

N.Sai Akash

2403a51|57

B-52

Lab 4

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

Task Description-1: Zero-shot Prompting

Prompt: Write a Python function to determine whether a given number is prime.

```
1  # Write a Python function to determine whether a given number is prime.
2  def is_prime(n):
3      if n <= 1:
4          return False
5      for i in range(2, int(n ** 0.5) + 1):
6          if n % i == 0:
7              return False
8      return True
9
10 print(is_prime(7))
11
12
13
```

OUTPUT:

The screenshot shows a Visual Studio Code (VS Code) interface. On the left, there are several icons: a file, a plus sign, a speech bubble, a lightning bolt, a user profile, a gear, and a search icon. The main area has a dark background. In the center, there is a code editor window with the following Python code:

```
15 ## Example: input: [1, 2, 3, 4], output: 10
16 ## Write a Python function to calculate the sum of elements in a list.
17 """def sum_list(lst):
18     total = 0
19     for num in lst:
20         total += num
21     return total
22 """
23
24 print(sum_list([1, 2, 3, 4]))
```

Below the code editor, there is a toolbar with buttons for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, AUGMENT NEXT EDIT, and a Python icon. To the right of the toolbar is a terminal window showing the following output:

```
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING & C:/Users/sarik/AppData/Local/Python/pythoncore-3.14.2/python.exe "c:/Users/sarik/OneDrive/Desktop/AI ASSISTED CODING/ASSIGN-4-2.py"
● True
○ PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING>
```

On the far right, there is a sidebar with a file icon, a plus sign, and the text "ASSIGN-4-2.py". Below it, there is a "Describe what to build" input field and some navigation icons. At the bottom, there is a taskbar with various icons and a status bar showing "LN 9, CUR 1", "SPACES 4", "CRLF", "Python 3.14.2", "Go Live", "BLACKBOXAI Open Chat", "Augment", and system information like "ENG IN", "20-01-2026", and "14:35".

Explanation:

1. Zero-shot prompting provides only instructions, no examples.
 2. The AI correctly implemented:

Prime definition logic

Square-root optimization

3. Demonstrates that simple logical problems work well with zero-shot prompts.

Task Description-2: One-shot Prompting

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

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

OUTPUT:

The screenshot shows a VS Code interface with the following details:

- File Explorer:** Shows files: ASSIGN-4-2.py, DAY-12.py, LAB-2.py, LAB(2)(PRACTICE SESSION), and Prime.py.
- Code Editor:** Displays Python code for summing a list:

```
total = 0
for num in lst:
    total += num
return total

print(sum_list([1, 2, 3, 4]))
```
- Terminal:** Shows an AI-assisted session:

```
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING & C:/Users/sarik/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/Users/sarik/OneDrive/Desktop/AI ASSISTED CODING/ASSIGN-4-2.py"
● 10
○ PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING>
```
- Bottom Status Bar:** Includes icons for file operations, a progress bar, and status text: Ln 34, Col 1, Spaces: 4, UTF-8, CRLF, Python 3.14.2, Go Live, BLACKBOXAI: Open Chat, and Augment.

Explanation:

