AI Assistant coding

Assignment-4.1

2303A51666

Sony Dodla

Batch-23

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.

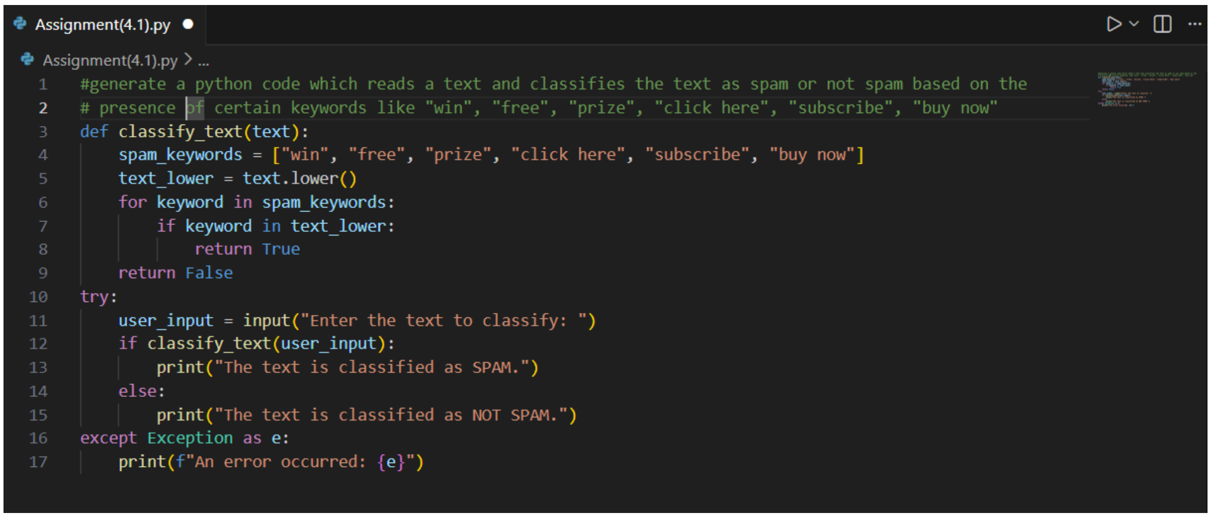
3. Display only the predicted label.

Input:

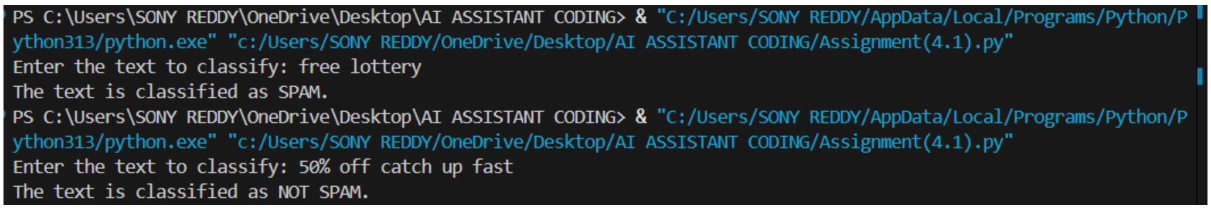
"Congratulations! You have won a free lottery ticket."

Expected Output:

Spam



```python
#generate a python code which reads a text and classifies the text as spam or not spam based on the
# presence of certain keywords like "win", "free", "prize", "click here", "subscribe", "buy now"
def classify_text(text):
    spam_keywords = ["win", "free", "prize", "click here", "subscribe", "buy now"]
    text_lower = text.lower()
    for keyword in spam_keywords:
        if keyword in text_lower:
            return True
    return False
try:
    user_input = input("Enter the text to classify: ")
    if classify_text(user_input):
        print("The text is classified as SPAM.")
    else:
        print("The text is classified as NOT SPAM.")
except Exception as e:
    print(f"An error occurred: {e}")
```



```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/P
ython313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the text to classify: free lottery
The text is classified as SPAM.
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/P
ython313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the text to classify: 50% off catch up fast
The text is classified as NOT SPAM.
```

Q2. One-Shot Prompting (Emotion detection)

Task:

Write a Python program that detects the emotion of a sentence

using one-shot prompting.

