

ASSIGNMENT-4.2

NAME-ARSHA VARDHINI

ROLL_NO:2303A51600

TASK-1: ZERO-SHOT PROMPTING

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

CODE:

The screenshot shows a code editor interface with multiple tabs at the top: 'ODE FOR EVEN ODD NUMBERS.py', 'AI.py', 'ASS-1.5.py', 'As-2.5.py', 'Assignment-3.1.py', and 'Assignment-4.2.py'. The 'Assignment-4.2.py' tab is active. The code in the editor is as follows:

```
#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
# Print statements
print(is_prime(7))
print(is_prime(10))
```

Below the code editor is a terminal window showing the output of the script:

```
/Python/Python314/python.exe" "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitant coding/Assignment-4.2.py"
True
False
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitant coding>
```

The terminal also displays the path and file name of the script being run.

OBSERVATION:

- AI model understands the concept of a prime number without being given any examples or additional guidance
- It applies correct mathematical reasoning purely from the instruction
- The model generates syntactically correct and logically sound Python code

TASK-2

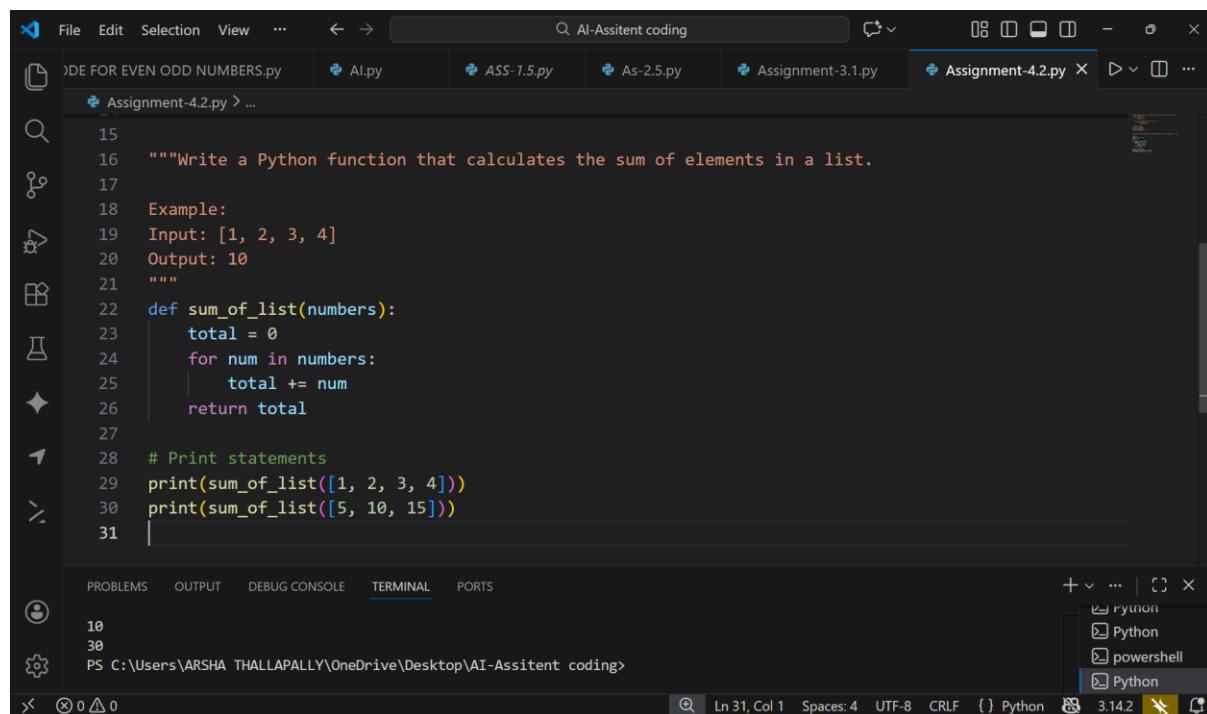
PROMPT: Write a Python function that calculates the sum of elements in a list.

