

# AIAC LAB

## Lab Assignment- 3.1

### Experiment – Prompt Engineering Techniques

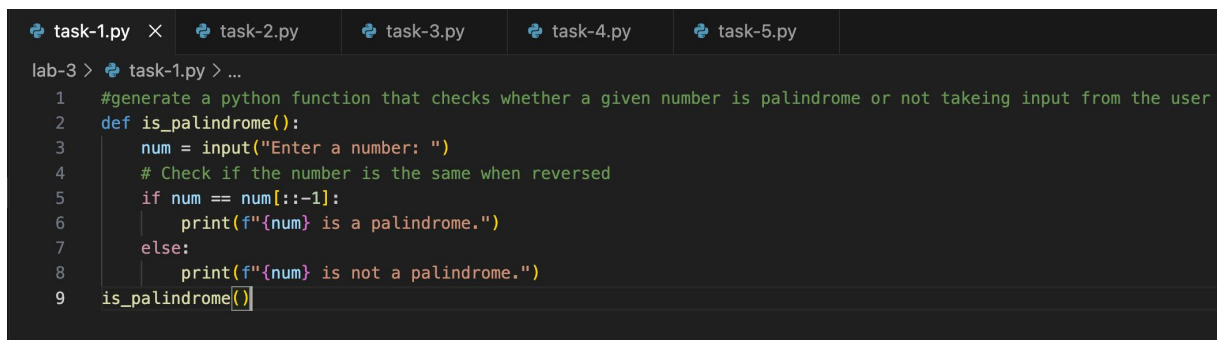
2303A51721

Batch-11

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

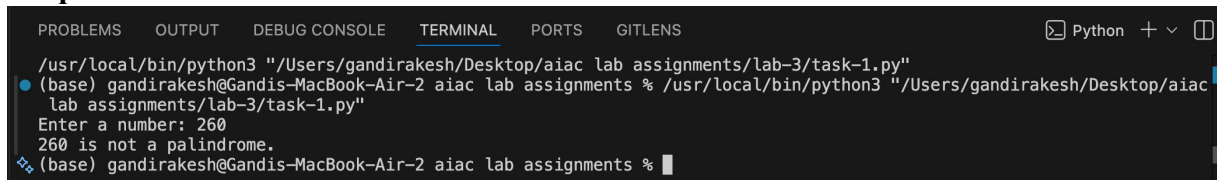
**Prompts:** Generate a Python function that checks whether a given number is a palindrome.

**Code:**



```
task-1.py x task-2.py task-3.py task-4.py task-5.py
lab-3 > task-1.py > ...
1 #generate a python function that checks whether a given number is palindrome or not takeing input from the user
2 def is_palindrome():
3     num = input("Enter a number: ")
4     # Check if the number is the same when reversed
5     if num == num[::-1]:
6         print(f"{num} is a palindrome.")
7     else:
8         print(f"{num} is not a palindrome.")
9 is_palindrome()
```

**Output:**



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS Python + v
/usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiac lab assignments/lab-3/task-1.py"
(base) gandirakesh@Gandis-MacBook-Air-2 aiac lab assignments % /usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiac
lab assignments/lab-3/task-1.py"
Enter a number: 260
260 is not a palindrome.
(base) gandirakesh@Gandis-MacBook-Air-2 aiac lab assignments %
```

**Explanation:** This code determines whether a number entered by the user is a palindrome, which means it reads the same forwards and backwards. When the user inputs a number, the code first converts it to a string so that it can easily compare the sequence of digits. It then checks if this string is identical to its reverse. If they match, the function returns True, indicating the number is a palindrome; otherwise, it returns False. The result is printed out, letting the user know whether their number is a palindrome or not.

#### Question 2: One-Shot Prompting (Factorial Calculation)

**Prompts:** Generate a Python function to compute the factorial of a given number.

**Code:**

```
task-1.py task-2.py × task-3.py task-4.py task-5.py
lab-3 > task-2.py > ...
1 #generate a python function to compute the factorial of a given number
2 def factorial():
3     num = int(input("Enter a number: "))
4     result = 1
5     for i in range(1, num + 1):
6         result *= i
7     print(f"The factorial of {num} is {result}.")
8 factorial()
```

**Output:**

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS Python + - 
/usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiaac lab assignments/lab-3/task-2.py"
(base) gandirakesh@Gandis-MacBook-Air-2 aiaac lab assignments % /usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiaac lab assignments/lab-3/task-2.py"
Enter a number: 6
The factorial of 6 is 720.
(base) gandirakesh@Gandis-MacBook-Air-2 aiaac lab assignments %
```

**Explanation:** This code defines a function to calculate the factorial of a given number using recursion. The factorial of a number (written as  $n!$ ) is the product of all positive integers up to that number. For example,  $5! = 5 \times 4 \times 3 \times 2 \times 1 = 120$ . The function checks if the input is 0 or 1, in which case it returns 1, since the factorial of both 0 and 1 is 1. The example usage shows how the function works for different inputs, printing the results for 5, 0, and 3.

