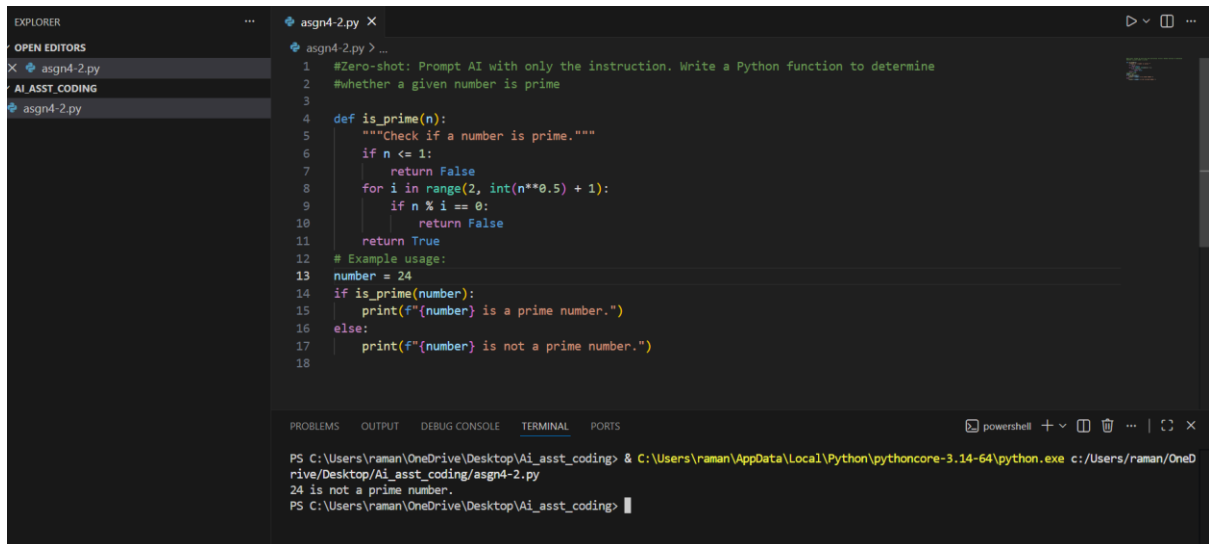


# AI Assisted Coding

## Assignment - 4.2

### Task – 1

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



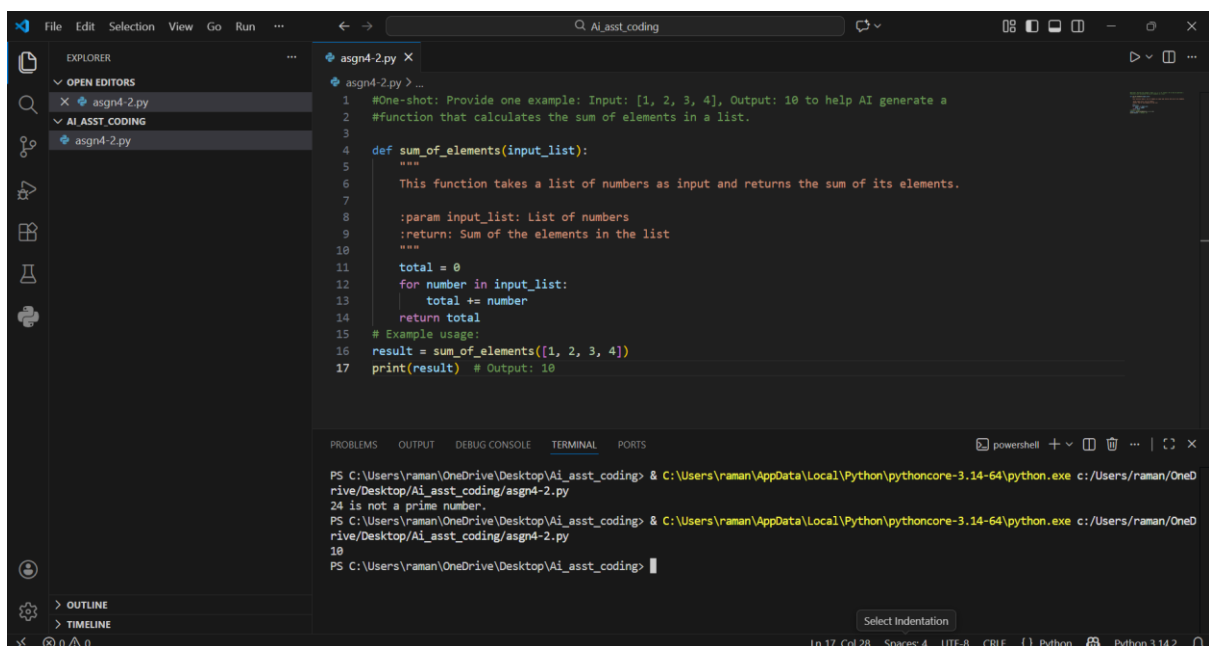
The screenshot shows a Visual Studio Code editor with a file named `asn4-2.py`. The code defines a function `is_prime(n)` that checks if a number is prime. It includes a docstring, a loop to check divisibility, and an example usage. The terminal shows the command to run the script and the output indicating that 24 is not a prime number.

```
1 #Zero-shot: Prompt AI with only the instruction. Write a Python function to determine
2 #whether a given number is prime
3
4 def is_prime(n):
5     """Check if a number is prime."""
6     if n <= 1:
7         return False
8     for i in range(2, int(n**0.5) + 1):
9         if n % i == 0:
10            return False
11    return True
12
13 # Example usage:
14 number = 24
15 if is_prime(number):
16     print(f"{number} is a prime number.")
17 else:
18     print(f"{number} is not a prime number.")
```

```
PS C:\Users\raman\OneDrive\Desktop\Ai_asst_coding> & C:\Users\raman\AppData\Local\Python\pythoncore-3.14-64\python.exe c:/Users/raman/OneDrive/Desktop/Ai_asst_coding/asn4-2.py
24 is not a prime number.
PS C:\Users\raman\OneDrive\Desktop\Ai_asst_coding>
```

