

AI ASSISTANT CODING

ASSIGNMENT-4

Name:B.Bhargava chary

Batch:24

Ht.No:2303A51747

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:

"Hey! you won a cash prize of 20,000 rupees"

Expected Output:

The text is spam.

A screenshot of a code editor window titled "C:\Users\BHARGAV> #Quetion 1.py". The code is as follows:

```
 1 #Quetion 1.py
 2 #write a python program to find whether a given text is spam or not
 3 def is_spam(text):
 4     spam_keywords = ["free", "win", "winner", "cash", "prize", "urgent", "click here"]
 5     text_lower = text.lower()
 6     for keyword in spam_keywords:
 7         if keyword in text_lower:
 8             return True
 9     return False
10 try:
11     text = input("Enter the text: ")
12     if is_spam(text):
13         print("The text is spam.")
14     else:
15         print("The text is not spam.")
16 except Exception as e:
17     print(f"An error occurred: {e}")
```

The terminal below shows the execution of the script:

```
PS C:\Users\BHARGAV> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Quetion 1.py"
Enter the text: Hey! you won a cash prize of 20,000 rupees
The text is spam.
PS C:\Users\BHARGAV> []
```

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

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

- File Explorer:** Shows files like Q30_2303A51747_Batch_24.ipynb, AI Assistance coding.py, #Question 1.py, and #Question 2.py.
- Code Editor:** Displays the Python code for emotion detection:

```
1 #Question 2
2 #write a python program that detects emotion of a person
3 """
4 sentence: I am very joyful today
5 output: happy
6 """
7 def detect_emotion(sentence):
8     text_lower = sentence.lower()
9     if "happy" in text_lower or "joyful" in text_lower or "excited" in text_lower:
10         return "happy"
11     elif "sad" in text_lower or "unhappy" in text_lower or "depressed" in text_lower:
12         return "sad"
13     elif "angry" in text_lower or "mad" in text_lower or "furious" in text_lower:
14         return "angry"
15     elif "fearful" in text_lower or "scared" in text_lower or "anxious" in text_lower:
16         return "fearful"
17     else:
18         return "neutral"
19 sentence = input("Enter a sentence: ")
20 emotion = detect_emotion(sentence)
21 print(f"The detected emotion is: {emotion}")
```
- Terminal:** Shows the command line output:

```
PS C:\Users\BHARGAV> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Question 2.py"
Enter a sentence: I am mad at todays incidents
The detected emotion is: angry
PS C:\Users\BHARGAV>
```
- Status Bar:** Shows the current file is Python, line 19, column 1, and other status information.

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.

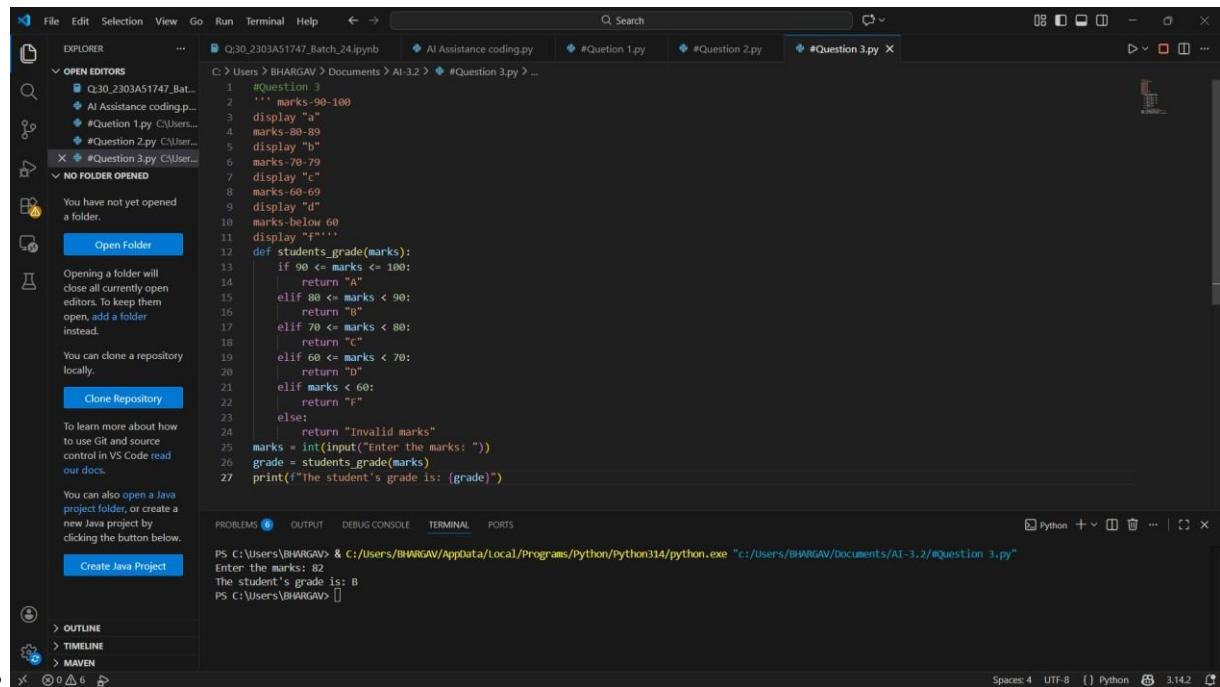
Grades:

