AI Assisted Coding (1425)
Assignment – 4.2

Name : K.Tejaswini

HTNO : 2303A51425

Batch : 21

• Zero-shot:PromptAIwithonlytheinstruction.WriteaPythonfunctiontodetermine

whether a given number is prime

ExpectedOutput-1

• AbasicPythonfunctiontocheckifanumberisprime,demonstratingcorrectlogical

conditions without relying on examples or additional context.

Code:

```python
#Write a Python function to determine whether a given number is prime.
def is_prime(num):
    if num <= 1:
        return False
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            return False
    return True
number = int(input("Enter a number to check if it's prime: "))
if is_prime(number):
    print(f"{number} is a prime number.")
else:
    print(f"{number} is not a prime number.")
```

Output:

```
(base) PS C:\Users\WINDOWS\OneDrive\Desktop\Teja 3_2\AI Assit coding> &
coding/task.py"
Enter a number to check if it's prime: 15
15 is not a prime number.
(base) PS C:\Users\WINDOWS\OneDrive\Desktop\Teja 3_2\AI Assit coding>
```

Final Observation:

Theis_prime(n)functioncheckswhetheranumberisprimebyfirstmakingsurethenumber is greater than 1, since 0, 1, and negative numbers are not prime. It then goes through the numbersfrom2ton−1 to see if anyof themdivide the givennumber exactly.If itfindseven onesuchnumber,itreturns False,meaningthenumberisnotprime.Ifnodivisorisfound,it returns True, confirming that the number is prime. Because this solution was generated usingzero-shotprompting,withoutgivinganyexamples,itshowsthattheAIcanunderstand the concept and produce a correct but basic prime-checking logic.

TaskDescription-2

• One-shot:Provideoneexample:Input:[1,2,3,4],Output:10tohelpAIgeneratea

function that calculates the sum of elements in a list.

ExpectedOutput-2

• Acorrectconversionfunctionguidedbythesingleexample.

Code:

```python
#write a python function to calculate the sum of elements in a list.
# Input: [1, 2, 3, 4].
# Output: 10
def sum_of_list(lst):
    total = 0
    for num in lst:
        total += num
    return total
# Example usage:
input_list = [1, 2, 3, 4]
result = sum_of_list(input_list)
print(f"The sum of {input_list} is {result}.")
```

Output:

```
(base) PS C:\Users\WINDOWS\OneDrive\Desktop\Teja 3_2\AI Assit coding>
coding/task.py"
The sum of [1, 2, 3, 4] is 10.
(base) PS C:\Users\WINDOWS\OneDrive\Desktop\Teja 3_2\AI Assit coding>
```

Final Observation:

The sum_of_elements(arr) function calculates the total sum of all elements in a list by startingwith avariable total setto zeroand thenaddingeachnumber inthe listone byone. After the loop finishes, the final value of total is returned as the sum of the list. Since this functionwasgeneratedusingone-shotprompting,whereasingleinput-outputexamplewas

provided,theexamplehelpedguidetheAItoclearlyunderstandthetaskandproducea correct and simple function for summing list elements.

TaskDescription-3

• Few-shot:Give2–3examplestocreateafunctionthatextractsdigitsfroman

alphanumeric string.

ExpectedOutput-3

• Accuratefunctionthatreturnsonlythedigitsfromalphanumericstring.

Code:

```python
#write a python function to extract only digits from an alphanumeric string.
def extract_digits(s):
    digits = ''
    for char in s:
        if char.isdigit():
            digits += char
    return digits
# Example usage:
input_string = "a1b2c3d4e5"
result = extract_digits(input_string)
print(f"The digits extracted from the string are: {result}")
```

Output:

```
exe "c:/Users/WINDOWS/OneDrive/Desktop/Teja 3_2/AI
The digits extracted from the string are: 12345
```

Final Observation:

The extract_digits(text) function goes through each character in the given string and checks whether it is a digit using the isdigit() method. If the character is a digit, it is added to the resultstring.Aftertheloopfinishes,thefunctionreturnsastringthatcontainsonlythedigits from the original input. Since this was generated using few-shot prompting, the multiple examplesclearlyguidedtheAItorecognizethepatternofremovinglettersandkeepingonly numbers, resulting in an accurate and reliable function.

