

ASSIGNMENT – 3.1

2303A51850

B-13

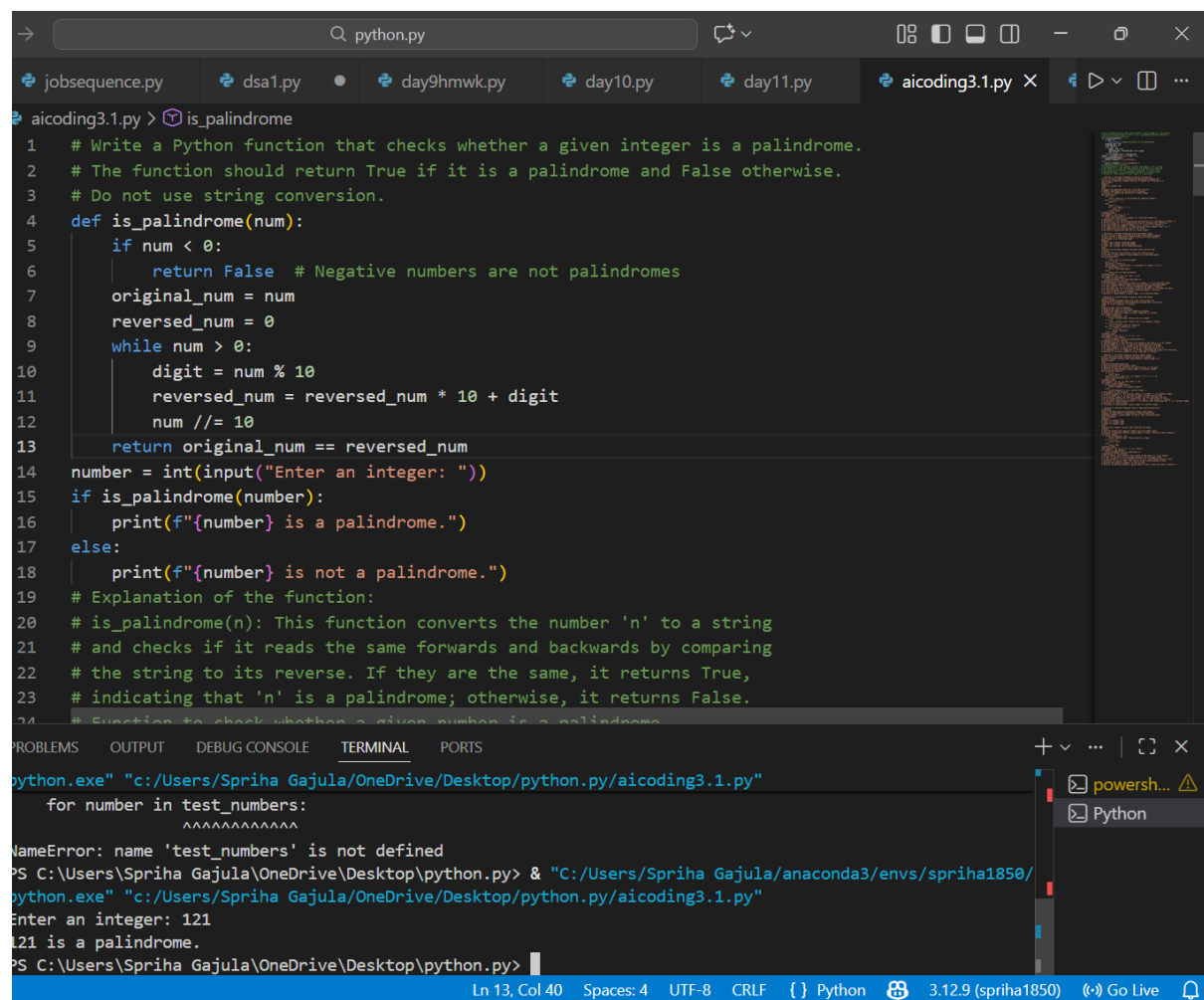
Question 1: Zero-Shot Prompting (Palindrome Number Program)

Write a zero-shot prompt (without providing any examples) to generate a Python function that checks whether a given number is a palindrome.

Task:

- Record the AI-generated code.
- Test the code with multiple inputs.
- Identify any logical errors or missing edge-case handling.

CODE:



```
python.py
jobsequence.py dsa1.py day9hmkw.py day10.py day11.py aicoding3.1.py X
aicoding3.1.py > is_palindrome
1 # Write a Python function that checks whether a given integer is a palindrome.
2 # The function should return True if it is a palindrome and False otherwise.
3 # Do not use string conversion.
4 def is_palindrome(num):
5     if num < 0:
6         return False # Negative numbers are not palindromes
7     original_num = num
8     reversed_num = 0
9     while num > 0:
10         digit = num % 10
11         reversed_num = reversed_num * 10 + digit
12         num //= 10
13     return original_num == reversed_num
14 number = int(input("Enter an integer: "))
15 if is_palindrome(number):
16     print(f"{number} is a palindrome.")
17 else:
18     print(f"{number} is not a palindrome.")
19 # Explanation of the function:
20 # is_palindrome(n): This function converts the number 'n' to a string
21 # and checks if it reads the same forwards and backwards by comparing
22 # the string to its reverse. If they are the same, it returns True,
23 # indicating that 'n' is a palindrome; otherwise, it returns False.
24 # Function to check whether a given number is a palindrome

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
python.exe "c:/Users/Spriha Gajula/OneDrive/Desktop/python.py/aicoding3.1.py"
for number in test_numbers:
    ^^^^^^^^^^^^^
NameError: name 'test_numbers' is not defined
PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py> & "C:/Users/Spriha Gajula/anaconda3/envs/spriha1850/python.exe" "c:/Users/Spriha Gajula/OneDrive/Desktop/python.py/aicoding3.1.py"
Enter an integer: 121
121 is a palindrome.
PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py>
```

