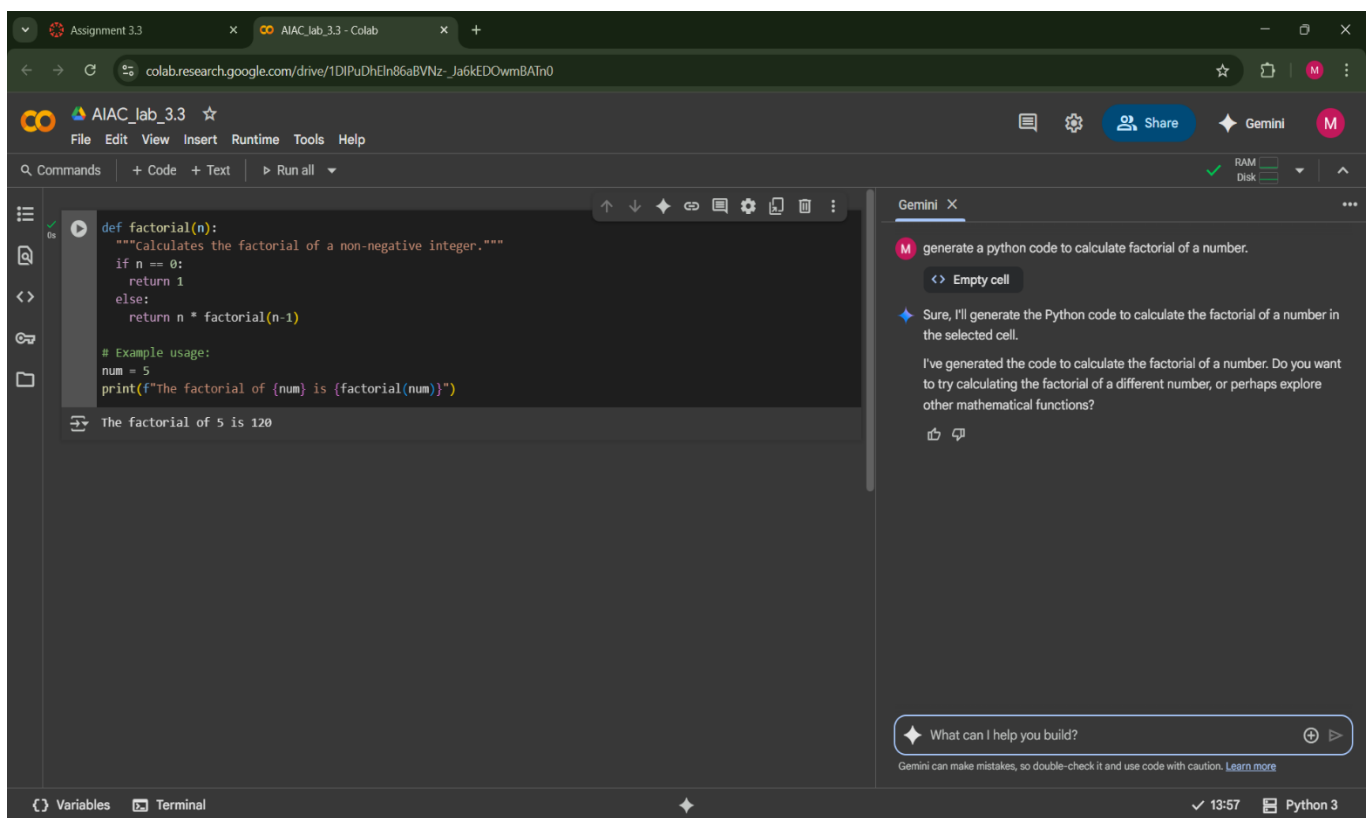


-----AI ASSISTED CODING-----

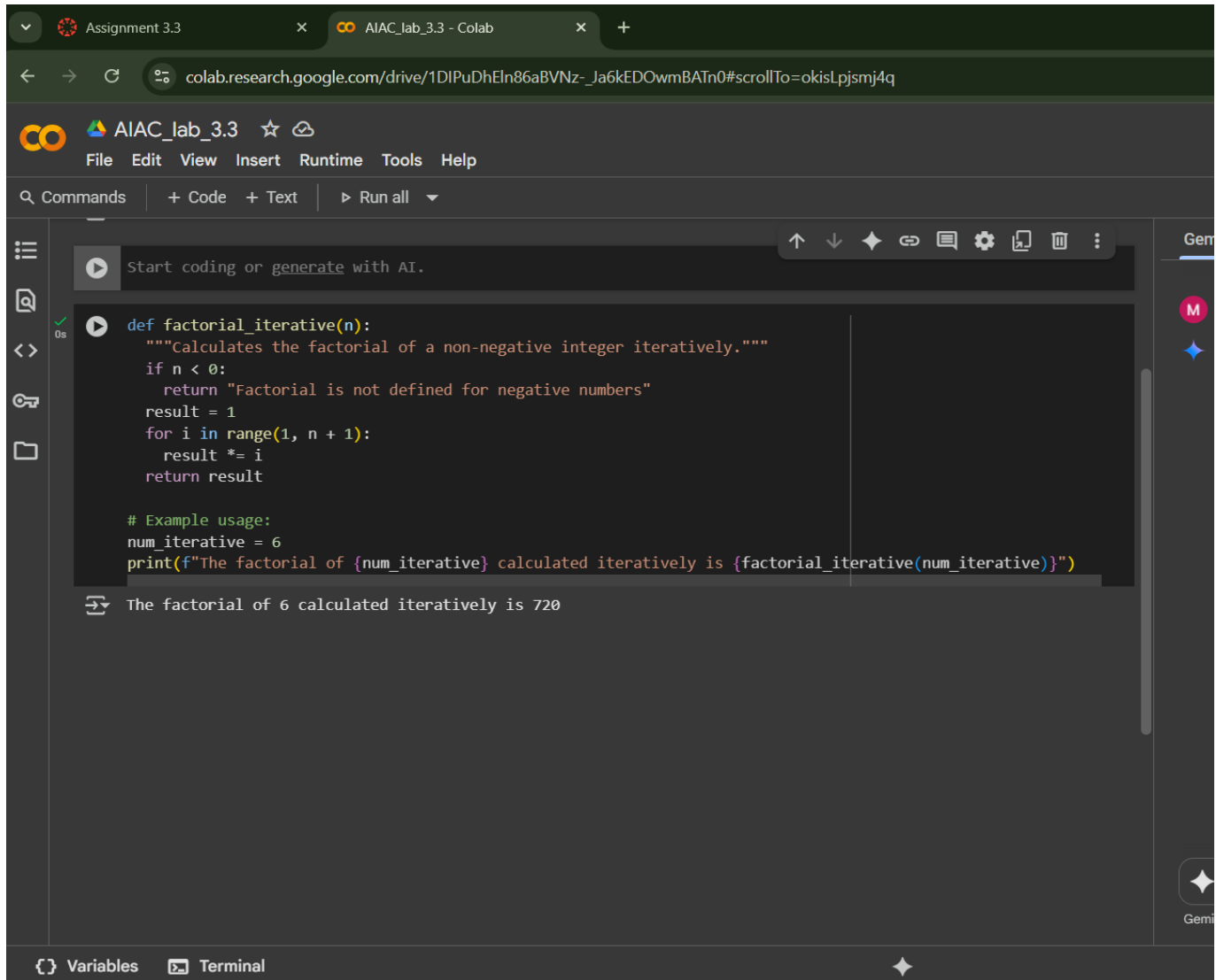
-----LAB ASSIGNMENT 3.3-----

TASK DESCRIPTION 1 : Try 3 different prompts to generate a factorial function

PROMPT 1 : generate a python code to print the factorial of a given number :



PROMPT 2 : generate a python code to print factorial of a number using iterations :



The screenshot shows a Google Colab notebook interface. The browser tabs at the top are 'Assignment 3.3' and 'AIAC_lab_3.3 - Colab'. The address bar shows the URL: colab.research.google.com/drive/1DIPuDhElN86aBVNz-_Ja6kEDOWmBATn0#scrollTo=okisLpjsmj4q. The notebook title is 'AIAC_lab_3.3'. The menu bar includes 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. Below the menu bar is a toolbar with 'Commands', '+ Code', '+ Text', and 'Run all'. The left sidebar contains icons for file explorer, search, and other notebook functions. The main code cell contains the following Python code:

```
def factorial_iterative(n):  
    """Calculates the factorial of a non-negative integer iteratively."""  
    if n < 0:  
        return "Factorial is not defined for negative numbers"  
    result = 1  
    for i in range(1, n + 1):  
        result *= i  
    return result  
  
# Example usage:  
num_iterative = 6  
print(f"The factorial of {num_iterative} calculated iteratively is {factorial_iterative(num_iterative)}")
```

Below the code cell, the output is displayed: 'The factorial of 6 calculated iteratively is 720'. The bottom status bar shows 'Variables' and 'Terminal' tabs.

PROMPT 3 : generate a python code to print factorial of a given number using conditional statements.

The screenshot shows a Google Colab notebook interface. The browser tabs at the top include 'Assignment 3.3' and 'AIAC_lab_3.3 - Colab'. The address bar shows a Google Drive link. The notebook title is 'AIAC_lab_3.3'. The menu bar includes 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. Below the menu is a 'Commands' bar with '+ Code', '+ Text', and 'Run all'. The code editor contains the following Python code:

```
n = int(input("Enter a non-negative integer to calculate its factorial: "))

if n < 0:
    print("Factorial is not defined for negative numbers.")
elif n == 0:
    print(f"The factorial of {n} is 1")
else:
    f = 1
    for i in range(1, n + 1):
        f *= i
    print(f"The factorial of {n} is {f}")
```

The output of the code is displayed below the editor:

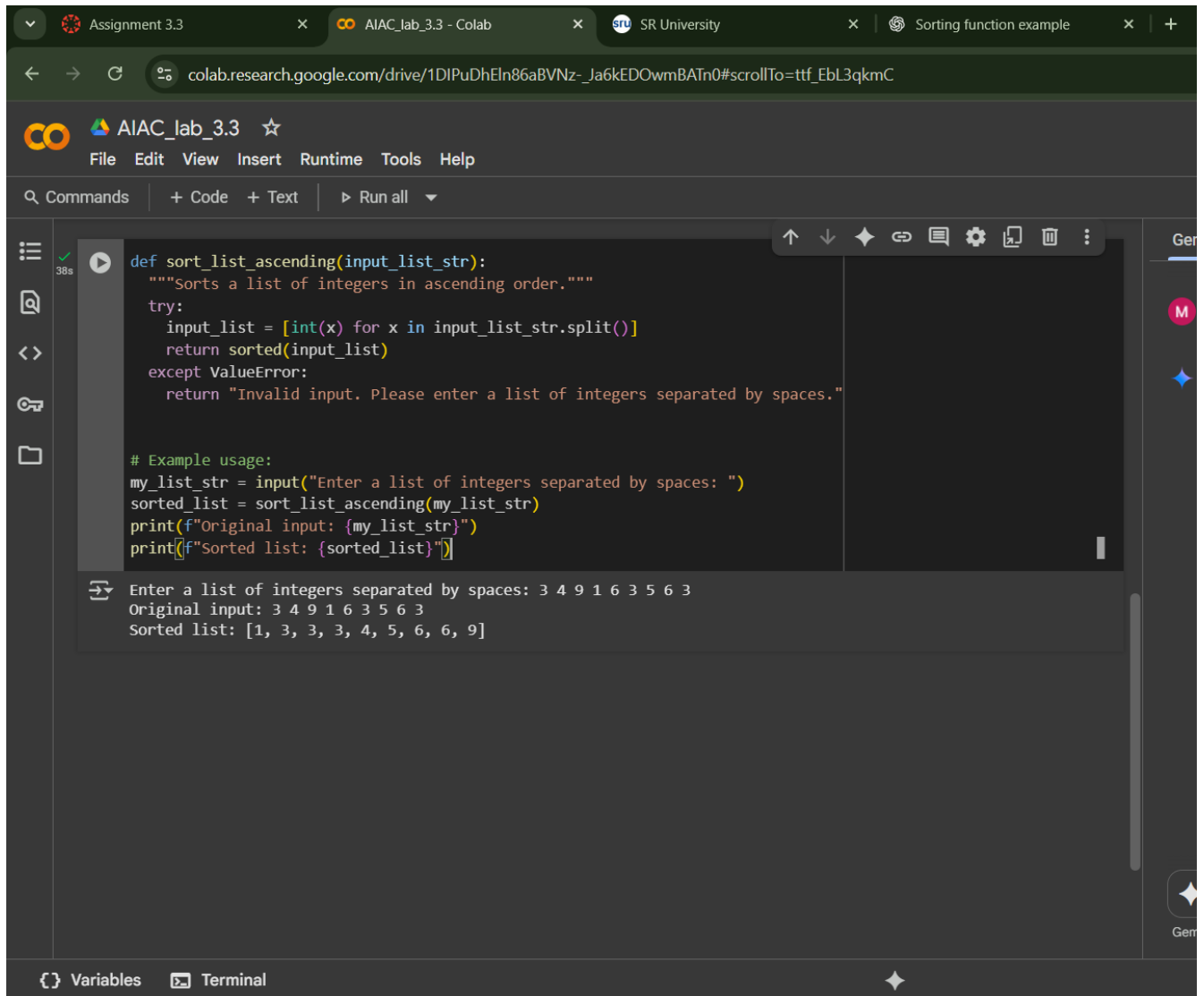
```
Enter a non-negative integer to calculate its factorial: 3
The factorial of 3 is 6
```

The bottom of the notebook shows tabs for 'Variables' and 'Terminal'.

Here are three codes which are different from each other in terms of logic . first one is using a function second one is iteration and third one is using conditional statements.so by this we can say that the same problem statement may has multiple codes based on the prompt given by the user.

TASK DESCRIPTION 2 : Provide a clear example input-output prompt to generate a sorting function

USED PROMPT : Write a Python code that takes a list of integers and returns the same list sorted in ascending order.



