

# ASSIGNMENT - 04

Hall Ticket Number : 2303a51027

Name : Aravind Reddy

Batch NO : 01

## Q1. Zero-Shot Prompting (Basic Lab Task)

The screenshot shows the VS Code interface with the file 'Assignment - 04.py' open in the editor. The code defines a function 'classify\_text' that checks if any keyword from a list is present in the input text. It then prints the classification ('Spam' or 'Not Spam'). The terminal window shows the execution of the script and its output for two different inputs.

```
1 # Question 1:
2 # Write a Python function that classifies a given text as Spam or Not
3 # Spam using zero-shot prompting.
4
5 def classify_text(text):
6     # This is a placeholder for zero-shot classification logic.
7     # In a real-world scenario, you would integrate with a pre-trained model or API.
8     spam_keywords = ['win', 'free', 'prize', 'click', 'buy now']
9     text_lower = text.lower()
10
11     if any(keyword in text_lower for keyword in spam_keywords):
12         return "Spam"
13     else:
14         return "Not Spam"
15
16 # Test the function
17 input_text = input('Enter the text to classify: ')
18 result = classify_text(input_text)
19 print(f"The text is classified as: {result}")
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
```

```
/usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment - 04.py"
Enter the text to classify: Congratulations! You have won a free lottery ticket!
The text is classified as: Spam
(base) [1]: /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment - 04.py"
Enter the text to classify: Your selected for the next Round.
The text is classified as: Not Spam
% (base) > AI Assisted Coding [1]
```

## Q2. One-Shot Prompting (Emotion detection)

The screenshot shows the VS Code interface with the file 'Assignment - 04.py' open in the editor. The code defines a function 'detect\_emotion' that uses several lists of keywords to detect the emotion in a sentence. It then prints the detected emotion. The terminal window shows the execution of the script and its output for two different sentences.

```
24
25 # Question 2:
26
27 # Write a Python program that detects the emotion of a sentence using one-shot prompting.
28 # Emotions: ['happy', 'sad', 'angry', 'excited', 'nervous', 'neutral']
29
30 def detect_emotion(sentence):
31     # This is a placeholder for one-shot emotion detection logic.
32     # In a real-world scenario, you would integrate with a pre-trained model or API.
33     happy_keywords = ["happy", "joy", "excited", "delighted"]
34     sad_keywords = ["sad", "depressed", "unhappy", "miserable"]
35     angry_keywords = ["angry", "mad", "annoyed", "irritated"]
36     excited_keywords = ["excited", "thrilled", "eager"]
37     nervous_keywords = ["nervous", "anxious", "worried"]
38
39     sentence_lower = sentence.lower()
40
41     if any(keyword in sentence_lower for keyword in happy_keywords):
42         return "Happy"
43     elif any(keyword in sentence_lower for keyword in sad_keywords):
44         return "Sad"
45     elif any(keyword in sentence_lower for keyword in angry_keywords):
46         return "Angry"
47     elif any(keyword in sentence_lower for keyword in excited_keywords):
48         return "Excited"
49     elif any(keyword in sentence_lower for keyword in nervous_keywords):
50         return "Nervous"
51     else:
52         return "Neutral"
53
54 # Test the function
55 input_sentence = input('Enter a sentence to detect emotion: ')
56 emotion = detect_emotion(input_sentence)
57 print(f"The detected emotion is: {emotion}")
58
```

```
#(base) [1]: /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment - 04.py"
#(base) > AI Assisted Coding [1]
Enter a sentence to detect emotion: I am feeling very happy today.
The detected emotion is: Happy
#(base) [2]: /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment - 04.py"
#(base) > AI Assisted Coding [2]
Enter a sentence to detect emotion: I am feeling very angry.
The detected emotion is: Angry
#(base) [3]: /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment - 04.py"
#(base) > AI Assisted Coding [3]
Enter a sentence to detect emotion: I am eating Apolis and feeling thrilled.
The detected emotion is: Excited
#(base) [4]: /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment - 04.py"
#(base) > AI Assisted Coding [4]
```

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

The screenshot shows a Jupyter Notebook environment with the following details:

- EXPLORER**: Shows a tree view of files and folders, including 'Assignment - 04.py' under 'ASSISTED CODING'.
- SOURCE CONTROL: CHANGES**: Shows a list of changes, such as 'Assignment - 04.py' being added.
- CELLS**: The main workspace contains the following Python code:

```
# Question 3:
# 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
def predict_grade(marks):
    if 90 <= marks <= 100:
        return 'A'
    elif 80 <= marks < 90:
        return 'B'
    elif 70 <= marks < 80:
        return 'C'
    elif 60 <= marks < 70:
        return 'D'
    else:
        return 'F'
# Test the function
student_marks = int(input('Enter the student marks (0-100): '))
predicted_grade = predict_grade(student_marks)
print("The predicted grade is: " + predicted_grade)
```