Explanation:

This function converts the number 'n' to a string and checks if it reads the same forwards and backwards by comparing the string to its reverse. If they are the same, it returns True, indicating that 'n' is a palindrome; otherwise, it returns False. Function to check whether a given number is a palindrome.

Question 2: One-Shot Prompting (Factorial Calculation)

Write a one-shot prompt by providing one input-output example and ask the AI to generate a Python function to compute the factorial of a given number.

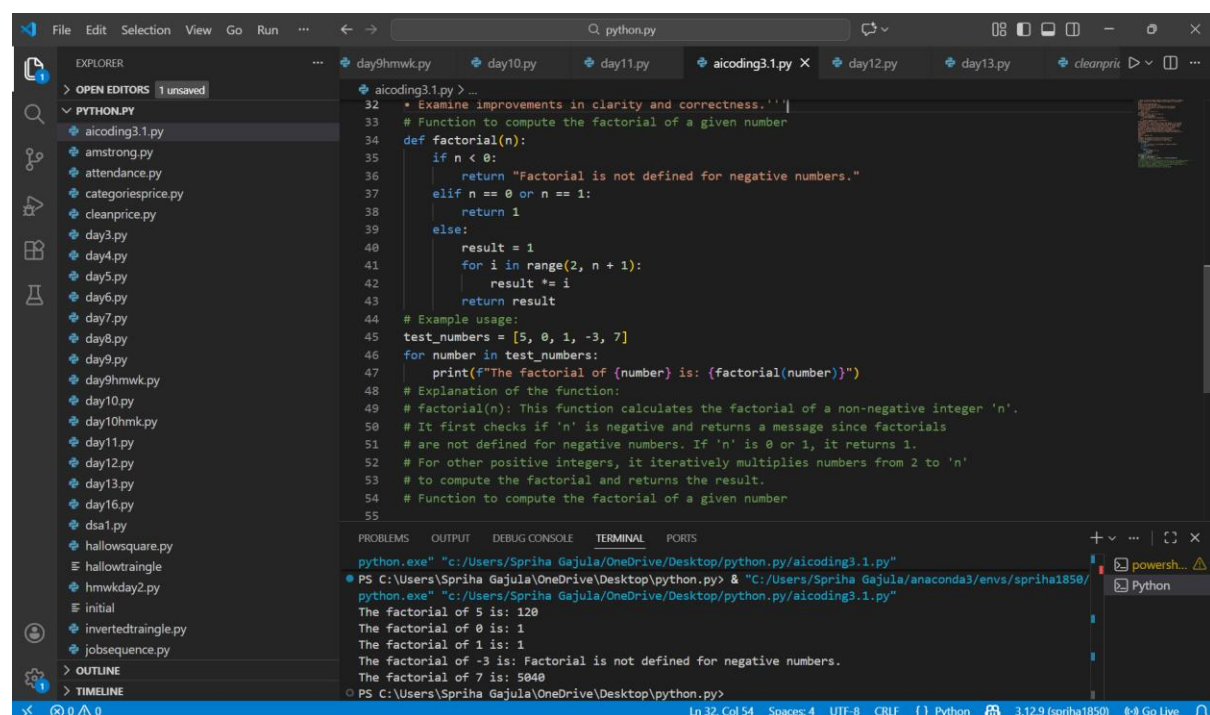
Example:

Input: 5 → Output: 120

Task:

- Compare the generated code with a zero-shot solution.
- Examine improvements in clarity and correctness.

One-shot code:



```
32 * Examine improvements in clarity and correctness.'']
33 # Function to compute the factorial of a given number
34 def factorial(n):
35     if n < 0:
36         return "Factorial is not defined for negative numbers."
37     elif n == 0 or n == 1:
38         return 1
39     else:
40         result = 1
41         for i in range(2, n + 1):
42             result *= i
43         return result
44 # Example usage:
45 test_numbers = [5, 0, 1, -3, 7]
46 for number in test_numbers:
47     print(f"The factorial of {number} is: {factorial(number)}")
48 # Explanation of the function:
49 # factorial(n): This function calculates the factorial of a non-negative integer 'n'.
50 # It first checks if 'n' is negative and returns a message since factorials
51 # are not defined for negative numbers. If 'n' is 0 or 1, it returns 1.
52 # For other positive integers, it iteratively multiplies numbers from 2 to 'n'
53 # to compute the factorial and returns the result.
54 # Function to compute the factorial of a given number
55
```

python.exe "C:/Users/Spriha Gajula/OneDrive/Desktop/python.py/aiencoding3.1.py"

PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py> & "C:/Users/Spriha Gajula/anaconda3/envs/spriha1850/python.exe" "C:/Users/Spriha Gajula/OneDrive/Desktop/python.py/aiencoding3.1.py"

The factorial of 5 is: 120
The factorial of 0 is: 1
The factorial of 1 is: 1
The factorial of -3 is: Factorial is not defined for negative numbers.
The factorial of 7 is: 5040

PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py>

Explanation:

factorial(n): This function calculates the factorial of a non-negative integer 'n'.

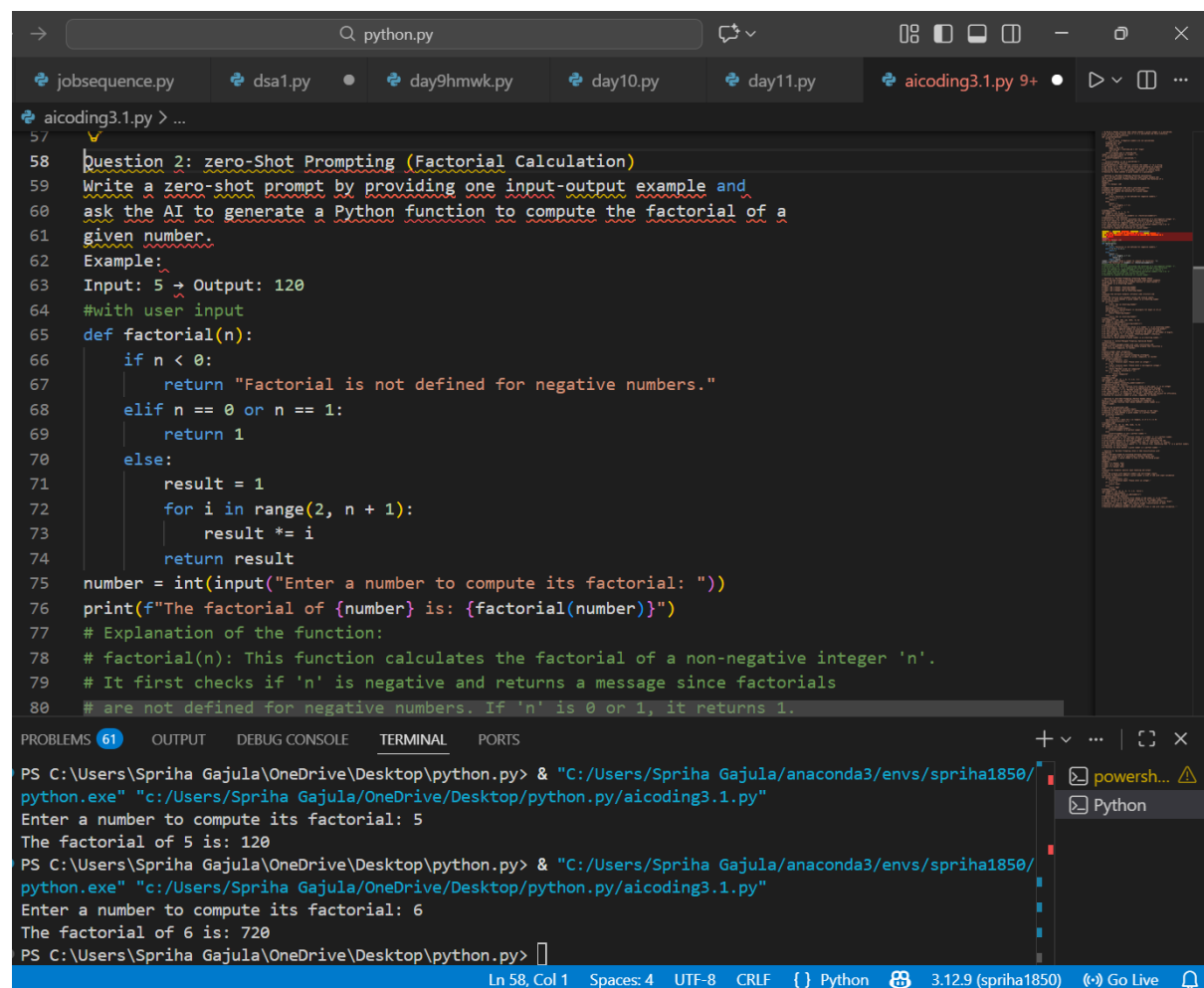
It first checks if 'n' is negative and returns a message since factorials

are not defined for negative numbers. If 'n' is 0 or 1, it returns 1.

For other positive integers, it iteratively multiplies numbers from 2 to 'n' to compute the factorial and returns the result.

