AI ASSIATED CODING

ASSIGNMENT : 3.3

PROGRAM : B.TECH\CSE

SPECIALISATION : AIML

NAME : MANDALA.TEJASRI

ENROLMENT NUMBER : 2403A51171

COURSE : AI ASSISTED CODING

DATE : 20 AUGUST 2025

BATCH NO : 01

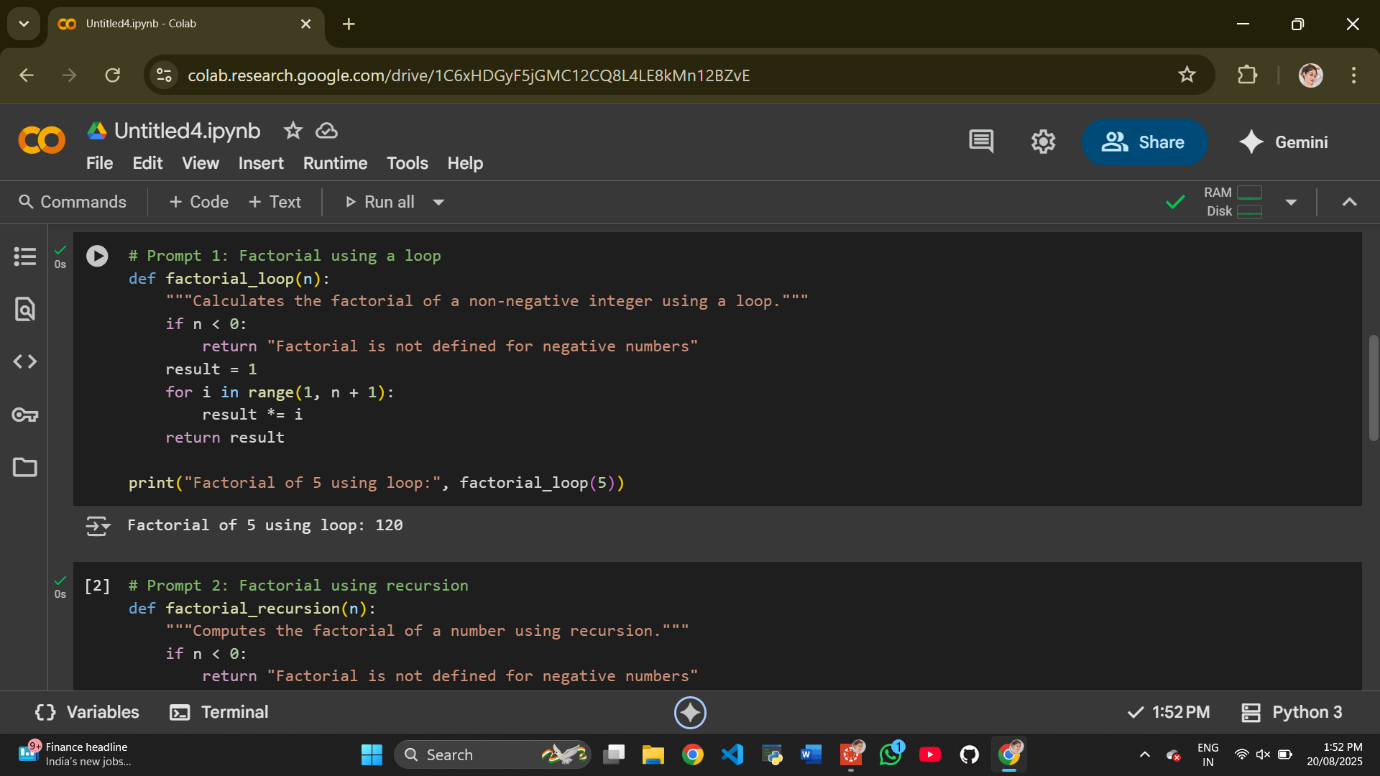
TASK DESCRIPTION#1

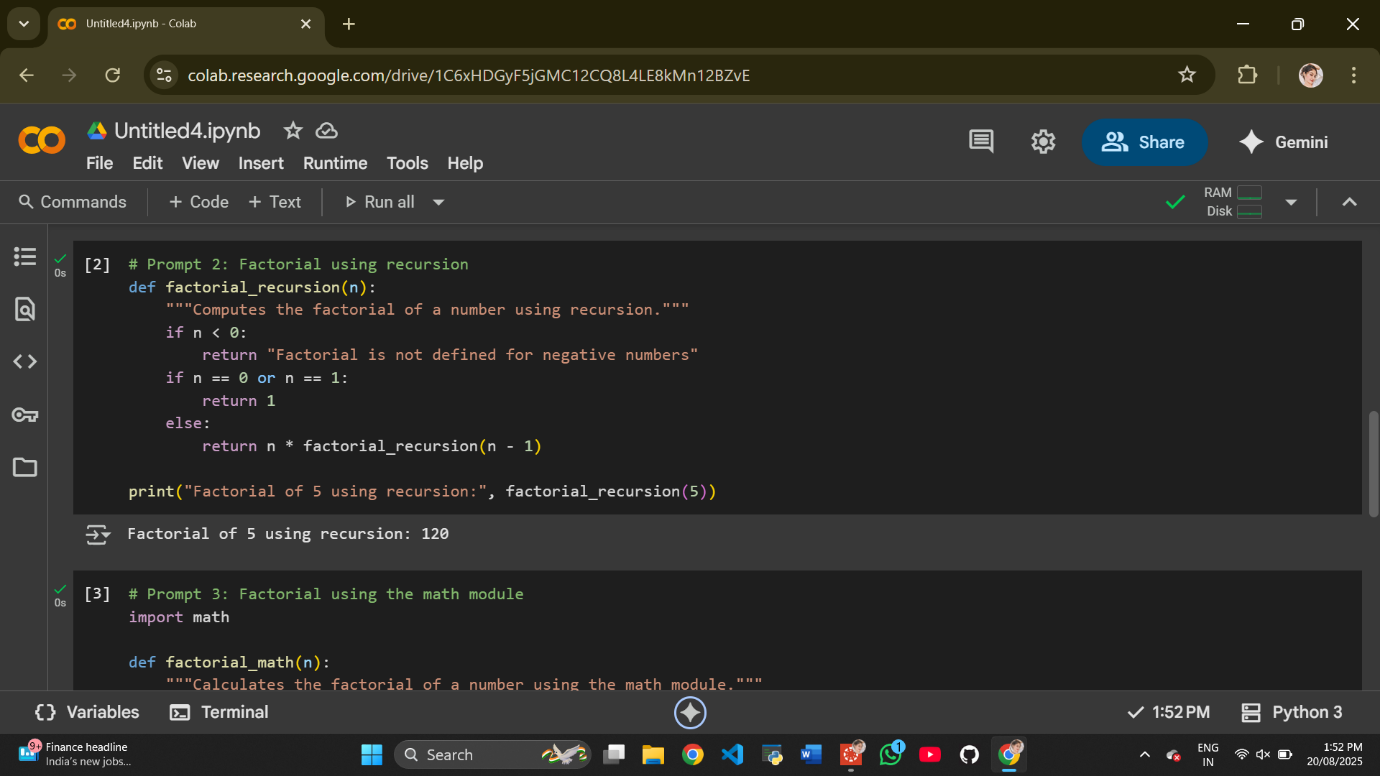
>> use gemini to generate 3 different prompts to generate a factorial function

