Lab 4: Advanced Prompt Engineering

Zero-shot, One-shot, and Few-shot Techniques, Chain of Thoughts, Prompt

NAME: CH VISHWANATH RAO

BATCH:02

HT NO:2303A51095

Q1. Zero-Shot Prompting (Basic Lab Task)

Task:

Write a Python function that classifies a given text as Spam or Not Spam

Spam using zero-shot prompting.

Steps:

1. Construct a prompt without any examples.

2. Clearly specify the output labels.

3. Display only the predicted label.

Input:

"Congratulations! You have won a free lottery ticket."

Expected Output:

Spam

PROMPT AND CODE:

```
#Write a Python function that classifies a given text as Spam or Not based on the presence of certain keywords like 'win, free, prize, click here
# user input
```

```
def classify_message(msg):
    keys = ['win', 'free', 'prize', 'click here']
    lowered = msg.lower()
    for token in keys:
        if token in lowered:
            return "Spam"
    return "Not Spam"

user_input = input("Please paste the text to analyze: ")
classification = classify_message(user_input)
print("Classification:", classification)
```

OUTPUT:

```
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING> & C:/Python314/python.exe "c:/Users/chvis/OneDrive/Desktop/AI ASSIST CODING/lab 3.
Please paste the text to analyze: hello, click the link below to claim the prizes
Classification: Spam
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING>
```

Q2. One-Shot Prompting (Emotion detection)

Task:

Write a Python program that detects the emotion of a sentence

using one-shot prompting.

Emotions: [‘happy', 'sad', 'angry', 'excited', 'nervous', ’neutral’]

Steps:

1. Provide one labelled example inside the prompt.

2. Take a sentence as input.

3. Print the predicted emotion

PROMPT:

```
'''
s=ohh!
display excietment
user input
'''
```

Code:

```
def classify_emotion(s):
    excitement_words = ['excited', 'excietment', 'thrilled', 'elated', 'overjoyed', 'ecstatic', 'euphoric', 'joyful']
    happiness_words = ['happy', 'joyful', 'content', 'pleased', 'cheerful', 'delighted', 'glad', 'satisfied']
    angry_words = ['angry', 'furious', 'irate', 'livid', 'outraged', 'resentful', 'annoyed', 'frustrated']
    sad_words = ['sad', 'unhappy', 'sorrowful', 'dejected', 'downcast', 'mournful', 'gloomy', 'disheartened']
    s_lower = s.lower()
    if any(word in s_lower for word in excitement_words):
        return "excietment"
    elif any(word in s_lower for word in happiness_words):
        return "happiness"
    elif any(word in s_lower for word in angry_words):
        return "angry"
    elif any(word in s_lower for word in sad_words):
        return "sad"
    else:
        return "unknown emotion"
user_input = input("Enter a sentence expressing your emotion: ")
emotion = classify_emotion(user_input)
print(f"The emotion expressed is: {emotion}")
```

Output:

```
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING> & C:/Python314/python.exe "c:/Users/chvis/OneDrive/Desktop/AI ASSIST C
Enter a sentence expressing your emotion: iam sad today
The emotion expressed is: sad
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING> & C:/Python314/python.exe "c:/Users/chvis/OneDrive/Desktop/AI ASSIST C
Enter a sentence expressing your emotion: iam happy today
The emotion expressed is: happiness
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING>
```

Q3. Few-Shot Prompting (Student Grading Based on Marks)

Task:

Write a Python program that predicts a student's grade based on

marks using few-shot prompting.

Grades:

['A', 'B', 'C', 'D', 'F']

Grading Criteria (to be inferred from examples):

- 90–100 → A
- 80–89 → B
- 70–79 → C
- 60–69 → D
- Below 60 → F

CODE:

```
90-100=A
80-89=B
70-79=C
60-69=D
below 60=F
write a python program that predicts a student's grade based on marks obtained and only accept positive integer values from the user otherwise d
try:
    student_marks = int(input("Enter the marks (0-100): "))
    if student_marks < 0 or student_marks > 100:
        print("Invalid input. Please enter a positive integer between 0 and 100.")
    else:
        if 90 <= student_marks <= 100:
            student_grade = 'A'
        elif 80 <= student_marks <= 89:
            student_grade = 'B'
        elif 70 <= student_marks <= 79:
            student_grade = 'C'
        elif 60 <= student_marks <= 69:
            studentstudent_grade = 'D'
        else:
            student_grade = 'F'
        print(f"The grade for {student_marks} is: {student_grade}")
except ValueError:
    print("Invalid input. Please enter a valid integer.")
```

Output:

```
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING> & C:/Python314/python.exe "c:/Users/chvis/OneDrive/Desktop/AI ASSIST CODING/lab 3.py/samp
Enter the marks (0-100): 100
The grade for 100 is: A
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING> & C:/Python314/python.exe "c:/Users/chvis/OneDrive/Desktop/AI ASSIST CODING/lab 3.py/samp
Enter the marks (0-100): 70
The grade for 70 is: C
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING> & C:/Python314/python.exe "c:/Users/chvis/OneDrive/Desktop/AI ASSIST CODING/lab 3.py/samp
Enter the marks (0-100): 35
The grade for 35 is: F
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING>
```

Q4. Multi-Shot Prompting (Indian Zodiac Sign Prediction using Month Name)

Task:

Write a Python program that predicts a person's Indian Zodiac sign (Rashi) based on the month of birth (month name) using multi-shot prompting.