Function to compute the factorial of a given number

Zero-shot code:



```
python.py
jobsequence.py dsa1.py day9hwmk.py day10.py day11.py aicoding3.1.py 9+
aicoding3.1.py > ...
57
58 Question 2: zero-Shot Prompting (Factorial Calculation)
59 Write a zero-shot prompt by providing one input-output example and
60 ask the AI to generate a Python function to compute the factorial of a
61 given number.
62 Example:
63 Input: 5 → Output: 120
64 #with user input
65 def factorial(n):
66     if n < 0:
67         return "Factorial is not defined for negative numbers."
68     elif n == 0 or n == 1:
69         return 1
70     else:
71         result = 1
72         for i in range(2, n + 1):
73             result *= i
74         return result
75 number = int(input("Enter a number to compute its factorial: "))
76 print(f"The factorial of {number} is: {factorial(number)}")
77 # Explanation of the function:
78 # factorial(n): This function calculates the factorial of a non-negative integer 'n'.
79 # It first checks if 'n' is negative and returns a message since factorials
80 # are not defined for negative numbers. If 'n' is 0 or 1, it returns 1.

PROBLEMS 61 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py> & "C:/Users/Spriha Gajula/anaconda3/envs/spriha1850/python.exe" "c:/Users/Spriha Gajula/OneDrive/Desktop/python.py/aicoding3.1.py"
Enter a number to compute its factorial: 5
The factorial of 5 is: 120
PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py> & "C:/Users/Spriha Gajula/anaconda3/envs/spriha1850/python.exe" "c:/Users/Spriha Gajula/OneDrive/Desktop/python.py/aicoding3.1.py"
Enter a number to compute its factorial: 6
The factorial of 6 is: 720
PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py>
Ln 58, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.12.9 (spriha1850) Go Live
```

Explanation:

factorial(n): This function calculates the factorial of a non-negative integer 'n'.

It first checks if 'n' is negative and returns a message since factorials

are not defined for negative numbers. If 'n' is 0 or 1, it returns 1.

For other positive integers, it iteratively multiplies numbers from 2 to 'n' to compute the factorial and returns the result. Function to compute the factorial of a given number

Question 3: Few-Shot Prompting (Armstrong Number Check)

Write a few-shot prompt by providing multiple input-output examples to guide the AI in generating a Python function to check whether a given number is an Armstrong number.

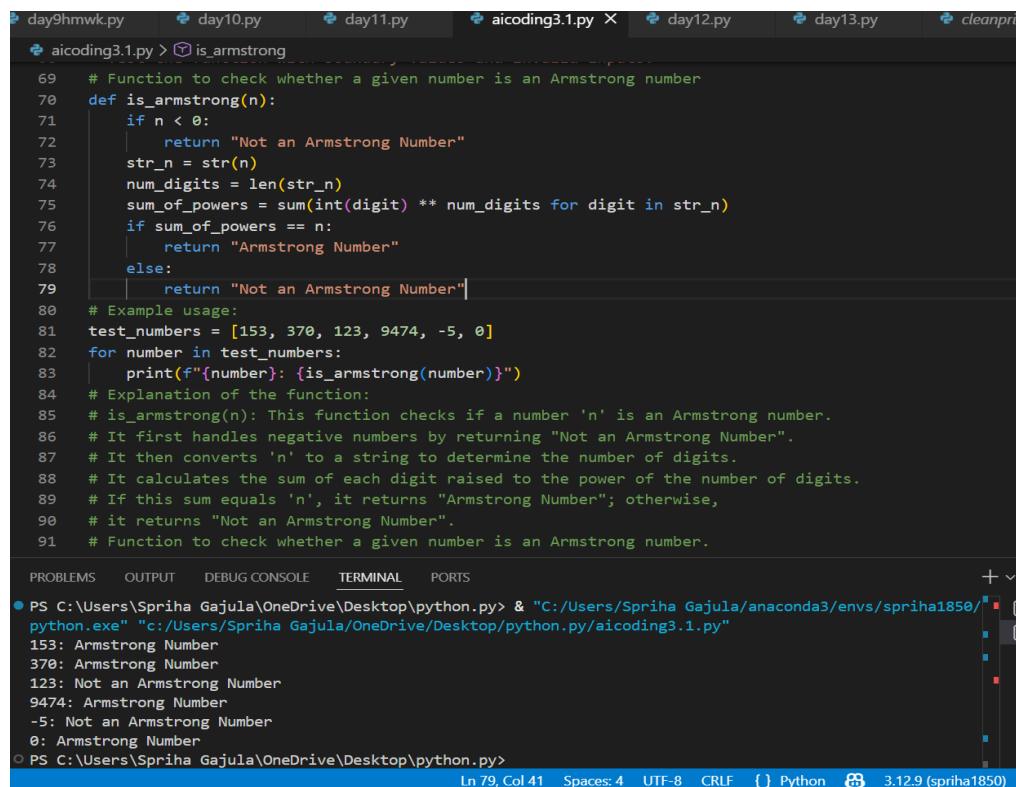
Examples:

- Input: 153 → Output: Armstrong Number
- Input: 370 → Output: Armstrong Number

Task:

- Analyze how multiple examples influence code structure and accuracy.
- Test the function with boundary values and invalid inputs.

