SR UNIVERSITY

Program	:B.tech(CSE)
Course Title	:AI Assisted Coding
Course Code	:24CS002PC215
Semester	:3rd sem
Name of Student	:Kaveti Manohar
En. No.	:2403A52079
Batch No.	:02
Date	:20/08/2025

#LAB ASSIGNMENT-3

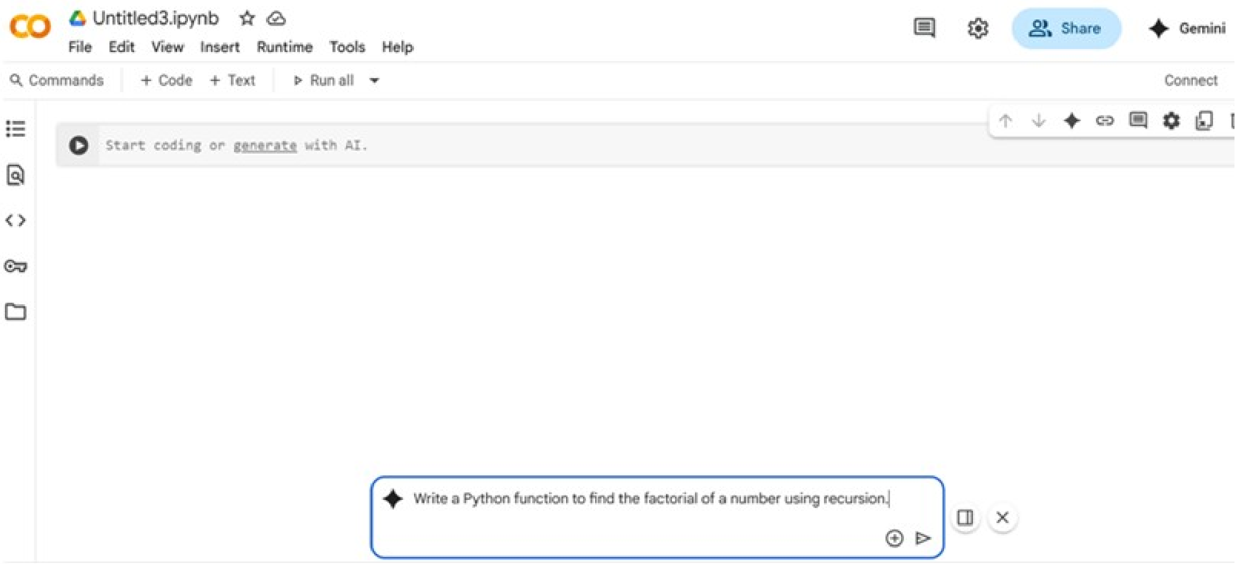
Task Description -1:

Try 3 different prompts to generate a factorial function.

Prompt-1

- Write a python function to find the factorial of a number using recursion.

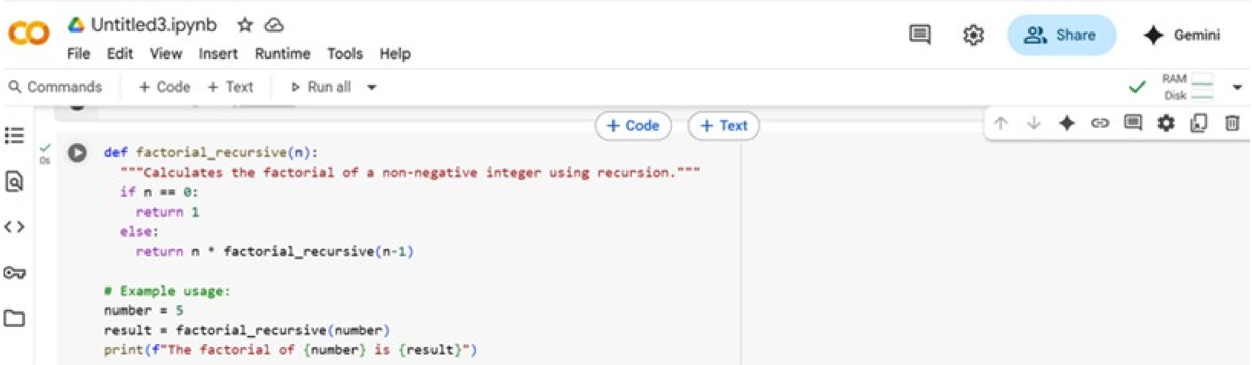
Question:



Start coding or generate with AI.

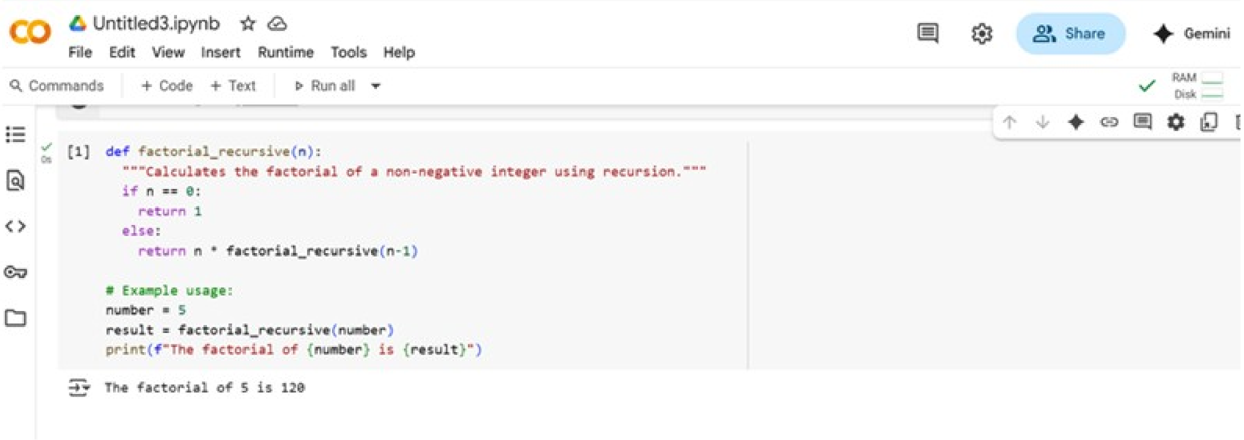
Write a Python function to find the factorial of a number using recursion.