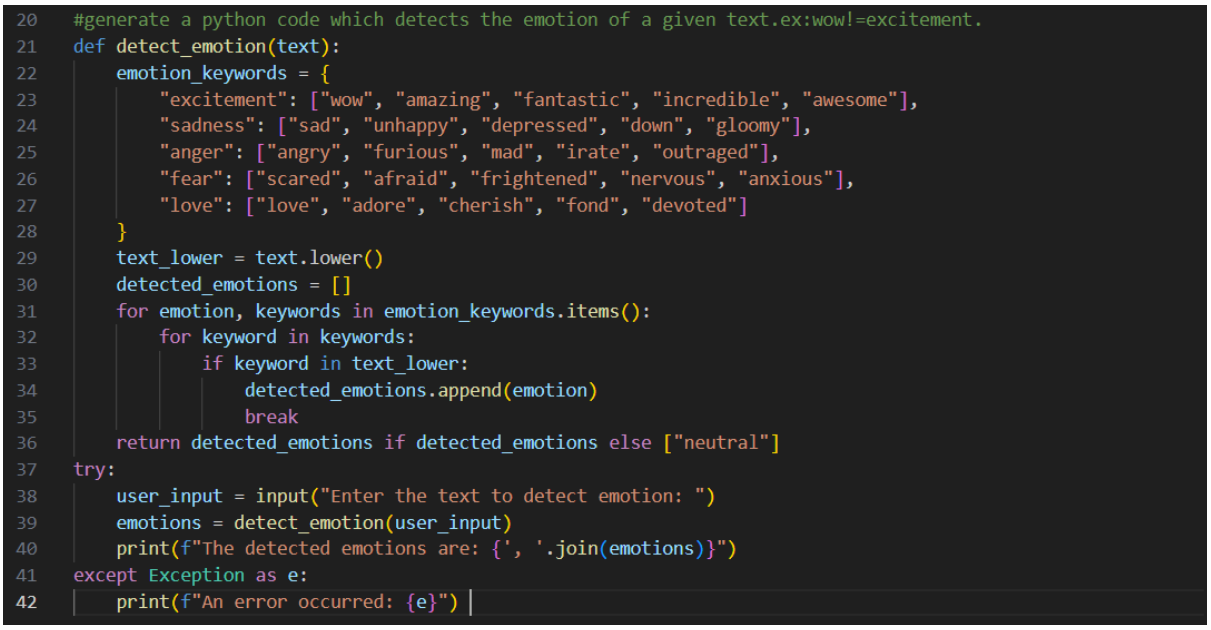
Emotions: [‘happy', 'sad', 'angry', 'excited', 'nervous', ’neutral’]

Steps:

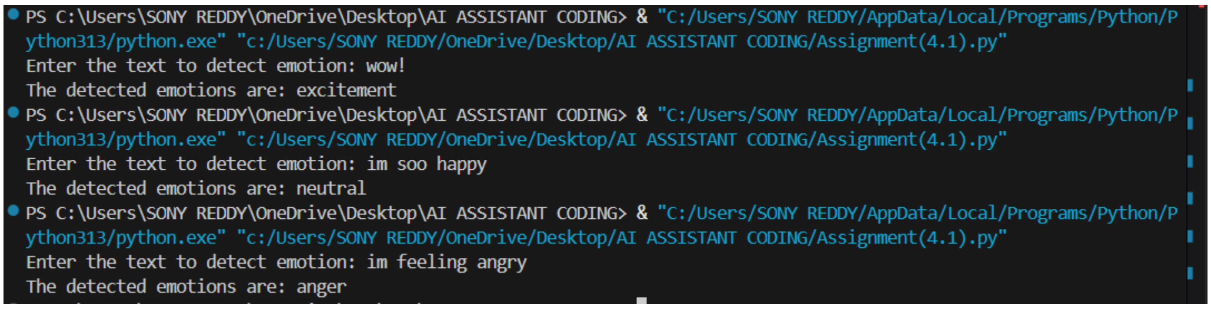
1. Provide one labeled example inside the prompt.

