

Lab Assignment – 3.1

Hall Ticket No.: 2303A510E6

Name – Anushka Boora

Batch – 29

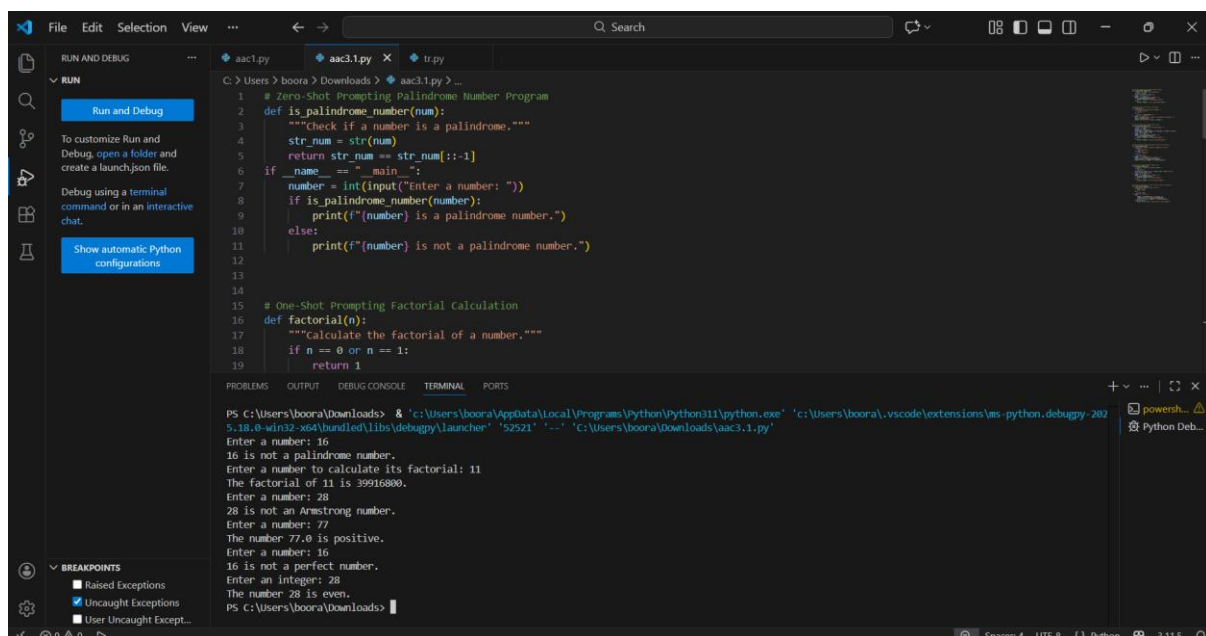
Question 1: Zero-Shot Prompting (Palindrome Number Program)

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 & OUTPUT



```
File Edit Selection View ... Search
aasc1.py aasc3.1.py tr.py
C:\Users\boora> Downloads > aasc3.1.py > ...
1 # Zero-Shot Prompting Palindrome Number Program
2 def is_palindrome_number(num):
3     """Check if a number is a palindrome."""
4     str_num = str(num)
5     return str_num == str_num[::-1]
6 if __name__ == "__main__":
7     number = int(input("Enter a number: "))
8     if is_palindrome_number(number):
9         print(f"{number} is a palindrome number.")
10    else:
11        print(f"{number} is not a palindrome number.")
12
13
14 # One-Shot Prompting Factorial Calculation
15 def factorial(n):
16     """Calculate the factorial of a number."""
17     if n == 0 or n == 1:
18         return 1
19
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\boora\Downloads> & 'c:\Users\boora\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\boora\.vscode\extensions\ms-python.debugpy-202
5.18.0-win32-x64\lib\debugpy\launcher' '52521' '-' 'c:\Users\boora\Downloads\aac3.1.py'
Enter a number: 16
16 is not a palindrome number.
Enter a number to calculate its factorial: 11
The factorial of 11 is 39916800.
Enter a number: 28
28 is not an Armstrong number.
Enter a number: 77
The number 77.0 is positive.
Enter a number: 16
16 is not a perfect number.
Enter an integer: 28
The number 28 is even.
PS C:\Users\boora\Downloads>
```

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.