Code:



```
def factorial_recursive(n):
  """Calculates the factorial of a non-negative integer using recursion."""
  if n == 0:
    return 1
  else:
    return n * factorial_recursive(n-1)

# Example usage:
number = 5
result = factorial_recursive(number)
print(f"The factorial of {number} is {result}")
```

Output:



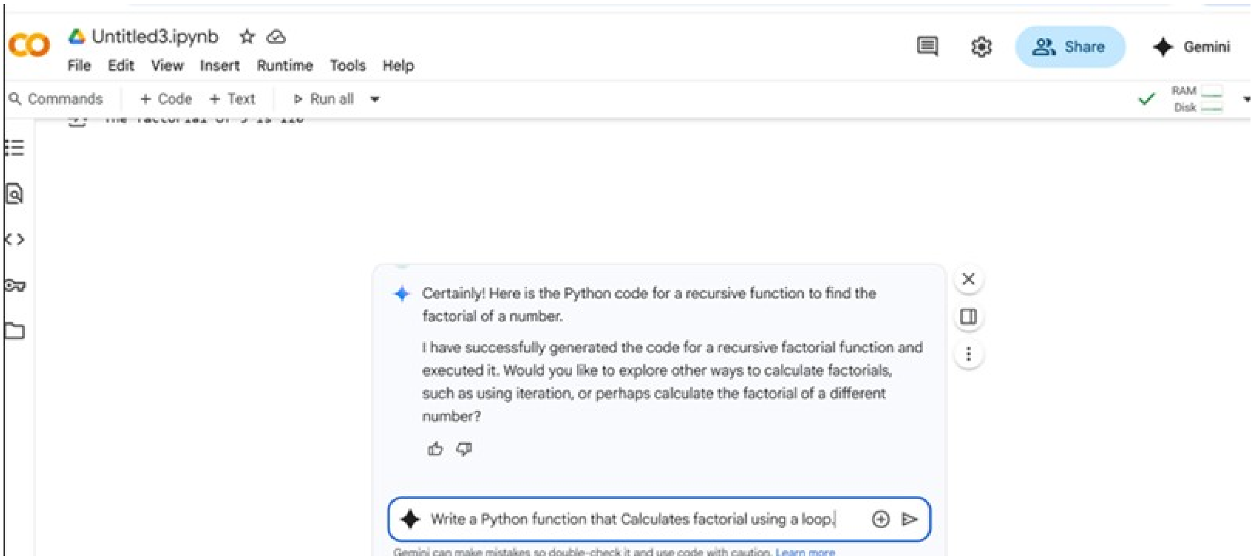
```
[1] def factorial_recursive(n):
      """Calculates the factorial of a non-negative integer using recursion."""
      if n == 0:
        return 1
      else:
        return n * factorial_recursive(n-1)

    # Example usage:
    number = 5
    result = factorial_recursive(number)
    print(f"The factorial of {number} is {result}")
```

```
The factorial of 5 is 120
```

Prompt-2:

- ➢ Write a python function that calculates factorial to find the factorial of a number.

Question:



Untitled3.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text ▷ Run all
Share
Gemini
RAM
Disk

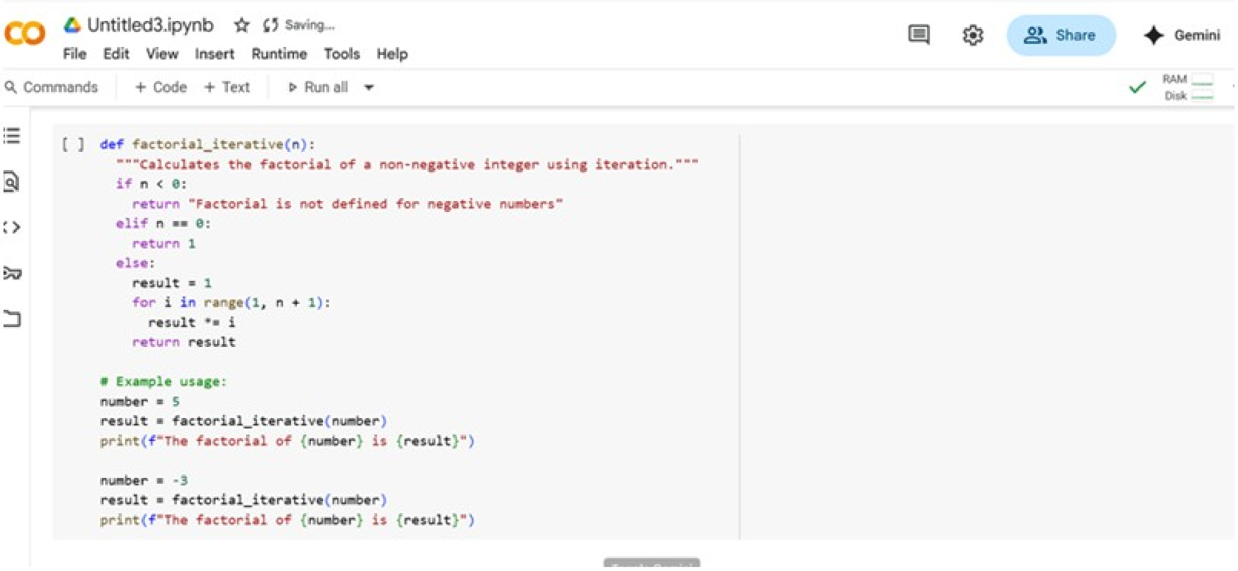
Certainly! Here is the Python code for a recursive function to find the factorial of a number.

I have successfully generated the code for a recursive factorial function and executed it. Would you like to explore other ways to calculate factorials, such as using iteration, or perhaps calculate the factorial of a different number?

Write a Python function that Calculates factorial using a loop.

Gemini can make mistakes so double-check it and use code with caution. Learn more

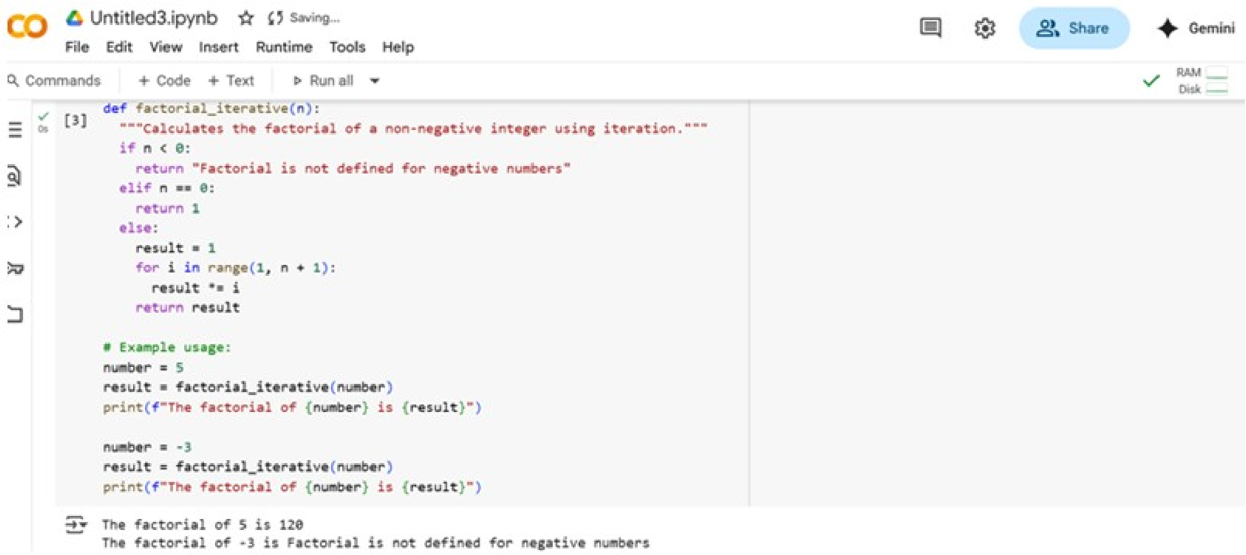
Code:



Untitled3.ipynb
Saving...
File Edit View Insert Runtime Tools Help
Commands + Code + Text ▷ Run all
Share
Gemini
RAM
Disk

```python
[ ] def factorial_iterative(n):
  """Calculates the factorial of a non-negative integer using iteration."""
  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

# Example usage:
number = 5
result = factorial_iterative(number)
print(f"The factorial of {number} is {result}")

number = -3
result = factorial_iterative(number)
print(f"The factorial of {number} is {result}")
```

Output:



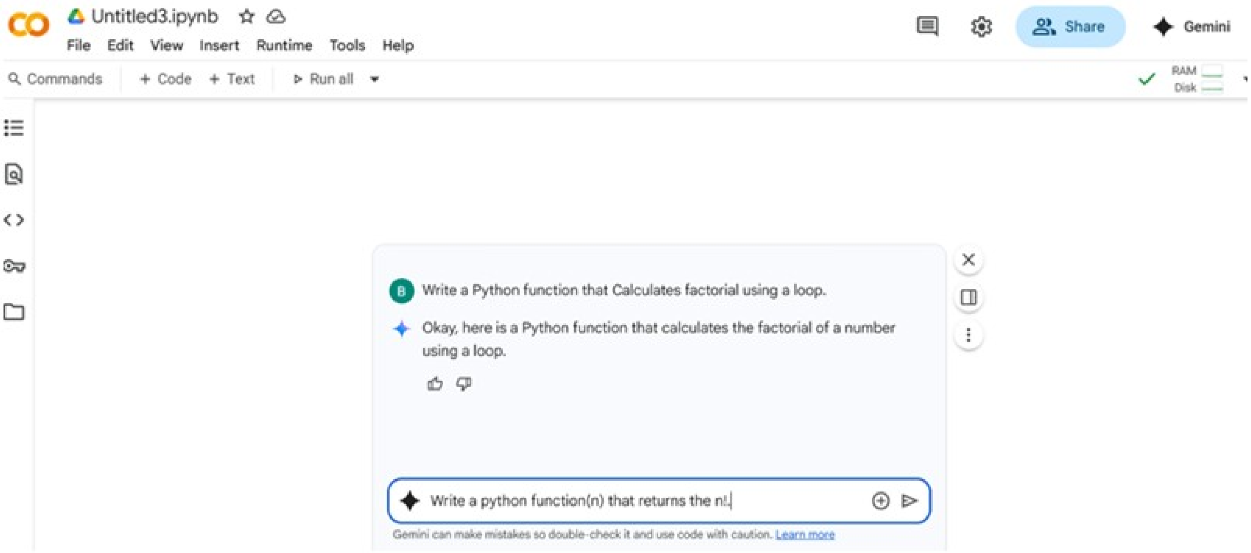
```
def factorial_iterative(n):
  """Calculates the factorial of a non-negative integer using iteration."""
  if n < 0:
    return "Factorial is not defined for negative numbers"
  elif n == 0:
    return 1
  else:
    result = 1
    for i in range(1, n + 1):
      result *= i
    return result

# Example usage:
number = 5
result = factorial_iterative(number)
print(f"The factorial of {number} is {result}")

number = -3
result = factorial_iterative(number)
print(f"The factorial of {number} is {result}")
```

```
The factorial of 5 is 120
The factorial of -3 is Factorial is not defined for negative numbers
```

Prompt-3:

- Write a python function() that returns the n!

Question:



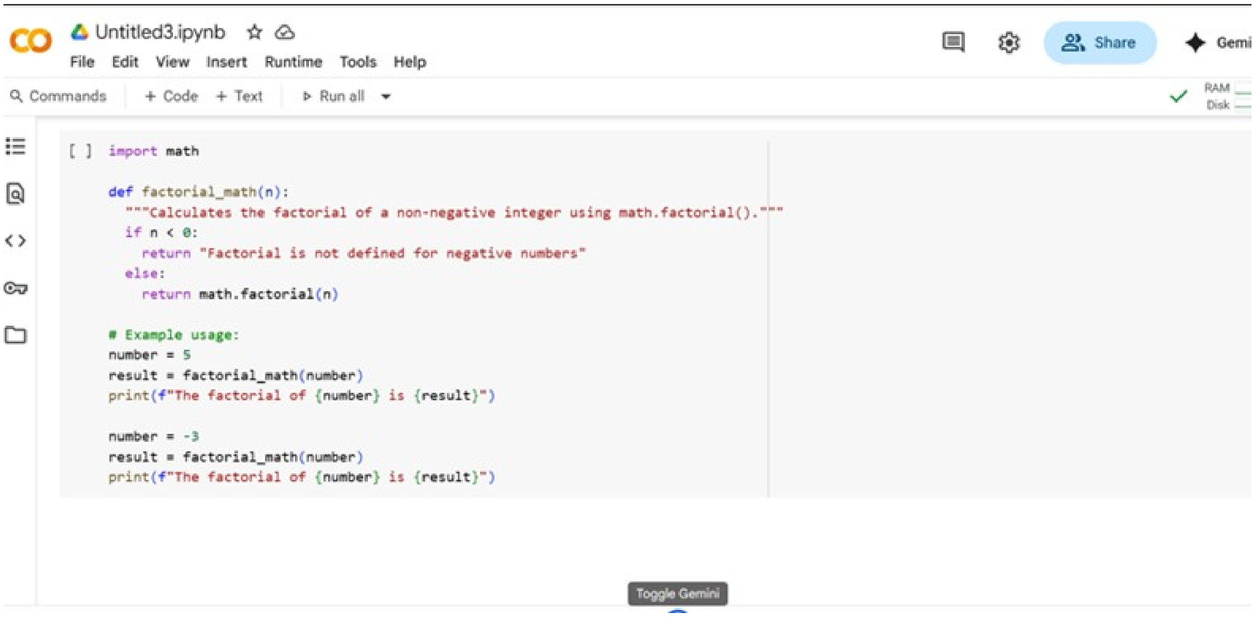
Write a Python function that Calculates factorial using a loop.