Code:



```
day9hmkw.py day10.py day11.py aicoding3.1.py X day12.py day13.py cleanpri
aicoding3.1.py > is_armstrong
69 # Function to check whether a given number is an Armstrong number
70 def is_armstrong(n):
71     if n < 0:
72         return "Not an Armstrong Number"
73     str_n = str(n)
74     num_digits = len(str_n)
75     sum_of_powers = sum(int(digit) ** num_digits for digit in str_n)
76     if sum_of_powers == n:
77         return "Armstrong Number"
78     else:
79         return "Not an Armstrong Number"
80 # Example usage:
81 test_numbers = [153, 370, 123, 9474, -5, 0]
82 for number in test_numbers:
83     print(f"{number}: {is_armstrong(number)}")
84 # Explanation of the function:
85 # is_armstrong(n): This function checks if a number 'n' is an Armstrong number.
86 # It first handles negative numbers by returning "Not an Armstrong Number".
87 # It then converts 'n' to a string to determine the number of digits.
88 # It calculates the sum of each digit raised to the power of the number of digits.
89 # If this sum equals 'n', it returns "Armstrong Number"; otherwise,
90 # it returns "Not an Armstrong Number".
91 # Function to check whether a given number is an Armstrong number.

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
• PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py> & "C:/Users/Spriha Gajula/anaconda3/envs/spriha1850/python.exe" "C:/Users/Spriha Gajula/OneDrive/Desktop/python.py/aicoding3.1.py"
153: Armstrong Number
370: Armstrong Number
123: Not an Armstrong Number
9474: Armstrong Number
-5: Not an Armstrong Number
0: Armstrong Number
PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py>
Ln 79, Col 41 Spaces: 4 UTF-8 CRLF Python 3.12.9 (spriha1850)
```

Explanation:

is_armstrong(n): This function checks if a number 'n' is an Armstrong number.

It first handles negative numbers by returning "Not an Armstrong Number".

It then converts 'n' to a string to determine the number of digits.

It calculates the sum of each digit raised to the power of the number of digits.

If this sum equals 'n', it returns "Armstrong Number"; otherwise,

it returns "Not an Armstrong Number".

Function to check whether a given number is an Armstrong number

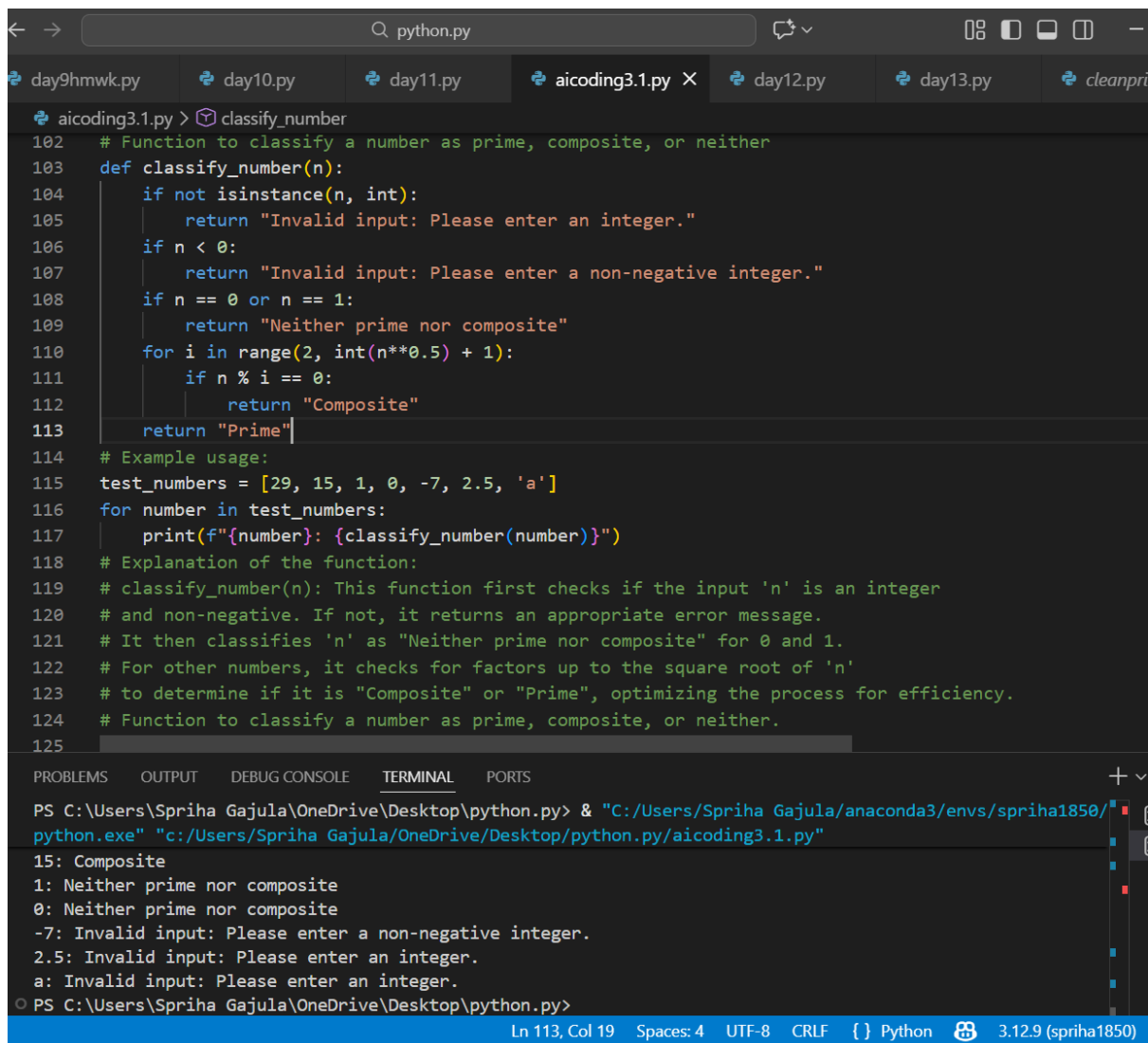
Question 4: Context-Managed Prompting (Optimized Number Classification)

