

## School of Computer Science and Artificial Intelligence

### Lab Assignment # 4.2

<b>Program</b>	<b>:</b> B. Tech (CSE)
<b>Specialization</b>	<b>:</b> --
<b>Course Title</b>	<b>:</b> AI Assisted coding
<b>Course Code</b>	<b>:</b>
<b>Semester</b>	<b>II</b>
<b>Academic Session</b>	<b>:</b> 2025-2026
<b>Name of Student</b>	<b>:</b> Akash Reddy
<b>Enrollment No.</b>	<b>:</b> 2403A51L30
<b>Batch No.</b>	<b>:</b> 51
<b>Date</b>	<b>:</b> 20-01-2026

### Task Description-1

Zero-shot: Prompt AI with only the instruction. Write a Python function to determine whether a given number is prime.

The screenshot shows a dark-themed instance of Visual Studio Code. In the Explorer sidebar, there are two open editors: 'lab\_1.2.py' and 'lab\_4.2.py'. The 'lab\_4.2.py' editor contains the following Python code:

```
# Task Description-1
# Zero-shot: Prompt AI with only the instruction. Write a Python function to determine whether a given number is prime.
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True
number = 29
print(f"{number} is prime: {is_prime(number)}")
```

In the bottom right corner, the terminal window shows the output of running the script:

```
/usr/local/bin/python3 /Users/akash/Desktop/ai_assis/lab_4.2.py
(base) akash@AKASH-MacBook-Air ai_assis % /usr/local/bin/python3 /Users/akash/Desktop/ai_assis/lab_4.2.py
29 is prime: True
(base) akash@AKASH-MacBook-Air ai_assis %
```

### Explanation:

```
# The function is_prime checks if a number n is prime by testing divisibility from 2 to the square root of n.
# If any divisor is found, it returns False; otherwise, it returns True.
# Few-shot: Provide AI with examples before the instruction.
# Example 1: 5 is prime because it has no divisors other than 1
# and itself.
```

### Task Description-2

- One-shot: Provide one example: Input: [1, 2, 3, 4], Output: 10 to help AI generate a function that calculates the sum of elements in a list.

```

11  # print(f'{number} is prime: {is_prime(number)}')
12
13 # Task Description-2
14 # One-shot: Provide one example: Input: [1, 2, 3, 4],
15 # Output: 10 to help AI generate a function that calculates the sum of elements in a list.
16 def sum_of_list(lst):
17     return sum(lst)
18 numbers = [1, 2, 3, 4]
19 print("Sum of list:", sum_of_list(numbers))
20
21
22
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

/usr/local/bin/python3 /Users/akash/Desktop/ai\_assis/lab\_4.2.py  
 ● (base) akash@AKASHs-MacBook-Air ai\_assis % /usr/local/bin/python3 /Users/akash/Desktop/ai\_assis/lab\_4.2.py  
 Sum of list: 10  
 ◇ (base) akash@AKASHs-MacBook-Air ai\_assis %

## Explanation

# The function `sum\_of\_list` takes a list as input and returns the sum of its elements using Python's built-in `sum()` function.

## Task Description-3

- Few-shot: Give 2–3 examples to create a function that extracts digits from an alphanumeric string.

```

20
21 # Task Description-3
22 # Few-shot: Give 2–3 examples to create a function that extracts digits from an alphanumeric string.
23 def extract_digits(s):
24     return ''.join(filter(str.isdigit, s))
25
26 # Example 1
27 input_str1 = "abc123def456"
28 output1 = extract_digits(input_str1)
29 print("Extracted digits:", output1)
30
31 # Example 2
32 input_str2 = "no_digits_here!"
33 output2 = extract_digits(input_str2)
34 print("Extracted digits:", output2)
35 # Example 3
36 input_str3 = "2024year!"
37 output3 = extract_digits(input_str3)
38 print("Extracted digits:", output3)
39
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

/usr/local/bin/python3 /Users/akash/Desktop/ai\_assis/lab\_4.2.py  
 ● (base) akash@AKASHs-MacBook-Air ai\_assis % /usr/local/bin/python3 /Users/akash/Desktop/ai\_assis/lab\_4.2.py  
 Extracted digits: 123456  
 Extracted digits:  
 Extracted digits: 2024  
 ◇ (base) akash@AKASHs-MacBook-Air ai\_assis %

## Explanation

# The function `extract\_digits` takes an alphanumeric string as input and uses the `filter` function along with `# str.isdigit` to extract only the digit characters from the string.  
 # It then joins these characters into a single string and returns it.

```
# The examples demonstrate the function's ability to extract digits from different types of input strings.
```

## Task Description-4

- Compare zero-shot vs few-shot prompting for generating a function that counts the number of vowels in a string.

```
40
41
42 # Task Description-4
43 # Compare zero-shot vs few-shot prompting for generating a function that counts the number of vowels in a string.
44 def count_vowels(s):
45     vowels = "aeiouAEIOU"
46     return sum(1 for char in s if char in vowels)
47
48 # Zero-shot example
49 input_str = "Hello World!"
50 print("Number of vowels (zero-shot):", count_vowels(input_str))
51
52 # Few-shot examples
53 input_str1 = "This is a test."
54 input_str2 = "Another example."
55 print("Number of vowels (few-shot):", count_vowels(input_str1))
56 print("Number of vowels (few-shot):", count_vowels(input_str2))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

/usr/local/bin/python3 /Users/akash/Desktop/ai\_assis/lab\_4.2.py  
● (base) akash@AKASHs-MacBook-Air ai\_assis % /usr/local/bin/python3 /Users/akash/Desktop/ai\_assis/lab\_4.2.py  
Number of vowels (zero-shot): 3  
Number of vowels (few-shot): 4  
Number of vowels (few-shot): 6  
◇ (base) akash@AKASHs-MacBook-Air ai\_assis %

## Comparison

```
# The zero-shot approach relies solely on the initial instruction, which may lead to less accurate or less optimized code.  
# The few-shot approach, by providing examples, helps the AI understand the context better and often results in  
# more accurate and efficient implementations. In this case, both approaches yield the same function,  
# but few-shot prompting can enhance understanding for more complex tasks.
```

## Task Description-5

Use few-shot prompting with 3 sample inputs to generate a function that determines the minimum of three numbers without using

```
59
60 # Task Description-5
61 # Use few-shot prompting with 3 sample inputs to generate a function that determines the minimum of three numbers
62 # without using the built-in min() function.
63 def find_minimum(a, b, c):
64     if a < b and a < c:
65         return a
66     elif b < c:
67         return b
68     else:
69         return c
70
71 # Example inputs
72 print("Minimum (few-shot):", find_minimum(3, 1, 2))
73 print("Minimum (few-shot):", find_minimum(5, 10, 7))
74 print("Minimum (few-shot):", find_minimum(-1, -5, -3))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

/usr/local/bin/python3 /Users/akash/Desktop/ai\_assis/lab\_4.2.py  
● (base) akash@AKASHs-MacBook-Air ai\_assis % /usr/local/bin/python3 /Users/akash/Desktop/ai\_assis/lab\_4.2.py  
Minimum (few-shot): 1  
Minimum (few-shot): 5  
Minimum (few-shot): -5  
◇ (base) akash@AKASHs-MacBook-Air ai\_assis %

## Explanation

```
# The function find_minimum takes three numbers as input and compares them using conditional statements to determine the smallest number.
```