Okay, here is a Python function that calculates the factorial of a number using a loop.

Write a python function(n) that returns the n!

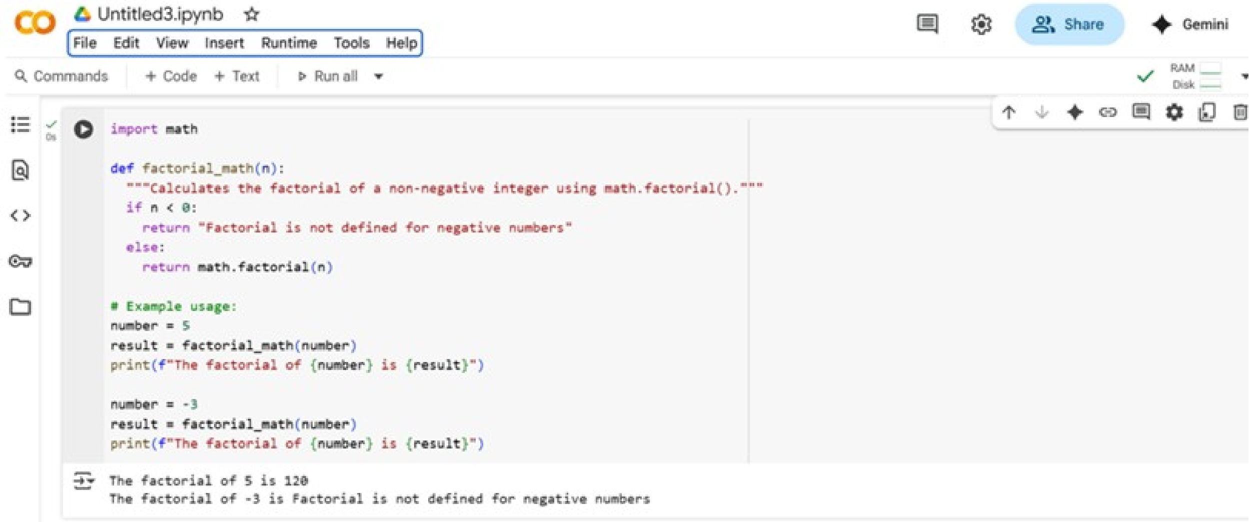
Gemini can make mistakes so double-check it and use code with caution. Learn more

Code:



```
[ ] import math

    def factorial_math(n):
      """Calculates the factorial of a non-negative integer using math.factorial()."""
      if n < 0:
        return "Factorial is not defined for negative numbers"
      else:
        return math.factorial(n)

    # Example usage:
    number = 5
    result = factorial_math(number)
    print(f"The factorial of {number} is {result}")

    number = -3
    result = factorial_math(number)
    print(f"The factorial of {number} is {result}")
```

Output:



```
import math

def factorial_math(n):
  """Calculates the factorial of a non-negative integer using math.factorial()."""
  if n < 0:
    return "Factorial is not defined for negative numbers"
  else:
    return math.factorial(n)

# Example usage:
number = 5
result = factorial_math(number)
print(f"The factorial of {number} is {result}")

number = -3
result = factorial_math(number)
print(f"The factorial of {number} is {result}")
```

```
The factorial of 5 is 120
The factorial of -3 is Factorial is not defined for negative numbers
```