### Task - 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.



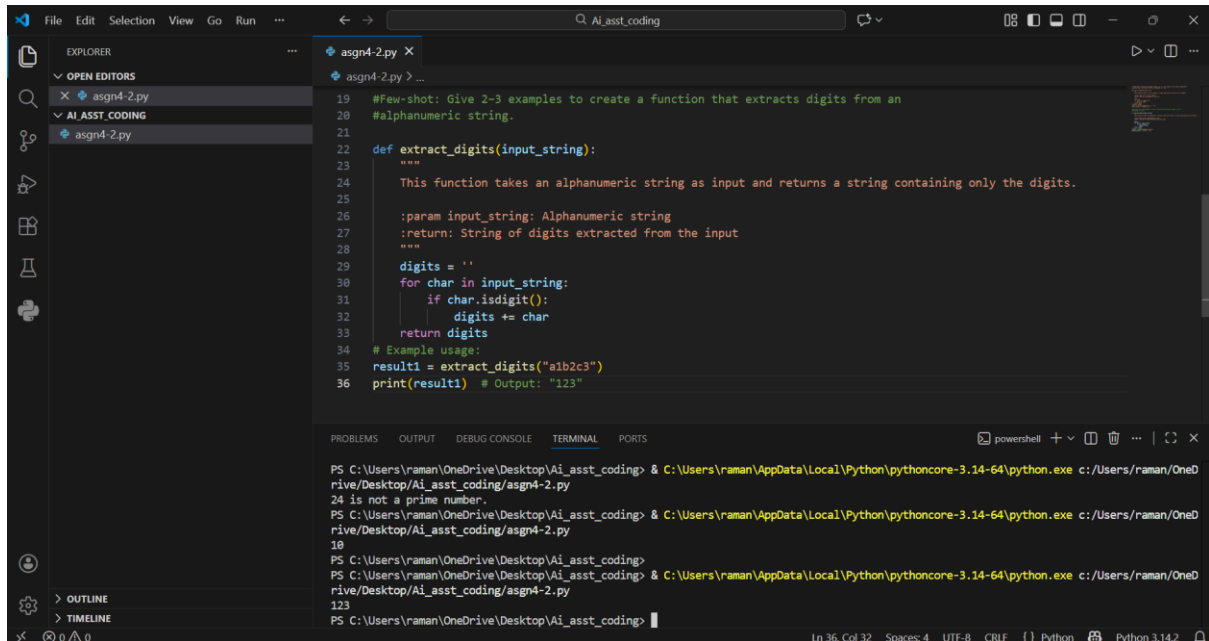
The screenshot shows a Visual Studio Code editor with a file named `asn4-2.py`. The code defines a function `sum_of_elements(input_list)` that calculates the sum of elements in a list. It includes a docstring, a loop to calculate the sum, and an example usage. The terminal shows the command to run the script and the output indicating that the sum of [1, 2, 3, 4] is 10.

```
1 #One-shot: Provide one example: Input: [1, 2, 3, 4], Output: 10 to help AI generate a
2 #function that calculates the sum of elements in a list.
3
4 def sum_of_elements(input_list):
5     """
6     This function takes a list of numbers as input and returns the sum of its elements.
7
8     :param input_list: List of numbers
9     :return: Sum of the elements in the list
10    """
11    total = 0
12    for number in input_list:
13        total += number
14    return total
15
16 # Example usage:
17 result = sum_of_elements([1, 2, 3, 4])
18 print(result) # Output: 10
```

```
PS C:\Users\raman\OneDrive\Desktop\Ai_asst_coding> & C:\Users\raman\AppData\Local\Python\pythoncore-3.14-64\python.exe c:/Users/raman/OneDrive/Desktop/Ai_asst_coding/asn4-2.py
10
PS C:\Users\raman\OneDrive\Desktop\Ai_asst_coding>
```