Indian Zodiac Order (Simplified Month-Based Model): The Indian Zodiac cycle starts in March with Mesha and follows this order:

March → Mesha

April → Vrishabha

May → Mithuna

June → Karka

July → Simha

August → Kanya

September → Tula

October → Vrischika

November → Dhanu

December → Makara

January → Kumbha

February → Meena

PROMPT:

```
'''march=mesha
april=vrishabha
may=mithuna
june=karka
july=simha
august=kanya
september=tula
october=vrischika
november=dhanus
december=makara
january=kumbha
february=meena
write a python code to accept month from the user and display the corresponding zodiac sign and only accept valid month names otherwise display inva
```

Code:

```
def display(sign):
    print("The zodiac sign is:", sign)
month = input("Enter the month: ").strip().lower()
if month == "march":
    display("Mesha")
elif month == "april":
    display("Vrishabha")
elif month == "may":
    display("Mithuna")
elif month == "june":
    display("Karka")
elif month == "july":
    display("Simha")
elif month == "august":
    display("Kanya")
elif month == "september":
    display("Tula")
elif month == "october":
    display("Vrischika")
elif month == "november":
    display("Dhanu")
elif month == "december":
    display("Makara")
elif month == "january":
    display("Kumbha")
elif month == "february":
    display("Meena")
else:
    print("Invalid month entered.")
```

Output:

```
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING> & C:/Python314/python.exe "c:/Users/chvis/OneDrive/
Enter the month: december
The zodiac sign is: Makara
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING>
```

Q5. Result Analysis Based on Marks

Task: Write a Python program that determines whether a student Passes or fails based on marks using Chain-of-Thought (CoT) prompting.

Result Categories:

['Pass', 'Fail']

Prompt:

```
'''
take marks as an input from the user
if marks is greater than or equal to 40 then print pass
if marks is less than 40 then print fail'''
```

Code:

```
marks = int(input("Enter marks: "))
if marks >= 40:
    print("Pass")
else:
    print("Fail")
```

Output:

```
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING> & C:/Python314/python.exe "c:/Users/chvis/OneDrive/Desktop/AI ASSIST CO
Enter marks: 41
Pass
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING> & C:/Python314/python.exe "c:/Users/chvis/OneDrive/Desktop/AI ASSIST CO
Enter marks: 29
Fail
```

Q6 Voting Eligibility Check (Chain-of-Thought Prompting)

Task: Write a Python program that determines whether a person is eligible to vote using Chain-of-Thought (CoT) prompting.

prompt:

```
'''
take age from the user
if age is equal or greater than 18
then print eligible to vote
if age is less than 18
print not eligible to vote'''
```

Code:

```
age = int(input("Enter your age: "))
if age >= 18:
    print("Eligible to vote")
else:
    print("Not eligible to vote")
```

Outputs:

```
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING> & C:/Python314/python.exe "c:/Users/chvis/OneDrive/Desktop/AI ASSIST CODING/lab 3.py/
Enter your age: 18
Eligible to vote
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING> & C:/Python314/python.exe "c:/Users/chvis/OneDrive/Desktop/AI ASSIST CODING/lab 3.py/
Enter your age: 16
Not eligible to vote
```

Q7 Prompt Chaining (String Processing – Palindrome Names)

Task: Write a Python program that uses the prompt chaining technique to identify palindrome names from a list of student names.

```
"""
take lidt of names form the user
if student names raa palindrome
then print those names in the form of list

"""
def is_palindrome(name):
    return name == name[::-1]
students = input("Enter student names separated by commas: ").split(",")
palindrome_students = [name.strip() for name in students if is_palindrome(name.strip())]
print("Palindrome names:", palindrome_students)


```

Output:

```
Enter student names separated by commas: Anand,aneesh,bob,bharath
Palindrome names: ['bob']
```

Q8 Prompt Chaining (String Processing – Word Length Analysis)

Task: Write a Python program that uses **prompt chaining** to analyse a list of words. In the first prompt, generate a list of words. In the second prompt, traverse the list and calculate the length of each word. In the third prompt, use the output of the previous step to determine whether each word is **Short** (length less than 5) or **Long** (length greater than or equal to 5), and display the result for each word

Prompt:

```
take list of words from the user
if the length of the individual word is greater than 5 then the the word is longs word
else the word is short
print the longs words and short words in the form of list'''
```

Code:

```
words = input("Enter the list of words: ").split()
long_words = []
short_words = []
for word in words:
    if len(word) > 5:
        long_words.append(word)
    else:
        short_words.append(word)

print("Long words:", long_words)
print("Short words:", short_words)
```

Outputs:


Enter the list of words: lion, cheetah, tiger
Long words: ['cheetah,']
Short words: ['lion,', 'tiger']
PS C:\Users\chvis\OneDrive\Desktop\AI ASSIST CODING>