TaskDescription-4

• Comparezero-shotvsfew-shotpromptingforgeneratingafunctionthatcountsthe

number of vowels in a string.

ExpectedOutput-4

• Outputcomparison+studentexplanationonhowexampleshelpedthemodel. Zero

Shot:

Code:

```python
#Write a Python function that counts the number of vowels in a given string.
def count_vowels(s):
    vowels = "aeiouAEIOU"
    count = 0
    for char in s:
        if char in vowels:
            count += 1
    return count
# Example usage:
input_string = input("Enter a string: ")
vowel_count = count_vowels(input_string)
print(f"The number of vowels in the string is: {vowel_count}")
```

Output:

```
exe "c:/Users/WINDOWS/OneDrive/Desktop/Teja 3_2/AI Ass:
Enter a string: " AI ASSISTANT"
The number of vowels in the string is: 5
```

FewShot:

Code:

```python
#Write a python function to count the number of vowels in a string.
def count_vowels(s):
    vowels = "aeiouAEIOU"
    count = 0
    for char in s:
        if char in vowels:
            count += 1
    return count
# Example usage:
input_string = "Hello Tejaswini"
vowel_count = count_vowels(input_string)
print(f"The number of vowels in the string is: {vowel_count}")
```

Output:


```
(base) PS C:\Users\WINDOWS\OneDrive\Desktop\Teja 3_2\AI Assit co
exe "c:/Users/WINDOWS/OneDrive/Desktop/Teja 3_2/AI Assit coding/
The number of vowels in the string is: 6
(base) PS C:\Users\WINDOWS\OneDrive\Desktop\Teja 3 2\AI Assit co
```

Output Comparison:

Zero-shotversion:
Thefunctionisclear,simple,andcorrect,usingaloopandcountervariable.

Few-shotversion:
Thefunctionismorerefinedandcompact.Theexampleshelpedthemodelclearly understandthatbothuppercaseandlowercasevowelsmustbecounted,leadingtoamore confident and optimized implementation.

Final Observation:

In the zero-shot approach, the AI generated a basic vowel-counting function based only on the instruction, which resulted in a straightforward loop-based solution. In the few-shot approach,thegivenexamplesclearlyshowedwhatshouldbeconsideredasvowelsandhow the output should look. These examples helped the AI better understand the pattern and expectations, which led to a cleaner and slightly optimized solution. This comparison shows that providing examples improves clarity and often results in better-structured and more accurateoutputs.

TaskDescription-5

• Usefew-shotpromptingwith3sampleinputstogenerateafunctionthatdetermines the

minimum of three numbers without using the built-in min() function.

ExpectedOutput-5

• Afunctionthathandlesallcaseswithcorrectlogicbasedonexamplepatterns

• **Code:**

```
#write a python function to find the minimum of three numbers without using the bulit-in min() function.
def find_minimum(a, b, c):
    if a <= b and a <= c:
        return a
    elif b <= a and b <= c:
        return b
    else:
        return c
# Example usage:
num1, num2, num3 = 7,18,5
minimum = find_minimum(num1, num2, num3)
print(f"The minimum of {num1}, {num2}, {num3} is {minimum}.")
```

Output:

```
(base) PS C:\Users\WINDOWS\OneDrive\Desktop\Teja 3_2\AI Assit coding>
coding/task.py"
The minimum of 7, 18, 5 is 5.
```

Final Observation:

The find_minimum(a, b, c) function compares the three given numbers using conditional statements.Itfirstcheckswhether a issmallerthan orequaltoboth band c.Ifnot,itthen checkswhetherbissmallerthanorequaltotheothertwo.Ifneitheroftheseconditionsis true, the function returns c as the minimum. Since this function was generated using few- shot prompting, the three examples clearly showed the expected pattern of comparing values and returning the smallest one. These examples helped the AI design logic that correctlyhandlesallcases,includingnegativenumbers.