

# AI ASSISTED CODING

## LAB-3.1

**Katta Lasya**

**2303A51724**

**Batch-11**

**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.

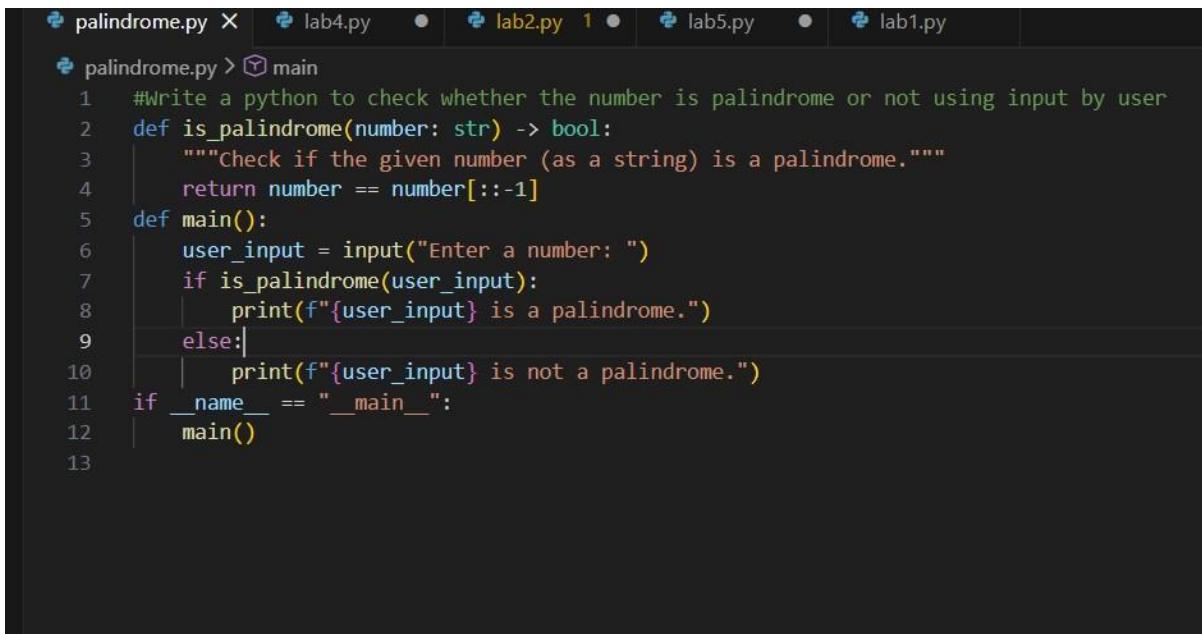
Task:

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

**PROMPT:**

#Write a python to check whether the number is palindrome or not using input by user

**CODE:**



```
palindrome.py X lab4.py ● lab2.py 1 ● lab5.py ● lab1.py
palindrome.py > main
1 #Write a python to check whether the number is palindrome or not using input by user
2 def is_palindrome(number: str) -> bool:
3     """Check if the given number (as a string) is a palindrome."""
4     return number == number[::-1]
5 def main():
6     user_input = input("Enter a number: ")
7     if is_palindrome(user_input):
8         print(f"{user_input} is a palindrome.")
9     else:
10        print(f"{user_input} is not a palindrome.")
11 if __name__ == "__main__":
12     main()
13
```

**OUTPUT:**

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS + v ... | ⌂ X
● PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py
Enter a number: 141
141 is a palindrome.
● PS C:\Users\thota\OneDrive\Desktop\AIAC> 142
142
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py
Enter a number: 141
141 is a palindrome.
● PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py
Enter a number: 142
142 is not a palindrome.
○ PS C:\Users\thota\OneDrive\Desktop\AIAC>

```

The terminal window shows several Python environments running simultaneously. The output of each environment is displayed sequentially. The script checks if the input number is a palindrome. For the first two runs, it correctly identifies 141 as a palindrome. For the third run, it correctly identifies 142 as not 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.

### PROMPT:

#Write a python to check whether the number is factorial or not give me the factorial of that number  
example input=5 and output=120 **CODE:**

```

palindrome.py ● lab4.py ● lab2.py 1 ● lab5.py ● lab1.py ▷ v ...
palindrome.py > factorial
1 #Write a python to check whether the number is factorial or not give me the factorial of that
2 def factorial(n):
3     if n < 0:
4         return "Factorial is not defined for negative numbers."
5     elif n == 0 or n == 1:
6         return 1
7     else:
8         result = 1
9         for i in range(2, n + 1):
10            result *= i
11    return result
12
13 # Example usage
14 number = 5
15 print(f"The factorial of {number} is {factorial(number)}")
16 # This program defines a function to calculate the factorial of a given number.
17

```

### OUTPUT:

```
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS + v ... |  
hon.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py  
The factorial of 5 is 120  
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/pyt  
hon.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py  
The factorial of 5 is 120  
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/pyt  
hon.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py  
The factorial of 5 is 120  
PS C:\Users\thota\OneDrive\Desktop\AIAC> []  
Ln 10, Col 24 Spaces: 4
```

### 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.

#### PROMPT:

```
#write a code to generate armstrong number checking functions with comments
```

#### CODE:

The screenshot shows a code editor window with several tabs at the top: palindrome.py, lab4.py, lab2.py (selected), lab5.py, and lab1.py. The current file is lab1.py, which contains the following Python code:

```
#write a code to generate armstrong number checking functions with comments
def is_armstrong_number(num):
    """
    This function checks if a number is an Armstrong number.
    An Armstrong number is a number that is equal to the sum of its own digits
    each raised to the power of the number of digits.
    For example, 153 is an Armstrong number because 1^3 + 5^3 + 3^3 = 153.
    """
    # Convert the number to string to easily iterate over digits
    digits = str(num)
    num_digits = len(digits)

    # Calculate the sum of each digit raised to the power of num_digits
    sum_of_powers = sum(int(digit) ** num_digits for digit in digits)

    # Check if the sum of powers is equal to the original number
    return sum_of_powers == num

# Example usage
if __name__ == "__main__":
    test_number = 153
    if is_armstrong_number(test_number):
        print(f"{test_number} is an Armstrong number.")
    else:
        print(f"{test_number} is not an Armstrong number.")
    test_number = 123
    if is_armstrong_number(test_number):
        print(f"{test_number} is an Armstrong number.")
    else:
        print(f"{test_number} is not an Armstrong number.")
```

## OUTPUT:

The screenshot shows a terminal window with the following output:

```
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab1.py
153 is an Armstrong number.
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab1.py
153 is an Armstrong number.
123 is not an Armstrong number.
PS C:\Users\thota\OneDrive\Desktop\AIAC>
```

## 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.

## PROMPT:

#Write a python code to check whether the number classifies as prime,composite or neither using input by user **CODE:**

```

❷ palindrome.py > ...
1  #Write a python code to check whether the number classifies as prime,composite or neither using input by user
2  def classify_number(num):
3      """
4          This function classifies a number as 'prime', 'composite', or 'neither'.
5          A prime number is greater than 1 and has no divisors other than 1 and itself.
6          A composite number is greater than 1 and has more than two divisors.
7          Numbers less than or equal to 1 are classified as 'neither'.
8      """
9      if num <= 1:
10          return "neither"
11      for i in range(2, int(num**0.5) + 1):
12          if num % i == 0:
13              return "composite"
14      return "prime"
15  # Example usage
16  if __name__ == "__main__":
17      user_input = int(input("Enter a number to classify: "))
18      classification = classify_number(user_input)
19      print(f"The number {user_input} is classified as: {classification}")
20  # This program defines a function to classify a number as prime, composite, or neither.
21

```

## OUTPUT:

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py
Enter a number to classify: 1
The number 1 is classified as: neither
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py
Enter a number to classify: 2
The number 2 is classified as: prime
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py
Enter a number to classify: 20
The number 20 is classified as: composite
PS C:\Users\thota\OneDrive\Desktop\AIAC>

```

## 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

## PROMPT:

#Write a python code to check whether the number is perfect number or not with inputbyuser **CODE**:

```

❸ palindrome.py > ...
1  #Write a python code to check whether the number is perfect number or not with inputbyuser
2  def is_perfect_number(n: int) -> bool:
3      """Check if a number is a perfect number."""
4      if n < 1:
5          return False
6      divisors_sum = sum(i for i in range(1, n) if n % i == 0)
7      return divisors_sum == n
8  def main():
9
10     try:
11         number = int(input("Enter a number to check if it is a perfect number: "))
12         if is_perfect_number(number):
13             print(f"{number} is a perfect number.")
14         else:
15             print(f"{number} is not a perfect number.")
16     except ValueError:
17         print("Invalid input. Please enter a positive integer.")
18  if __name__ == "__main__":
19      main()
20

```

## OUTPUT:

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS
❸ PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py
Enter a number to check if it is a perfect number: 6
6 is a perfect number.
❸ PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py
Enter a number to check if it is a perfect number: 10
10 is not a perfect number.
❸ PS C:\Users\thota\OneDrive\Desktop\AIAC>

```

## 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 no n-integer inputs.

### PROMPT:

"""Write a python code to check whether the number is even or odd

Examples Input:8,Output:Even

Input:15,Output:odd

Input:0,Output:Even"""

### CODE:

```
palindrome.py > ...
1  """Write a python code to check whether the number is even or odd
2  Examples Input:8,Output:Even
3  Input:15,Output:odd
4  Input:0,Output:Even"""
5  def check_even_odd(number: int) -> str:
6      if number % 2 == 0:
7          return "Even"
8      else:
9          return "Odd"
10 if __name__ == "__main__":
11     test_numbers = [8, 15, 0]
12     for num in test_numbers:
13         result = check_even_odd(num)
14         print(f"Input:{num}, Output:{result}")
15 # This program checks whether a given number is even or odd and prints the result.
16 # Examples:
```

## OUTPUT:

```
palindrome.py > ...
1  """Write a python code to check whether the number is even or odd
2  Examples Input:8,Output:Even
3  Input:15,Output:odd
4  Input:0,Output:Even"""
5  def check_even_odd(number: int) -> str:
6      if number % 2 == 0:
7          return "Even"
8      else:
9          return "Odd"
10 if __name__ == "__main__":
11     test_numbers = [8, 15, 0]
12     for num in test_numbers:
13         result = check_even_odd(num)
14         print(f"Input:{num}, Output:{result}")
15 # This program checks whether a given number is even or odd and prints the result.
16 # Examples:
```

PROBLEMS | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS

- PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py  
Enter a number to check if it is a perfect number: 10  
10 is not a perfect number.
- PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py  
Enter a number: 6  
The number 6 is even.
- PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/palindrome.py  
Input:8, Output:Even  
Input:15, Output:odd  
Input:0, Output:Even
- PS C:\Users\thota\OneDrive\Desktop\AIAC>

Ln 4, Col 23 | Spaces: 4 | UTF-8 | CRLF | Python | 3.13.0 | 1423 | 23-01-2026 | 1423 | ENG | IN | 28°C | Sunny