2. Take a sentence as input.

3. Print the predicted emotion



```python
#generate a python code which detects the emotion of a given text.ex:wow!=excitement.
def detect_emotion(text):
    emotion_keywords = {
        "excitement": ["wow", "amazing", "fantastic", "incredible", "awesome"],
        "sadness": ["sad", "unhappy", "depressed", "down", "gloomy"],
        "anger": ["angry", "furious", "mad", "irate", "outraged"],
        "fear": ["scared", "afraid", "frightened", "nervous", "anxious"],
        "love": ["love", "adore", "cherish", "fond", "devoted"]
    }
    text_lower = text.lower()
    detected_emotions = []
    for emotion, keywords in emotion_keywords.items():
        for keyword in keywords:
            if keyword in text_lower:
                detected_emotions.append(emotion)
                break
    return detected_emotions if detected_emotions else ["neutral"]
try:
    user_input = input("Enter the text to detect emotion: ")
    emotions = detect_emotion(user_input)
    print(f"The detected emotions are: {', '.join(emotions)}")
except Exception as e:
    print(f"An error occurred: {e}")
```



```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/P
ython313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the text to detect emotion: wow!
The detected emotions are: excitement
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/P
ython313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the text to detect emotion: im soo happy
The detected emotions are: neutral
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/P
ython313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the text to detect emotion: im feeling angry
The detected emotions are: anger
```

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):

