SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
ProgramName:B. Tech		**Assignment Type: Lab**	**AcademicYear:**2025-2026
CourseCoordinatorName		Venkataramana Veeramsetty	
Instructor(s)Name		1. Dr. Mohammed Ali Shaik 2. Dr. T Sampath Kumar 3. Mr. S Naresh Kumar 4. Dr. V. Rajesh 5. Dr. Brij Kishore 6. Dr Pramoda Patro 7. Dr. Venkataramana 8. Dr. Ravi Chander 9. Dr. Jagjeeth Singh	
CourseCode	24CS002PC215	**CourseTitle**	AI Assisted Coding
Year/Sem	II/I	**Regulation**	R24
Date and Day of Assignment		**Time(s)**	
Duration	2 Hours	**Applicableto Batches**	
AssignmentNumber:3.3(Present assignment number)/**24**(Total number of assignments)			

Q.No.	Question	*Expected Time to complete*
1	Lab 3: Prompt Engineering – Improving Prompts and Context Management **Lab Objectives:** • To understand how prompt structure and wording influence AI-generated code. • To explore how context (like comments and function names) helps AI generate relevant output. • To evaluate the quality and accuracy of code based on prompt clarity. • To develop effective prompting strategies for AI-assisted programming. **Lab Outcomes (LOs):** After completing this lab, students will be able to: • Generate Python code using Google Gemini in Google Colab. • Analyze the effectiveness of code explanations and suggestions by Gemini. • Set up and use Cursor AI for AI-powered coding assistance. • Evaluate and refactor code using Cursor AI features. • Compare AI tool behavior and code quality across different platforms.	03.08.2025 EOD

