

Assignment-4.2

Name:M.Harshith

Ht.no:2303A51444

Batch.No:21

Question1:Zero-ShotPrompting(PrimeNumberCheck)

Task

Write a zero-shot prompt to generate a Python function that checks whether a given number is prime.

Prompt used:

Write a Python function that checks whether a given number is prime or not.

Code:

```
#Write a Python function that checks whether a given number is prime or not
def is_prime(n):
    """Return True if n is a prime number, else False."""
    if n <= 1:
        return False
    if n <= 3:
        return True
    if n % 2 == 0 or n % 3 == 0:
        return False
    i = 5
    while i * i <= n:
        if n % i == 0 or n % (i + 2) == 0:
            return False
        i += 6
    return True
# Example usage:
number = int(input("Enter a number to check if it is prime: "))
if is_prime(number):
    print(f"{number} is a prime number.")
else:
    print(f"{number} is not a prime number.")
```

Output:

```
P -- C:\Users\dell\OneDrive\Desktop\AI\20-01-2026
Enter a number to check if it is prime: 29
29 is a prime number.
PS C:\Users\dell\OneDrive\Desktop\AI>
```

Justification:

This zero-shot prompt provided only an instruction without examples. The AI generated a basic and logically correct prime-checking function. However, the solution is not optimized because it checks divisibility up to $n-1$. This demonstrates that zero-shot prompting works but may lack efficiency and edge-case optimization.

Question 2: One-Shot Prompting (Sum of List Elements)

Task:

Provide one example to guide AI in generating a function to sum elements of a list.

Prompt Used:

Write a Python function to calculate the sum of elements in a list. Example:

Input: [1, 2, 3, 4]

Output: 10

Code:

```
#Write a Python function to calculate the sum of elements in a list.
#Example:
#Input: [1, 2, 3, 4]
#Output: 10
def sum_list(numbers):
    total = 0
    for num in numbers:
        total += num
    return total

lst = [1, 2, 3, 4]
print("Sum:", sum_list(lst))
```

Output:

```
132-x64\bundled\libs\debugpy\launcher 50582 -- C:\Users\dell\OneDrive\
Desktop\AI\20-01-2026.py'
Sum: 10
PS C:\Users\dell\OneDrive\Desktop\AI>
```

Justification:

The one-shot example clearly guided the AI to understand:

- Input format (list)
- Expected output (sum)

Compared to zero-shot prompting, one-shot prompting improves clarity and correctness by giving a reference pattern.

Question 3: Few-Shot Prompting (Extract Digits from Alphanumeric String)

Task

Use 2–3 examples to generate a function that extracts digits from an alphanumeric string.

Prompt Used

Write a Python function that extracts digits from an alphanumeric string. Examples:

Input: "a1b2c3" → Output: "123"

Input: "abc123xyz" → Output: "123"

Input: "9a8b7" → Output:

"987" Code:

```
#Write a Python function that extracts digits from an alphanumeric string
#Examples:
#Input: "a1b2c3" → Output: "123"
#Input: "abc123xyz" → Output: "123"
#Input: "9a8b7" → Output: "987"

def extract_digits(s):
    digits = ""
    for char in s:
        if char.isdigit():
            digits += char
    return digits

# Example usage:
input_str = input("Enter an alphanumeric string: ")
result = extract_digits(input_str)
print("Extracted digits:", result)
```

Output:

```
ve\Desktop\AI\20-01-2026.py'
Enter an alphanumeric string: a1b2c3
Extracted digits: 123
PS C:\Users\dell\OneDrive\Desktop\AI>
```

Justification:

Few-shot prompting helped the AI:

- Identify the pattern across multiple examples
- Use `isdigit()` effectively
- Produce accurate and consistent output

This shows few-shot prompting significantly improves reliability and pattern recognition.

Question 4: Zero-Shot vs Few-Shot Prompting (Vowel Count Comparison)

Zero-ShotPrompt:

Write a Python function to count the number of vowels in a string. Code:

```
#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_str = input("Enter a string: ")
vowel_count = count_vowels(input_str)
print("Number of vowels:", vowel_count)
```

Output:

```
32-x64\bundled\libs\debugpy\launcher'
ve\Desktop\AI\20-01-2026.py'
Enter a string: hello
Number of vowels: 2
S C:\Users\dell\OneDrive\Desktop\AI>
```

Few-ShotPrompt:

Write a Python function to count vowels in a string. Examples:

Input: "hello" → Output: 2

Input: "education" → Output: 5

Few-Shot Code:

```
#Few-Shot Code Write a Python function to count vowels in a string.
#Examples:
#Input: "hello" → Output: 2
#Input: "education" → Output: 5
def count_vowels(s):
    vowels = "aeiouAEIOU"
    return sum(1 for ch in s if ch in vowels)
# Example usage:
input_str = input("Enter a string: ")
vowel_count = count_vowels(input_str)
print("Number of vowels:", vowel_count)
```

Output:

```
n32-x64\bundled\libs\debugpy\launcher' '4930
ive\Desktop\AI\20-01-2026.py'
Enter a string: hello
Number of vowels: 2
PS C:\Users\dell\OneDrive\Desktop\AI>
```

ComparisonOutput

PromptType	Accuracy	CaseHandling	Efficiency
Zero-shot	Medium	No	Low
Few-shot	High	Yes	High

Justification:

Few-shotpromptinghelpedtheAI:

- Handleuppercasevowels
- Writemoreoptimizedandconcisecode
- Improveoveralllogicquality

ThisclearlyprovesexamplessignificantlyenhanceAIunderstanding

Question5:Few-ShotPrompting(MinimumofThreeNumbers)

Task

Generate a function to find the minimum of three numbers **without using min()**. **Prompt**

Used:

Write a Python function to find the minimum of three numbers without using min().

Examples:

Input: 3, 7, 5 → Output: 3

Input: 10, 2, 8 → Output: 2

Input: -1, -5, 0 → Output: -5

Code:

```
#Write a Python function to find the minimum of three numbers without u
#Examples:
#Input: 3, 7, 5 → Output: 3
#Input: 10, 2, 8 → Output: 2
#Input: -1, -5, 0 → Output: -5
def min_of_three(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 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
num3 = float(input("Enter third number: "))
minimum = min_of_three(num1, num2, num3)
print("The minimum of the three numbers is:", minimum)
```

Output:


```
ve\Desktop\AI\20-01-2026.py'  
Enter first number: 3  
Enter second number: 7  
Enter second number: 7  
Enter third number: 5  
The minimum of the three numbers is: 3.0  
PS C:\Users\dell\OneDrive\Desktop\AI>
```

Justification

Few-shot prompting guided the AI to:

- Handle negative numbers
- Use logical comparisons correctly
- Avoid built-in functions as instructed

This resulted in **accurate and well-structured logic**.

Conclusion

- **Zero-shot prompting** is useful for simple tasks but may lack optimization.
- **One-shot prompting** improves clarity and correctness.
- **Few-shot prompting** delivers the most accurate, efficient, and reliable results.
- Providing examples greatly enhances AI code quality and understanding.