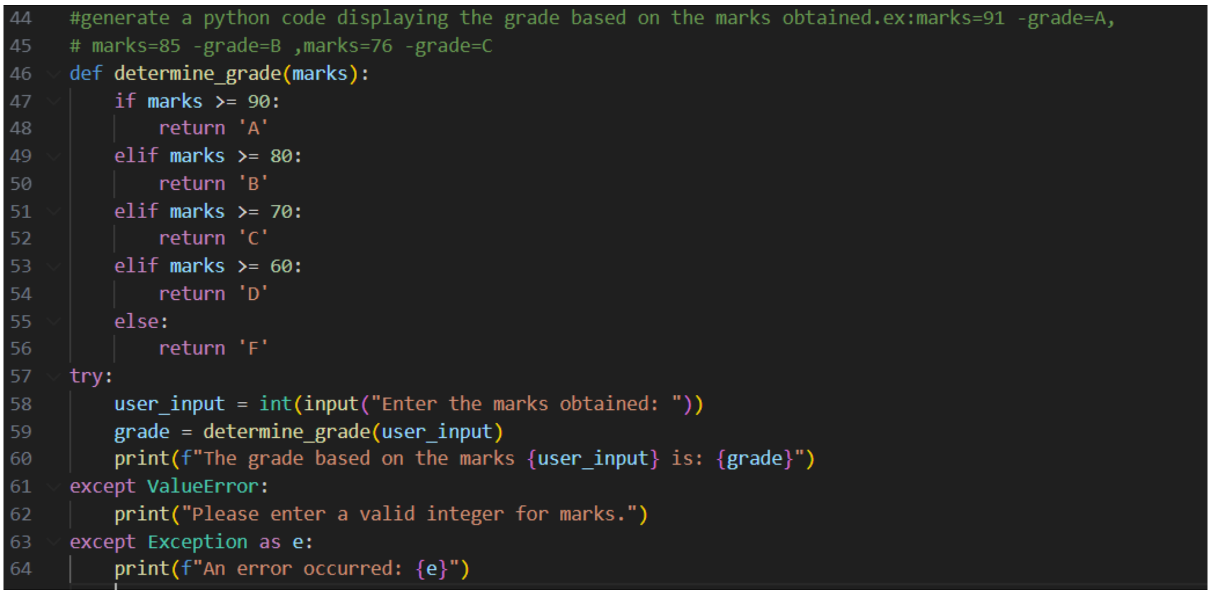
• 90–100 → A

• 80–89 → B

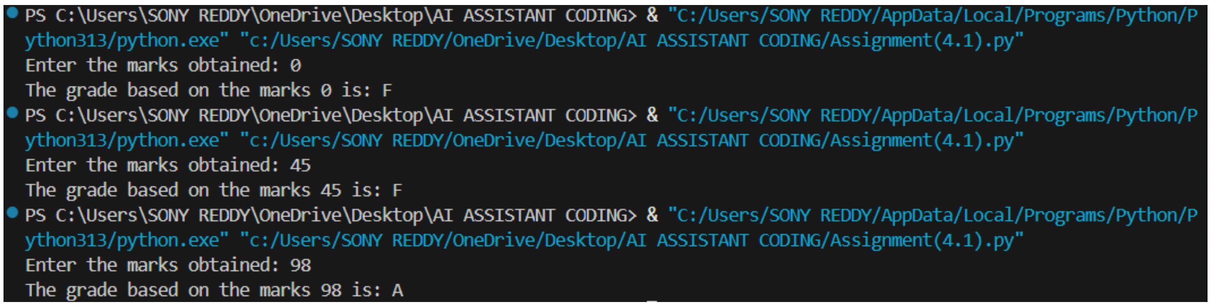
• 70–79 → C

• 60–69 → D

• Below 60 → F



```python
#generate a python code displaying the grade based on the marks obtained.ex:marks=91 -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 occurred: {e}")
```



```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/P
ython313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the marks obtained: 0
The grade based on the marks 0 is: F
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/P
ython313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the marks obtained: 45
The grade based on the marks 45 is: F
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/P
ython313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the marks obtained: 98
The grade based on the marks 98 is: A
```