['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



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

- File Explorer:** Shows a folder named "Q30_2303A51747_Batch_24.ipynb" and several Python files: "AI Assistance coding.py", "#Question 1.py", "#Question 2.py", and "#Question 3.py".
- Code Editor:** Displays a Python script for calculating student grades based on marks.
- Terminal:** Shows the command line output of running the script, asking for marks and printing the grade as "B".
- Bottom Status Bar:** Shows "Spaces: 4", "UTF-8", "Python", and "3.14.2".

```

1 #Question 3
2   ''' marks-90-100
3   display "A"
4   marks-80-89
5   display "B"
6   marks-70-79
7   display "C"
8   marks-60-69
9   display "D"
10  marks-below 60
11  display "F"
12
13 def students_grade(marks):
14     if 90 <= marks <= 100:
15         return "A"
16     elif 80 <= marks < 90:
17         return "B"
18     elif 70 <= marks < 80:
19         return "C"
20     elif 60 <= marks < 70:
21         return "D"
22     elif marks < 60:
23         return "F"
24     else:
25         return "Invalid marks"
26 marks = int(input("Enter the marks: "))
27 grade = students_grade(marks)
28 print("The student's grade is: (grade)")

```

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

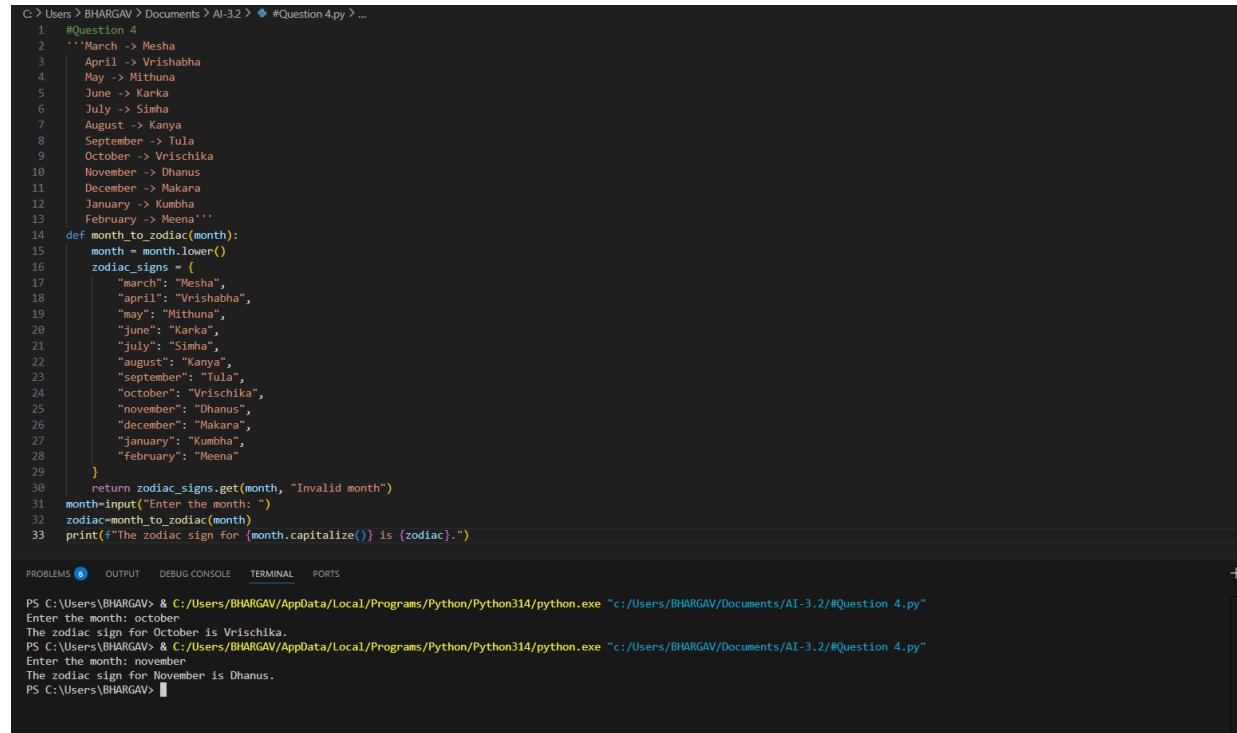
October → Vrischika

November → Dhanu

December → Makara

January → Kumbha

February → Meena



```
C:\Users\BHARGAV> Documents > AI-3.2 > #Question 4.py > ...
1  #Question 4
2  '''March -> Mesha
3  April -> Vrishabha
4  May -> Mithuna
5  June -> Karka
6  July -> Simha
7  August -> Kanya
8  September -> Tula
9  October -> Vrischika
10 November -> Dhanus
11 December -> Makara
12 January -> Kumbha
13 February -> Meena'''
14 def month_to_zodiac(month):
15     month = month.lower()
16     zodiac_signs = {
17         "march": "Mesha",
18         "april": "Vrishabha",
19         "may": "Mithuna",
20         "june": "Karka",
21         "july": "Simha",
22         "august": "Kanya",
23         "september": "Tula",
24         "october": "Vrischika",
25         "november": "Dhanus",
26         "december": "Makara",
27         "january": "Kumbha",
28         "february": "Meena"
29     }
30     return zodiac_signs.get(month, "Invalid month")
31 month=input("Enter the month: ")
32 zodiac=month_to_zodiac(month)
33 print(f"The zodiac sign for {month.capitalize()} is {zodiac}.")
```