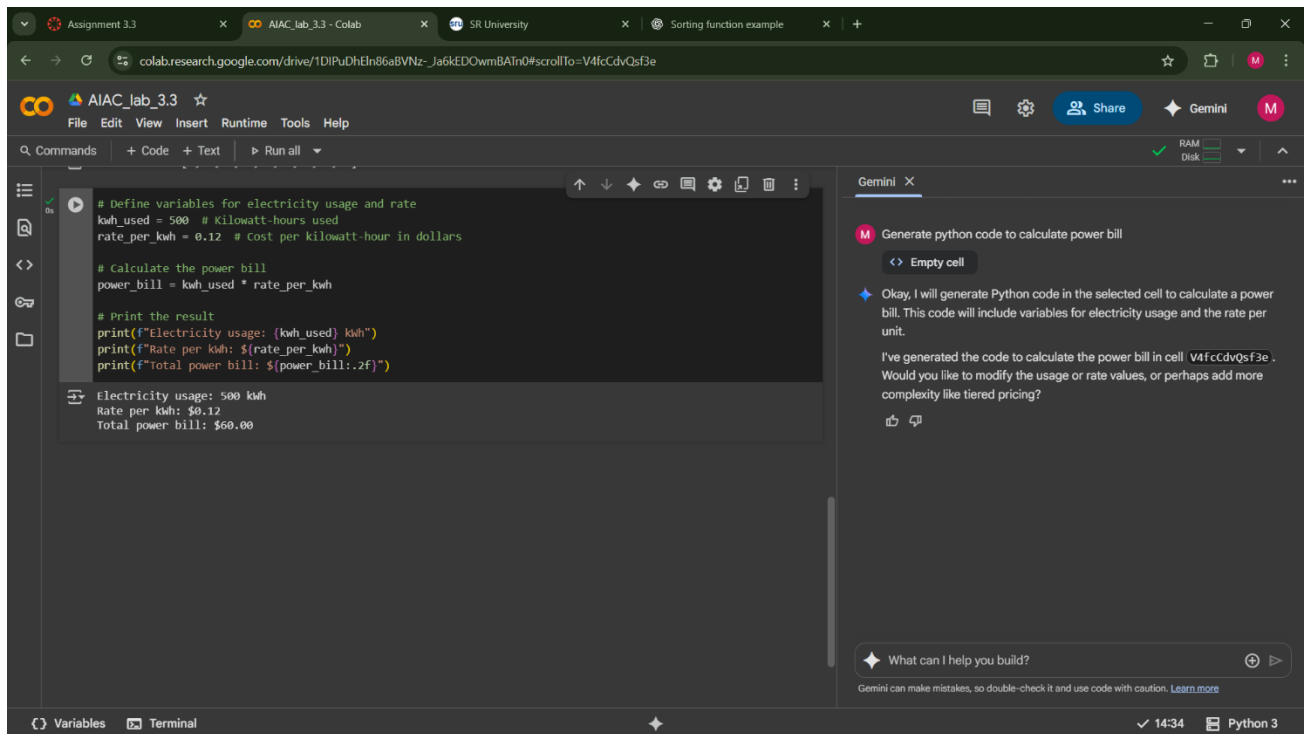
The screenshot shows a Google Colab notebook titled "AIAC_lab_3.3". The code defines a function `sort_list_ascending` that takes a string of integers separated by spaces and returns a sorted list. The function uses a try-except block to handle invalid input. An example usage is provided, showing the input string "3 4 9 1 6 3 5 6 3" being converted to a list, sorted, and printed as "Sorted list: [1, 3, 3, 3, 4, 5, 6, 6, 9]".

```
def sort_list_ascending(input_list_str):  
    """Sorts a list of integers in ascending order."""  
    try:  
        input_list = [int(x) for x in input_list_str.split()]  
        return sorted(input_list)  
    except ValueError:  
        return "Invalid input. Please enter a list of integers separated by spaces."  
  
# Example usage:  
my_list_str = input("Enter a list of integers separated by spaces: ")  
sorted_list = sort_list_ascending(my_list_str)  
print(f"Original input: {my_list_str}")  
print(f"Sorted list: {sorted_list}")
```

