LAB ASSIGNMENT – 4.2

B.Varun

2303A52020

Batch- 38

Lab 4: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques

Lab Objectives:

- To explore and apply different levels of prompt examples in AI-assisted code generation.
- To understand how zero-shot, one-shot, and few-shot prompting affect AI output quality.
- To evaluate the impact of context richness and example quantity on AI performance.
- To build awareness of prompt strategy effectiveness for different problem types

Lab Outcomes (LOs):

After completing this lab, students will be able to:

- Use zero-shot prompting to instruct AI with minimal context.
- Use one-shot prompting with a single example to guide AI code generation.
- Apply few-shot prompting using multiple examples to improve AI responses.
- Compare AI outputs across the three prompting strategies.

Task Description-1

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

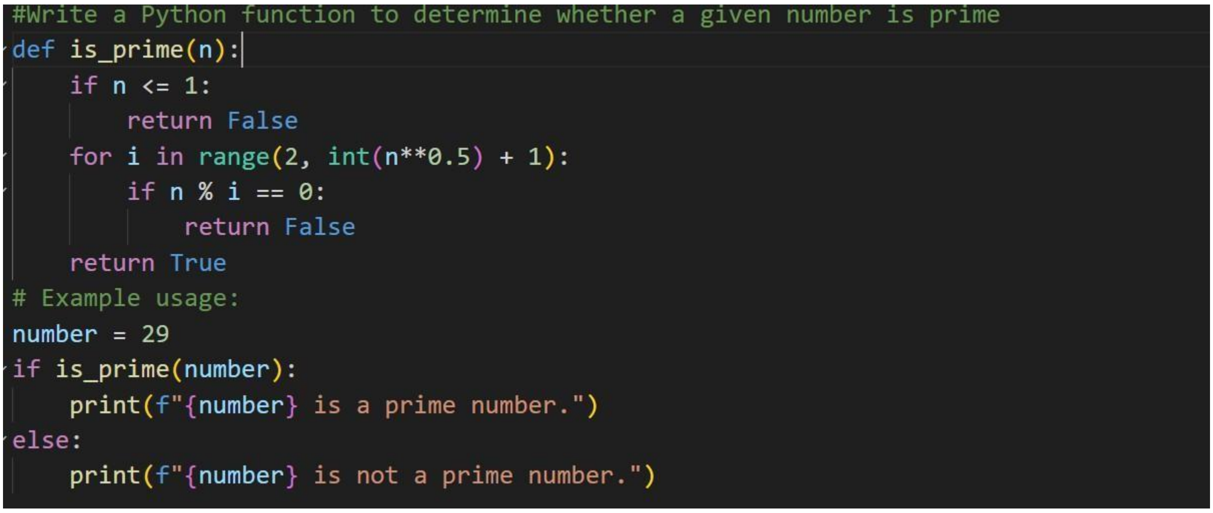
Expected Output-1

- A basic Python function to check if a number is prime, demonstrating correct logical conditions without relying on examples or additional context

PROMPT:

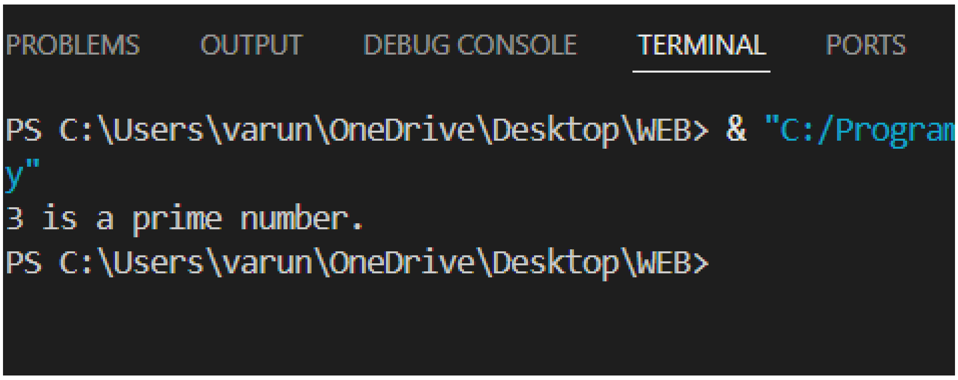
Write a Python function to determine whether a given number is prime

CODE:



```python
#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
# Example usage:
number = 29
if is_prime(number):
    print(f"{number} is a prime number.")
else:
    print(f"{number} is not a prime number.")
```

OUTPUT:



```
PS C:\Users\varun\OneDrive\Desktop\WEB> & "C:/Program
y"
3 is a prime number.
PS C:\Users\varun\OneDrive\Desktop\WEB>
```

EXPLANATION:

This program checks whether a number is prime or not. A number is prime if it is divisible only by 1 and itself. The program checks divisibility from 2 up to the square root of the number. If any number divides it, the number is not prime; otherwise, it is prime.