**PROBLEMS**: Shows no problems.

**OUTPUT**: Shows the execution results of the code:

```
/usr/local/bin/python "/Users/aravindreddy/Desktop/My-Information/College/AT Assisted Coding/Assignments/Assignment - 04.py"
* base = At Assisted Coding "/usr/local/bin/python" "/Users/aravindreddy/Desktop/My-Information/College/AT Assisted Coding/Assignments/Assignment - 04.py"
  Enter the student marks (0-100): 95
  The predicted grade is: A
* base = At Assisted Coding "/usr/local/bin/python" "/Users/aravindreddy/Desktop/My-Information/College/AT Assisted Coding/Assignments/Assignment - 04.py"
  Enter the student marks (0-100): 87
  The predicted grade is: B
* base = At Assisted Coding "/usr/local/bin/python" "/Users/aravindreddy/Desktop/My-Information/College/AT Assisted Coding/Assignments/Assignment - 04.py"
  Enter the student marks (0-100): 68
  The predicted grade is: D
* base = At Assisted Coding "/usr/local/bin/python" "/Users/aravindreddy/Desktop/My-Information/College/AT Assisted Coding/Assignments/Assignment - 04.py"
  Enter the student marks (0-100): 54
  The predicted grade is: F
% (base) = At Assisted Coding
```

**DEBUG CONSOLE**: Shows no output.

**TERMINAL**: Shows no output.

**PORTS**: Shows no output.

**SPILL CHECKER**: Shows no output.

**Python**: Shows the Python logo and version information.

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

The screenshot shows a Python code editor with the following details:

- File Path:** /Users/aravindreddy/Desktop/My-Information/Catalog/AI Assisted Coding/Assignments/Assignment - 04.py
- Code Content:** A Python script for predicting the Indian Zodiac sign (Rashi) based on the month of birth. It uses a dictionary to map months to their corresponding Rashi names.
- Output Terminal:** The terminal shows the script being run and a sample test case where the user inputs 'January' and the output is 'Kumbha'.
- Bottom Status Bar:** Shows file paths, line numbers (Ln 80, Col 1), character count (Spaces: 4), encoding (UTF-8), file type (LF), Python version (3.13.2), Go Live, and a preview icon.

```
EXPLORER          ⌂ Welcome  ⌂ Assignment - 04.py ×
SOURCE CONTROL: CHANGES
A/ AI ASSISTED CODING
  > vscode
    assignments
      ⌂ Assignment - 01.py
      ⌂ Assignment - 03.py
      ⌂ Assignment - 04.py
      ⌂ Assignment - 05.py
  > Programs

Assignments 2  ⌂ Assignment - 04.py ...
  01 # Question 4:
  02
  03 # This is a Python program which predicts a person's Indian Zodiac sign (Rashi) based on the month of birth (month name) using multi-shot prompting.
  04 # Indian Zodiac Order (Simplified Month-Based Model): The Indian
  05 # Zodiac cycle starts in March with Mesha and follows this order:
  06 # March - Mesha
  07 # April - Vrischika
  08 # May - Mithuna
  09 # June - Karka
  10 # July - Simha
  11 # August - Kanya
  12 # September - Tula
  13 # October - Vrischika
  14 # November - Dhanu
  15 # December - Makara
  16 # January - Kumbha
  17 # February - Meena
  18
  19 from calendar import month
  20
  21
  22 def predict_rashi(month):
  23     rashi_dict = {
  24         'march': 'Mesha',
  25         'april': 'Vrischika',
  26         'may': 'Mithuna',
  27         'june': 'Karka',
  28         'july': 'Simha',
  29         'august': 'Kanya',
  30         'september': 'Tula',
  31         'october': 'Vrischika',
  32         'november': 'Dhanu',
  33         'december': 'Makara',
  34         'january': 'Kumbha',
  35         'february': 'Meena'
  36     }
  37
  38     month_lower = month.lower()
  39     return rashi_dict.get(month_lower, "Invalid month name")
  40
  41 # Test the function
  42 birth_month = input("Enter the month of birth: ")
  43 rashi = predict_rashi(birth_month)
  44 print(f"The Indian Zodiac sign (Rashi) is: {rashi}")

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER 1
/usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/Catalog/AI Assisted Coding/Assignments/Assignment - 04.py"
A local Python environment has been created at /Users/aravindreddy/Desktop/My-Information/Catalog/AI Assisted Coding/Assignments/Assignment - 04.py
Enter the month of birth: January
The Indian Zodiac sign (Rashi) is: Kumbha
% (base) ⌂ AI Assisted Coding 1

Python 3.13.2 64 Go Live ... x
Ln 80, Col 1 Spaces: 4 UTF-8 LF ⌂ Python 3.13.2 64 Go Live ⌂ Preview
```