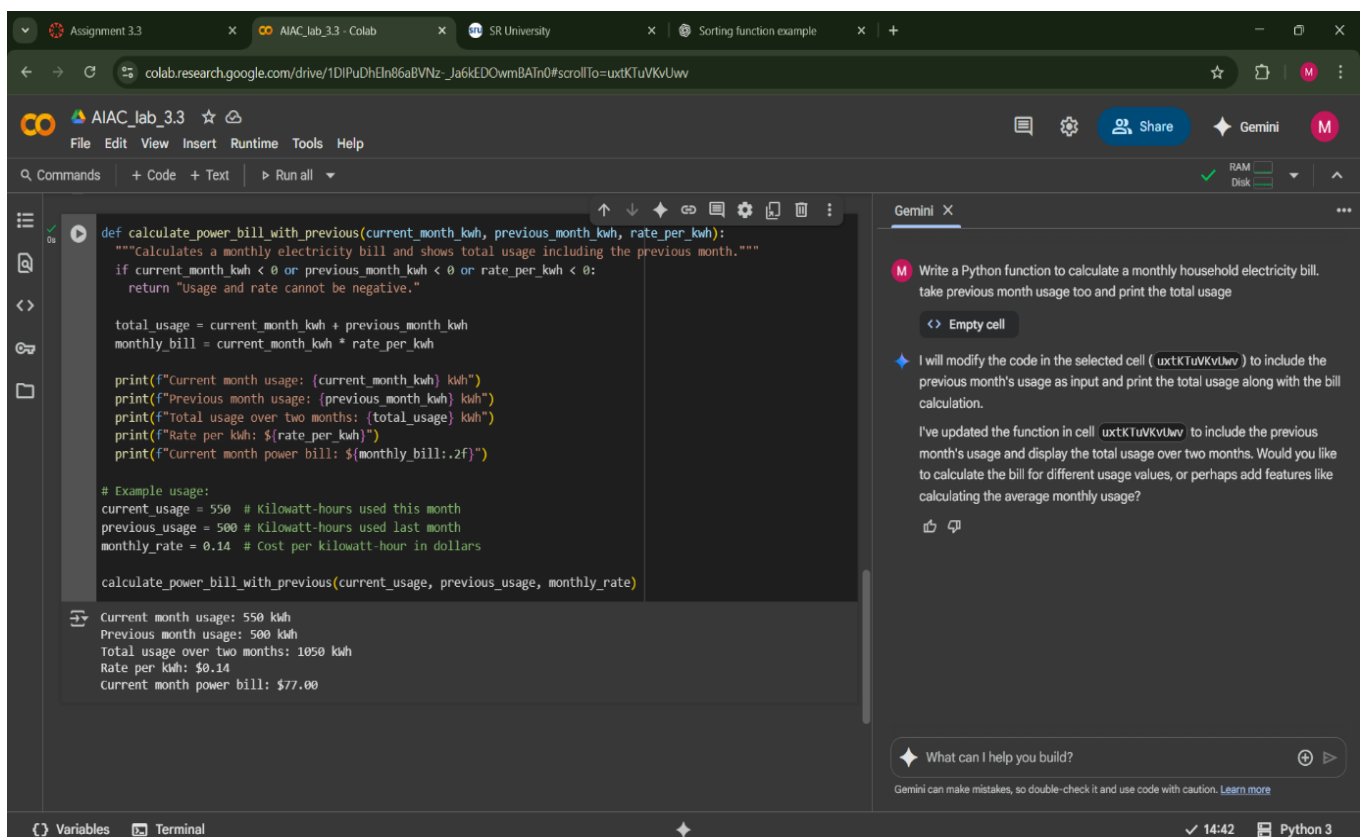
Enter a list of integers separated by spaces: 3 4 9 1 6 3 5 6 3
Original input: 3 4 9 1 6 3 5 6 3
Sorted list: [1, 3, 3, 3, 4, 5, 6, 6, 9]

TASK DESCRIPTION 3 : Start with the vague prompt “Generate python code to calculate power bill” and improve it step-by-step

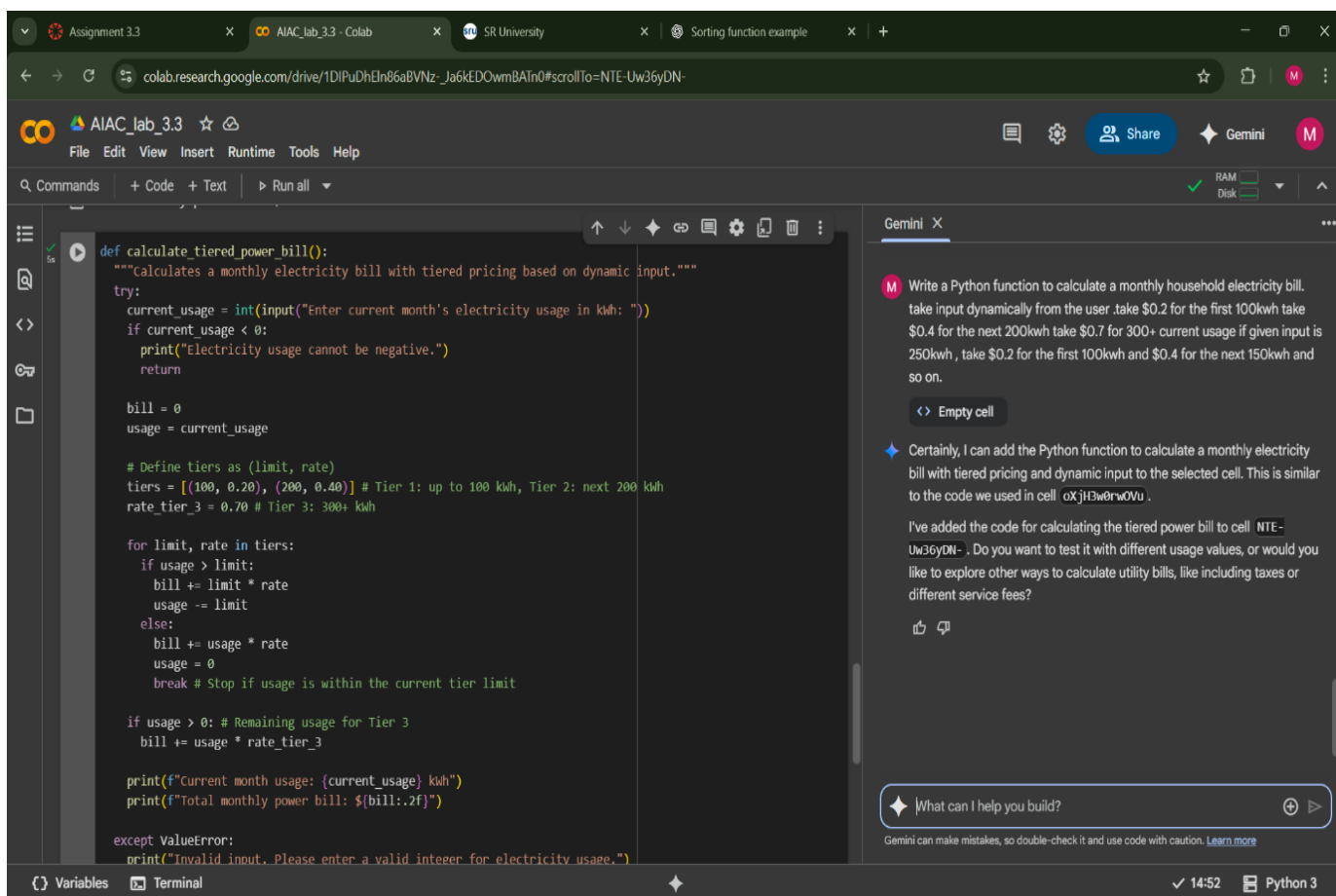
PROMPT 1 : generate a python code to calculate power bill :