## Task – 3

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



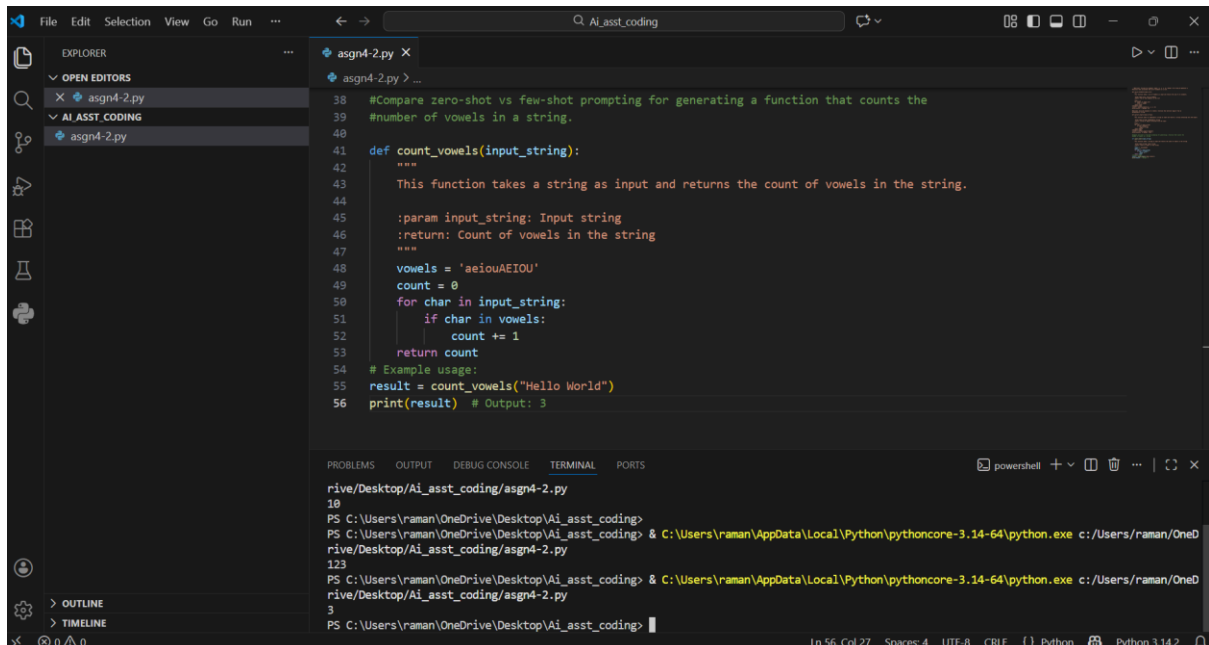
The screenshot shows a Visual Studio Code editor with a file named `asn4-2.py` open. The code defines a function `extract_digits` that takes an alphanumeric string as input and returns a string containing only the digits. The function uses a loop to iterate over each character in the input string and checks if it is a digit using `char.isdigit()`. If it is a digit, it is added to the `digits` string. The function is tested with the input `"a1b2c3"`, which returns `"123"`.

```
19 #Few-shot: Give 2-3 examples to create a function that extracts digits from an
20 #alphanumeric string.
21
22 def extract_digits(input_string):
23     """
24     This function takes an alphanumeric string as input and returns a string containing only the digits.
25
26     :param input_string: Alphanumeric string
27     :return: String of digits extracted from the input
28     """
29     digits = ''
30     for char in input_string:
31         if char.isdigit():
32             digits += char
33     return digits
34 # Example usage:
35 result1 = extract_digits("a1b2c3")
36 print(result1) # Output: "123"
```

The terminal output shows the execution of the script, confirming the output is `123`.

