

AI-ASSISTANT-COADING

Lab3.3

2303A51247

Batch-18

Question1:

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)

Prompt:

```
#accept a number as input and check wheather it a leap year or not
```

Code:

```
def is_leap_year(year):  
    if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):  
        return True  
    else:  
        return False  
year = int(input("Enter a year to check if it's a leap year: "))  
if is_leap_year(year):  
    print(f"{year} is a leap year.")  
else:  
    print(f"{year} is not a leap year.")
```

Output:

```
PS C:\Users\nagas\OneDrive\Desktop\AI Lab> & C:/Users/nagas/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/nagas/OneDrive/Desktop/AI Lab/Lab3.3.py"
Enter a year to check if it's a leap year: 2025
2025 is not a leap year.
PS C:\Users\nagas\OneDrive\Desktop\AI Lab> 
```

Analysis:

The AI generated correct leap year logic using only instructions without examples.

Question2:

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

Prompt:

```
#1cm=0.393701 inches
#accept a number in cm and convert it into inches
```

Code:

```
def cm_to_inches(cm):
    inches = cm * 0.393701
    return inches
cm = float(input("Enter length in centimeters: "))
print(f"{cm} cm is equal to {cm_to_inches(cm)} inches.")
```

Output:

```
Enter length in centimeters: 30
30.0 cm is equal to 11.81103 inches.
PS C:\Users\nagas\OneDrive\Desktop\AI Lab> 
```

Analysis:

Providing one example helped the AI give accurate unit conversion and output format.

Question3:

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

Prompt:

```
#"John Smith" -> "Smith, John"
#"Anita Rao" -> "Rao, Anita"
#accept a full name and print it in "last name, first name" format
```

Code:

```
def format_name(full_name):
    name_parts = full_name.split()
    if len(name_parts) >= 2:
        first_name = " ".join(name_parts[:-1])
        last_name = name_parts[-1]
        return f"{last_name}, {first_name}"
    else:
        return full_name # Return as is if there's no space
```

```
full_name = input("Enter a full name: ")
print(f"Formatted name: {format_name(full_name)}")
```

Output:

```
Enter a full name: Nagasri Reddy
Formatted name: Reddy, Nagasri
PS C:\Users\nagas\OneDrive\Desktop\AI Lab>
```

Analysis:

Multiple examples helped the AI correctly follow the required name format.

Question4: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:

- o Accuracy

- o Readability

- o Logical clarity

Expected Output

- Two vowel-counting functions
- Comparison table or short reflection paragraph
- Conclusion on prompt effectiveness

Zero-Shot:

Prompt:

```
#generate a function that counts vowels in a string
```

Code:

```
def count_vowels(input_string):
    vowels = "aeiouAEIOU"
    count = 0
    for char in input_string:
        if char in vowels:
            count += 1
    return count

input_string = input("Enter a string: ")
print(f"Number of vowels in the string: {count_vowels(input_string)}")
```

Output:

```
Enter a string: nagasrireddy
Number of vowels in the string: 4
PS C:\Users\nagas\OneDrive\Desktop\AI Lab>
```

Few-Shot:

```
#"nagasri"->count=3
#"hello"->count=2
#write code for counting vowels in a string
```

Code:

```
def count_vowels(input_string):
    vowels=set("aeiouAEIOU")
    return sum(1 for char in input_string if char in vowels)
input_string = input("Enter a string: ")
print(f"Number of vowels in the string: {count_vowels(input_string)}")
```

Output:

```
Enter a string: hellow world
Number of vowels in the string: 3
PS C:\Users\nagas\OneDrive\Desktop\AI Lab>
```

Analysis:

	Zero-Shot	Few-Shot
Accuracy	Correct	Correct
Readability	Moderate	High
Logical Clarity	Clear loop	Better

Question5: 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

Prompt:

```
#Reads a .txt file
#counts the number of lines in the file
```

```
#returns the line count
```

Code:

```
def count_lines_in_file(file_path):  
    try:  
        with open(file_path, 'r') as file:  
            lines = file.readlines()  
            return len(lines)  
    except FileNotFoundError:  
        print("The specified file was not found.")  
        return None  
file_path = input("Enter the path to the .txt file: ")  
line_count = count_lines_in_file(file_path)  
if line_count is not None:  
    print(f"The number of lines in the file: {line_count}")
```

Output:

```
Enter the path to the .txt file: C:\Users\nagas\OneDrive  
e\Desktop\game_tictactao.txt  
The number of lines in the file: 40  
PS C:\Users\nagas\OneDrive\Desktop\AI Lab> 
```

Analysis:

Examples helped the AI correctly understand file reading and line counting logic.