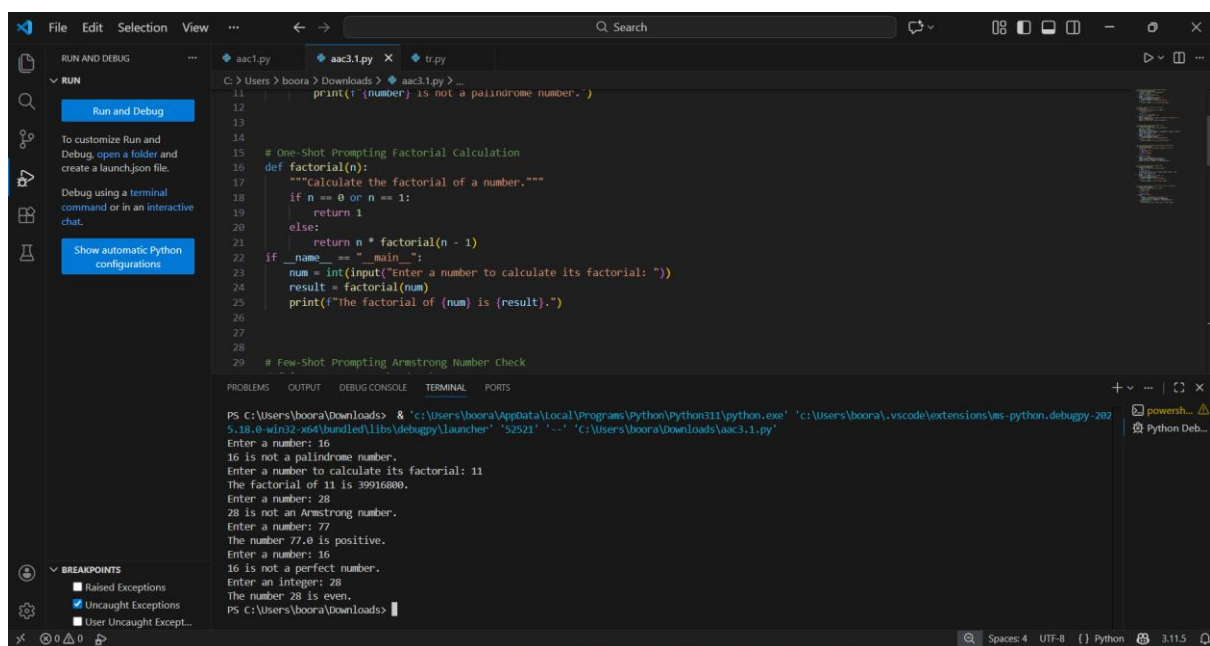
Example:

Input: 5 → Output: 120

Task:

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

CODE & OUTPUT



The screenshot shows a VS Code editor with a Python file named `aac3.1.py`. The code defines a `factorial(n)` function and includes a main block for user input. The terminal output shows the program running and processing several inputs.

```
11 print(f'{number} is not a palindrome number.')
12
13
14
15 # One-Shot Prompting Factorial Calculation
16 def factorial(n):
17     """Calculate the factorial of a number."""
18     if n == 0 or n == 1:
19         return 1
20     else:
21         return n * factorial(n - 1)
22 if __name__ == "__main__":
23     num = int(input("Enter a number to calculate its factorial: "))
24     result = factorial(num)
25     print(f"The factorial of {num} is {result}.")
26
27
28 # Few-Shot Prompting Armstrong Number Check
```

Terminal Output:

```
PS C:\Users\boora\Downloads> & "c:\Users\boora\AppData\Local\Programs\Python\Python311\python.exe" "c:\Users\boora\.vscode\extensions\ms-python.debugpy-202
5.18.0-win32-x64\bundled\libs\debugpy\launcher" "52521" "--" "c:\Users\boora\Downloads\aac3.1.py"
Enter a number: 16
16 is not a palindrome number.
Enter a number to calculate its factorial: 11
The factorial of 11 is 39916800.
Enter a number: 28
28 is not an Armstrong number.
Enter a number: 77
The number 77.0 is positive.
Enter a number: 16
16 is not a perfect number.
Enter an integer: 28
The number 28 is even.
PS C:\Users\boora\Downloads>
```

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
- Input: 123 → Output: Not an Armstrong Number

Task:

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

CODE & OUTPUT

The screenshot shows a VS Code editor with a Python file named `aac3.1.py`. The code defines a function `is_armstrong_number(num)` that checks if a number is an Armstrong number. The function converts the number to a string, calculates the sum of each digit raised to the power of the number of digits, and compares it to the original number. The script also includes a main loop that prompts the user for a number and prints the result.

```

26
27
28
29 # Few-Shot Prompting Armstrong Number Check
30 def is_armstrong_number(num):
31     """Check if a number is an Armstrong number."""
32     num_str = str(num)
33     num_digits = len(num_str)
34     sum_of_powers = sum(int(digit) ** num_digits for digit in num_str)
35     return sum_of_powers == num
36
37 if __name__ == "__main__":
38     number = int(input("Enter a number: "))
39     if is_armstrong_number(number):
40         print(f"{number} is an Armstrong number.")
41     else:
42         print(f"{number} is not an Armstrong number.")
43
44

```

The terminal output shows the execution of the script. It prompts the user for a number and prints the result. The output is as follows:

```

PS C:\Users\boora\Downloads> & 'c:\Users\boora\AppData\Local\Programs\Python\Python111\python.exe' 'c:\Users\boora\.vscode\extensions\ms-python.debugpy-202
5.18.0-win32-x64\bundle\libs\debugpy\launcher' '52521' '...' 'C:\Users\boora\Downloads\aac3.1.py'
Enter a number: 16
16 is not a palindrome number.
Enter a number to calculate its factorial: 11
The factorial of 11 is 39916800.
Enter a number: 28
28 is not an Armstrong number.
Enter a number: 77
The number 77.0 is positive.
Enter a number: 16
16 is not a perfect number.
Enter an integer: 28
The number 28 is even.
PS C:\Users\boora\Downloads>

```

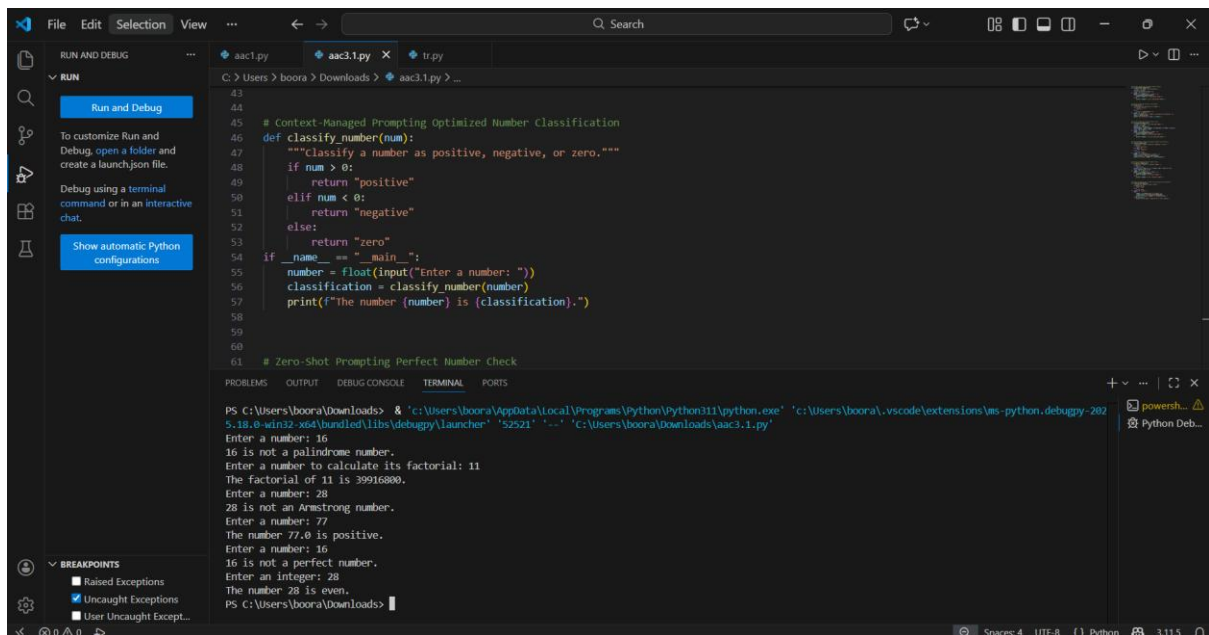
Question 4: Context-Managed Prompting (Optimized Number Classification)