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\BHARGAV> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Question 4.py"
Enter the month: october
The zodiac sign for October is Vrischika.
PS C:\Users\BHARGAV> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Question 4.py"
Enter the month: november
The zodiac sign for November is Dhanus.
PS C:\Users\BHARGAV>
```

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']

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

- File Explorer:** Shows files like Q30_2303A51747_Batch_24.ipynb, AI Assistance coding.py, #Question 1.py, #Question 2.py, #Question 3.py, #Question 4.py, and #Question 5.py.
- Code Editor:** Displays the following Python code for Question 5:

```
1 #Question 5
2 '''read marks of students from range 0-100
3     check if marks are greater than 40
4     if yes display "pass"
5     if no display "fail"...
6 def check_pass_fail(marks):
7     if 0 <= marks <= 100:
8         if marks > 40:
9             return "pass"
10        else:
11            return "fail"
12    else:
13        return "Invalid marks. Please enter a value between 0 and 100."
14 marks=int(input("Enter the marks of the student: "))
15 result=check_pass_fail(marks)
16 print(f"The student has {result}.")
```

Terminal: Shows the command line output:

```
PS C:\Users\BHARGAV> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Question 5.py"
Enter the marks of the student: 56
The student has pass.
PS C:\Users\BHARGAV>
```

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.

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

- File Explorer:** Shows files like Q30_2303A51747_Batch_24.ipynb, AI Assistance coding.py, #Question 1.py, #Question 2.py, #Question 3.py, #Question 4.py, and #Question 6.py.
- Code Editor:** Displays the following Python code for Question 6:

```
1 #Question 6
2 '''read the age of person from range 1-100
3     check if age is greater than or equal to 18
4     if yes print "eligible to vote"
5     else print "not eligible to vote"...
6 age=int(input("Enter the age of the person: "))
7 if age>=18:
8     print("eligible to vote")
9 else:
10    print("not eligible to vote")
```

Terminal: Shows the command line output:

```
PS C:\Users\BHARGAV> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Question 6.py"
Enter the age of the person: 21
eligible to vote
PS C:\Users\BHARGAV> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Question 6.py"
Enter the age of the person: 14
not eligible to vote
PS C:\Users\BHARGAV>
```

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 > BHARGAV > Documents > AI-3.2 > #Quetion 7.py > ...
1  #Quetion 7
2  """ read student names from user
3  if name is palindrome store it in a list
4  handle case sensitivity
5  handle invalid inputs
6  display list of palindromic names"""
7  def is_palindrome(name):
8      name = name.strip()
9      if not name.isalpha():
10         return False
11      name_lower = name.lower()
12      return name_lower == name_lower[::-1]
13  palindromic_names = []
14  while True:
15      name = input("Enter a student name (or type 'exit' to finish): ")
16      if name.lower() == 'exit':
17          break
18      if is_palindrome(name):
19          palindromic_names.append(name)
20  print("Palindromic names:", palindromic_names)

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS Python + ▾
```

```
n/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Quetion 7.py"
Enter a student name (or type 'exit' to finish): vikramaditya
Enter a student name (or type 'exit' to finish): bindu
Enter a student name (or type 'exit' to finish): eve
Enter a student name (or type 'exit' to finish): gani
Enter a student name (or type 'exit' to finish): chanti
Enter a student name (or type 'exit' to finish): 1747
Enter a student name (or type 'exit' to finish): exit
Palindromic names: ['eve']
PS C:\Users\BHARGAV>
```

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 > BHARGAV > Documents > AI-3.2 > #Quetion 8.py > ...
1  #Quetion 8
2  '''read words from user
3  count the length of each and store it in a variable
4  if variable <5 display as short
5  otherwise display as long
6  display the result of each word'''
7  def classify_word_length(word):
8      length = len(word)
9      if length < 5:
10         return "short"
11     else:
12         return "long"
13 words = input("Enter words separated by spaces: ").split()
14 for word in words:
15     classification = classify_word_length(word)
16     print(f"The word '{word}' is {classification}.")


```

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Python +

```
PS C:\Users\BHARGAV> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Quetion 8.py"
Enter words separated by spaces: mango
The word 'mango' is long.
PS C:\Users\BHARGAV> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/#Quetion 8.py"
Enter words separated by spaces: cat
The word 'cat' is short.
PS C:\Users\BHARGAV>
```