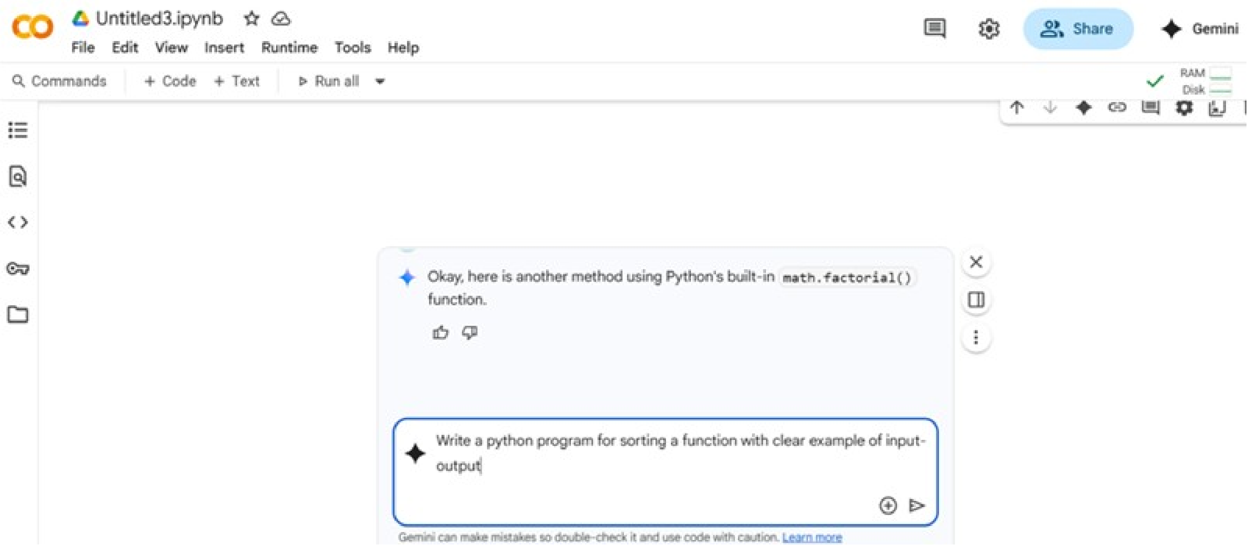
Tas Description-2

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

Prompt-2

- Write a python program using python built-in math.factorial() function

Question-

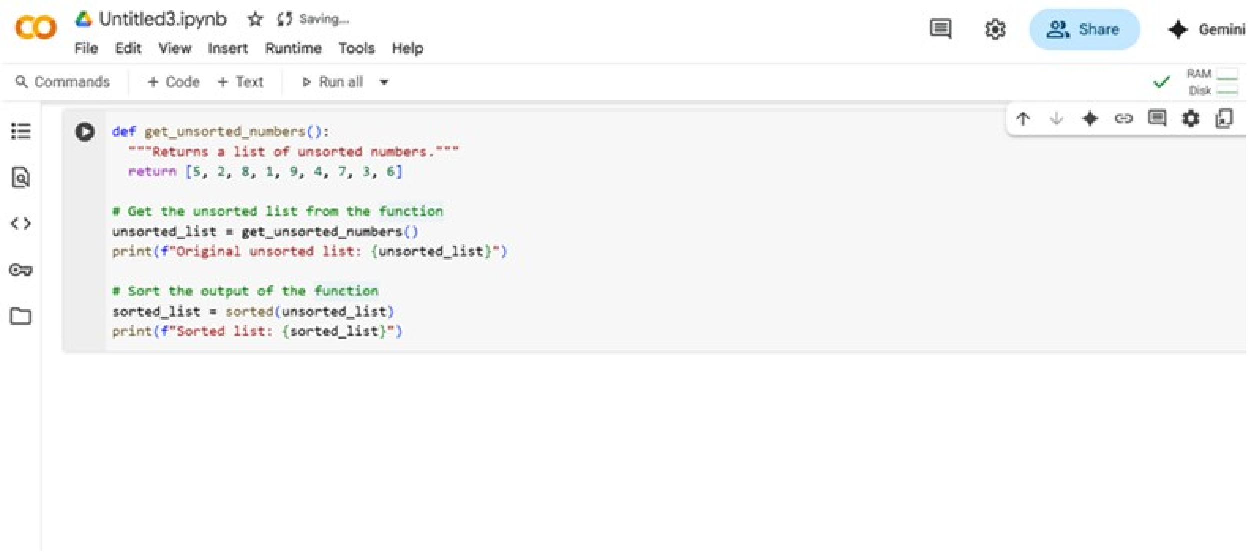

Untitled3.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Share
Gemini
RAM
Disk

Okay, here is another method using Python's built-in `math.factorial()` function.

Write a python program for sorting a function with clear example of input-output

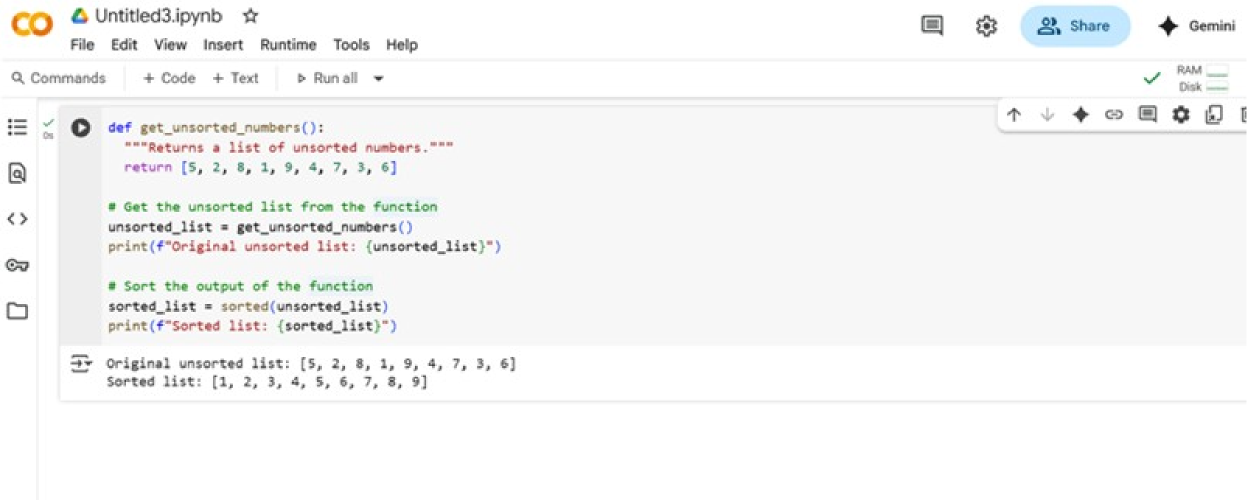
Gemini can make mistakes so double-check it and use code with caution. Learn more

Code-


Untitled3.ipynb
Saving...
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Share
Gemini
RAM
Disk

```python
def get_unsorted_numbers():
  """Returns a list of unsorted numbers."""
  return [5, 2, 8, 1, 9, 4, 7, 3, 6]

# Get the unsorted list from the function
unsorted_list = get_unsorted_numbers()
print(f"Original unsorted list: {unsorted_list}")

# Sort the output of the function
sorted_list = sorted(unsorted_list)
print(f"Sorted list: {sorted_list}")
```

Output-



```python
def get_unsorted_numbers():
  """Returns a list of unsorted numbers."""
  return [5, 2, 8, 1, 9, 4, 7, 3, 6]

# Get the unsorted list from the function
unsorted_list = get_unsorted_numbers()
print(f"Original unsorted list: {unsorted_list}")

# Sort the output of the function
sorted_list = sorted(unsorted_list)
print(f"Sorted list: {sorted_list}")
```

```
Original unsorted list: [5, 2, 8, 1, 9, 4, 7, 3, 6]
Sorted list: [1, 2, 3, 4, 5, 6, 7, 8, 9]
```

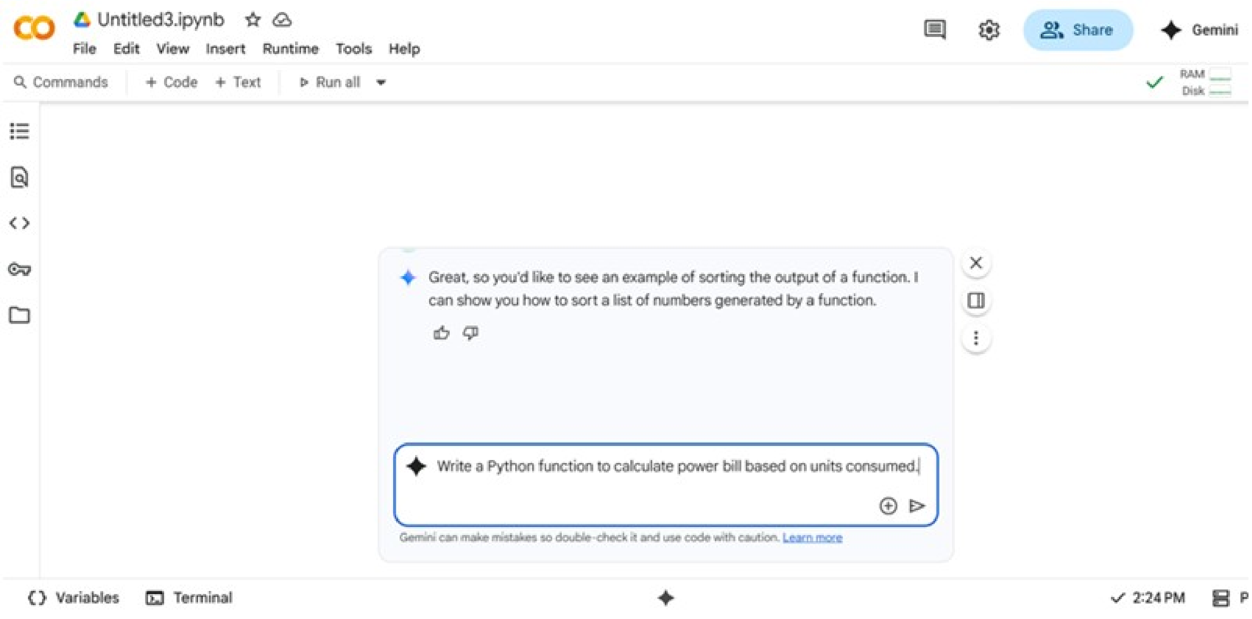
Task Description-3:

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

Prompt-

- Write a python function to calculate power bill based on units consumed.