Design a context-managed prompt with clear instructions and constraints to generate an optimized Python program that classifies a number as prime, composite, or neither.

Task:

- Ensure proper input validation.
- Optimize the logic for efficiency.
- Compare the output with earlier prompting strategies.

Code:



```
python.py
day9hwmk.py day10.py day11.py aicoding3.1.py X day12.py day13.py cleanpri
aicoding3.1.py > classify_number
102 # Function to classify a number as prime, composite, or neither
103 def classify_number(n):
104     if not isinstance(n, int):
105         return "Invalid input: Please enter an integer."
106     if n < 0:
107         return "Invalid input: Please enter a non-negative integer."
108     if n == 0 or n == 1:
109         return "Neither prime nor composite"
110     for i in range(2, int(n**0.5) + 1):
111         if n % i == 0:
112             return "Composite"
113     return "Prime"
114 # Example usage:
115 test_numbers = [29, 15, 1, 0, -7, 2.5, 'a']
116 for number in test_numbers:
117     print(f"{number}: {classify_number(number)}")
118 # Explanation of the function:
119 # classify_number(n): This function first checks if the input 'n' is an integer
120 # and non-negative. If not, it returns an appropriate error message.
121 # It then classifies 'n' as "Neither prime nor composite" for 0 and 1.
122 # For other numbers, it checks for factors up to the square root of 'n'
123 # to determine if it is "Composite" or "Prime", optimizing the process for efficiency.
124 # Function to classify a number as prime, composite, or neither.
125
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py> & "C:/Users/Spriha Gajula/anaconda3/envs/spriha1850/python.exe" "c:/Users/Spriha Gajula/OneDrive/Desktop/python.py/aicoding3.1.py"
15: Composite
1: Neither prime nor composite
0: Neither prime nor composite
-7: Invalid input: Please enter a non-negative integer.
2.5: Invalid input: Please enter an integer.
a: Invalid input: Please enter an integer.
PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py>
Ln 113, Col 19 Spaces: 4 UTF-8 CRLF {} Python 3.12.9 (spriha1850)
```

Explanation:

`classify_number(n)`: This function first checks if the input 'n' is an integer and non-negative. If not, it returns an appropriate error message.

It then classifies 'n' as "Neither prime nor composite" for 0 and 1.

For other numbers, it checks for factors up to the square root of 'n'

to determine if it is "Composite" or "Prime", optimizing the process for efficiency.

Function to classify a number as prime, composite, or neither.

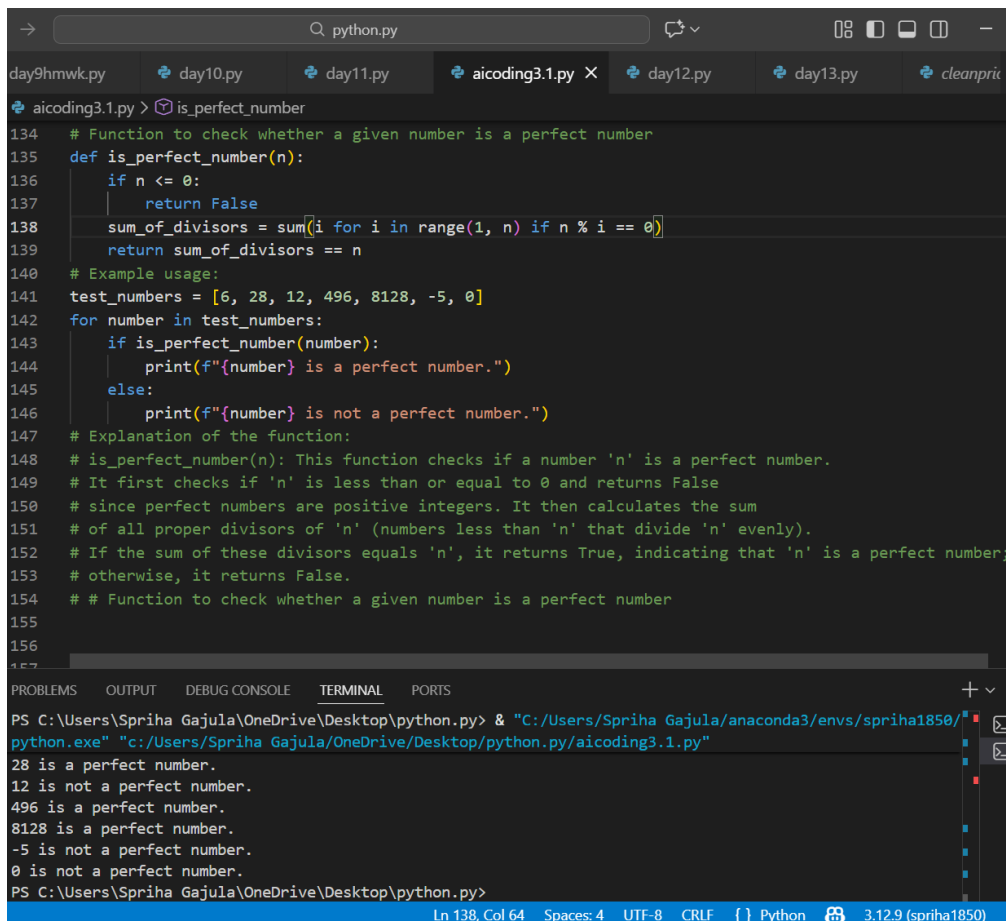
Question 5: Zero-Shot Prompting (Perfect Number Check)