## **Q5. Result Analysis Based on Marks**

A screenshot of a code editor interface, likely VS Code, showing a Python file named 'Assignment - 04.py'. The code is a script to calculate grades based on marks. It includes comments explaining steps and logic. The terminal below shows the script running and outputting 'Pass' for a mark of 28.

```
139 ...
140 step 1: Read input from the user and store it in a variable marks
141 step 2: The marks variable value should be in between 0 - 100
142 step 3: Based on the marks value, display the grade using the following criteria:
143 if marks >= 40 and marks <= 100 display "Pass"
144 if marks < 40 and marks >= 0 display "Fail"
145 ...
146 # marks = int(input("Enter the marks (0-100): "))
147 # if 0 <= marks <= 100:
148 #     if marks >= 40:
149 #         print("Pass")
150 #     else:
151 #         print("Fail")
152 # else:
153 #     print("Invalid marks. Please enter a value between 0 and 100.")
154 #
155 # print("Invalid marks. Please enter a value between 0 and 100.")
156 #
157
```

```
/usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/Collage/AI Assisted Coding/Assignments/Assignment - 04.py"
Assignment - 04.py
Enter the marks (0-100): 28
# (base) = AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/Collage/AI Assisted Coding/Assignments/Assignment - 04.py"
# (base) = AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/Collage/AI Assisted Coding/Assignments/Assignment - 04.py"
# (base) = AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/Collage/AI Assisted Coding/Assignments/Assignment - 04.py"
# (base) = AI Assisted Coding
```

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

A screenshot of a code editor interface, likely VS Code, showing a Python file named 'Assignment - 04.py'. The code checks if a user is eligible to vote based on their age. The terminal below shows the script running and outputting 'You are eligible to vote.' for an age of 21.

```
157
158
159 # Question 6:
160
161 ...
162 step 1: Read input from user and store it in a variable age
163 step 2: The age variable value should be in between 0 - 120
164 step 3: Based on the age value, display the life stage using the following criteria:
165 if age >= 18 you are eligible to vote
166 if age < 18 and age >= 0 you are not eligible to vote
167 ...
168
169 age = int(input("Enter your age (0-120): "))
170
171 if 0 <= age <= 120:
172     if age >= 18:
173         print("You are eligible to vote.")
174     else:
175         print("You are not eligible to vote.")
176 else:
177     print("Invalid age. Please enter a value between 0 and 120.")


```

```
/usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/Collage/AI Assisted Coding/Assignments/Assignment - 04.py"
Assignment - 04.py
Enter your age (0-120): 21
You are eligible to vote.
# (base) = AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/Collage/AI Assisted Coding/Assignments/Assignment - 04.py"
# (base) = AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/Collage/AI Assisted Coding/Assignments/Assignment - 04.py"
# (base) = AI Assisted Coding
```

## Q7 Prompt Chaining (String Processing – Palindrome Names)

A screenshot of a code editor interface, likely VS Code, showing a Python script named `Assignment - 04.py`. The code defines a list of student names and prints out those that are palindromes. The output window shows the names Alice, Bob, Civic, Dad, Eve, Level, and Madam.

```
179
180
181
182 # Question 7:
183
184 ...
185 step 1: from the given list of names of students.
186 step 2: display the student names whose names are palindromes.
187
188 student_names = ["Alice", "Bob", "Civic", "Dad", "Eve", "Level", "Madam"]
189
190 palindrome_names = [name for name in student_names if name.lower() == name.lower()[::-1]]
191
192 print("Student names that are palindromes:")
193 for name in palindrome_names:
194     print(name)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER

```
/usr/local/bin/python3 "/Users/aravindreddy/Desktop/My Information/Collage/AI Assisted Coding/Assignments/Assignment - 04.py"
/usr/local/bin/python3 "/Users/aravindreddy/Desktop/My Information/Collage/AI Assisted Coding/Assignments/Assignment - 04.py"
Student names that are palindromes
Alice
Bob
Civic
Dad
Eve
Level
Madam
(base) - AI Assisted Coding
```

## Q8 Prompt Chaining (String Processing – Word Length Analysis)

A screenshot of a code editor interface, likely VS Code, showing a Python script named `Assignment - 04.py`. The code processes a list of words, printing each word as either 'Long' or 'Short' based on its length. The output window shows the words apple, bat, banana, cat, elephant, dog, and fish categorized as Long or Short.

```
196
197
198
199 # Question 8:
200
201 ...
202 step 1: Generate a list of words in a list.
203 step 2: Traverse through the list and find the length of the word.
204 step 3: Finally, display the words as 'Long' whose length is 5 or more else display as 'Short'.
205 ...
206
207 words = ["apple", "bat", "banana", "cat", "elephant", "dog", "fish"]
208
209 for word in words:
210     if len(word) >= 5:
211         print(f"(word): Long")
212     else:
213         print(f"(word): Short")
214
215
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER

```
/usr/local/bin/python3 "/Users/aravindreddy/Desktop/My Information/Collage/AI Assisted Coding/Assignments/Assignment - 04.py"
/usr/local/bin/python3 "/Users/aravindreddy/Desktop/My Information/Collage/AI Assisted Coding/Assignments/Assignment - 04.py"
apple: Long
bat: Short
banana: Long
cat: Short
elephant: Long
dog: Short
fish: Short
(base) - AI Assisted Coding
```