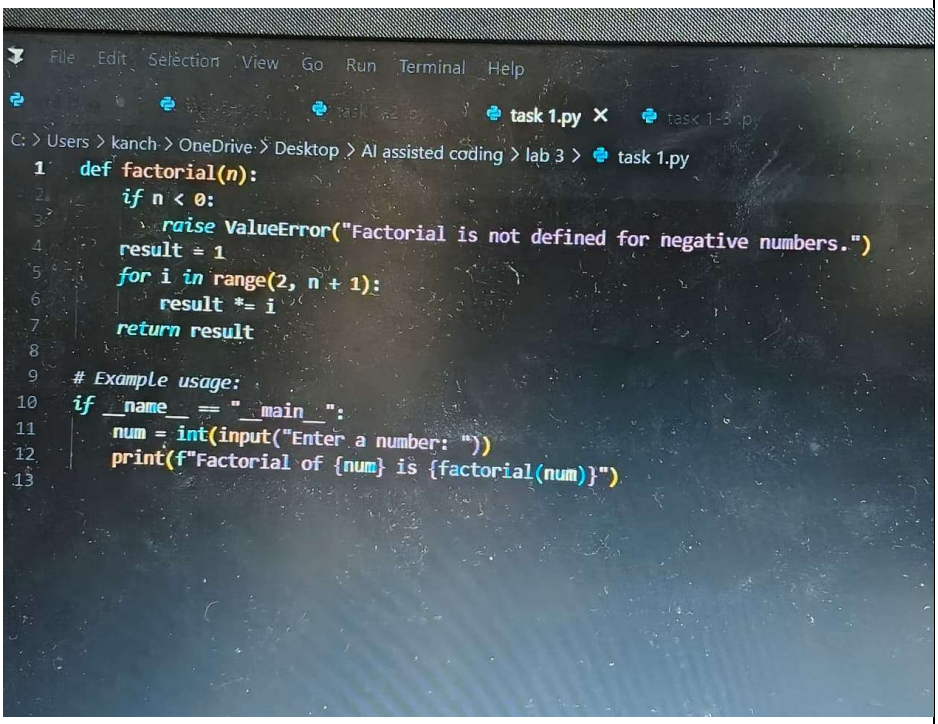
Task Description#1

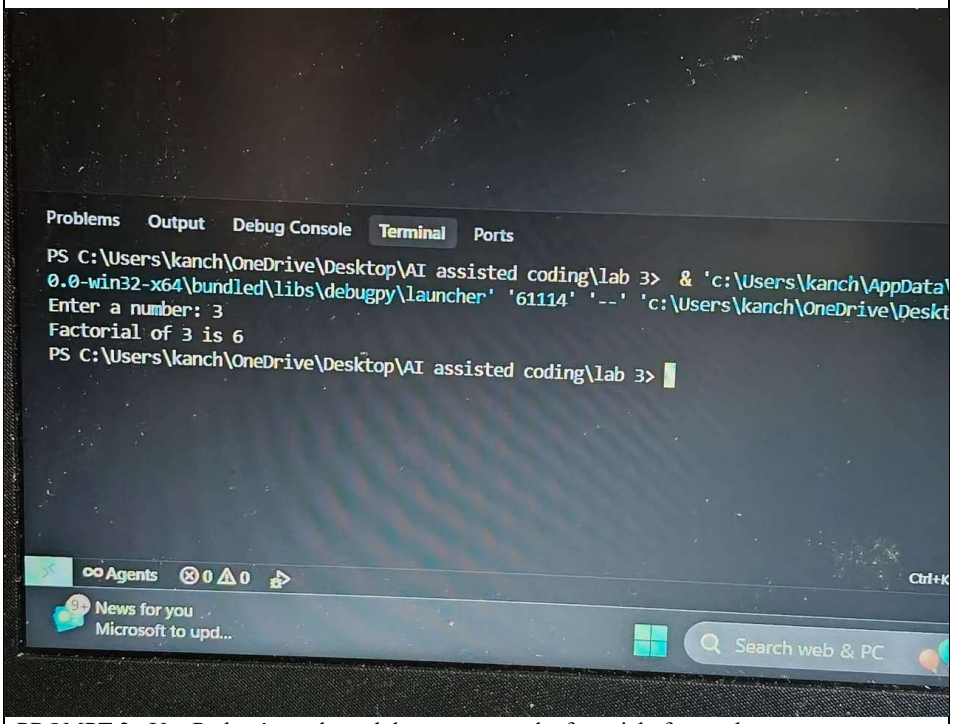
- Try 3 different prompts to generate a factorial function.

Expected Output#1

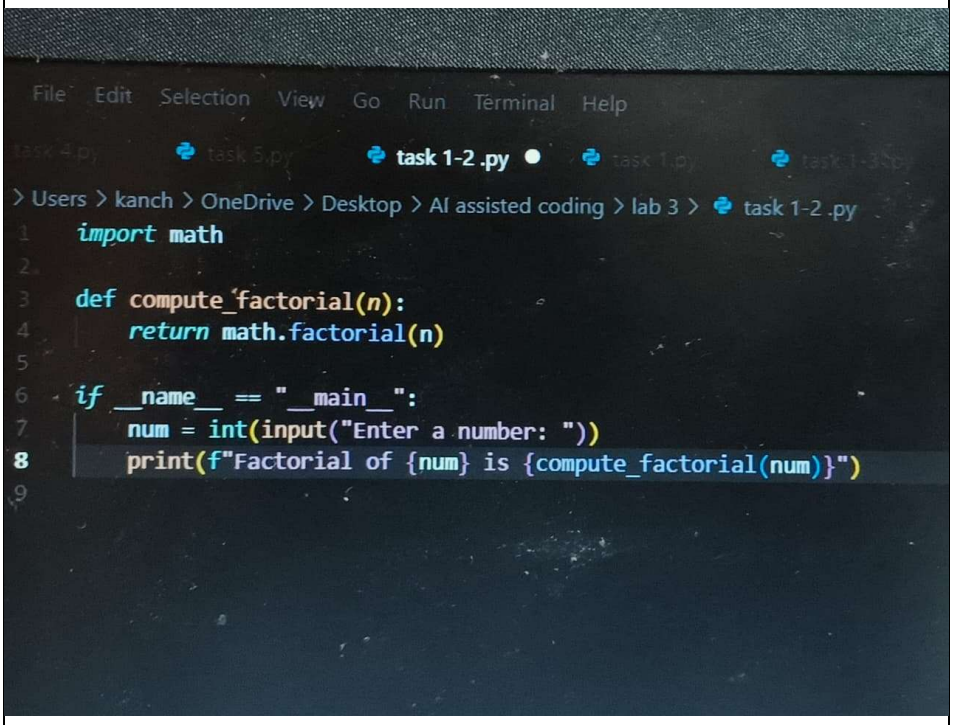
- Comparison of AI-generated code styles