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.

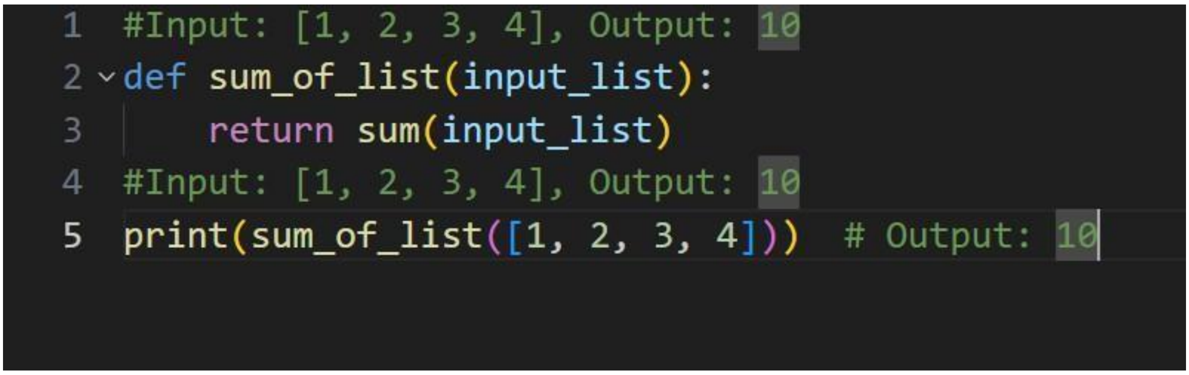
Expected Output-2

- A correct conversion function guided by the single example.

PROMPT:

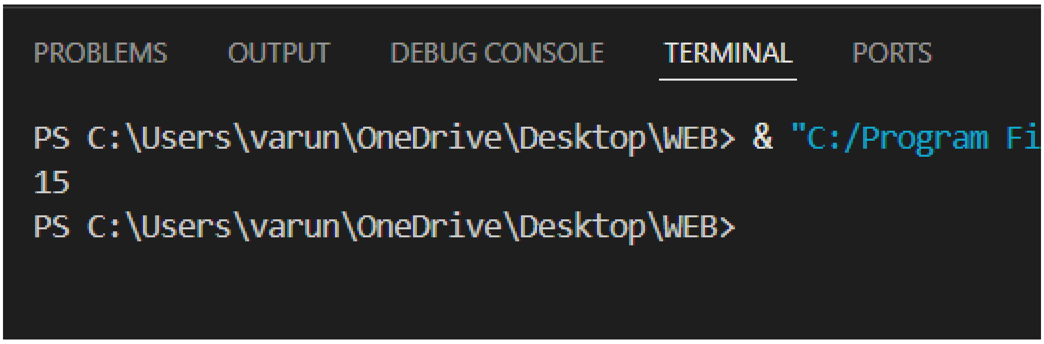
Input: [1, 2, 3, 4], Output: 10

CODE:



```python
#Input: [1, 2, 3, 4], Output: 10
def sum_of_list(input_list):
    return sum(input_list)
#Input: [1, 2, 3, 4], Output: 10
print(sum_of_list([1, 2, 3, 4]))  # Output: 10
```

OUTPUT:



```
PS C:\Users\varun\OneDrive\Desktop\WEB> & "C:/Program Fi
15
PS C:\Users\varun\OneDrive\Desktop\WEB>
```

EXPLANATION:

This program defines a function that finds the sum of all elements in a list. The function takes a list as input and uses the built-in sum () function to add all the values. When the list [1, 2, 3, 4] is passed to the function, it returns 10, which is printed as the output**.**

Task Description-3

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

alphanumeric string.