PROMPT 2 : generate a python code to calculate power bill take previous month usage too from the user and print the total usage



PROMPT 3 : generate a python code to calculate power bill take input dynamically from the user . take \$0.2 for the first 100kWh , take \$0.4 dollars for the next 200kWh , take \$0.7 for 300+ current usage.



```
def calculate_tiered_power_bill():
    """Calculates a monthly electricity bill with tiered pricing based on dynamic input."""
    try:
        current_usage = int(input("Enter current month's electricity usage in kWh: "))
        if current_usage < 0:
            print("Electricity usage cannot be negative.")
            return

        bill = 0
        usage = current_usage

        # Define tiers as (limit, rate)
        tiers = [(100, 0.20), (200, 0.40)] # Tier 1: up to 100 kWh, Tier 2: next 200 kWh
        rate_tier_3 = 0.70 # Tier 3: 300+ kWh

        for limit, rate in tiers:
            if usage > limit:
                bill += limit * rate
                usage -= limit
            else:
                bill += usage * rate
                usage = 0
                break # Stop if usage is within the current tier limit

        if usage > 0: # Remaining usage for Tier 3
            bill += usage * rate_tier_3

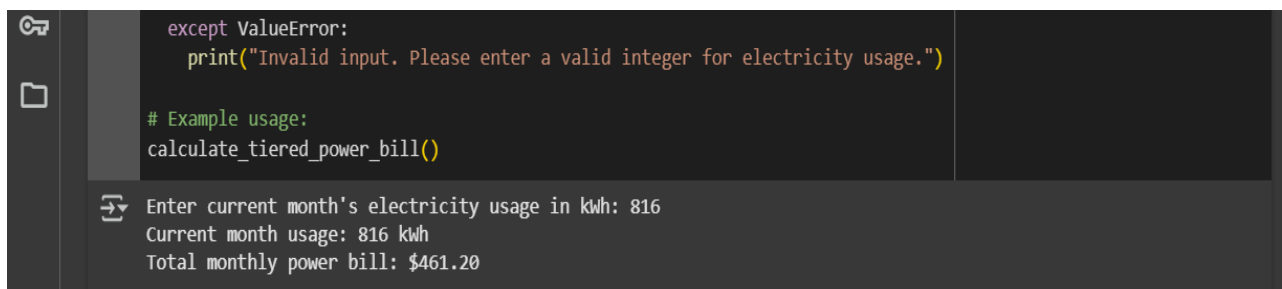
        print(f"Current month usage: {current_usage} kWh")
        print(f"Total monthly power bill: ${bill:.2f}")

    except ValueError:
        print("Invalid input. Please enter a valid integer for electricity usage.")
```

Write a Python function to calculate a monthly household electricity bill. take input dynamically from the user. take \$0.2 for the first 100kwh take \$0.4 for the next 200kwh take \$0.7 for 300+ current usage if given input is 250kwh, take \$0.2 for the first 100kwh and \$0.4 for the next 150kwh and so on.

Certainly, I can add the Python function to calculate a monthly electricity bill with tiered pricing and dynamic input to the selected cell. This is similar to the code we used in cell oXjh3w0rw0Vu .

I've added the code for calculating the tiered power bill to cell NTE-Uw36yDN- . Do you want to test it with different usage values, or would you like to explore other ways to calculate utility bills, like including taxes or different service fees?



```
except ValueError:
    print("Invalid input. Please enter a valid integer for electricity usage.")

# Example usage:
calculate_tiered_power_bill()

Enter current month's electricity usage in kWh: 816
Current month usage: 816 kWh
Total monthly power bill: $461.20
```

here , as the 1st description we make 3 prompts starting from very basic to advanced ones to calculate the biil of the household

TASK DESCRIPTION 4 : Write structured comments to help AI generate two linked functions (e.g., login_user() and register_user()).