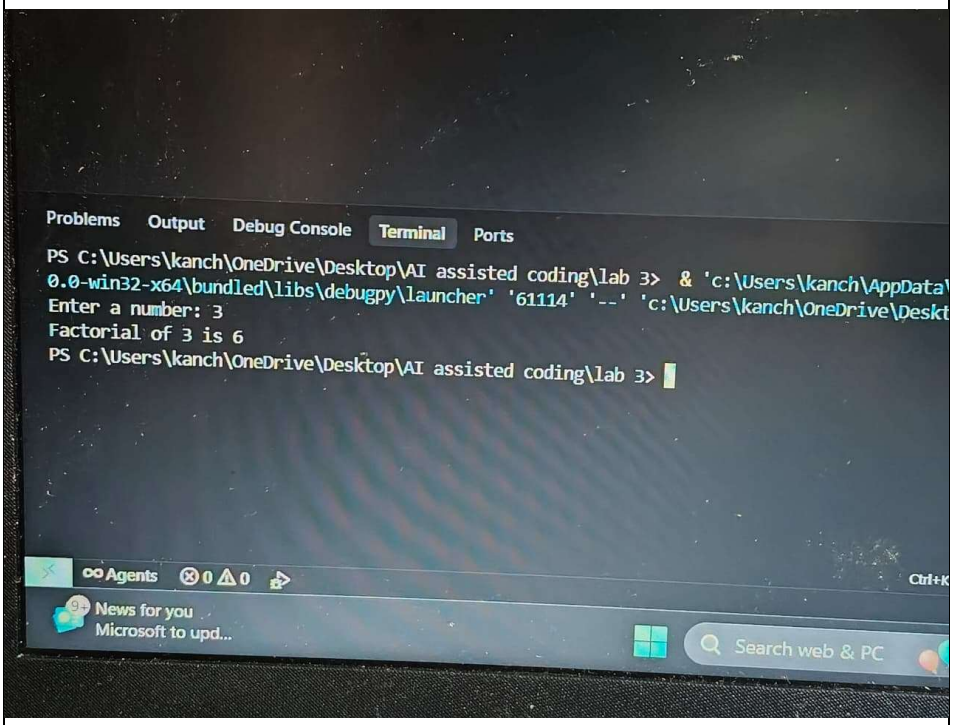
PROMPT 1: Write a python code to calculate factorial of a number





PROMPT 2 : Use Python's math module to compute the factorial of a number





PROMPT 3 : Implement factorial using static programming in python

```python
def factorial(n):
    if n < 0:
        raise ValueError("Factorial is not defined for negative numbers.")
    dp = [1] * (n + 1)
    for i in range(2, n + 1):
        dp[i] = dp[i - 1] * i
    return dp[n]

if __name__ == "__main__":
    num = int(input("Enter a number: "))
    print(f"Factorial of {num} is {factorial(num)}")
```

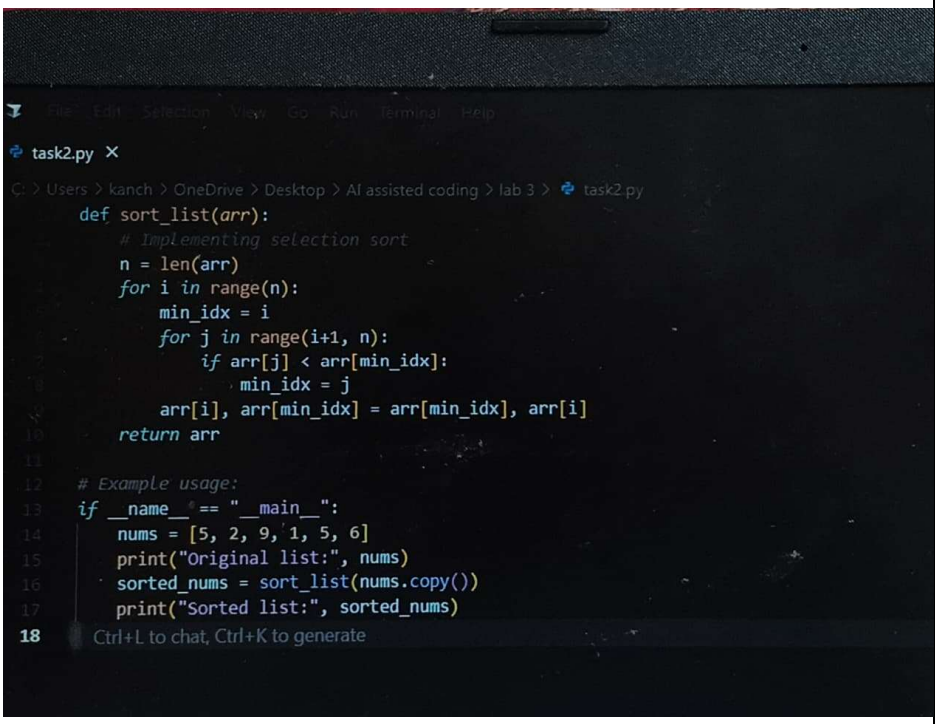
```
Problems  Output  Debug Console  Terminal  Ports
PS C:\Users\kanch\OneDrive\Desktop\AI assisted coding\lab 3> & 'c:\Users\kanch\AppData\
0.0-win32-x64\bundled\libs\debugpy\launcher' '61114' '--' 'c:\Users\kanch\OneDrive\Deskt
Enter a number: 3
Factorial of 3 is 6
PS C:\Users\kanch\OneDrive\Desktop\AI assisted coding\lab 3>
```