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

**Prompts:** Generating a Python function to check whether a given number is an Armstrong number.

**Code:**

```
lab-3 > task-3.py > ...
1 #generate a python function to checl whether the given number is armstrong number or not
2 def is_armstrong():
3     num = int(input("Enter a number: "))
4     # Calculate the number of digits
5     num_digits = len(str(num))
6     sum_of_powers = 0
7     temp = num
8     while temp > 0:
9         digit = temp % 10
10        sum_of_powers += digit ** num_digits
11        temp //= 10
12    if sum_of_powers == num:
13        print(f"{num} is an Armstrong number.")
14    else:
15        print(f"{num} is not an Armstrong number.")
16    is_armstrong()
```

**Output:**

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS Python + v []
/usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiac lab assignments/lab-3/task-3.py"
(base) gandirakesh@Gandis-MacBook-Air-2 aiac lab assignments % /usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiac
lab assignments/lab-3/task-3.py"
Enter a number: 65
65 is not an Armstrong number.
❖ (base) gandirakesh@Gandis-MacBook-Air-2 aiac lab assignments %
```

**Explanation:** This code defines a function to check 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. The function works by first converting the number to a string so it can easily access each digit. It then calculates the number of digits and computes the sum of each digit raised to that power. If this sum matches the original number, the function returns True; otherwise, it returns False

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

**Prompts:** Generate an optimized Python program that classifies a number as prime, composite, or neither.

**Code:**

```
task-1.py task-2.py task-3.py task-4.py X task-5.py
lab-3 > task-4.py > ...
1 #Generate an optimized Python program that classifies a number as prime, composite, or neither
2 def classify_number():
3     num = int(input("Enter a number: "))
4     if num <= 1:
5         print(f"{num} is neither prime nor composite.")
6         return
7     for i in range(2, int(num**0.5) + 1):
8         if num % i == 0:
9             print(f"{num} is a composite number.")
10            return
11        print(f"{num} is a prime number.")
12    classify_number()
```

**Output:**

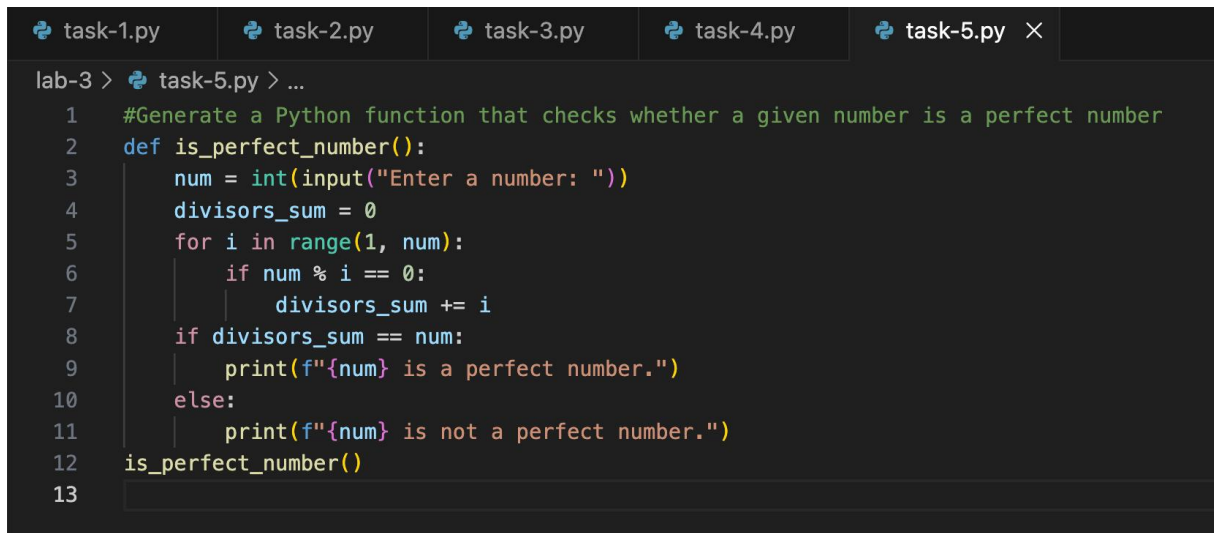
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS Python + v []
/usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiac lab assignments/lab-3/task-4.py"
(base) gandirakesh@Gandis-MacBook-Air-2 aiac lab assignments % /usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiac
lab assignments/lab-3/task-4.py"
Enter a number: 23
23 is a prime number.
❖ (base) gandirakesh@Gandis-MacBook-Air-2 aiac lab assignments %
```

**Explanation:** This code defines a function that classifies a given number as "prime," "composite," or "neither." If the input number is less than or equal to 1, it returns "neither" because 0 and 1 are not considered prime or composite. For numbers greater than 1, it checks if the number is divisible by any integer from 2 up to the square root of the number. If it finds a divisor, the number is "composite".