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

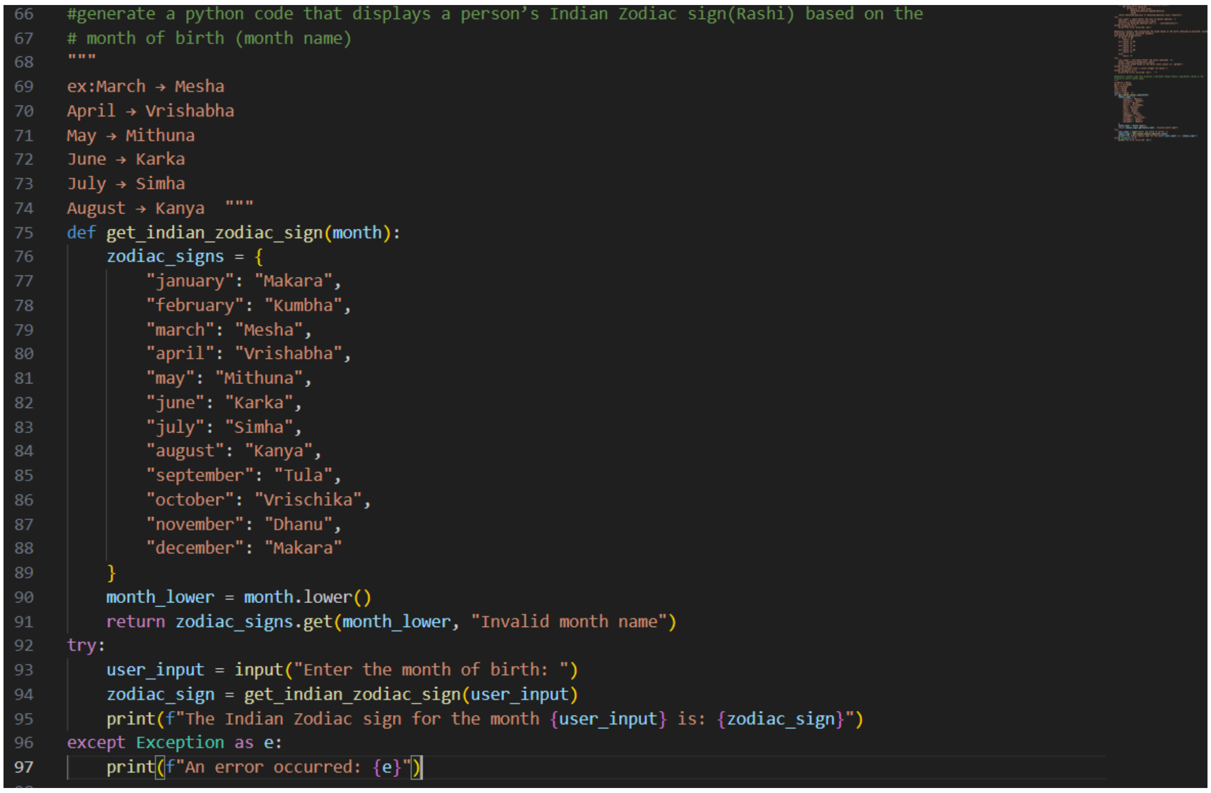
October → Vrischika

November → Dhanu

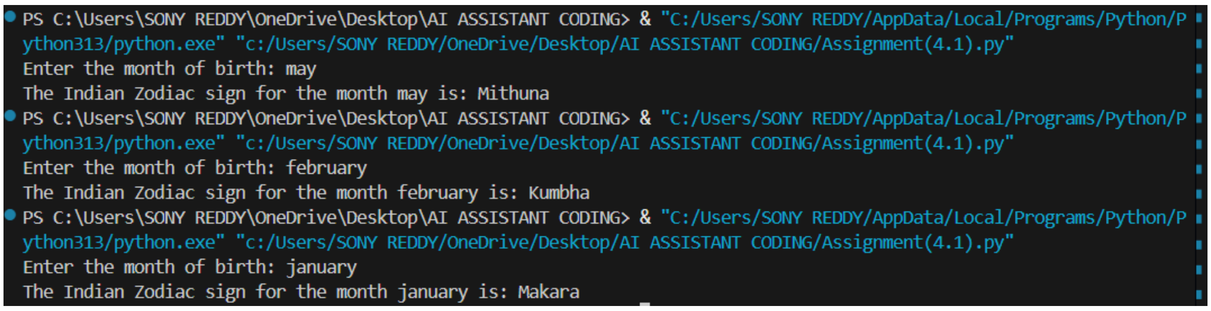
December → Makara

January → Kumbha

February → Meena



```python
#generate a python code that displays a person's Indian Zodiac sign(Rashi) based on the
# month of birth (month name)
"""
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(f"The Indian Zodiac sign for the month {user_input} is: {zodiac_sign}")
except Exception as e:
    print(f"An error occurred: {e}")
```



```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/P
ython313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the month of birth: may
The Indian Zodiac sign for the month may is: Mithuna
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/P
ython313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the month of birth: february
The Indian Zodiac sign for the month february is: Kumbha
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/P
ython313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the month of birth: january
The Indian Zodiac sign for the month january is: Makara
```

Q5. Result Analysis Based on Marks

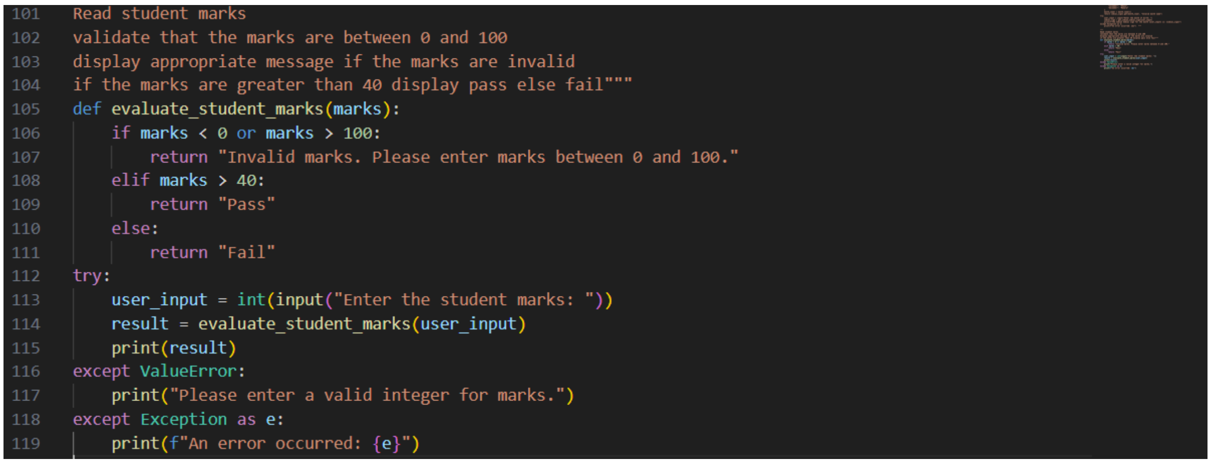
Task: Write a Python program that determines whether a student

Passes or Fails based on marks using Chain-of-Thought (CoT)

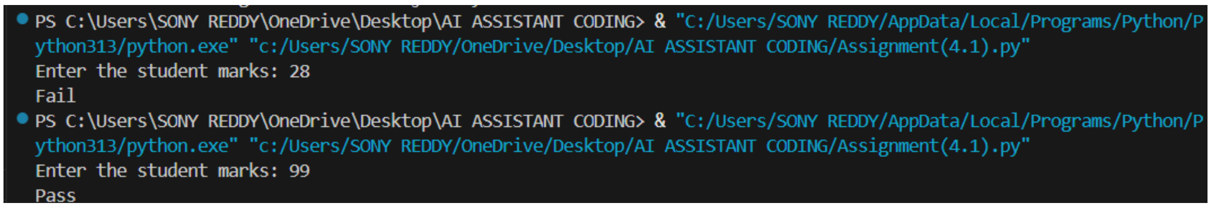
prompting.