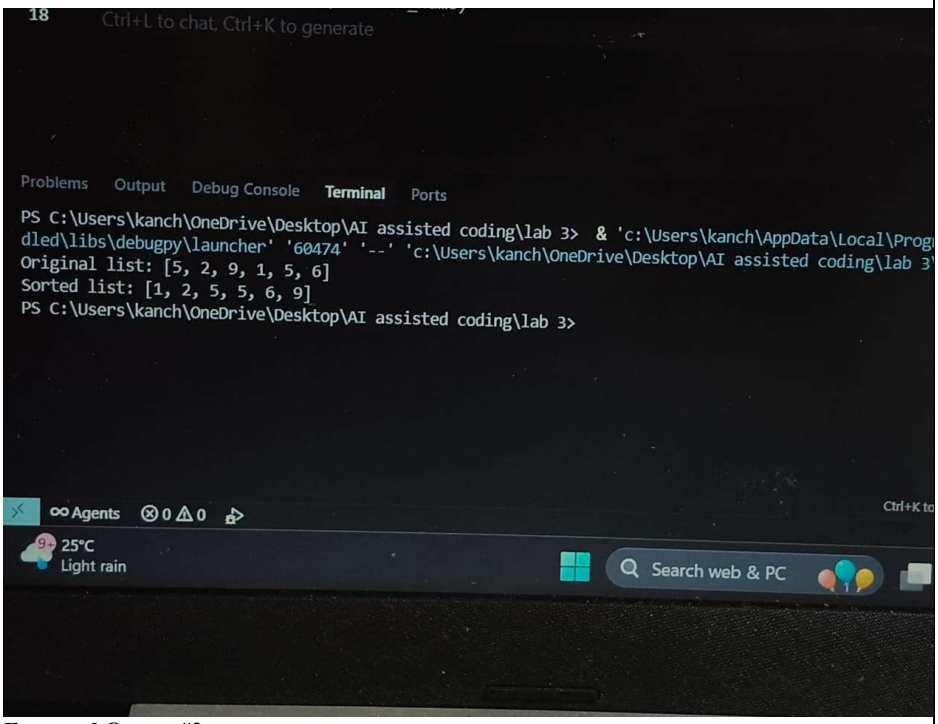
CONCLUSION: This Python script defines a function factorial(n) that calculates the **factorial** of a non-negative integer n. The script is a robust and efficient implementation of the factorial function using an iterative approach. It includes input validation for negative values and demonstrates safe, user-friendly handling of edge cases. It is suitable for basic factorial computation tasks and can be extended or integrated into larger applications.

Task Description#2

- Provide a clear example input-output prompt to generate a sorting function.

PROMPT : Given a list of integers, write a function that returns the list sorted in ascending order by using built-in sorting functions like sort() or sorted()