Question-

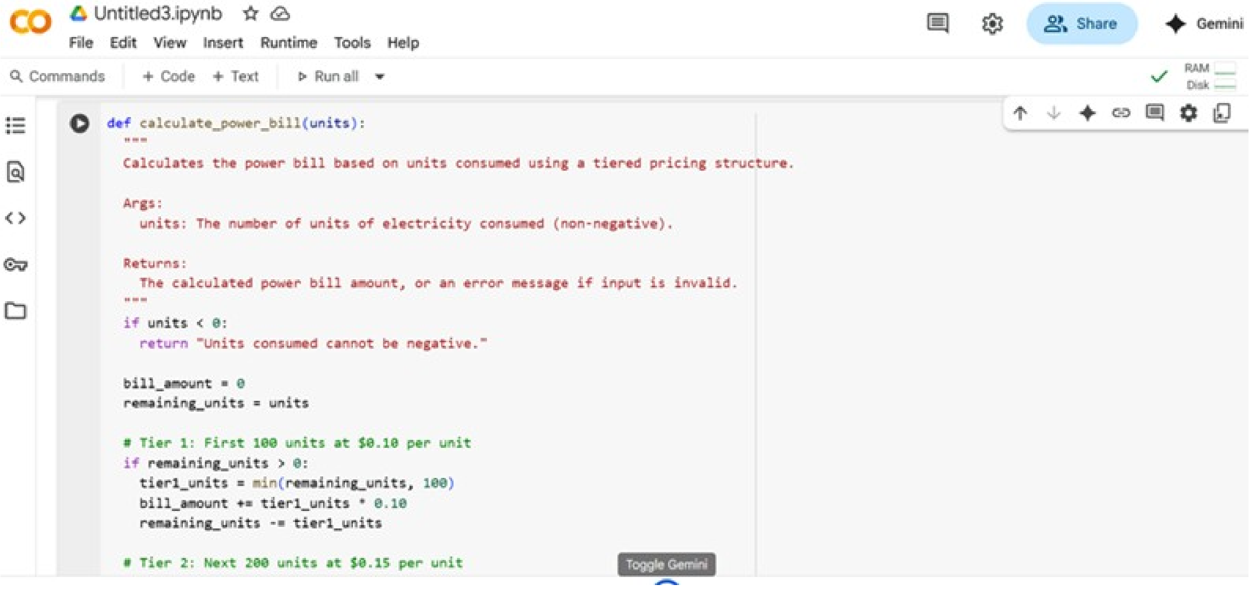


Great, so you'd like to see an example of sorting the output of a function. I can show you how to sort a list of numbers generated by a function.

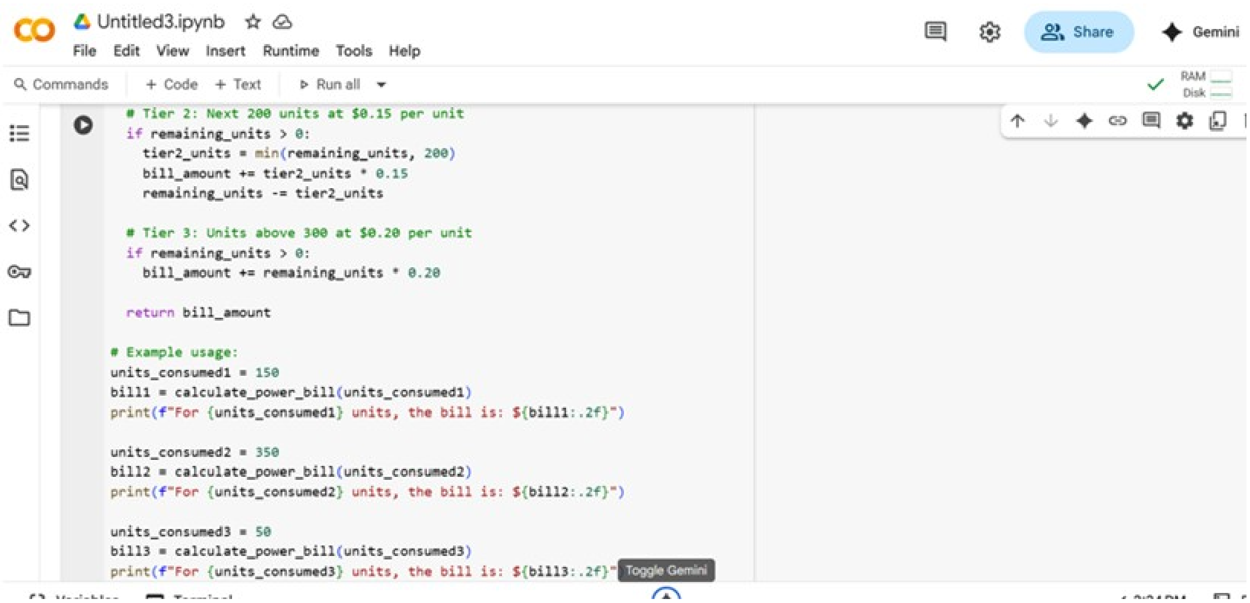
Write a Python function to calculate power bill based on units consumed.

