# AI ASSISTED CODING

# ASSIGNMENT-4.1

**Name: Hari Priya**

**H.T.No:2303A51104**

**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:

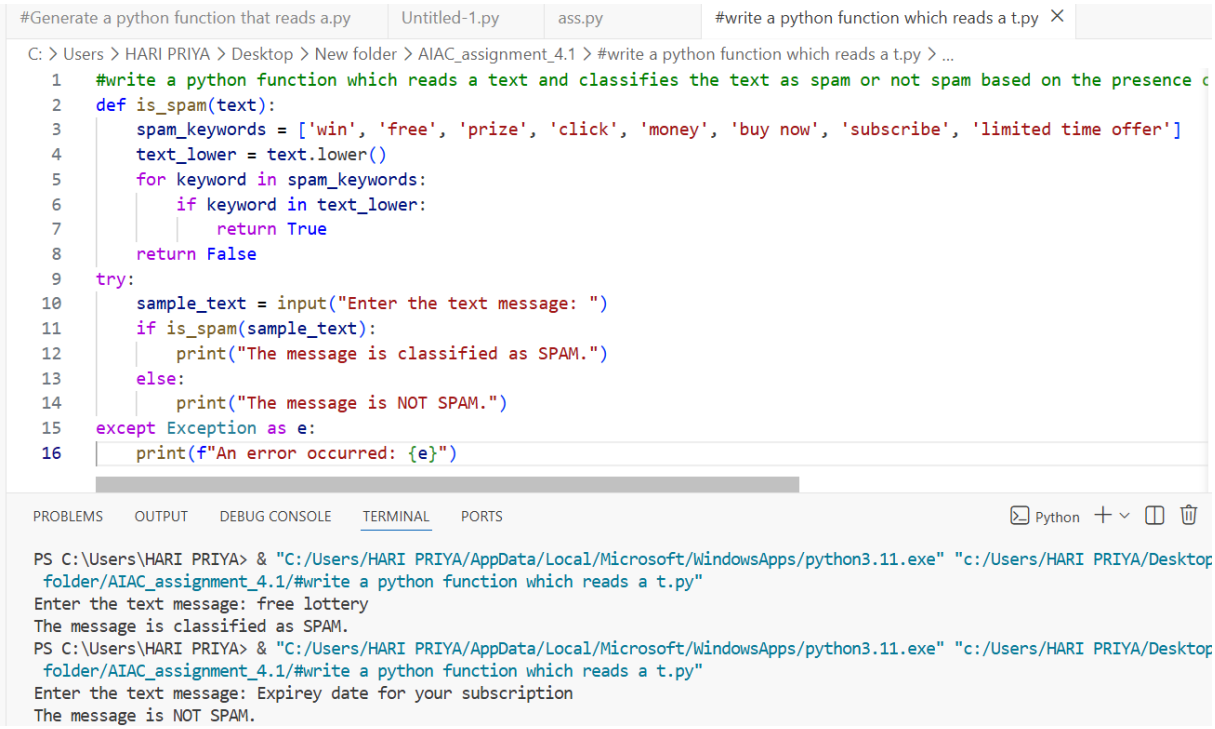
1. Construct a prompt without any examples.
2. Clearly specify the output labels.
3. Display only the predicted label.

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

C: > Users > HARI PRIYA > Desktop > New folder > AIAC_assignment_4.1 > Untitled-1.py > ...

```
'''
s=Ohh! what a suprise
display excitement
'''
def classify_emotions(text):
    excitement_keywords = ['ohh', 'wow', 'amazing', 'fantastic', 'incredible', 'surprise', 'exciting']
    happiness_keywords = ['happy', 'joy', 'pleased', 'delighted', 'content', 'cheerful']
    sad_keywords = ['sad', 'unhappy', 'sorrow', 'dejected', 'downcast', 'mournful']
    anger_keywords = ['angry', 'furious', 'irate', 'livid', 'outraged', 'mad']
    text_lower = text.lower()
    if any(word in text_lower for word in excitement_keywords):
        return "excitement"
    elif any(word in text_lower for word in happiness_keywords):
        return "happiness"
    elif any(word in text_lower for word in sad_keywords):
        return "sad"
    elif any(word in text_lower for word in anger_keywords):
        return "anger"
    else:
        return "neutral"
try:
    user_input = input("Enter the text: ")
    emotion = classify_emotions(user_input)
    print(emotion)
except Exception as e:
    print("An error occurred:", e)
```

```
Enter the text: she is good
neutral
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop/New
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop/New
 folder/AIAC_assignment_4.1/Untitled-1.py"
Enter the text: ohh!
excitement
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop/New
 folder/AIAC_assignment_4.1/Untitled-1.py"
Enter the text: I am so happy
happiness
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop/New
 folder/AIAC_assignment_4.1/Untitled-1.py"
Enter the text: so sorrow
sad
```

**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

C: > Users > HARI PRIYA > Desktop > New folder > AIAC_assignment_4.1 > Untitled-2.py > ...

```
'''
m=85
diaplay B
m=55
display F
m=78
display C
m=61
display D
'''
def display_grade(marks):
    if marks < 0 or marks > 100:
        return "Invalid marks"
    elif marks >= 90:
        return "A"
    elif marks >= 80:
        return "B"
    elif marks >= 70:
        return "C"
    elif marks >= 60:
        return "D"
    else:
        return "F"
try:
    marks = int(input("Enter the student marks: "))
    grade = display_grade(marks)
    print(grade)
except ValueError:
    print("Invalid marks")
```

```
Enter the student marks: 91
A
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop/New
 folder/AIAC_assignment_4.1/Untitled-2.py"
Enter the student marks: 55
F
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop/New
 folder/AIAC_assignment_4.1/Untitled-2.py"
Enter the student marks: 89
B
Enter the student marks: 89
B
B
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop/New
 folder/AIAC_assignment_4.1/Untitled-2.py"
Enter the student marks: -2
Invalid marks
```