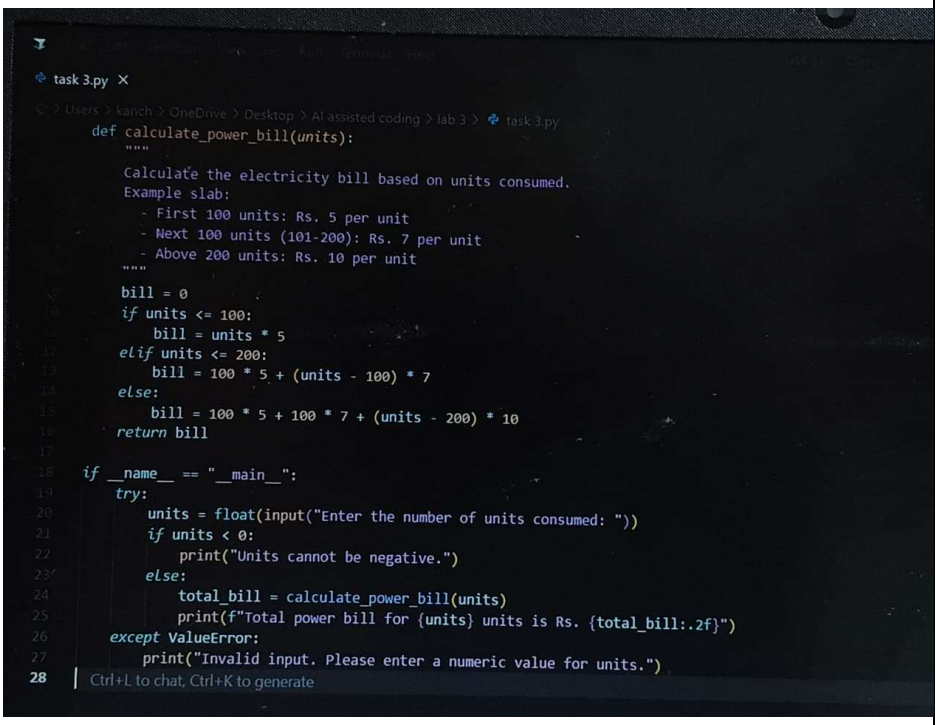
Expected Output#2

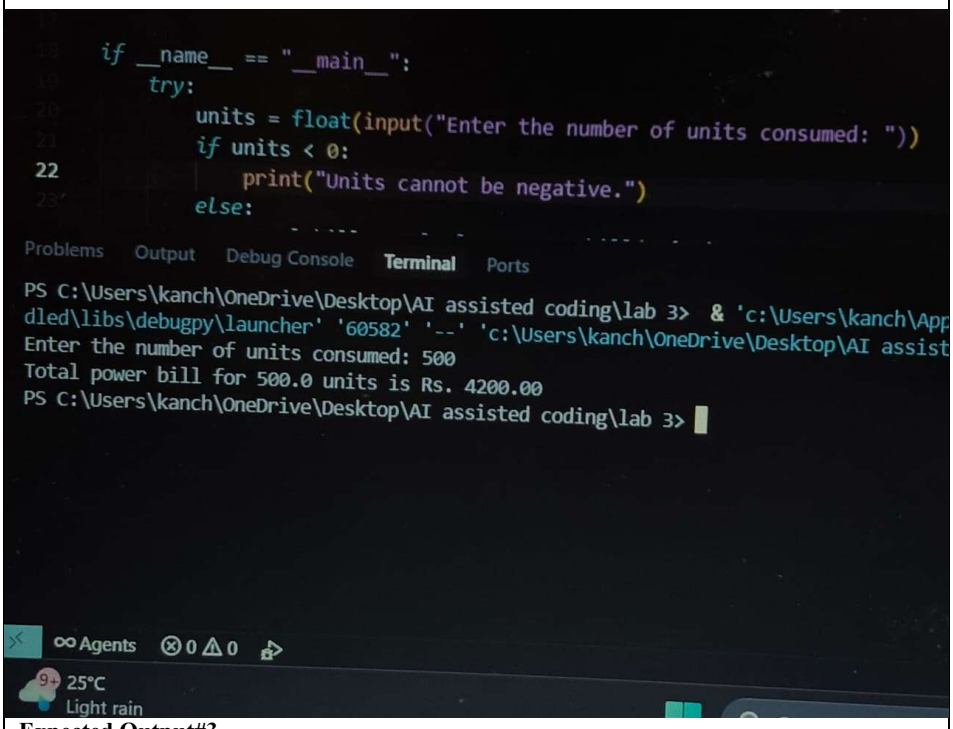
- Functional sorting code from AI

Task Description#3

- Start with the vague prompt “Generate python code to calculate power bill” and improve it step-by-step