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 & OUTPUT



The screenshot shows a VS Code editor with a Python file named `aac3.1.py` open. The code defines a function `classify_number` that takes a number and returns a string classification: "positive", "negative", or "zero". The function is then used in a `main` block to prompt the user for a number and print the classification. Below the code, the terminal output shows the program being run, with the user entering several numbers and seeing the corresponding classifications.

```
43
44
45 # Context-Managed Prompting Optimized Number Classification
46 def classify_number(num):
47     """Classify a number as positive, negative, or zero."""
48     if num > 0:
49         return "positive"
50     elif num < 0:
51         return "negative"
52     else:
53         return "zero"
54 if __name__ == "__main__":
55     number = float(input("Enter a number: "))
56     classification = classify_number(number)
57     print(f"The number {number} is {classification}.")
58
59
60
61 # Zero-Shot Prompting Perfect Number Check
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\boora\Downloads> & 'c:\Users\boora\AppData\Local\Programs\Python\Python111\python.exe' 'c:\Users\boora\.vscode\extensions\ms-python.debugpy-202
5.18.0-win32-x64\bin\debugpy\launcher' '52521' '-' 'c:\Users\boora\Downloads\aac3.1.py'
Enter a number: 16
16 is not a palindrome number.
Enter a number to calculate its factorial: 11
The factorial of 11 is 39916800.
Enter a number: 28
28 is not an Armstrong number.
Enter a number: 77
The number 77.0 is positive.
Enter a number: 16
16 is not a perfect number.
Enter an integer: 28
The number 28 is even.
PS C:\Users\boora\Downloads>
```

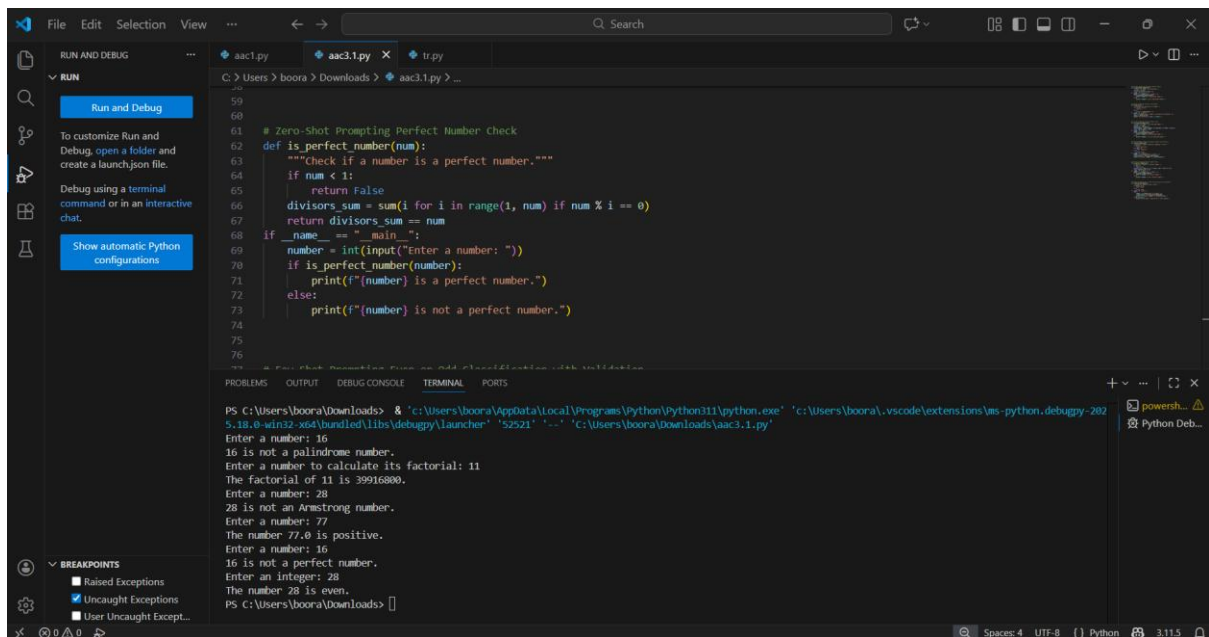
Question 5: Zero-Shot Prompting (Perfect Number Check)

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

Task:

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

CODE & OUTPUT



The screenshot shows a VS Code editor with a Python file named `aac3.1.py` open. The code is a function `is_perfect_number(num)` that checks if a number is a perfect number. The terminal output shows the program running and prompting the user to enter a number. The user enters 16, and the program outputs "16 is not a perfect number." The user then enters 28, and the program outputs "28 is not a perfect number." The user then enters 77, and the program outputs "The number 77.0 is positive." The user then enters 16, and the program outputs "16 is not a perfect number." The user then enters 28, and the program outputs "The number 28 is even." The terminal output is as follows:

```
PS C:\Users\boora\Downloads> & 'c:\Users\boora\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\boora\.vscode\extensions\ms-python.debugpy-2023.18.0-win32-aarch64\libs\debugpy\launcher' '52521' '-' 'c:\Users\boora\Downloads\aac3.1.py'
Enter a number: 16
16 is not a perfect number.
Enter a number to calculate its factorial: 11
The factorial of 11 is 39916800.
Enter a number: 28
28 is not an Armstrong number.
Enter a number: 77
The number 77.0 is positive.
Enter a number: 16
16 is not a perfect number.
Enter an integer: 28
The number 28 is even.
PS C:\Users\boora\Downloads>
```

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.