Example:

Input: [1, 2, 3, 4]

Output: 10

CODE:



The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows files like "CODE FOR EVEN ODD NUMBERS.py", "AI.py", "ASS-1.5.py", "As-2.5.py", "Assignment-3.1.py", and "Assignment-4.2.py".
- Code Editor:** Displays the following Python code:

```
15
16     """Write a Python function that calculates the sum of elements in a list.
17
18 Example:
19 Input: [1, 2, 3, 4]
20 Output: 10
21 """
22 def sum_of_list(numbers):
23     total = 0
24     for num in numbers:
25         total += num
26     return total
27
28 # Print statements
29 print(sum_of_list([1, 2, 3, 4]))
30 print(sum_of_list([5, 10, 15]))
```
- Terminal:** Shows the output of the code execution:

```
10
30
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitant coding>
```
- Status Bar:** Shows "Ln 31, Col 1" and "Spaces: 4" and "UTF-8" and "CRLF".

OBSERVATION:

The single example guides the AI model to understand the expected input and output relationship
The model correctly generalizes the pattern from the example to any list of numbers

TASK-3

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

Examples:

Input: "a1b2c3"

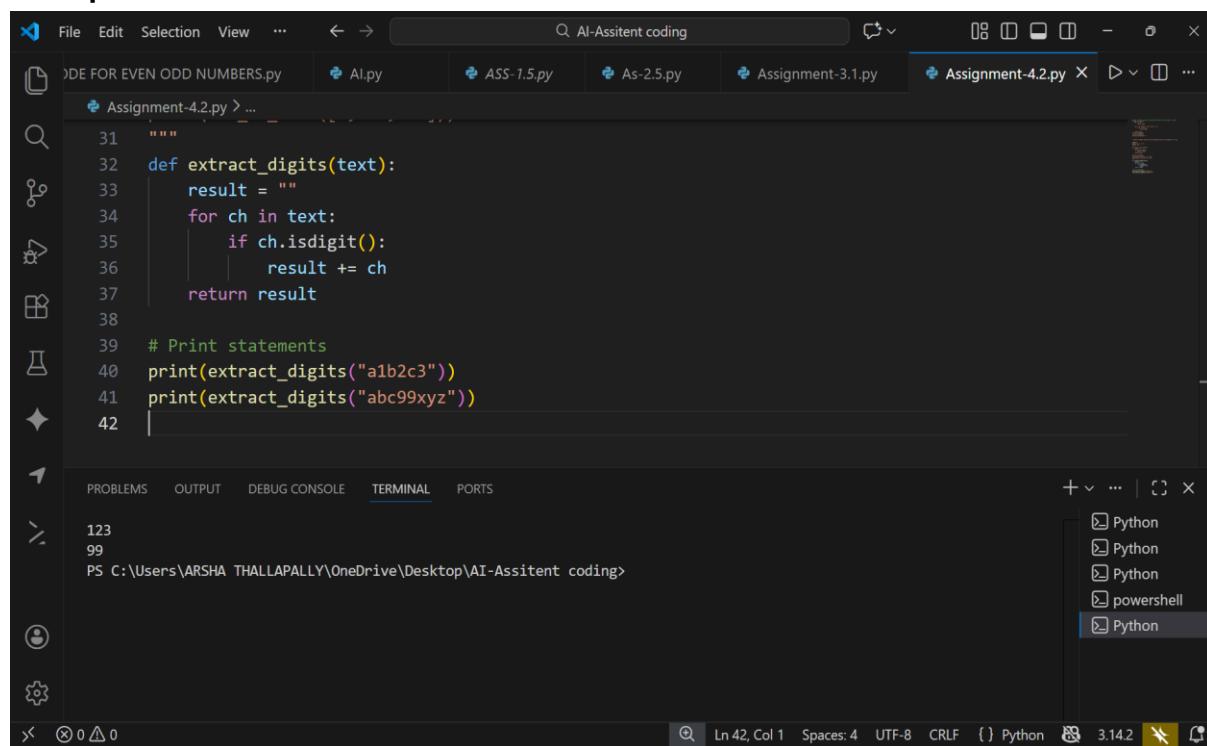
Output: "123"

Input: "x9y8z7"

Output: "987"

Input: "abc123def"

Output: "123"



The screenshot shows a code editor interface with a dark theme. The top bar includes 'File', 'Edit', 'Selection', 'View', and other standard menu items. A search bar at the top right contains the text 'AI-Assitant coding'. Below the search bar is a tab bar with several files: 'CODE FOR EVEN ODD NUMBERS.py', 'AI.py', 'ASS-1.5.py', 'As-2.5.py', 'Assignment-3.1.py', and 'Assignment-4.2.py' (which is currently selected). The main workspace displays the following Python code:

```
31 """"
32 def extract_digits(text):
33     result = ""
34     for ch in text:
35         if ch.isdigit():
36             result += ch
37     return result
38
39 # Print statements
40 print(extract_digits("a1b2c3"))
41 print(extract_digits("abc99xyz"))
42 |
```

Below the code editor is a 'TERMINAL' panel showing the output of the code execution:

```
123
99
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitant coding>
```

The bottom status bar indicates the current file is 'Assignment-4.2.py', the line is 'Ln 42, Col 1', and the column is 'Spaces: 4'. It also shows the Python version as '3.14.2' and other system information.