PROMPT : Generate power bill in python





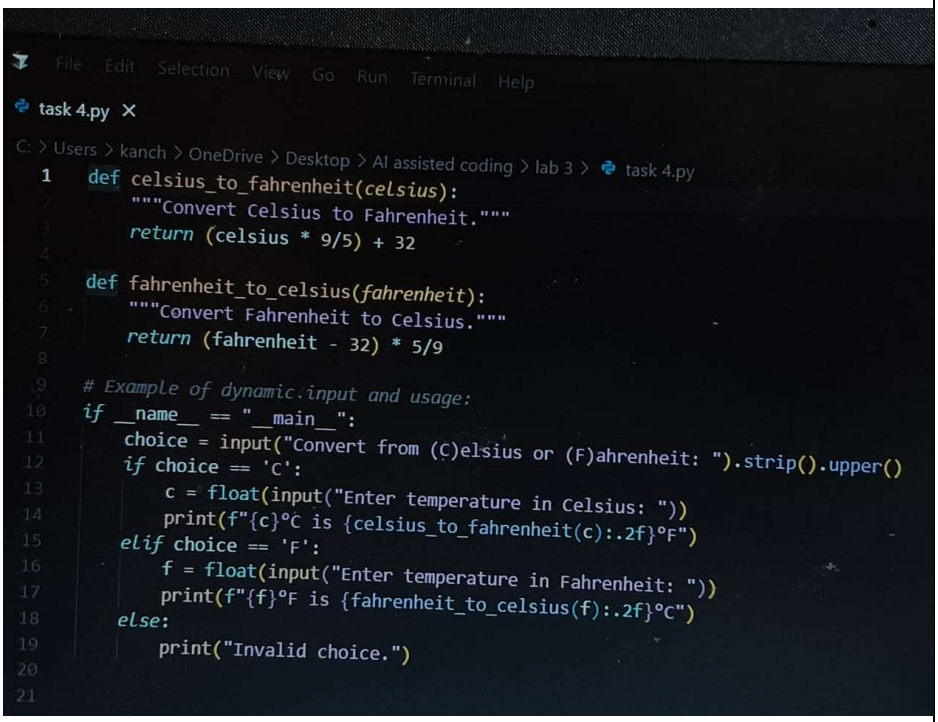
Expected Output#3

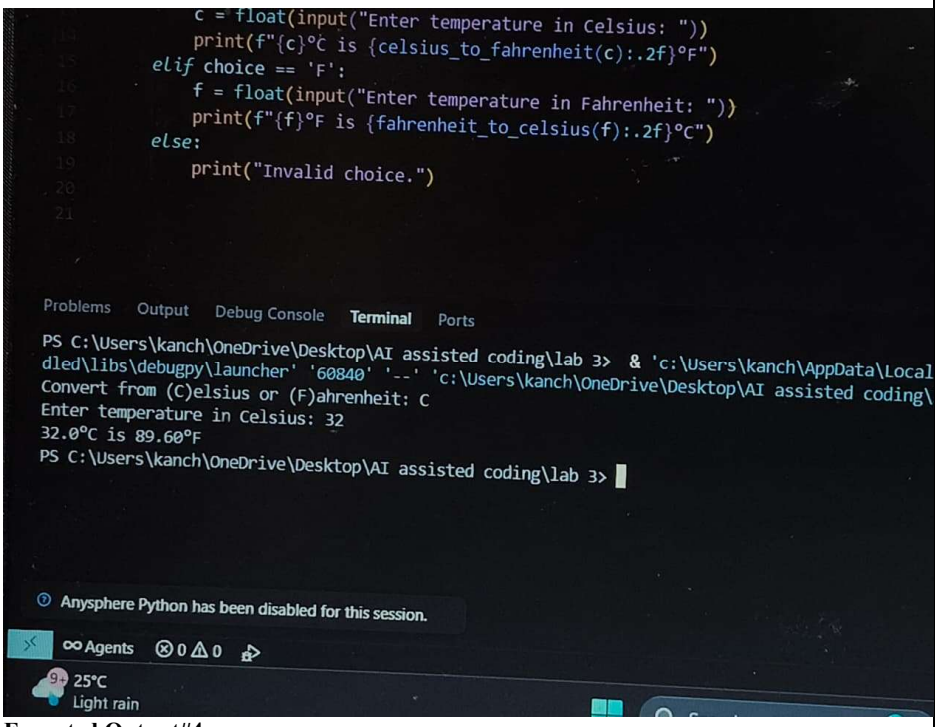
- Enhanced AI output with clearer prompts

Task Description#4

- Write structured comments to help AI generate two linked functions (e.g., login_user() and register_user()).