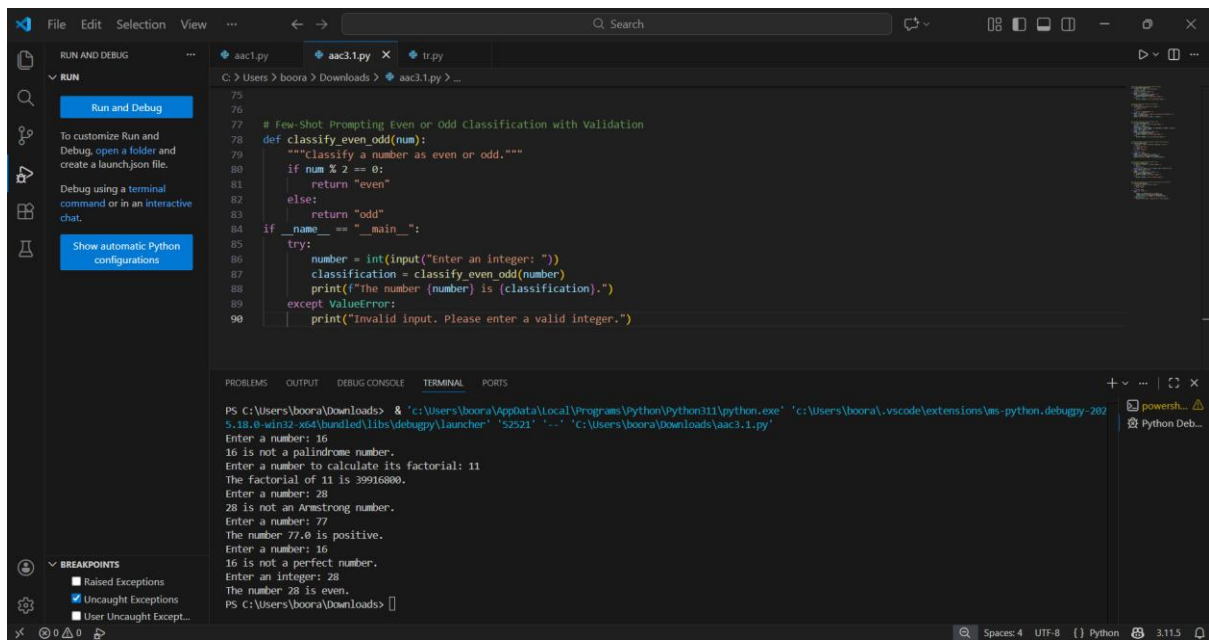
Examples:

- 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 & OUTPUT



The screenshot displays the Visual Studio Code interface with a Python file named `aac3.1.py` open. The code defines a function `classify_even_odd(num)` that checks if a number is even or odd. The main block prompts the user to enter an integer and prints the classification. The output window shows the execution results, including the prompt and the classification for the input 16.

```
75
76
77 # Few-Shot Prompting Even or Odd Classification with Validation
78 def classify_even_odd(num):
79     """Classify a number as even or odd."""
80     if num % 2 == 0:
81         return "even"
82     else:
83         return "odd"
84 if __name__ == "__main__":
85     try:
86         number = int(input("Enter an integer: "))
87         classification = classify_even_odd(number)
88         print(f"The number {number} is {classification}.")
89     except ValueError:
90         print("Invalid input. Please enter a valid integer.")
```

PS C:\Users\boora\Downloads> & 'c:\Users\boora\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\boora\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle\libs\debugpy\launcher' '52521' '-' 'c:\Users\boora\Downloads\aac3.1.py'

Enter a number: 16
16 is a palindrome number.
Enter a number to calculate its factorial: 11
The factorial of 11 is 39916800.
Enter a number: 28
28 is not an Armstrong number.
Enter a number: 77
The number 77.0 is positive.
Enter a number: 16
16 is not a perfect number.
Enter an integer: 28
The number 28 is even.
PS C:\Users\boora\Downloads> |