Result Categories:

['Pass', 'Fail']



```python
Read student marks
validate that the marks are between 0 and 100
display appropriate message if the marks are invalid
if the marks are greater than 40 display pass else fail"""
def evaluate_student_marks(marks):
    if marks < 0 or marks > 100:
        return "Invalid marks. Please enter marks between 0 and 100."
    elif marks > 40:
        return "Pass"
    else:
        return "Fail"
try:
    user_input = int(input("Enter the student marks: "))
    result = evaluate_student_marks(user_input)
    print(result)
except ValueError:
    print("Please enter a valid integer for marks.")
except Exception as e:
    print(f"An error occurred: {e}")
```

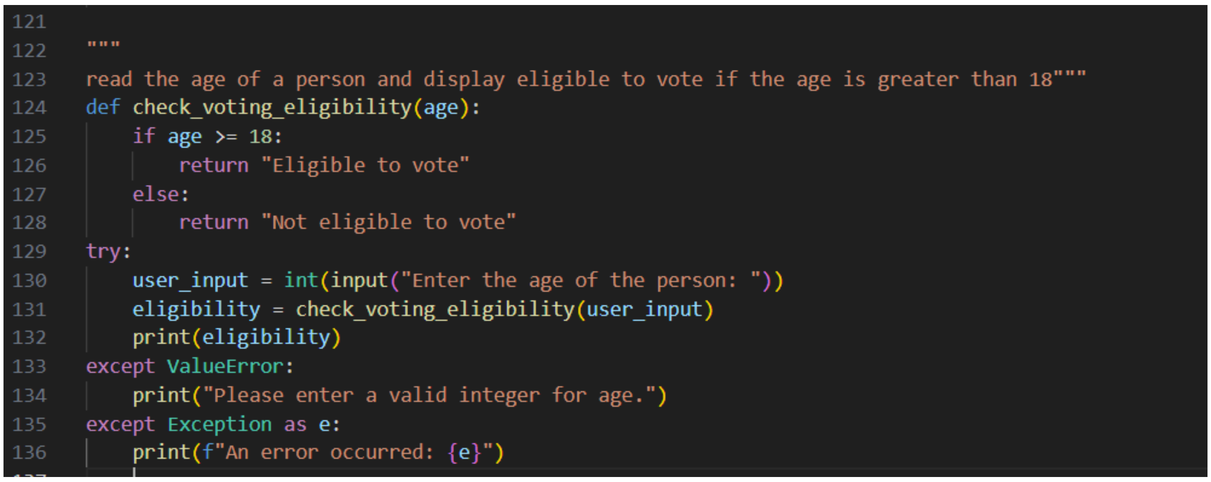


```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the student marks: 28
Fail
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the student marks: 99
Pass
```

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

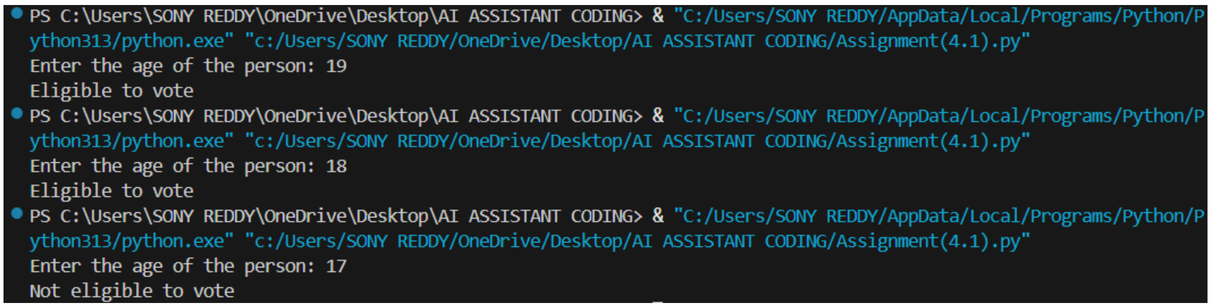
Task: Write a Python program that determines whether a person is

