

Assignment-4.3

Name: B.Shravya

Hall ticket No:2303A51492

Batch:08

Lab 4: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques

Task 1: Zero-Shot Prompting – Leap Year Check

The screenshot shows a code editor window for a file named '10_2.py'. The code defines a function 'is_leap_year' that checks if a given year is a leap year based on specific rules. It handles years divisible by 400, 100, and 4. A test case section at the bottom uses a list of years from 2000 to 2400 to print their status as leap or non-leap years. The right side of the screen displays AI-generated documentation and rules for leap years.

```
def is_leap_year(year):
    if year % 400 == 0:
        return True
    if year % 100 == 0:
        return False
    if year % 4 == 0:
        return True
    return False

# Test cases
if __name__ == "__main__":
    test_cases = [2000, 1900, 2004, 2001, 2020, 2021, 2024, 1996, 2100, 2400]

    for year in test_cases:
        result = is_leap_year(year)
        status = "Leap Year" if result else "Not a Leap Year"
        print(f"{year} → {status}")
```

Output:

The terminal window shows the command to run the script and its output. The script prints the status of each year in the test list: 2000 is a leap year, 1900 is not, 2004 is a leap year, etc.

```
PS D:\AI_assistant_coding> & 'c:\Users\yarav\AppData\Local\Programs\Python\Python310\python.exe' 'c:\Users\yarav\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '61916' '--' 'd:\AI_assistant_coding\10_2.py'
2000 → Leap Year
1900 → Not a Leap Year
2004 → Leap Year
2001 → Not a Leap Year
2020 → Leap Year
2021 → Not a Leap Year
2024 → Leap Year
1996 → Leap Year
2100 → Not a Leap Year
2400 → Leap Year
PS D:\AI_assistant_coding>
```

Task 2: One-Shot Prompting

Centimeters to Inches Conversion

The screenshot shows the VS Code interface with the following details:

- Editor:** The code editor displays a Python file named `10_2.py`. It contains a function `cm_to_inches` and a series of test cases for it.
- Terminal:** The terminal window shows the output of running the script, displaying various centimeter-to-inch conversions.
- Output:** The output panel shows a summary of the task completed.
- Debug Console:** The Python Debug Console shows the command used to run the script.
- Chat:** A sidebar titled "PALINDROME CHECK FUNCTION IN PYT..." is visible.
- Task Bar:** A task bar at the top indicates "Task 2: One-Shot Prompting - Centimeters to Inches Conversion".
- Key Features:**
 - Uses the standard conversion: 1 inch = 2.54 cm
 - Takes centimeters as input and returns inches
 - Rounds the result to 2 decimal places for accurate measurements
 - Includes test cases demonstrating the conversion
- Test Cases:**
 - 10 cm → 3.94 inches
 - 25 cm → 9.84 inches
 - 50 cm → 19.69 inches
 - 100 cm → 39.37 inches
 - 154 cm → 60.63 inches
 - 5.5 cm → 2.17 inches
 - 30.48 cm → 12.0 inches

Task 3: Few-Shot Prompting – Name Formatting

The screenshot shows the VS Code interface with the following details:

- Editor:** The code editor displays a Python file named `10_2.py`. It contains a function `format_name` and a series of test cases for it.
- Terminal:** The terminal window shows the output of running the script, displaying various name-formatting examples.
- Output:** The output panel shows a summary of the task completed.
- Debug Console:** The Python Debug Console shows the command used to run the script.
- Chat:** A sidebar titled "PALINDROME CHECK FUNCTION IN PYT..." is visible.
- Task Bar:** A task bar at the top indicates "Task 2: One-Shot Prompting - Centimeters to Inches Conversion".
- Key Features:**
 - Takes first and last names as separate inputs
 - Converts names to title case (first letter capitalized)
 - Handles mixed input cases (lowercase, uppercase, or mixed)
 - Returns a properly formatted full name
- Few-Shot Examples Included:**
 - john, doe → John Doe
 - jane, smith → Jane Smith
 - MARY, JOHNSON → Mary Johnson
 - robert, brown → Robert Brown
 - alice, williams → Alice Williams
 - michael, jones → Michael Jones
- Description:** A note states: "This demonstrates the few-shot prompting pattern with multiple examples showing the expected input-output format."

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

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files like "10.2.py" and "10.2.py" under "OPEN EDITORS".
- Editor:** Displays the code for "10.2.py". The code compares zero-shot and few-shot prompting for sentiment classification.
- AI Assistant:** A sidebar on the right provides "PALINDROME CHECK FUNCTION IN PY..." and two comparison tables:
 - Setup Time:** Compares "Faster" vs "Slower".
 - Token Usage:** Compares "Lower" vs "Higher".
- Zero-Shot Prompting:** A list of bullet points:
 - No examples provided; just the instruction
 - Direct approach
 - Faster to set up but less consistent
 - Relies on model's general knowledge
- Few-Shot Prompting:** A list of bullet points:
 - Provides examples demonstrating the pattern
 - Guides the model through similar cases
 - More consistent and accurate results
 - Uses more tokens/computation

```
# Few-Shot approach
print("FEW-SHOT APPROACH:")
print("." * 70)
print("✓ Multiple examples provided (typically 2-5)")
print("✓ Demonstrates the expected pattern")
print("✓ Guides the model through similar examples")
print("✓ Produces more consistent and accurate results")
print("✓ Requires more tokens/computation")
print()
result_few = classify_sentiment_few_shot("This is amazing!")
print(f'Result: {result_few}')
print()

# Key Differences
print("KEY DIFFERENCES:")
print("." * 70)
comparison_data = [
    ("Examples", "None", "2-5+ examples"),
    ("Consistency", "Lower", "Higher"),
    ("Accuracy", "Lower", "Higher"),
    ("Setup Time", "Faster", "Slower"),
    ("Token Usage", "Lower", "Higher"),
    ("Use case", "Simple tasks", "Complex/nuanced tasks"),
]
for feature, zero, few in comparison_data:
    print(f'{feature:15} | Zero-Shot: {zero:20} | Few-Shot: {few}')


Usage
Best For
Zero-Shot

• No examples required
• Instructive examples
• Direct
• Faster
• Relies on context

Few-Shot

• Provides multiple examples
• Consistent pattern
• Guides the model
• More tokens
• Uses more computation

The file includes an implementation of a few-shot learning model which would structure the input data for the classifier based on the few examples provided.
```