PROMPT : Generate two linked functions in python





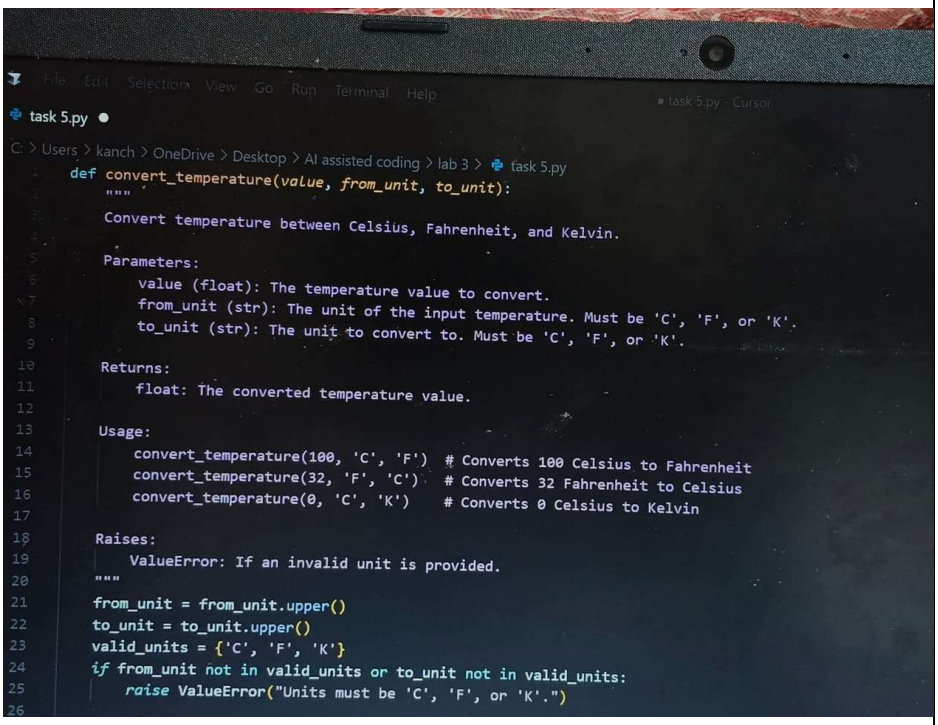
Expected Output#4

- Consistent functions with shared logic

Task Description#5

- Analyzing Prompt Specificity: Improving Temperature Conversion Function with Clear Instructions

PROMPT : Generate a python code to improve temperature conversion function with clear instruction



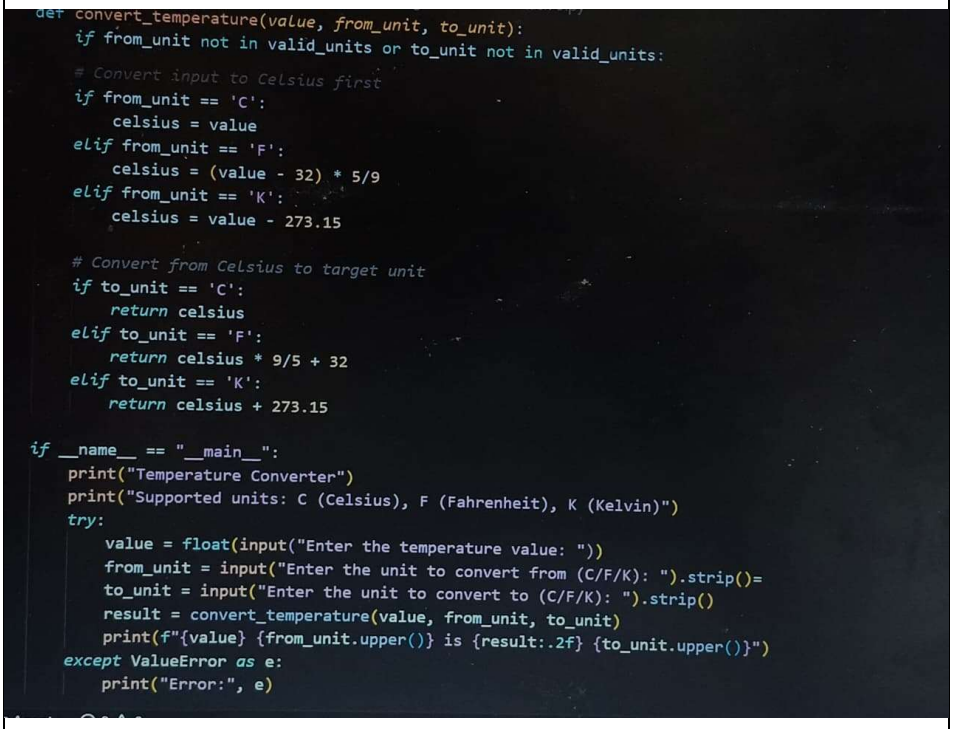
```
def convert_temperature(value, from_unit, to_unit):
    """
    Convert temperature between Celsius, Fahrenheit, and Kelvin.

    Parameters:
        value (float): The temperature value to convert.
        from_unit (str): The unit of the input temperature. Must be 'C', 'F', or 'K'.
        to_unit (str): The unit to convert to. Must be 'C', 'F', or 'K'.

    Returns:
        float: The converted temperature value.

    Usage:
        convert_temperature(100, 'C', 'F')  # Converts 100 Celsius to Fahrenheit
        convert_temperature(32, 'F', 'C')   # Converts 32 Fahrenheit to Celsius
        convert_temperature(0, 'C', 'K')    # Converts 0 Celsius to Kelvin

    Raises:
        ValueError: If an invalid unit is provided.
    """
    from_unit = from_unit.upper()
    to_unit = to_unit.upper()
    valid_units = {'C', 'F', 'K'}
    if from_unit not in valid_units or to_unit not in valid_units:
        raise ValueError("Units must be 'C', 'F', or 'K'.")
```