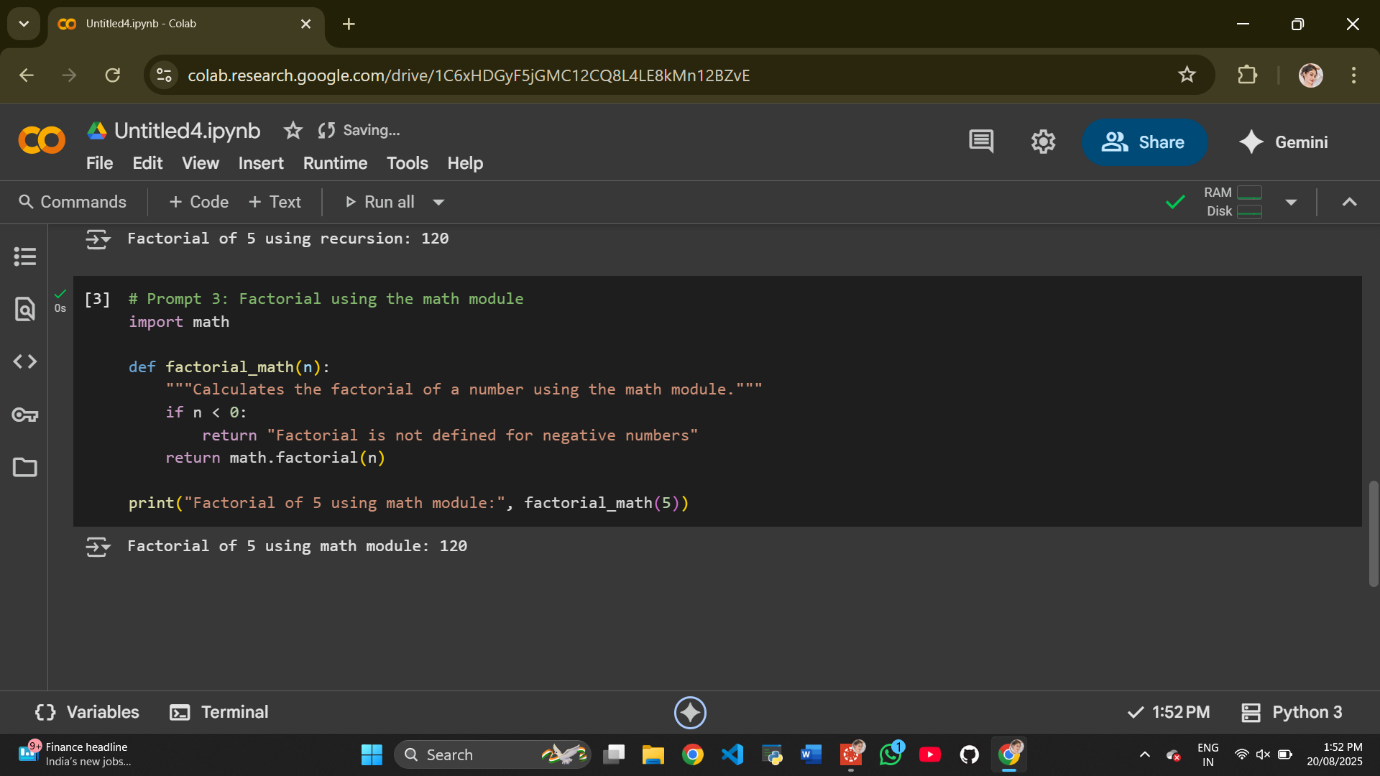
PROMPT :

>> Develop a code to generate 3 different prompts to generate a factorial function

EXPECTED OUTPUT AND SCREENSHOT :







EXPLANATION :

def factorial1(n):

if n == 0:

return 1

else:

return n \* factorial1(n-1)

* This function uses **recursion** to calculate the factorial.
* **Base Case:** The if n == 0: condition is the base case of the recursion. The factorial of 0 is defined as 1, so it returns 1 when n is 0. This stops the recursion.
* **Recursive Step:** The else: block handles the recursive calls. It returns n multiplied by the factorial of n-1. This process continues until the base case (n == 0) is reached.

**Function 2: factorial2(n)**

def factorial2(n):  
  if n < 0:  
    return "Factorial is not defined for negative numbers"  
  elif n == 0:  
    return 1  
  else:  
    result = 1  
    for i in range(1, n + 1):  
      result \*= i  
    return result

* This function uses an **iterative** approach (a loop) to calculate the factorial.
* It first checks for negative input, as factorial is not defined for negative numbers.
* It handles the base case for n == 0, returning 1.
* For positive n, it initializes a result variable to 1.
* The for loop iterates from 1 up to n (inclusive). In each iteration, it multiplies the result by the current value of i.
* Finally, it returns the accumulated result.

**Function 3: factorial3(n: int) -> int**

def factorial3(n: int) -> int:  
  """Calculates the factorial of a non-negative integer.  
  
  Args:  
    n: A non-negative integer.  
  
  Returns:  
    The factorial of n.  
  """  
  if not isinstance(n, int) or n < 0:  
      raise ValueError("Input must be a non-negative integer")  
  if n == 0:  
    return 1  
  else:  
    return n \* factorial3(n-1)

* This function is also **recursive**, similar to factorial1, but it includes **docstrings** and **type hints** for better code readability and maintainability.
* **Type Hints:** n: int indicates that the input n is expected to be an integer, and -> int indicates that the function is expected to return an integer.
* **Docstring:** The triple-quoted string is a docstring that explains what the function does, its arguments (Args), and what it returns (Returns).
* It includes an additional check to ensure the input is a non-negative integer and raises a ValueError if it's not.
* The recursive logic for calculating the factorial is the same as in factorial1.