Expected Output-3

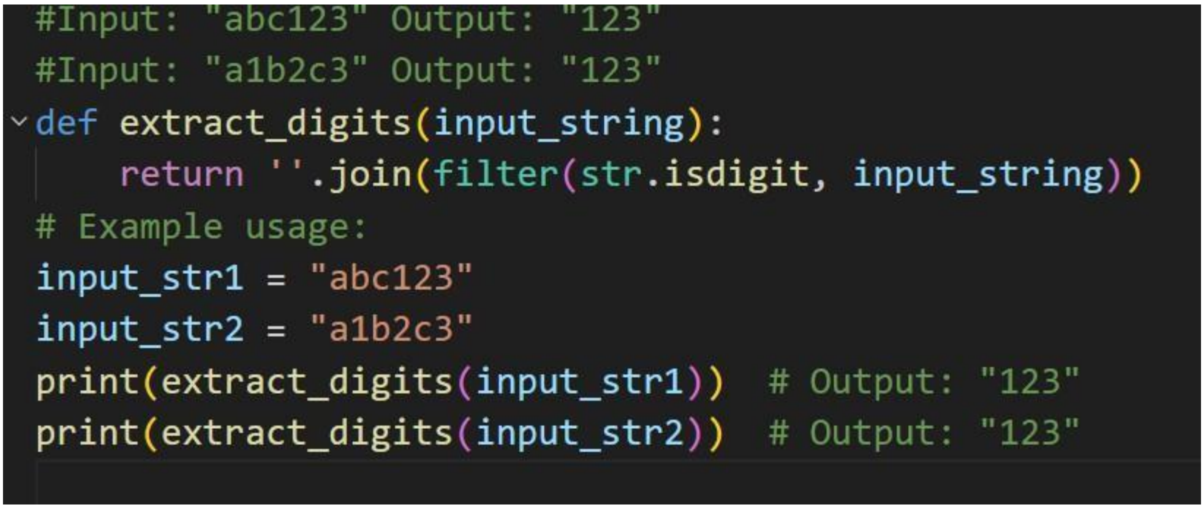
• Accurate function that returns only the digits from alphanumeric string

PROMPT:

Input: "abc123" Output: "123"

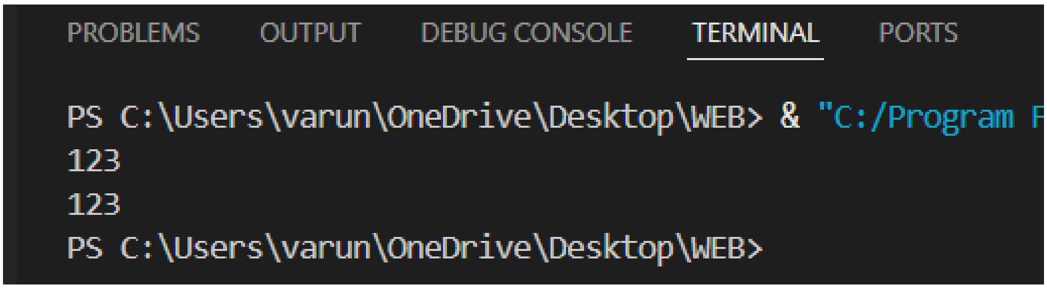
Input: "a1b2c3" Output: "123"

CODE:



```python
#Input: "abc123" Output: "123"
#Input: "a1b2c3" Output: "123"
def extract_digits(input_string):
    return ''.join(filter(str.isdigit, input_string))
# Example usage:
input_str1 = "abc123"
input_str2 = "a1b2c3"
print(extract_digits(input_str1))  # Output: "123"
print(extract_digits(input_str2))  # Output: "123"
```

OUTPUT:



```
PS C:\Users\varun\OneDrive\Desktop\WEB> & "C:/Program F
123
123
PS C:\Users\varun\OneDrive\Desktop\WEB>
```

EXPLANATION:

This function takes an alphanumeric string and extracts only the digits from it. It checks each character in the string, keeps only the numeric characters, and joins them together to form the output string.
For example, from "abc123" or "a1b2c3", it returns "123"