Write a zero-shot prompt (without providing any examples) to generate a Python function that checks whether a given number is perfect number.

Task:

- Record the AI-generated code.
- Test the program with multiple inputs.
- Identify any missing conditions or inefficiencies in the logic.

Code:

The image is a screenshot of a Python IDE, likely VS Code, with a dark theme. The editor window shows a file named 'aicoding3.1.py' with a Python script. The script defines a function 'is_perfect_number(n)' that checks if a number is a perfect number. It includes comments explaining the logic: checking for non-positive numbers, summing proper divisors, and comparing the sum to the original number. Test cases are provided in a list, and the script prints the result for each. The terminal at the bottom shows the output of the script, confirming that 28, 12, 496, and 8128 are perfect numbers, while -5 and 0 are not. The status bar at the bottom indicates the current position is at line 138, column 64, with 4 spaces, UTF-8 encoding, CRLF line endings, and Python 3.12.9 (spriha1850) interpreter.

```
→ python.py
day9hwmk.py day10.py day11.py aicoding3.1.py X day12.py day13.py cleanpric
aicoding3.1.py > is_perfect_number
134 # Function to check whether a given number is a perfect number
135 def is_perfect_number(n):
136     if n <= 0:
137         return False
138     sum_of_divisors = sum(i for i in range(1, n) if n % i == 0)
139     return sum_of_divisors == n
140 # Example usage:
141 test_numbers = [6, 28, 12, 496, 8128, -5, 0]
142 for number in test_numbers:
143     if is_perfect_number(number):
144         print(f"{number} is a perfect number.")
145     else:
146         print(f"{number} is not a perfect number.")
147 # Explanation of the function:
148 # is_perfect_number(n): This function checks if a number 'n' is a perfect number.
149 # It first checks if 'n' is less than or equal to 0 and returns False
150 # since perfect numbers are positive integers. It then calculates the sum
151 # of all proper divisors of 'n' (numbers less than 'n' that divide 'n' evenly).
152 # If the sum of these divisors equals 'n', it returns True, indicating that 'n' is a perfect number;
153 # otherwise, it returns False.
154 # # Function to check whether a given number is a perfect number
155
156
157
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py> & "C:/Users/Spriha Gajula/anaconda3/envs/spriha1850/python.exe" "c:/Users/Spriha Gajula/OneDrive/Desktop/python.py/aicoding3.1.py"
28 is a perfect number.
12 is not a perfect number.
496 is a perfect number.
8128 is a perfect number.
-5 is not a perfect number.
0 is not a perfect number.
PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py>
Ln 138, Col 64 Spaces: 4 UTF-8 CRLF {} Python 3.12.9 (spriha1850)
```

Explanation:

is_perfect_number(n): This function checks if a number 'n' is a perfect number.

It first checks if 'n' is less than or equal to 0 and returns False since perfect numbers are positive integers. It then calculates the sum of all proper divisors of 'n' (numbers less than 'n' that divide 'n' evenly).

If the sum of these divisors equals 'n', it returns True, indicating that 'n' is a perfect number; otherwise, it returns False.

Function to check whether a given number is a perfect number

Question 6: Few-Shot Prompting (Even or Odd Classification with Validation)

Write a few-shot prompt by providing multiple input-output examples to guide the AI in generating a Python program that determines whether a given number is even or odd, including proper input validation.