Gemini can make mistakes so double-check it and use code with caution. Learn more

Code-

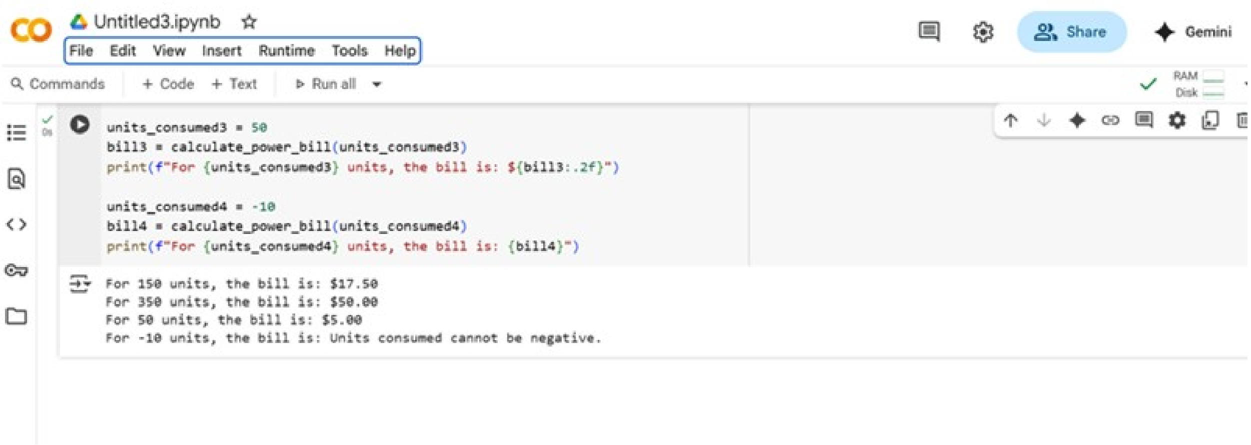


```
def calculate_power_bill(units):
  """
  Calculates the power bill based on units consumed using a tiered pricing structure.

  Args:
    units: The number of units of electricity consumed (non-negative).

  Returns:
    The calculated power bill amount, or an error message if input is invalid.
  """
  if units < 0:
    return "Units consumed cannot be negative."

  bill_amount = 0
  remaining_units = units

  # Tier 1: First 100 units at $0.10 per unit
  if remaining_units > 0:
    tier1_units = min(remaining_units, 100)
    bill_amount += tier1_units * 0.10
    remaining_units -= tier1_units

  # Tier 2: Next 200 units at $0.15 per unit
```



```
  # Tier 2: Next 200 units at $0.15 per unit
  if remaining_units > 0:
    tier2_units = min(remaining_units, 200)
    bill_amount += tier2_units * 0.15
    remaining_units -= tier2_units

  # Tier 3: Units above 300 at $0.20 per unit
  if remaining_units > 0:
    bill_amount += remaining_units * 0.20

  return bill_amount

# Example usage:
units_consumed1 = 150
bill1 = calculate_power_bill(units_consumed1)
print(f"For {units_consumed1} units, the bill is: ${bill1:.2f}")

units_consumed2 = 350
bill2 = calculate_power_bill(units_consumed2)
print(f"For {units_consumed2} units, the bill is: ${bill2:.2f}")

units_consumed3 = 50
bill3 = calculate_power_bill(units_consumed3)
print(f"For {units_consumed3} units, the bill is: ${bill3:.2f}"
```

Output-



```
units_consumed3 = 50
bill3 = calculate_power_bill(units_consumed3)
print(f"For {units_consumed3} units, the bill is: ${bill3:.2f}")

units_consumed4 = -10
bill4 = calculate_power_bill(units_consumed4)
print(f"For {units_consumed4} units, the bill is: {bill4}")
```

```
For 150 units, the bill is: $17.50
For 350 units, the bill is: $50.00
For 50 units, the bill is: $5.00
For -10 units, the bill is: Units consumed cannot be negative.
```

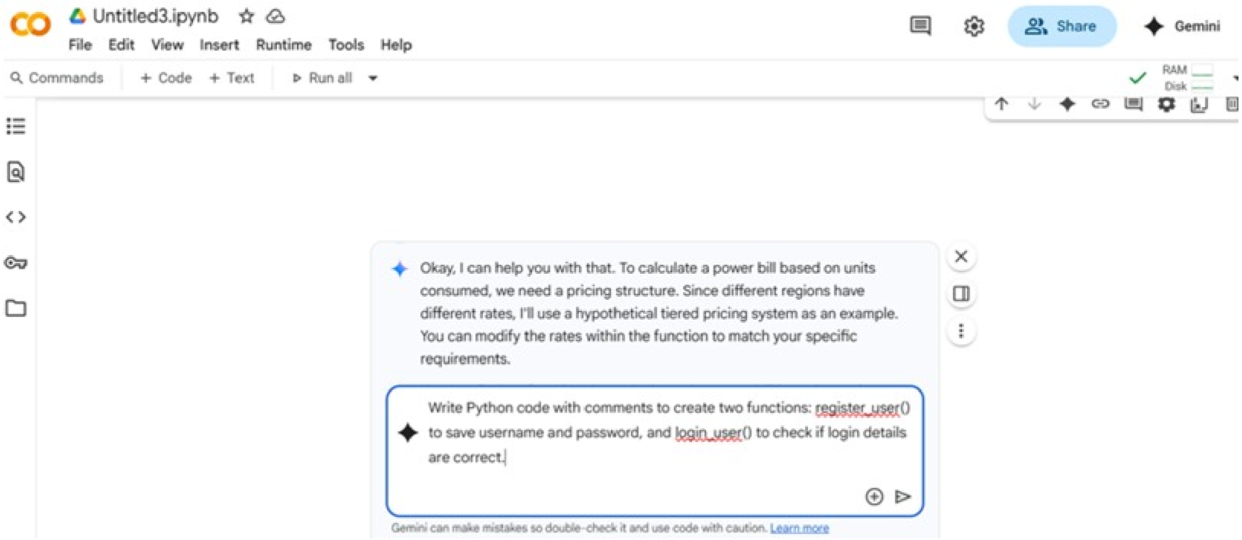
Task Description-4:

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

Prompt-4

- Write Python code with comments to create two functions :register user() to save username and password ,and login_user() to check if login details are correct.

