraph
- Conclusion on prompt effectiveness



The screenshot shows a VS Code editor with a Python file named `AAC A(4.3).py`. The script defines a function `count_vowels(text)` that counts the number of vowels in a given string. The function uses a set of vowels `'aeiouAEIOU'` and iterates through each character in the input string, incrementing a counter if the character is a vowel. The script also includes a test case where the function is called with the string `"Shashidhar Ashadapu"` and the result is printed.

```
1 def count_vowels(text):
2     """
3     Count the number of vowels in a given text.
4
5     Args:
6     text (str): The input string.
7
8     Returns:
9     int: Number of vowels (a, e, i, o, u, case-insensitive).
10    """
11    vowels = 'aeiouAEIOU'
12    count = 0
13    for char in text:
14        if char in vowels:
15            count += 1
16    return count
17
18 # Test
19 test_text = "Shashidhar Ashadapu"
20 result = count_vowels(test_text)
21 print(f"Vowels in '{test_text}': {result}")
```

The terminal output shows the execution of the script, displaying the result of the vowel count for the test string:

```
Formatted: Ashadapu, Shashidhar
PS C:\Users\shash\Downloads> c:; cd 'c:\Users\shash\Downloads'; & 'c:\
undled\libs\debugpy\launcher' '50181' '--' 'c:\Users\shash\Downloads\AAC
Vowels in 'hello world': 3
PS C:\Users\shash\Downloads> c:; cd 'c:\Users\shash\Downloads'; & 'c:\
undled\libs\debugpy\launcher' '50890' '--' 'c:\Users\shash\Downloads\AAC
Vowels in 'Shashidhar Ashadapu': 7
PS C:\Users\shash\Downloads>
```

**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
- Correct line count
- Sample .txt input and output
- AI-assisted logic explanation

```
os.html # resume devops.css JS resume devops.js <> Tavinios.html # Tav
C: > Users > shash > Downloads > AAC A(4.3).py > ...
1 def count_lines_in_file(file_path):
2     """
3     Count the number of lines in a text file.
4
5     Args:
6     file_path (str): Path to the .txt file.
7
8     Returns:
9     int: Number of lines, or 0 if file not found.
10
11     Raises:
12     FileNotFoundError: If the file doesn't exist.
13     """
14     try:
15         with open(file_path, 'r') as file:
16             lines = file.readlines()
17             return len(lines)
18     except FileNotFoundError:
19         print(f"Error: File '{file_path}' not found.")
20         return 0
21
22 # Create a sample file for testing (run this once)
23 with open('sample.txt', 'w') as f:
24     f.write(["Line1\nLine2\nLine3\nLine4\nLine5"])
25
26 # Test the function
27 result = count_lines_in_file('sample.txt')
28 print(f"Line count in 'sample.txt': {result}")

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS
undled\libs\debugpy\launcher' '50181' '--' 'c:\Users\shash\Downloads\AAC
Vowels in 'hello world': 3
PS C:\Users\shash\Downloads> c;; cd 'c:\Users\shash\Downloads'; & 'c:\U
undled\libs\debugpy\launcher' '50890' '--' 'c:\Users\shash\Downloads\AAC
Vowels in 'Shashidhar Ashadapu': 7
● PS C:\Users\shash\Downloads> c;; cd 'c:\Users\shash\Downloads'; & 'c:\U
undled\libs\debugpy\launcher' '61017' '--' 'c:\Users\shash\Downloads\AAC
Line count in 'sample.txt': 3
● PS C:\Users\shash\Downloads> c;; cd 'c:\Users\shash\Downloads'; & 'c:\U
undled\libs\debugpy\launcher' '64519' '--' 'c:\Users\shash\Downloads\AAC
Line count in 'sample.txt': 5
○ PS C:\Users\shash\Downloads> 
```

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