eligible to vote using Chain-of-Thought (CoT) prompting.



```python

"""
read the age of a person and display eligible to vote if the age is greater than 18"""
def check_voting_eligibility(age):
    if age >= 18:
        return "Eligible to vote"
    else:
        return "Not eligible to vote"
try:
    user_input = int(input("Enter the age of the person: "))
    eligibility = check_voting_eligibility(user_input)
    print(eligibility)
except ValueError:
    print("Please enter a valid integer for age.")
except Exception as e:
    print(f"An error occurred: {e}")
```

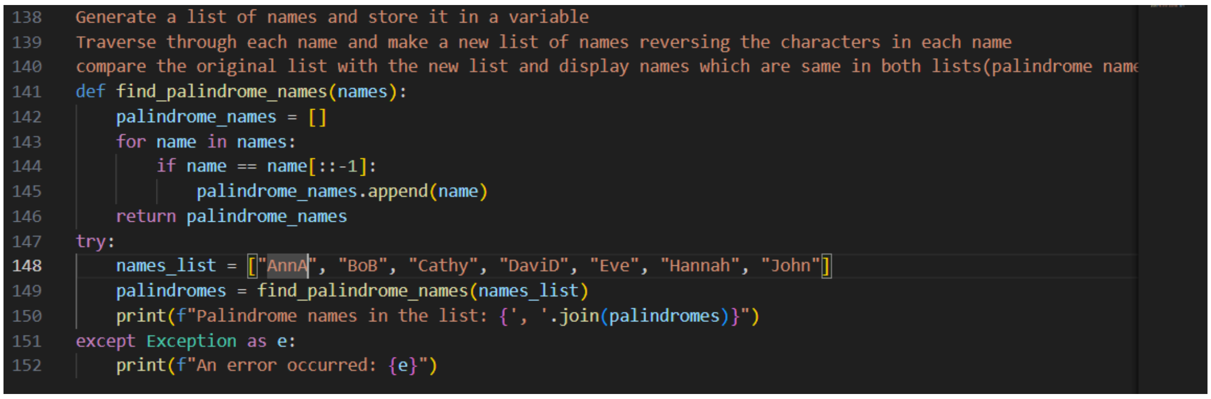

```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the age of the person: 19
Eligible to vote
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the age of the person: 18
Eligible to vote
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Enter the age of the person: 17
Not eligible to vote
```

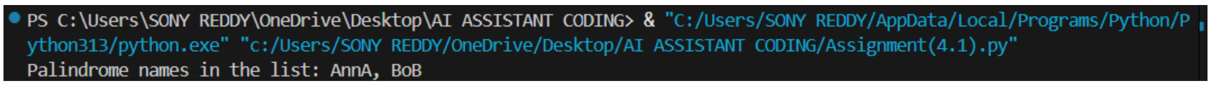
Q7 Prompt Chaining (String Processing – Palindrome Names)

Task: Write a Python program that uses the prompt chaining

technique to identify palindrome names from a list of student

names.


```
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 name
def find_palindrome_names(names):
    palindrome_names = []
    for name in names:
        if name == name[::-1]:
            palindrome_names.append(name)
    return palindrome_names
try:
    names_list = ["AnnA", "BoB", "Cathy", "DaviD", "Eve", "Hannah", "John"]
    palindromes = find_palindrome_names(names_list)
    print(f"Palindrome names in the list: {', '.join(palindromes)}")
except Exception as e:
    print(f"An error occurred: {e}")
```


```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Palindrome names in the list: AnnA, BoB
```

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.