Task Description-4

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

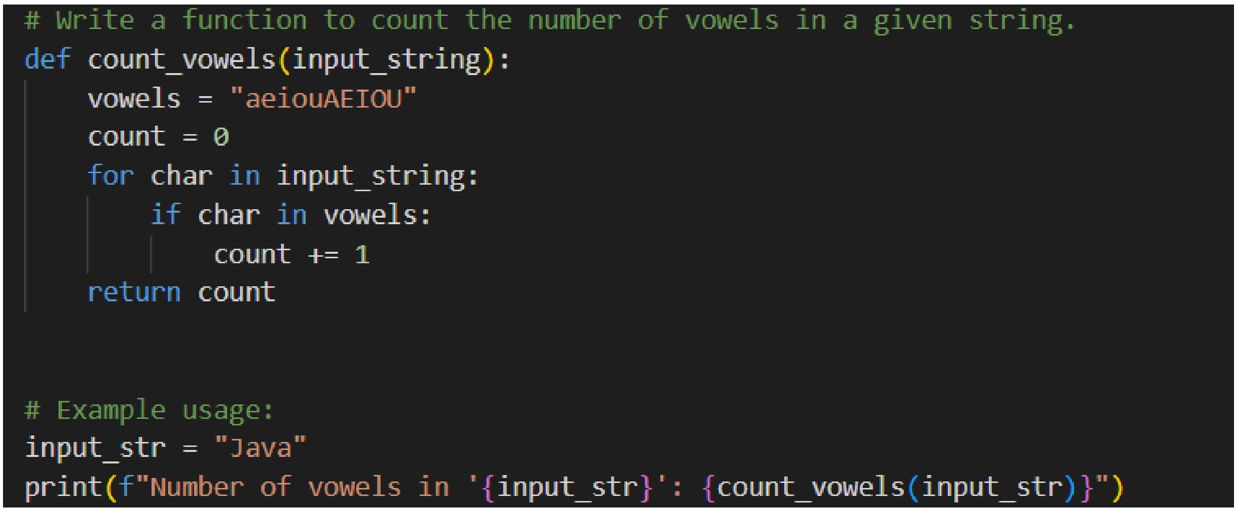
Expected Output-4

- Output comparison + student explanation on how examples helped the model.

PROMPT:

Write a function to count the number of vowels in a given string.

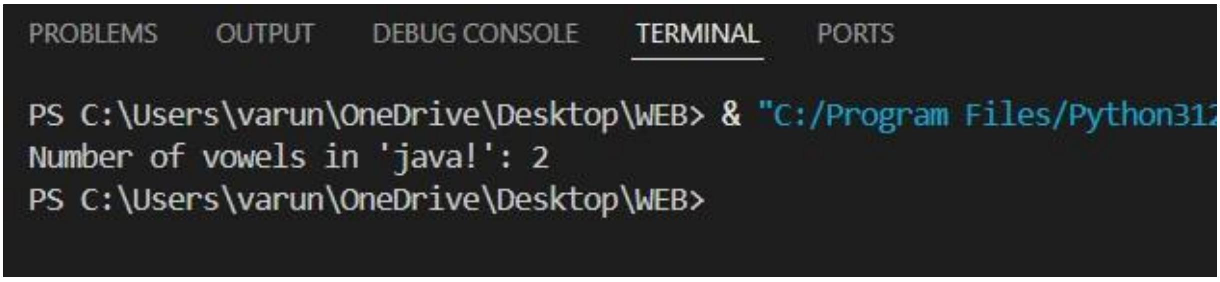
CODE:



```python
# Write a function to count the number of vowels in a given string.
def count_vowels(input_string):
    vowels = "aeiouAEIOU"
    count = 0
    for char in input_string:
        if char in vowels:
            count += 1
    return count


# Example usage:
input_str = "Java"
print(f"Number of vowels in '{input_str}': {count_vowels(input_str)}")
```

OUTPUT:

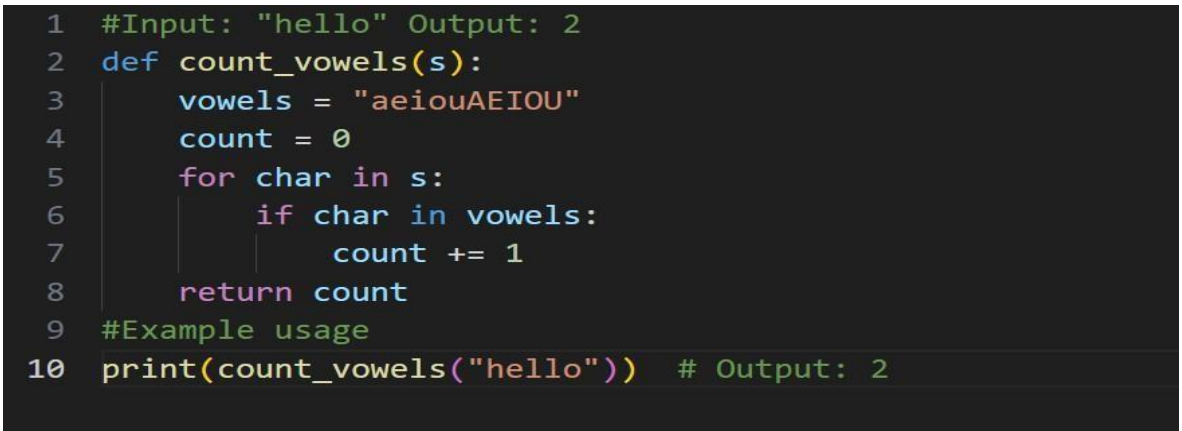


```
PS C:\Users\varun\OneDrive\Desktop\WEB> & "C:/Program Files/Python31
Number of vowels in 'java!': 2
PS C:\Users\varun\OneDrive\Desktop\WEB>
```

PROMPT:

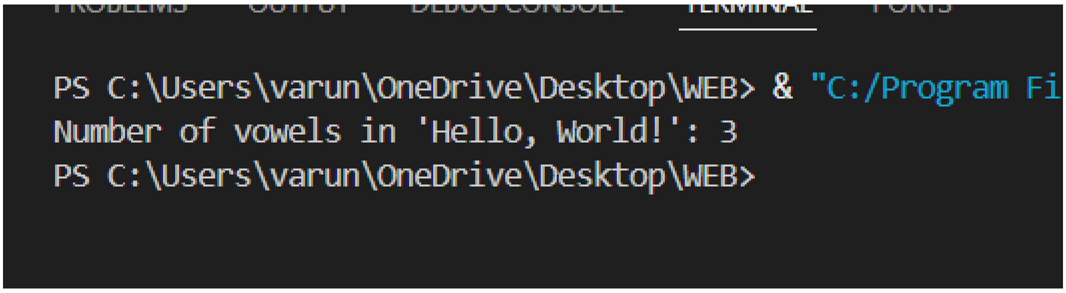
Example 1:
Input: "hello"
Output: 2

CODE:



```python
#Input: "hello" Output: 2
def count_vowels(s):
    vowels = "aeiouAEIOU"
    count = 0
    for char in s:
        if char in vowels:
            count += 1
    return count
#Example usage
print(count_vowels("hello"))  # Output: 2
```

OUTPUT:



```
PS C:\Users\varun\OneDrive\Desktop\WEB> & "C:/Program Fi
Number of vowels in 'Hello, World!': 3
PS C:\Users\varun\OneDrive\Desktop\WEB>
```

EXPLANATION:

In zero-shot prompting, the model works only with the instruction, so the answer may not be very clear.
In few-shot prompting, examples show what vowels are and what output is expected.
Because of these examples, the model understands better and gives a more accurate result.

Task Description-5

• Use few-shot prompting with 3 sample inputs to generate a function that determines

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

Expected Output-5

• A function that handles all cases with correct logic based on example patterns.

PROMPT:

Create a function that finds the minimum of three numbers without using the built-in min() function.