OBSERVATION:

- Multiple examples help the AI model clearly identify the pattern to be learned
- The model focuses only on digit characters and ignores alphabetic content
- The AI demonstrates improved confidence and reduced ambiguity compared to zero shot and one shot prompting

TASK-4

PROMPT: ZERO-SHOT: Write a Python function that counts the number of vowels in a string.

FEW-SHOT: Write a Python function that counts the number of vowels in a string.

Examples:

Input: "hello"

Output: 2

Input: "AEIOU"

Output: 5

Input: "chatgpt"

Output: 2

CODE:

ZERO-SHOT:

A screenshot of the Visual Studio Code interface. The top bar shows the title "AI-Assitant coding". The left sidebar has icons for file operations like Open, Save, Find, and Delete. The main editor area contains the following Python code:

```
Assignment-4.2.py > ...
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
6     for i in range(2, int(n ** 0.5) + 1):
7         if n % i == 0:
8             return False
9
10    return True
11 # Print statements.
12 print(is_prime(7))
13 print(is_prime(10))
```

The bottom status bar shows the file path "Python/Python314/python.exe" and the command "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitant coding/Assignment-4.2.py", along with terminal output "True" and "False". The bottom right corner shows the version "3.14.2".

FEW-SHOT:

A screenshot of the Visual Studio Code interface. The top bar shows the title "AI-Assitant coding". The left sidebar has icons for file operations like Open, Save, Find, and Delete. The main editor area contains the following Python code:

```
Assignment-4.2.py > ...
31 """
32 def extract_digits(text):
33     result = ""
34     for ch in text:
35         if ch.isdigit():
36             result += ch
37     return result
38
39 # Print statements
40 print(extract_digits("a1b2c3"))
41 print(extract_digits("abc99xyz"))
42 |
```

The bottom status bar shows the file path "Python/Python314/python.exe" and the command "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitant coding", along with terminal output "123" and "99". The bottom right corner shows the version "3.14.2".

`def count_vowels(text):`

`vowels = "aeiouAEIOU"`

```
count = 0  
for ch in text:  
    if ch in vowels:  
        count += 1  
return count
```

OBSERVATION:

FEW-SHOT OBSERVATION

The provided examples clearly define what characters should be counted as vowels

The model confidently includes both uppercase and lowercase vowels due to examples

ZERO SHOT:

zero shot prompting the AI guesses the intent based on general knowledge which may vary for ambiguous tasks

TASK-5

PROMPT:

Write a Python function that determines the minimum of three numbers without using the built-in min() function.

Examples:

Input: 3, 7, 5

Output: 3

Input: 10, 2, 8

Output: 2

Input: 4, 4, 9

Output: 4

CODE:

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

- File Explorer:** Shows files like "CODE FOR EVEN ODD NUMBERS.py", "AI.py", "ASS-1.5.py", "As-2.5.py", "Assignment-3.1.py", and "Assignment-4.2.py".
- Code Editor:** Displays the following Python code:

```
42
43
44 def find_min(a, b, c):
45     if a <= b and a <= c:
46         return a
47     elif b <= a and b <= c:
48         return b
49     else:
50         return c
51
52 # Print statements
53 print(find_min(3, 7, 5))
54 print(find_min(10, 2, 8))
55 print(find_min(4, 4, 9))
```
- Terminal:** Shows the output of running the script:

```
PS C:\Users\ARSHA TH> & "C:/Users/ARSHA THALLAPALLY/AppData/Local/Programs/Python/Python314/python.exe" "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitant coding/Assignment-4.2.py"
3
2
4
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitant coding>
```
- Bottom Status Bar:** Shows "Ln 56, Col 1" and "Python 3.14.2".

OBSERVATION:

The examples clearly establish the comparison pattern needed to identify the smallest value

The AI model infers the requirement to handle equality cases correctly

Conditional logic is generated without relying on built-in functions