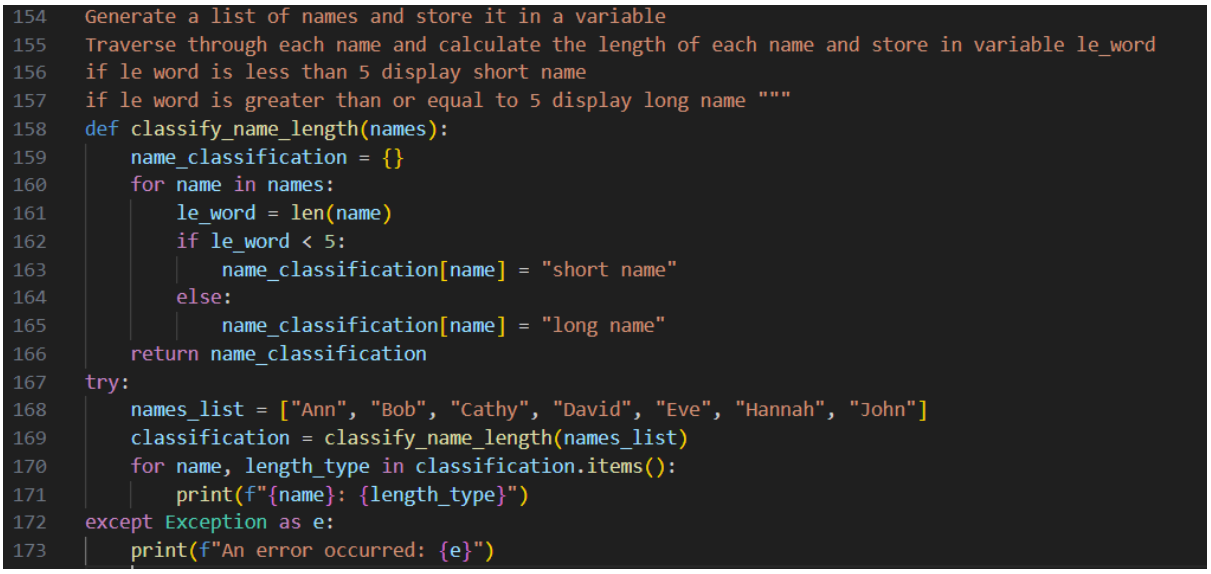
In the second prompt, traverse the list and calculate the length of

each word. In the third prompt, use the output of the previous step

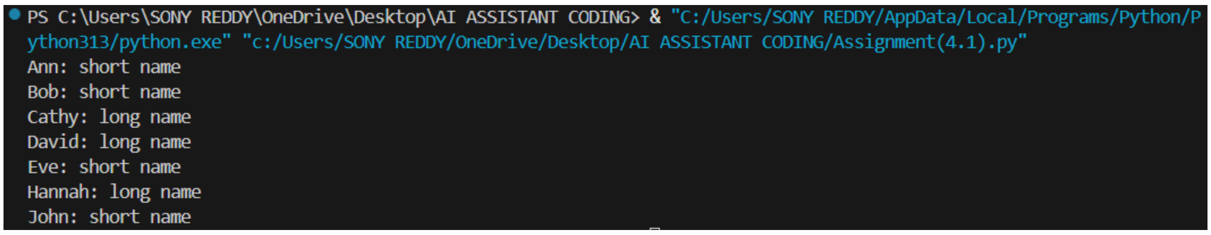
to determine whether each word is Short (length less than 5) or

Long (length greater than or equal to 5), and display the result for

each word



```
Generate a list of names and store it in a variable
Traverse through each name and calculate the length of each name and store in variable le_word
if le word is less than 5 display short name
if le word is greater than or equal to 5 display long name """
def classify_name_length(names):
    name_classification = {}
    for name in names:
        le_word = len(name)
        if le_word < 5:
            name_classification[name] = "short name"
        else:
            name_classification[name] = "long name"
    return name_classification
try:
    names_list = ["Ann", "Bob", "Cathy", "David", "Eve", "Hannah", "John"]
    classification = classify_name_length(names_list)
    for name, length_type in classification.items():
        print(f"{name}: {length_type}")
except Exception as e:
    print(f"An error occurred: {e}")
```



```
PS C:\Users\SONY REDDY\OneDrive\Desktop\AI ASSISTANT CODING> & "C:/Users/SONY REDDY/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/SONY REDDY/OneDrive/Desktop/AI ASSISTANT CODING/Assignment(4.1).py"
Ann: short name
Bob: short name
Cathy: long name
David: long name
Eve: short name
Hannah: long name
John: short name
```