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

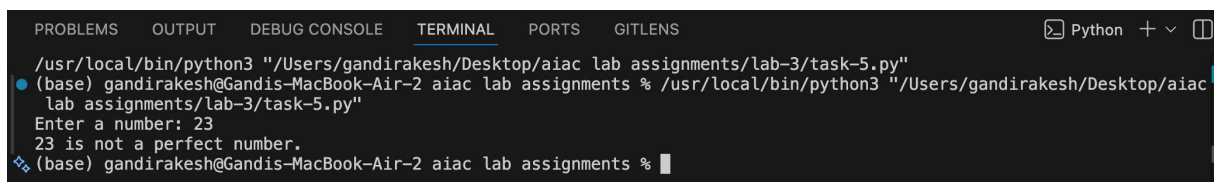
**Prompts:** Generate a Python function that checks whether a given number is a perfect number.

**Code:**

A screenshot of a code editor with a dark theme. At the top, there are tabs for 'task-1.py', 'task-2.py', 'task-3.py', 'task-4.py', and 'task-5.py' (which is active and has a close button). The main editor area shows a Python script. Line 1 is a comment: '#Generate a Python function that checks whether a given number is a perfect number'. Line 2 defines the function 'def is\_perfect\_number()'. Line 3 takes user input: 'num = int(input("Enter a number: "))'. Line 4 initializes a sum: 'divisors\_sum = 0'. Line 5 starts a loop: 'for i in range(1, num):'. Line 6 checks for divisibility: 'if num % i == 0:'. Line 7 adds the divisor to the sum: 'divisors\_sum += i'. Line 8 checks if the sum equals the number: 'if divisors\_sum == num:'. Line 9 prints a message if it's a perfect number: 'print(f"{num} is a perfect number.")'. Line 10 is the 'else:' branch. Line 11 prints a message if it's not a perfect number: 'print(f"{num} is not a perfect number.")'. Line 12 calls the function: 'is\_perfect\_number()'. Line 13 is an empty line.

```
lab-3 > task-5.py > ...
1  #Generate a Python function that checks whether a given number is a perfect number
2  def is_perfect_number():
3      num = int(input("Enter a number: "))
4      divisors_sum = 0
5      for i in range(1, num):
6          if num % i == 0:
7              divisors_sum += i
8      if divisors_sum == num:
9          print(f"{num} is a perfect number.")
10     else:
11         print(f"{num} is not a perfect number.")
12     is_perfect_number()
13
```

**Output:**

A screenshot of a terminal window. The title bar shows 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL' (which is active), 'PORTS', and 'GITLENS'. The terminal shows the command to run the script: '/usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiac lab assignments/lab-3/task-5.py"'. The prompt is '(base) gandirakesh@Gandis-MacBook-Air-2 aiac lab assignments %'. The user enters '23'. The output is '23 is not a perfect number.' followed by a new prompt: '(base) gandirakesh@Gandis-MacBook-Air-2 aiac lab assignments %'.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS Python + v
/usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiac lab assignments/lab-3/task-5.py"
(base) gandirakesh@Gandis-MacBook-Air-2 aiac lab assignments % /usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiac
lab assignments/lab-3/task-5.py"
Enter a number: 23
23 is not a perfect number.
(base) gandirakesh@Gandis-MacBook-Air-2 aiac lab assignments %
```

**Explanation:** This code defines a function to check if a number is a perfect number. A perfect number is a positive integer that is equal to the sum of all its proper divisors (excluding itself). The function first checks if the number is less than 1, returning False since perfect numbers must be positive. It then calculates the sum of all divisors of the number from 1 up to (but not including) the number itself. If this sum equals the original number, the function returns True.

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

**Prompts:** Determines whether a given number is even or odd, including proper input validation.

**Code:**

```
task-1.py task-2.py task-3.py task-4.py task-5.py task-6.py X
lab-3 > task-6.py > ...
1  #Determines whether a given number is even or odd, including proper input validation
2  def is_even_or_odd():
3      while True:
4          try:
5              num = int(input("Enter a number: "))
6              break
7          except ValueError:
8              print("Invalid input. Please enter an integer.")
9      if num % 2 == 0:
10         print(f"{num} is even.")
11     else:
12         print(f"{num} is odd.")
13 is_even_or_odd()
14
```

## Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS Python + - []
/usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiac lab assignments/lab-3/task-6.py"
(base) gandirakesh@Gandis-MacBook-Air-2 aiac lab assignments % /usr/local/bin/python3 "/Users/gandirakesh/Desktop/aiac
lab assignments/lab-3/task-6.py"
Enter a number: 765
765 is odd.
(base) gandirakesh@Gandis-MacBook-Air-2 aiac lab assignments %
```

**Explanation:** This code defines a function that determines whether a given number is even or odd, with input validation. The function first checks if the input is an integer; if not, it returns a message asking for a valid integer. If the input is an integer, it checks if the number is divisible by 2. If it is, the function returns "even"; otherwise, it returns "odd."