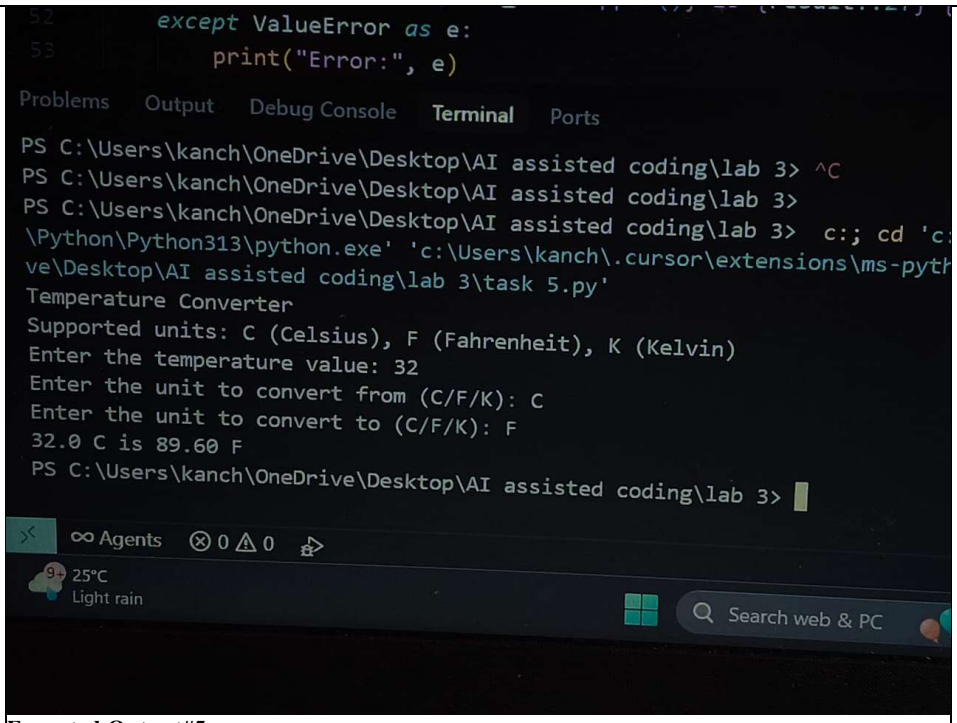


```
def convert_temperature(value, from_unit, to_unit):
    if from_unit not in valid_units or to_unit not in valid_units:

    # Convert input to Celsius first
    if from_unit == 'C':
        celsius = value
    elif from_unit == 'F':
        celsius = (value - 32) * 5/9
    elif from_unit == 'K':
        celsius = value - 273.15

    # Convert from Celsius to target unit
    if to_unit == 'C':
        return celsius
    elif to_unit == 'F':
        return celsius * 9/5 + 32
    elif to_unit == 'K':
        return celsius + 273.15

if __name__ == "__main__":
    print("Temperature Converter")
    print("Supported units: C (Celsius), F (Fahrenheit), K (Kelvin)")
    try:
        value = float(input("Enter the temperature value: "))
        from_unit = input("Enter the unit to convert from (C/F/K): ").strip()=
        to_unit = input("Enter the unit to convert to (C/F/K): ").strip()
        result = convert_temperature(value, from_unit, to_unit)
        print(f"{value} {from_unit.upper()} is {result:.2f} {to_unit.upper()}")
    except ValueError as e:
        print("Error:", e)
```



Expected Output#5

- Code quality difference analysis for various prompts

Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots

Evaluation Criteria:

Criteria	Max Marks
Factorial Function (Task#1)	0.5
Sorting Function (Task#2)	0.5
Vogue Vs. Specific Prompting (Task #3)	0.5
Linked Functions (Task #4)	0.5
Temperature Conversion Function (Task #5)	0.5
Total	**2.5 Marks**