## Task – 4

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



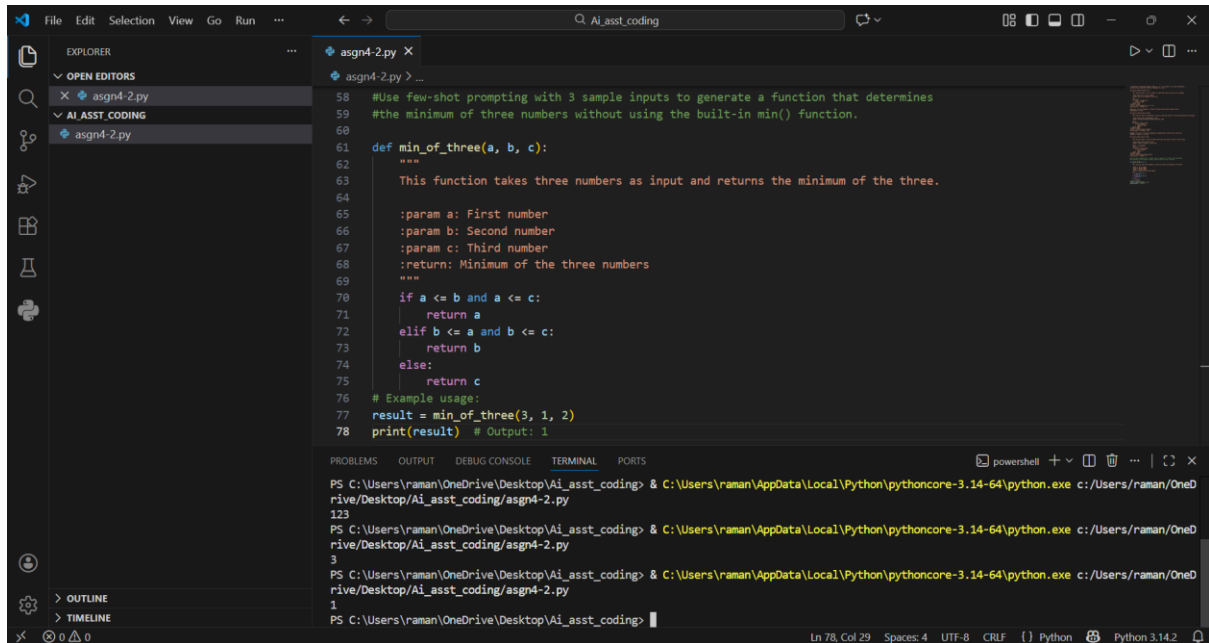
The screenshot shows a Visual Studio Code editor with a file named `asn4-2.py` open. The code defines a function `count_vowels` that takes a string as input and returns the count of vowels in the string. The function uses a loop to iterate over each character in the input string and checks if it is a vowel by comparing it to a set of vowels `'aeiouAEIOU'`. If it is a vowel, the count is incremented by 1. The function is tested with the input `"Hello World"`, which returns `3`.

```
38 #Compare zero-shot vs few-shot prompting for generating a function that counts the
39 #number of vowels in a string.
40
41 def count_vowels(input_string):
42     """
43     This function takes a string as input and returns the count of vowels in the string.
44
45     :param input_string: Input string
46     :return: Count of vowels in the string
47     """
48     vowels = 'aeiouAEIOU'
49     count = 0
50     for char in input_string:
51         if char in vowels:
52             count += 1
53     return count
54 # Example usage:
55 result = count_vowels("Hello World")
56 print(result) # Output: 3
```

The terminal output shows the execution of the script, confirming the output is `3`.

## Task – 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.



The screenshot shows a Visual Studio Code editor window with a file named 'AI\_asst\_coding'. The Explorer sidebar on the left shows the file structure with 'AI\_ASST\_CODING' and 'asn4-2.py'. The main editor area displays the content of 'asn4-2.py', which includes a few-shot prompt and a Python function. The terminal at the bottom shows the command to run the script and its output.

```
58 #Use few-shot prompting with 3 sample inputs to generate a function that determines
59 #the minimum of three numbers without using the built-in min() function.
60
61 def min_of_three(a, b, c):
62     """
63     This function takes three numbers as input and returns the minimum of the three.
64
65     :param a: First number
66     :param b: Second number
67     :param c: Third number
68     :return: Minimum of the three numbers
69     """
70     if a <= b and a <= c:
71         return a
72     elif b <= a and b <= c:
73         return b
74     else:
75         return c
76
77 # Example usage:
78 result = min_of_three(3, 1, 2)
79 print(result) # Output: 1
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\raman\OneDrive\Desktop\AI_asst_coding> & C:\Users\raman\AppData\Local\Python\pythoncore-3.14-64\python.exe c:/Users/raman/OneD
rive/Desktop/Ai_asst_coding/asn4-2.py
123
PS C:\Users\raman\OneDrive\Desktop\AI_asst_coding> & C:\Users\raman\AppData\Local\Python\pythoncore-3.14-64\python.exe c:/Users/raman/OneD
rive/Desktop/Ai_asst_coding/asn4-2.py
3
PS C:\Users\raman\OneDrive\Desktop\AI_asst_coding> & C:\Users\raman\AppData\Local\Python\pythoncore-3.14-64\python.exe c:/Users/raman/OneD
rive/Desktop/Ai_asst_coding/asn4-2.py
1
PS C:\Users\raman\OneDrive\Desktop\AI_asst_coding>
```

Ln 78, Col 29 Spaces: 4 UTF-8 CRLF {} Python Python 3.14.2

NIHARIKA PATHAK

2303A51955(B-15)