Input: a = 3, b = 7, c = 5

Output: 3

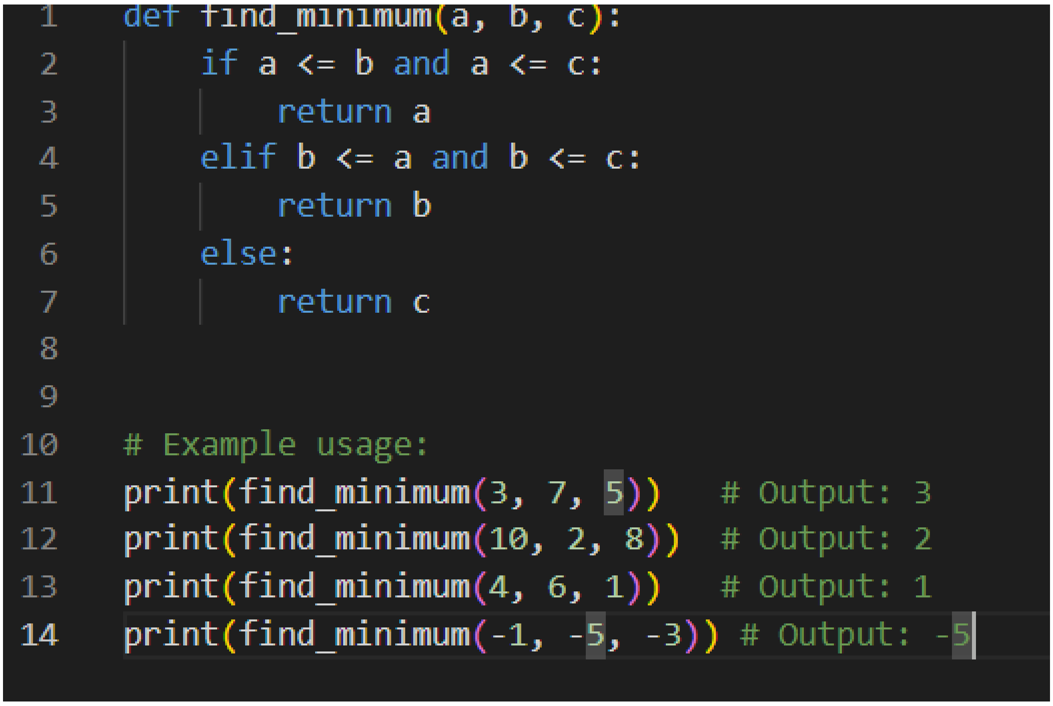
Input: a = 10, b = 2, c = 8

Output: 2

Input: a = 4, b = 6, c = 1

Output: 1

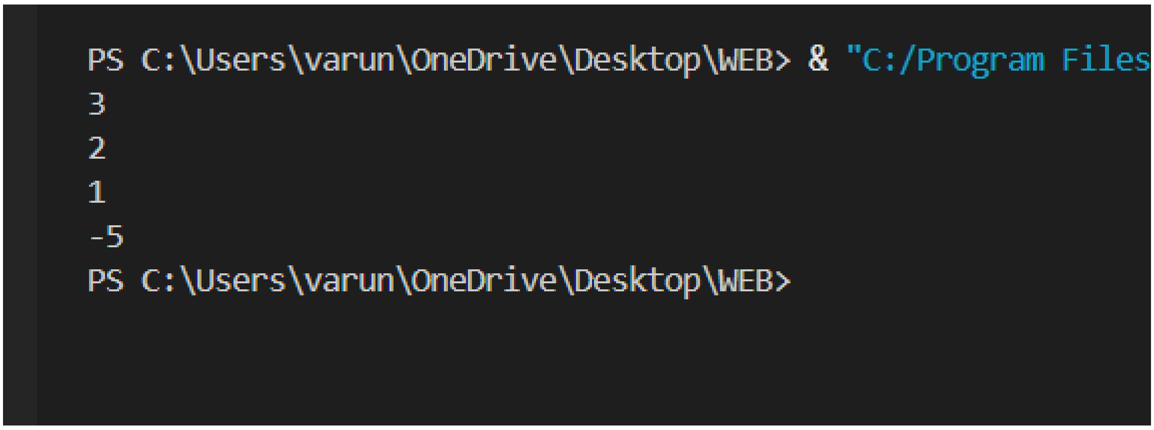
CODE:



```python
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:
print(find_minimum(3, 7, 5))   # Output: 3
print(find_minimum(10, 2, 8))  # Output: 2
print(find_minimum(4, 6, 1))   # Output: 1
print(find_minimum(-1, -5, -3)) # Output: -5
```

OUTPUT:



```
PS C:\Users\varun\OneDrive\Desktop\WEB> & "C:/Program Files
3
2
1
-5
PS C:\Users\varun\OneDrive\Desktop\WEB>
```

EXPLANATION:

In this task, few-shot prompting is used by giving example inputs and their outputs. These examples show how to compare three numbers and choose the smallest one without using the min() function.