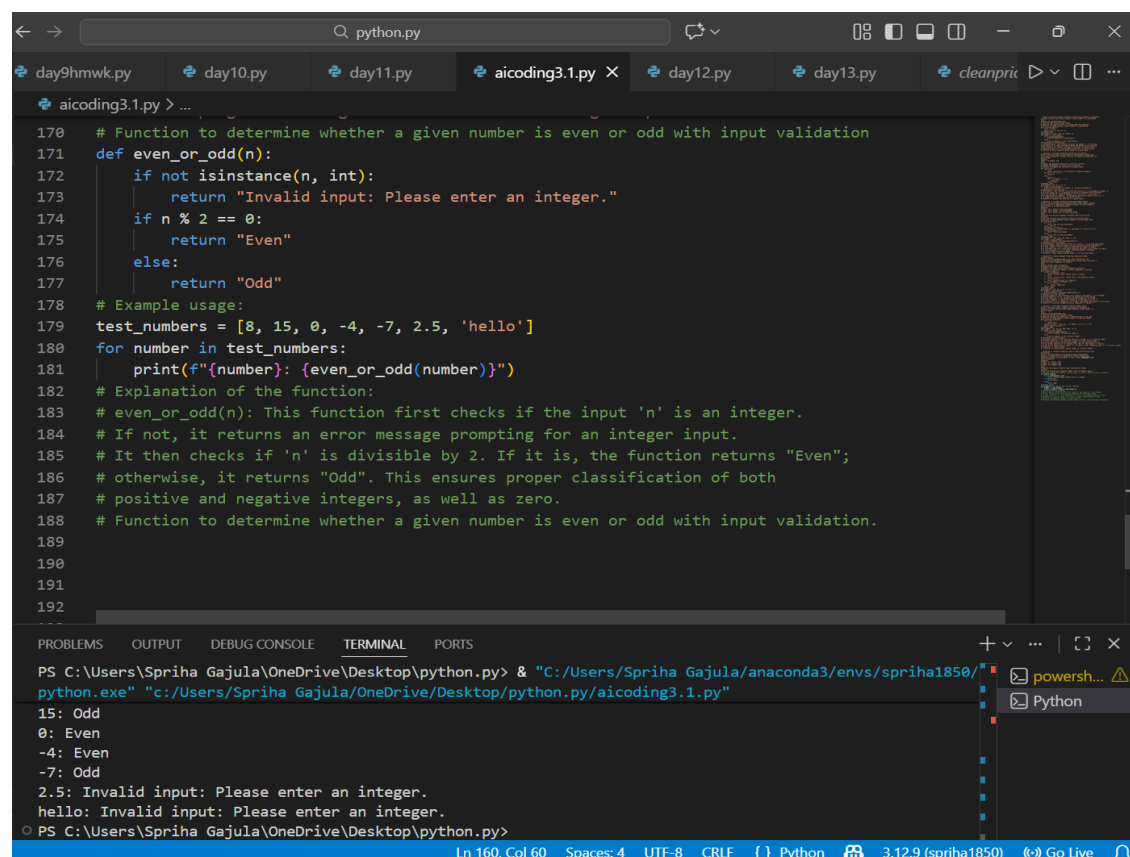
Examples:

- Input: 8 → Output: Even
- Input: 15 → Output: Odd
- Input: 0 → Output: Even

Task:

- Analyze how examples improve input handling and output clarity.
- Test the program with negative numbers and non-integer inputs.

Code:



```
python.py
day9hmkw.py day10.py day11.py aicoding3.1.py X day12.py day13.py cleanprie
aicoding3.1.py > ...
170 # Function to determine whether a given number is even or odd with input validation
171 def even_or_odd(n):
172     if not isinstance(n, int):
173         return "Invalid input: Please enter an integer."
174     if n % 2 == 0:
175         return "Even"
176     else:
177         return "Odd"
178 # Example usage:
179 test_numbers = [8, 15, 0, -4, -7, 2.5, 'hello']
180 for number in test_numbers:
181     print(f"{number}: {even_or_odd(number)}")
182 # Explanation of the function:
183 # even_or_odd(n): This function first checks if the input 'n' is an integer.
184 # If not, it returns an error message prompting for an integer input.
185 # It then checks if 'n' is divisible by 2. If it is, the function returns "Even";
186 # otherwise, it returns "Odd". This ensures proper classification of both
187 # positive and negative integers, as well as zero.
188 # Function to determine whether a given number is even or odd with input validation.
189
190
191
192
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py> & "C:/Users/Spriha Gajula/anaconda3/envs/spriha1850/python.exe" "c:/Users/Spriha Gajula/OneDrive/Desktop/python.py/aicoding3.1.py"
15: Odd
0: Even
-4: Even
-7: Odd
2.5: Invalid input: Please enter an integer.
hello: Invalid input: Please enter an integer.
PS C:\Users\Spriha Gajula\OneDrive\Desktop\python.py>
Ln 160, Col 60 Spaces: 4 UTF-8 CRLF Python 3.12.9 (spriha1850) Go Live
```


Explanation:

`even_or_odd(n)`: This function first checks if the input 'n' is an integer.

If not, it returns an error message prompting for an integer input.

It then checks if 'n' is divisible by 2. If it is, the function returns "Even"; otherwise, it returns "Odd". This ensures proper classification of both positive and negative integers, as well as zero.

Function to determine whether a given number is even or odd with input validation.