1. One example clarifies the expected behavior.
 2. The AI correctly inferred:

Iteration over list

Accumulation of sum

3. The example helped remove ambiguity.

Task Description-3: Few-shot Prompting

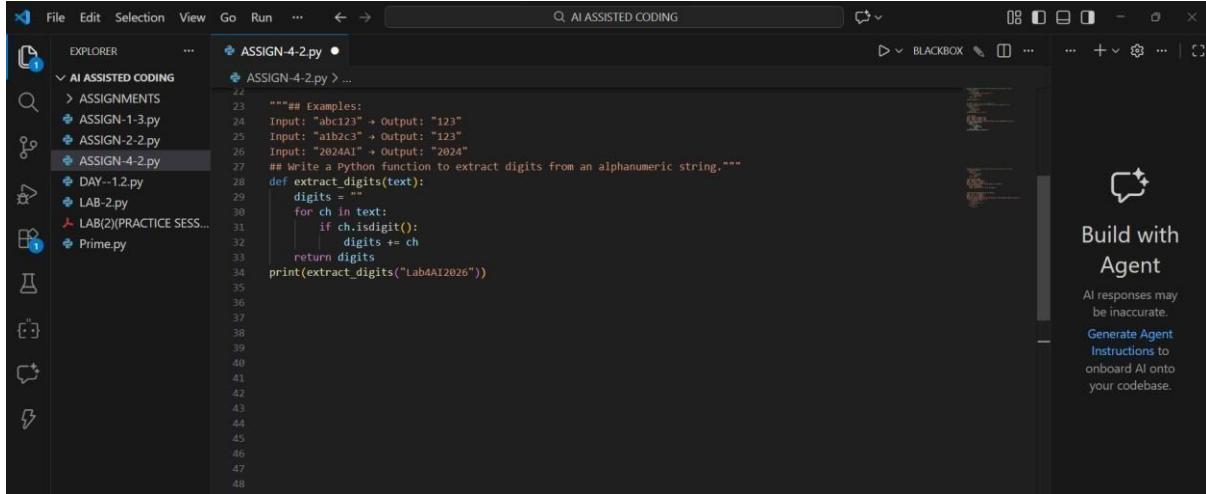
Prompt: Write a Python function to extract digits from an alphanumeric string.

Examples:

Input: "abc123" → Output: "123"

Input: "a1b2c3" → Output: "123"

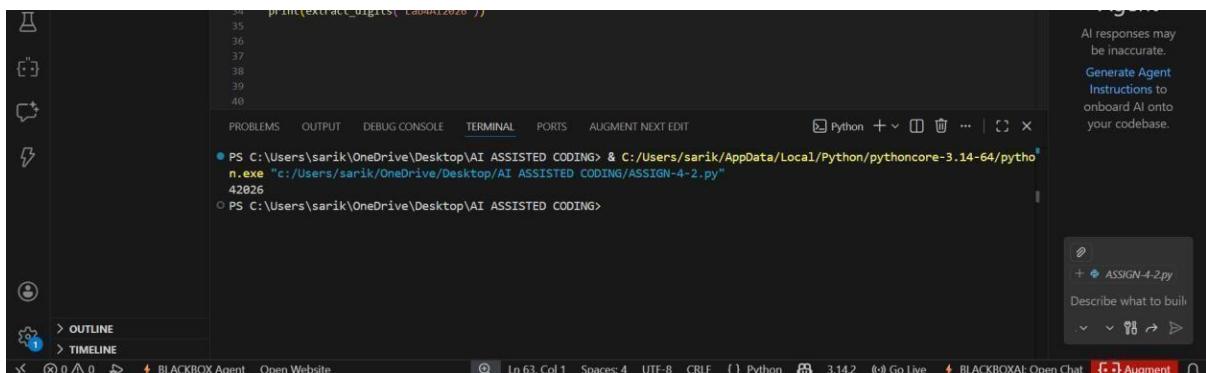
Input: "2024AI" → Output: "2024"



```
File Edit Selection View Go Run ... ← → Q AI ASSISTED CODING
EXPLORER ASSIGN-4-2.py
ASSIGN-4-2.py
    """# Examples:
    Input: "abc123" → Output: "123"
    Input: "a1b2c3" → Output: "123"
    Input: "2024AI" → Output: "2024"
    ## Write a Python function to extract digits from an alphanumeric string."""
    def extract_digits(text):
        digits = ""
        for ch in text:
            if ch.isdigit():
                digits += ch
        return digits
    print(extract_digits("Lab4AI2026"))

Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
```

OUTPUT:



```
print(extract_digits("Lab4AI2026"))
35
36
37
38
39
40
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AUGMENT NEXT EDIT Python + ...
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING> & C:/Users/sarik/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/Users/sarik/OneDrive/Desktop/AI ASSISTED CODING/ASSIGN-4-2.py"
42026
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING>
```

Explanation:

1. Few-shot prompting provides pattern recognition.

2. AI correctly:

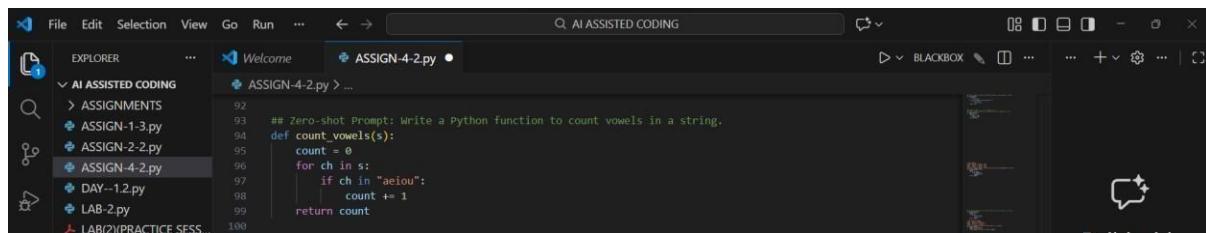
Identified digit extraction rule

Ignored alphabetic characters

3. Output accuracy improved due to multiple examples.

Task Description-4: Comparison Zero-shot vs Few-shot Prompting

Zero-shot Prompt: Write a Python function to count vowels in a string.



```
92     ## Zero-shot Prompt: Write a Python Function to count vowels in a string.
93     def count_vowels(s):
94         count = 0
95         for ch in s:
96             if ch in "aeiou":
97                 count += 1
98         return count
99
100
```

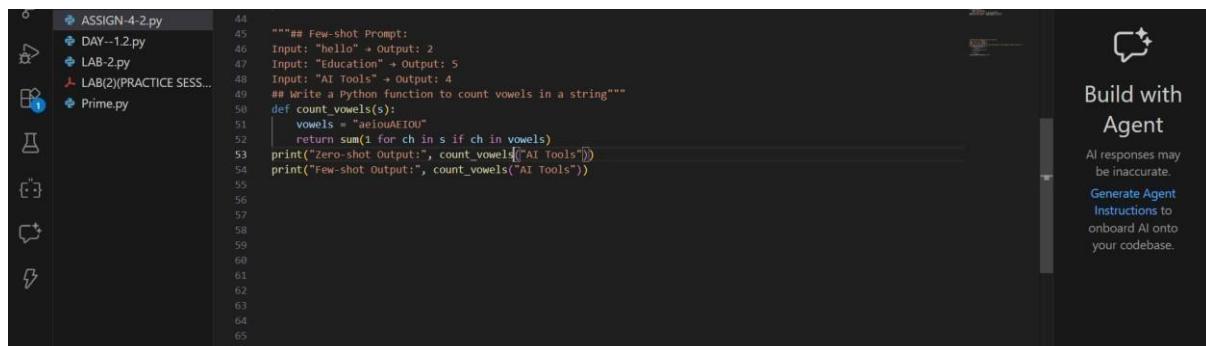
Few-shot Prompt: Write a Python function to count vowels in a string

Examples:

Input: "hello" → Output: 2

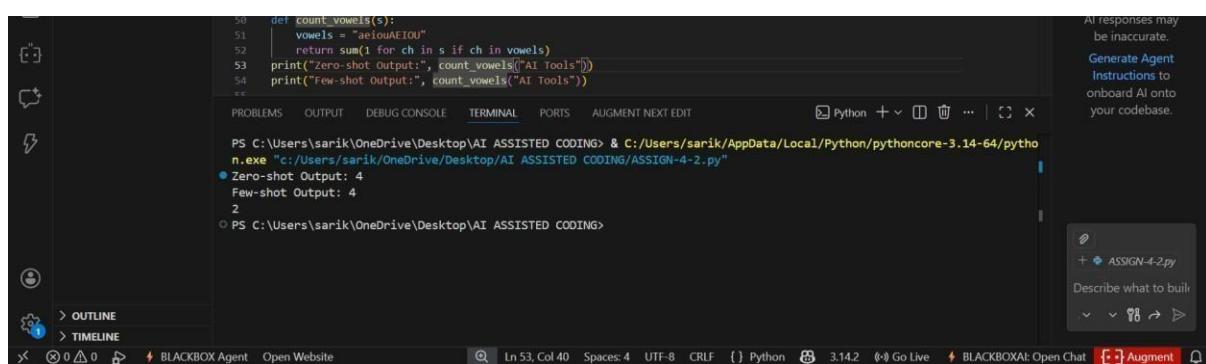
Input: "Education" → Output: 5

Input: "AI Tools" → Output: 4



```
44     """## Few-shot Prompt:
45     Input: "hello" -> Output: 2
46     Input: "Education" -> Output: 5
47     Input: "AI Tools" -> Output: 4
48     ## Write a Python function to count vowels in a string"""
49     def count_vowels(s):
50         vowels = "aeiouAEIOU"
51         return sum(1 for ch in s if ch in vowels)
52     print("Zero-shot Output:", count_vowels("AI Tools"))
53     print("Few-shot Output:", count_vowels("AI Tools"))
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

OUTPUT:



```
50     def count_vowels(s):
51         vowels = "aeiouAEIOU"
52         return sum(1 for ch in s if ch in vowels)
53     print("Zero-shot Output:", count_vowels("AI Tools"))
54     print("Few-shot Output:", count_vowels("AI Tools"))
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
89
90
91
92
93
94
95
96
97
98
99
100
```

PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING & C:/Users/sarik/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/Users/sarik/OneDrive/Desktop/AI ASSISTED CODING/ASSIGN-4-2.py"

Zero-shot Output: 4
Few-shot Output: 4
2

PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING>

Comparison Table:

Feature	Zero-shot	Few-shot
		Upper &

Case handling	Only lowercase	
	lowercase	
Accuracy	Moderate	High
Robustness	Basic	Improved
Readability	Simple	Optimized

Explanation:

1. Few-shot prompting improved the output by providing examples that showed:

Upper and lowercase handling

Realistic input patterns

This helped the AI generate a more accurate and generalized solution.

Task Description-5: Few-shot Prompting (No min() function)

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

Examples:

Input: (3, 5, 1) → Output: 1

Input: (10, 2, 7) → Output: 2

Input: (4, 4, 9) → Output: 4

The screenshot shows the VS Code interface with the following code in the editor:

```

57     """# Few-shot Prompting (No min() function)
58     Input: (3, 5, 1) → Output: 1
59     Input: (10, 2, 7) → Output: 2
60     Input: (4, 4, 9) → Output: 4
61     ## Write a Python function to find the minimum of three numbers without using min()."""
62     def minimum_of_three(a, b, c):
63         if a <= b and a <= c:
64             return a
65         elif b <= a and b <= c:
66             return b
67         else:
68             return c
69
70 print(minimum_of_three(6, 2, 9))

```

The code is part of a file named `ASSIGN-4-2.py`. The code defines a function `minimum_of_three` that takes three arguments and returns the smallest one. It uses a series of if-elif-else statements to compare the values. The code is preceded by a multi-line comment containing three examples of inputs and their corresponding outputs.

OUTPUT:

The screenshot shows the Microsoft Visual Studio Code (VS Code) interface. On the left is the sidebar with icons for file operations, search, and navigation. The main area is a code editor with the following Python code:

```
67     else:
68         return c
69     print(minimum_of_three(6, 2, 9))
70 
```

Below the code editor are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, and AUGMENT NEXT EDIT. The TERMINAL tab is active, showing a Windows PowerShell window (PS) with the command "python.exe "c:/Users/sarik/OneDrive/Desktop/AI ASSISTED CODING/ASSIGN-4-2.py"" and its output "2".

The status bar at the bottom shows various settings: BLACKBOX Agent, Open Website, Ln 43, Col 17 (122 selected), Spaces: 4, UTF-8, CRLF, Python, 3.14.2, Go Live, BLACKBOXAII: Open Chat, Augment, and a profile icon.

A right-hand sidebar titled "Agent" contains the text: "AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase." Below this is a small input field with placeholder text "Describe what to build" and a "Describe what to build" button.

Explanation:

1. Few-shot examples guided logical comparisons.
2. Handles: Equal values

All ordering cases

3. Does not use built-in min() as instructed.