These three functions demonstrate different ways to implement the factorial calculation in Python, showcasing recursion, iteration, and the use of docstrings and type hints.

Would you like me to summarize the generated code and explanations, or is there anything else you'd like to know?

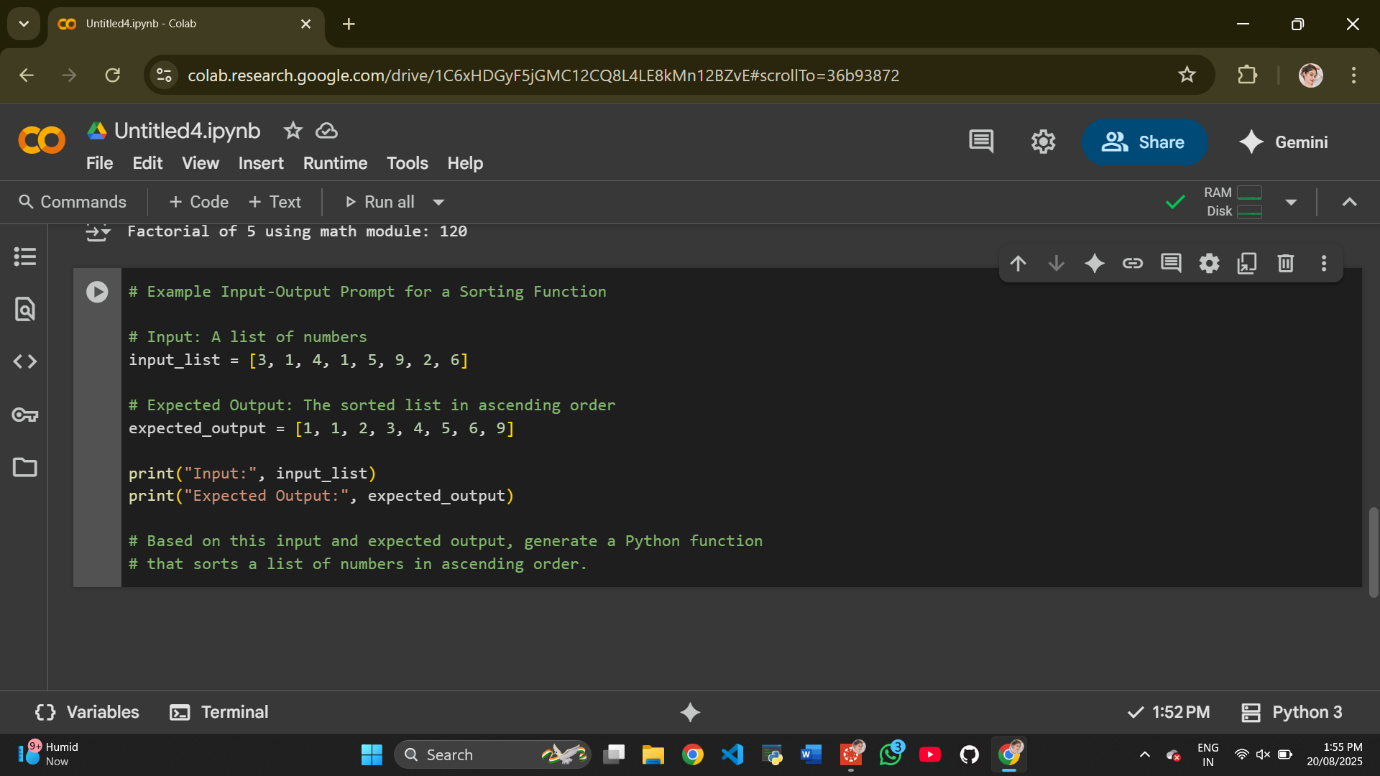
TASK DESCRIPTION#2

>>Use gemini to generate a clear example input-output prompt to generate a sorting function

PROMPT :

>>Develop a code to generate a clear example input-output prompt to generate a sorting function

EXPECTED OUTPUT AND SCREENSHOT :



EXPLANATION :