USED PROMPT : generate a python code to take input from the user as 1 or 2 (1 for login and 2 for register) if it is a login user then ask the user to enter the username and password only and if it is a register user ask to enter all the details necessary to register into a particular app/website

The screenshot displays a Google Colab notebook titled "AIAC_lab_3.3". The notebook contains a Python function named `handle_user_action()` that handles user login and registration based on input. The function uses `input()` to collect user details and `print()` to display the collected information. It includes comments indicating where validation and logic should be implemented.

```
def handle_user_action():
    """Handles user login or registration based on input."""
    try:
        action = int(input("Enter 1 for login or 2 for register: "))

        if action == 1:
            username = input("Enter your username: ")
            password = input("Enter your password: ")
            print("\n--- Login Details ---")
            print(f"Username: {username}")
            print(f>Password: {password}")
            # In a real application, you would add login validation here
            print("Login functionality would be implemented here.")
        elif action == 2:
            print("\nPlease enter your registration details:")
            username = input("Enter a username: ")
            password = input("Enter a password: ")
            email = input("Enter your email address: ")
            full_name = input("Enter your full name: ")
            print("\n--- Registration Details ---")
            print(f"Username: {username}")
            print(f>Password: {password}")
            print(f>Email: {email}")
            print(f"Full Name: {full_name}")
            # In a real application, you would add registration logic here
            print("Registration functionality would be implemented here.")
        else:
            print("Invalid input. Please enter 1 for login or 2 for register.")

    except ValueError:
        print("Invalid input. Please enter a number (1 or 2).")
```

The Gemini AI chat interface on the right shows the following interaction:

Generate a linked code to take the input as 1 or 2 (1 for login user) and (2 for register user) if it is a login user then ask the user to enter the username and password only or if the user is a register user ask to enter all the necessary details to register into a particular app/website.

Empty cell

Okay, I will generate code in the selected cell to handle user login or registration based on your input.

I have generated the code to handle user login and registration based on your input.

What would you like to do next? Here are a few suggestions:

- **Add validation:** Implement checks to ensure usernames and passwords meet certain criteria (e.g., minimum length, complexity).
- **Store user data:** Set up a way to store registered user information (e.g., in a dictionary, file, or database).
- **Implement login logic:** Add code to verify the entered username and password against stored data.
- **Expand registration details:** Include more fields for registration, such as address or phone number.
- **Explore different input methods:** Instead of command line input, consider using a graphical user interface (GUI) library like Tkinter or PyQt.

What can I help you build?

Gemini can make mistakes, so double-check it and use code with caution. [Learn more](#)

Variables Terminal 18:45 Python 3

```
print(f"Email: {email}")
print(f"Full Name: {full_name}")
# In a real application, you would add registration logic here
print("Registration functionality would be implemented here.")
else:
    print("Invalid input. Please enter 1 for login or 2 for register.")

except ValueError:
    print("Invalid input. Please enter a number (1 or 2).")

# Example usage:
handle_user_action()
```

Enter 1 for login or 2 for register: 1
Enter your username: mustafa
Enter your password: mustafa2006

--- Login Details ---
Username: mustafa
Password: mustafa2006
Login functionality would be implemented here.

TASK DESCRIPTION 5 : Analysing Prompt Specificity: Improving Temperature Conversion Function with Clear Instructions.

PROMPT 1 : generate a python code to convert the temperature from Celsius to Fahrenheit .

The screenshot shows a Google Colab notebook interface. The top bar includes the Google Colab logo, the file name 'AIAC_lab_3.3', and a 'Saving...' status. The menu bar contains 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. The toolbar has icons for 'Commands', '+ Code', '+ Text', 'Run all', and a dropdown menu. The main code cell contains the following Python code:

```
[4] # Example usage:
celsius_temp = 25
fahrenheit_temp = (celsius_temp * 9/5) + 32
print(f"{celsius_temp}°C is equal to {fahrenheit_temp}°F")
```

The output of the code is displayed below the cell:

```
25°C is equal to 77.0°F
```

On the right side of the notebook, there is a 'Gemini' chat window. It contains a prompt: 'generate a basic python code to convert the temperature from celcius to fahrenheit.' and a response: 'Sure, I can help with that! Here is a basic Python code to convert temperature from Celsius to Fahrenheit.'

PROMPT 2 : generate a python code to convert the temperature from Celsius to Fahrenheit or Fahrenheit to Celsius. Take input accordingly.

