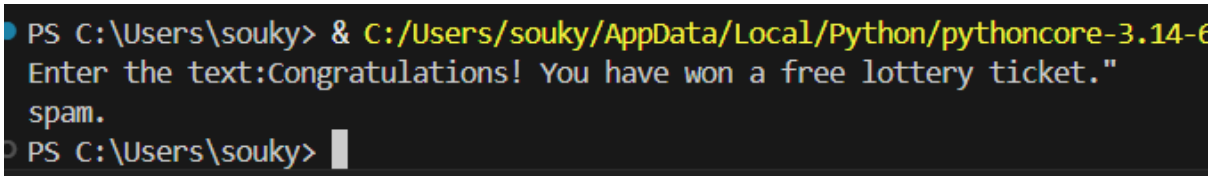
Lab Assignment -4

N. Soukya

Batch-24

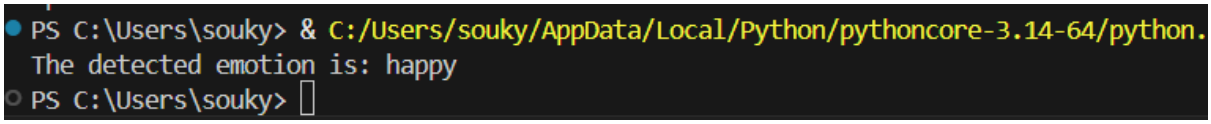
2303A51401

Q1. Zero-Shot Prompting (Basic Lab Task)
Task:
Write a Python function that classifies a given text as Spam or Not 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



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 labeled example inside the prompt.
2. Take a sentence as input.
3. Print the predicted emotion



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

Users > souky > OneDrive > Documents > AI Lab > grde.py > determine_grade

```python
#write a python code displaying the grade based on the marks obtained.ex:marks=92, grade=A
#marks=85 -grade=B, marks=76 -grade=C
def determine_grade(marks):
    if marks >= 90:
        return 'A'
    elif marks >= 80:
        return 'B'
    elif marks >= 70:
        return 'C'
    elif marks >= 60:
        return 'D'
    else:
        return 'F'
try:
    user_input = int(input("Enter the marks obtained: "))
    grade = determine_grade(user_input)
    print(f"The grade based on the marks {user_input} is: {grade}")
except ValueError:
    print("Please enter a valid integer for marks.")
except Exception as e:
    print(f"An error occured: {e}")
```

```
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-6
Enter the marks obtained: 0
The grade based on the marks 0 is: F
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-6
Enter the marks obtained: 45
The grade based on the marks 45 is: F
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-6
Enter the marks obtained: 96
The grade based on the marks 96 is: A
PS C:\Users\souky>
```

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

Users > souky > OneDrive > Documents > AI Lab > zodiac.py > ...

```python
#generate a python code that displays a person's Indian Zodiac sign(Rashi) based on the month of birth
'''ex:March → Mesha
April → Vrishabha
May → Mithuna
June → Karka
July → Simha
August → Kanya'''
def get_indian_zodiac_sign(month):
    zodiac_signs = {
        "january": "Makara",
        "february":"Kumbha",
        "march": " Mesha",
        "april": "Vrishabha",
        "may": "Mithuna",
        "june": "Karka",
        "july": "Simha",
        "august": "Kanya",
        "september":"Tula",
        "october": "Vrischika",
        "november": "Dhanu",
        "december": "Makara"
    }
    month_lower = month.lower()
    return zodiac_signs.get(month_lower, "Invalid month name")
try:
    user_input = input("Enter the month of birth:")
    zodiac_sign = get_indian_zodiac_sign(user_input)
    print("The Indian zodiac sign for the month {user_input} is:  {zodiac_sign}")
except Exception as e:
    print(f"An error occured: {e}")
```

```
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-64/python.ex
Enter the month of birth:august
The Indian zodiac sign for the month {user_input} is:  {zodiac_sign}
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-64/python.ex
Enter the month of birth:february
The Indian zodiac sign for the month {user_input} is:  {zodiac_sign}
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-64/python.ex
Enter the month of birth:january
The Indian zodiac sign for the month {user_input} is:  {zodiac_sign}
PS C:\Users\souky>
0 ⚠ 0
```

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']

C: > Users > souky > OneDrive > Documents > AI Lab > ass 4 Q5.py > ...

```python
'''
read marks of students from range 0-100
check if marks are greater than 40
if yes print pass otherwise print fail
'''
def check_pass_fail(marks):
    if 0 <= marks <= 100:
        if marks > 40:
            return "Pass"
        else:
            return "Fail"
    else:
        return "Invalid marks"
marks = int(input("Enter marks (0-100): "))
print(check_pass_fail(marks))
```

```
PS C:\Users\souky> & C:/Users/souky/AppDa
Enter marks (0-100): 60
Pass
PS C:\Users\souky> & C:/Users/souky/AppDa
Enter marks (0-100): 35
Fail
PS C:\Users\souky> & C:/Users/souky/AppDa
Enter marks (0-100): 0
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.

> Users > souky > OneDrive > Documents > AI Lab > cot age.py > ...

```python
'''
read the age of person from range 1-100
check if age is greater than or equal to 18
if yes,print eligible to vote
otherwise print not eligible
'''

age = int(input("Enter the age of the person (1-100): "))
if age >= 18:
    print("Eligible to vote")
else:
    print("Not eligible to vote")
```

```
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncor
Enter the age of the person (1-100): 16
Not eligible to vote
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncor
Enter the age of the person (1-100): 20
Eligible to vote
PS C:\Users\souky>
```

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.

C: > Users > souky > OneDrive > Documents > AI Lab > palindromic names.py > ...

```python
'''
read student names from user and store in a list of student names
if name is palindrome store it in a list
handle case sensitivity
handle invalid inputs
display list of palindrome names
'''
def is_palindrome(name):
    name = name.strip()
    if not name.isalpha():
        return False
    name_lower = name.lower()
    return name_lower == name_lower[::-1]
palindrome_names = []
while True:
    name = input("Enter a student name (or 'quit' to stop): ")
    if name.lower() == 'quit':
        break
    if is_palindrome(name):
        palindrome_names.append(name)
print("Palindrome names:", palindrome_names)
```

```
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pyt
Enter a student name (or 'quit' to stop): soukya
Enter a student name (or 'quit' to stop): madam
Enter a student name (or 'quit' to stop): harini
Enter a student name (or 'quit' to stop): navya
Enter a student name (or 'quit' to stop): mom
Enter a student name (or 'quit' to stop): quit
Palindrome names: ['madam', 'mom']
PS C:\Users\souky>
```

Q8 Prompt Chaining (String Processing – Word Length Analysis)

Task: Write a Python program that uses prompt chaining to analyze 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