**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

C: > Users > HARI PRIYA > Desktop > New folder > AIAC_assignment_4.1 > Untitled-3.py > ...

```
'''
March
display Mesha
April
display Vrishabha
May
display Mithuna
June
display Karka
July
display Simha
August
display Kanya
September
display Tula
October
display Vrischika
November
display Dhanu
December
display Makara
January
display Kumbha
February
display Meena
'''
def get_zodiac_sign(month):
    month = month.lower()
    zodiac_signs = {
        'march': 'Mesha',
        'april': 'Vrishabha',
        'may': 'Mithuna',
        'june': 'Karka',
        'july': 'Simha',
        'august': 'Kanya',
        'september': 'Tula',
        'october': 'Vrischika',
        'november': 'Dhanu',
        'december': 'Makara',
        'january': 'Kumbha',
        'february': 'Meena'
    }
    return zodiac_signs.get(month, 'Invalid month')
try:
    user_input = input("Enter the month: " )
    zodiac_sign = get_zodiac_sign(user_input)
    print(zodiac_sign)
except Exception as e:
    print("An error occurred:", e)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python

```
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop
 folder/AIAC_assignment_4.1/Untitled-3.py"
Enter the month: may
Mithuna
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop
 folder/AIAC_assignment_4.1/Untitled-3.py"
Enter the month: september
Tula
```

**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 > HARI PRIYA > Desktop > New folder > AIAC_assignment_4.1 > Untitled-4.py > ...

```
'''
Read student marks
validate that the marks are between 0 and 100
display appropriate message if invalid
if marks are greater than 40 display pass else display fail
'''
def check_marks(marks):
    if marks < 0 or marks > 100:
        return "Invalid marks"
    elif marks > 40:
        return "Pass"
    else:
        return "Fail"
try:
    student_marks = int(input("Enter student marks: "))
    result = check_marks(student_marks)
    print(result)
except ValueError:
    print("Invalid marks")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python

```
Enter student marks: 35
Fail
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop/
 folder/AIAC_assignment_4.1/Untitled-4.py"
Enter student marks: 70
Pass
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop/
 folder/AIAC_assignment_4.1/Untitled-4.py"
Enter student marks: 111
Invalid marks
```

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

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

C: > Users > HARI PRIYA > Desktop > New folder > AIAC_assignment_4.1 > Untitled-5.py > ...

```python
'''
read the age of a person
display eligibility to vote if age is greater than or equal to 18
'''
def check_voting_eligibility(age):
    if age < 0:
        return "Invalid age"
    elif age >= 18:
        return "Eligible to vote"
    else:
        return "Not eligible to vote"
try:
    age = int(input("Enter your age of the person: "))
    eligibility = check_voting_eligibility(age)
    print(eligibility)
except ValueError:
    print("Invalid age")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                                                    Python

PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop
 folder/AIAC_assignment_4.1/Untitled-5.py"
Enter your age of the person: 18
Eligible to vote
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop
 folder/AIAC_assignment_4.1/Untitled-5.py"
Enter your age of the person: 10
Not eligible to vote
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop
 folder/AIAC_assignment_4.1/Untitled-5.py"
Enter your age of the person: -4
Invalid age
```

### 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 > HARI PRIYA > Desktop > New folder > AIAC_assignment_4.1 > Untitled-6.py > ...

```python
'''
Generate a list of names and store it in a variable
Traverse through each name and make a new list of names reversing the characters in each name
compare the original list with the new list and display names which are same in both lists(palindrome names)
'''
names=['anna', 'bob','vasav', 'ram', 'ramar', 'Hannah']
reversed_names = [name[::-1] for name in names]
palindrome_names = [name for name in names if name == name[::-1]]
print("Palindrome names:",palindrome_names)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                                                    Python

PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop/New folder/AIAC_as
ment_4.1/Untitled-6.py"
Palindrome names: ['anna', 'bob', 'vasav', 'ramar']
```

### 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

C: > Users > HARI PRIYA > Desktop > New folder > AIAC_assignment_4.1 > Untitled-7.py > ...

```python
'''
Generate a list of names
Traverse through each name and caluculate the length of each name and store in variable le_names
if le_names is less than 5 display short names
if le_names is greater than or equal to 5 display long names
'''
names=['Asha', 'Ravi', 'Kavitha', 'Suresh', 'Priya', 'Vikram']
for name in names:
    le_names = len(name)
    if le_names < 5:
        print(f"{name} is a short name")
    else:
        print(f"{name} is a long name")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python

```
PS C:\Users\HARI PRIYA> & "C:/Users/HARI PRIYA/AppData/Local/Microsoft/WindowsApps/python3.11.exe" "c:/Users/HARI PRIYA/Desktop/New folder/AIAC_
ment_4.1/Untitled-7.py"
Asha is a short name
Ravi is a short name
Kavitha is a long name
Suresh is a long name
Priya is a long name
Vikram is a long name
```