The screenshot displays a Google Colab notebook environment. The top bar shows the notebook name 'AIAC_lab_3.3' and various icons for file management, runtime, and sharing. The main code cell contains a Python function `convert_temperature()` that prompts the user to choose between Celsius to Fahrenheit ('C') or Fahrenheit to Celsius ('F'). It then takes a temperature value and performs the conversion using the formulas $F = C \times \frac{9}{5} + 32$ and $C = (F - 32) \times \frac{5}{9}$. The function includes error handling for invalid inputs and an example usage section. The output of the code shows the user entering 'C' and a temperature of 39, resulting in 102.2°F.

On the right side, the Gemini AI chat interface is visible. It shows a prompt: 'generate a python code to convert celsius to fahrenheit or fahrenheit to celsius . take input from the user accordingly.' The AI responds with a confirmation and provides several suggestions for enhancing the code, such as adding input validation, supporting multiple conversions, including other temperature scales, and creating a more user-friendly interface using libraries like Tkinter or PyQt.

```
def convert_temperature():  
    """Converts temperature between Celsius and Fahrenheit based on user input."""  
    try:  
        choice = input("Enter 'C' to convert Celsius to Fahrenheit or 'F' to convert Fahrenheit to Celsius: ").upper()  
  
        if choice == 'C':  
            celsius = float(input("Enter temperature in Celsius: "))  
            fahrenheit = (celsius * 9/5) + 32  
            print(f"{celsius}°C is equal to {fahrenheit}°F")  
        elif choice == 'F':  
            fahrenheit = float(input("Enter temperature in Fahrenheit: "))  
            celsius = (fahrenheit - 32) * 5/9  
            print(f"{fahrenheit}°F is equal to {celsius}°C")  
        else:  
            print("Invalid input. Please enter 'C' or 'F'.")  
  
    except ValueError:  
        print("Invalid input. Please enter a valid number for temperature.")  
  
# Example usage:  
convert_temperature()
```

Enter 'C' to convert Celsius to Fahrenheit or 'F' to convert Fahrenheit to Celsius: C
Enter temperature in Celsius: 39
39.0°C is equal to 102.2°F

Gemini X

M generate a python code to convert celsius to fahrenheit or fahrenheit to celsius . take input from the user accordingly.

Empty cell

Okay, I will generate code in the selected cell to convert between Celsius and Fahrenheit based on your input.

I have generated the code to convert temperature between Celsius and Fahrenheit based on user input.

Here are a few ideas for what you could do next:

- **Add input validation:** Implement more robust checks to ensure the user enters valid numerical input for the temperature.
- **Loop for multiple conversions:** Modify the code to allow the user to perform multiple conversions without rerunning the cell.
- **Include other temperature scales:** Add options to convert to or from Kelvin.
- **Create a more user-friendly interface:** If you're interested in building applications with a graphical interface, you could use libraries like Tkinter or PyQt to create a window with input fields and buttons for the conversion.

What can I help you build?

Gemini can make mistakes, so double-check it and use code with caution. [Learn more](#)

Variables Terminal 18:59 Python 3

PROMPT 3 : generate a python code to take two inputs from the user a number (temperature) and a string F to C (Fahrenheit to Celsius) or C to F(Celsius to Fahrenheit)

The screenshot shows a Google Colab notebook interface. The top bar includes tabs for 'Assignment 3.3', 'AIAC_lab_3.3 - Colab', and 'Sorting function example'. The address bar shows a Google Drive link. The notebook's menu bar includes 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. Below the menu is a toolbar with 'Commands', '+ Code', '+ Text', and 'Run all'. The main code cell contains a Python function `convert_temperature_with_inputs()` that takes a temperature value and a conversion type string as input. The function uses `try/except` blocks to handle `ValueError` and invalid conversion types. The output of the function is displayed in a terminal window at the bottom of the cell, showing the input '98' and the output '98.0°C is equal to 208.4°F'. On the right side, the Gemini chat interface is visible, showing a prompt to generate the Python code and a response that includes the code and suggestions for improvement.

```
def convert_temperature_with_inputs():
    """Converts temperature based on a number and a string indicating the conversion type."""
    try:
        temperature = float(input("Enter the temperature: "))
        conversion_type = input("Enter 'C to F' for Celsius to Fahrenheit or 'F to C' for Fahrenheit to Celsius: ")

        if conversion_type == 'C to F':
            fahrenheit = (temperature * 9/5) + 32
            print(f"{temperature}°C is equal to {fahrenheit}°F")
        elif conversion_type == 'F to C':
            celsius = (temperature - 32) * 5/9
            print(f"{temperature}°F is equal to {celsius}°C")
        else:
            print("Invalid conversion type. Please enter 'C to F' or 'F to C'.")

    except ValueError:
        print("Invalid input. Please enter a valid number for temperature.")

# Example usage:
convert_temperature_with_inputs()
```

Enter the temperature: 98
Enter 'C to F' for Celsius to Fahrenheit or 'F to C' for Fahrenheit to Celsius: C to F
98.0°C is equal to 208.4°F

This task is same as the 1st one and the 3rd one. first code contains a basic code to convert Celsius to Fahrenheit and then the next two codes contain increasingly advanced logic to convert Celsius to Fahrenheit or Fahrenheit to Celsius