Question-

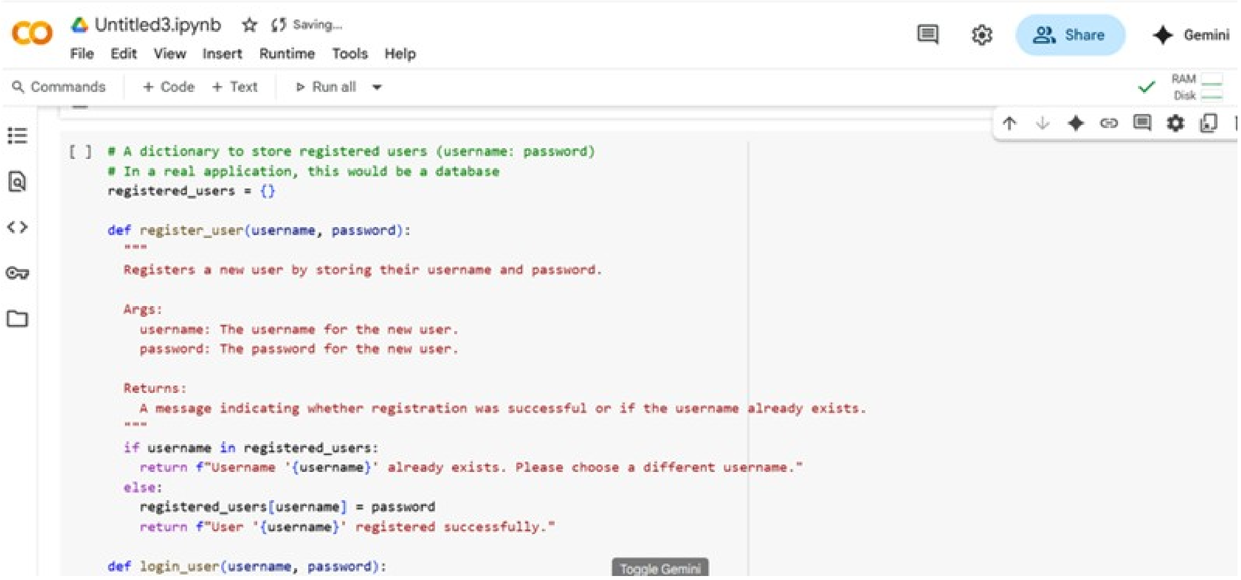

Untitled3.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Share
Gemini
RAM
Disk

Okay, I can help you with that. To calculate a power bill based on units consumed, we need a pricing structure. Since different regions have different rates, I'll use a hypothetical tiered pricing system as an example. You can modify the rates within the function to match your specific requirements.

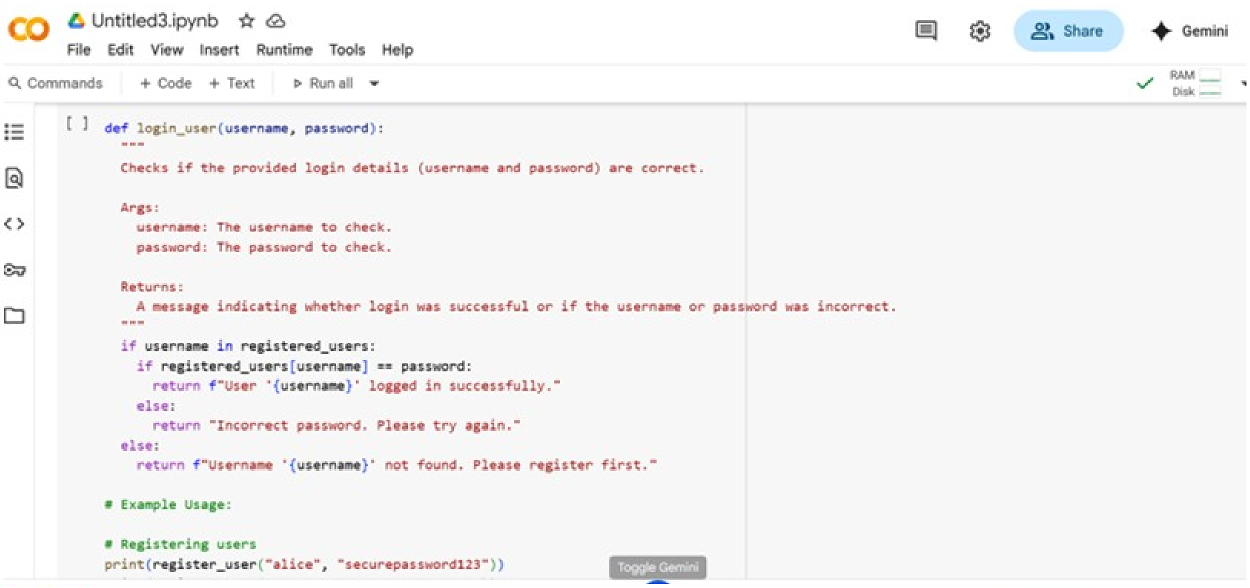
Write Python code with comments to create two functions: register_user() to save username and password, and login_user() to check if login details are correct.

Gemini can make mistakes so double-check it and use code with caution. Learn more

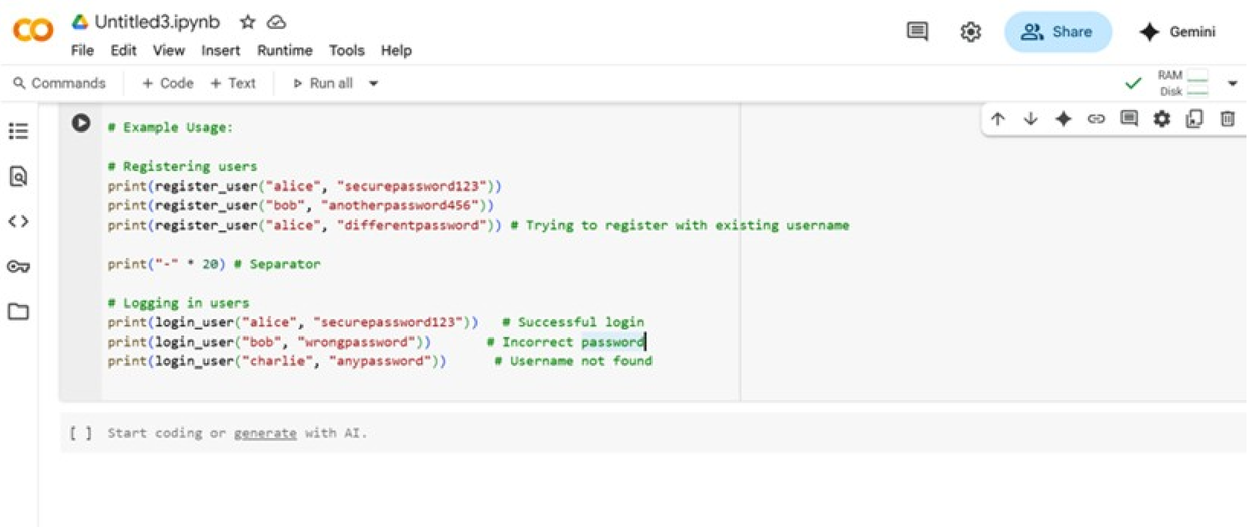
Code-


Untitled3.ipynb
Saving...
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Share
Gemini
RAM
Disk
Toggle Gemini

```
[ ] # A dictionary to store registered users (username: password)
    # In a real application, this would be a database
    registered_users = {}

    def register_user(username, password):
      """
      Registers a new user by storing their username and password.

      Args:
        username: The username for the new user.
        password: The password for the new user.

      Returns:
        A message indicating whether registration was successful or if the username already exists.
      """
      if username in registered_users:
        return f"Username '{username}' already exists. Please choose a different username."
      else:
        registered_users[username] = password
        return f"User '{username}' registered successfully."

    def login_user(username, password):
```

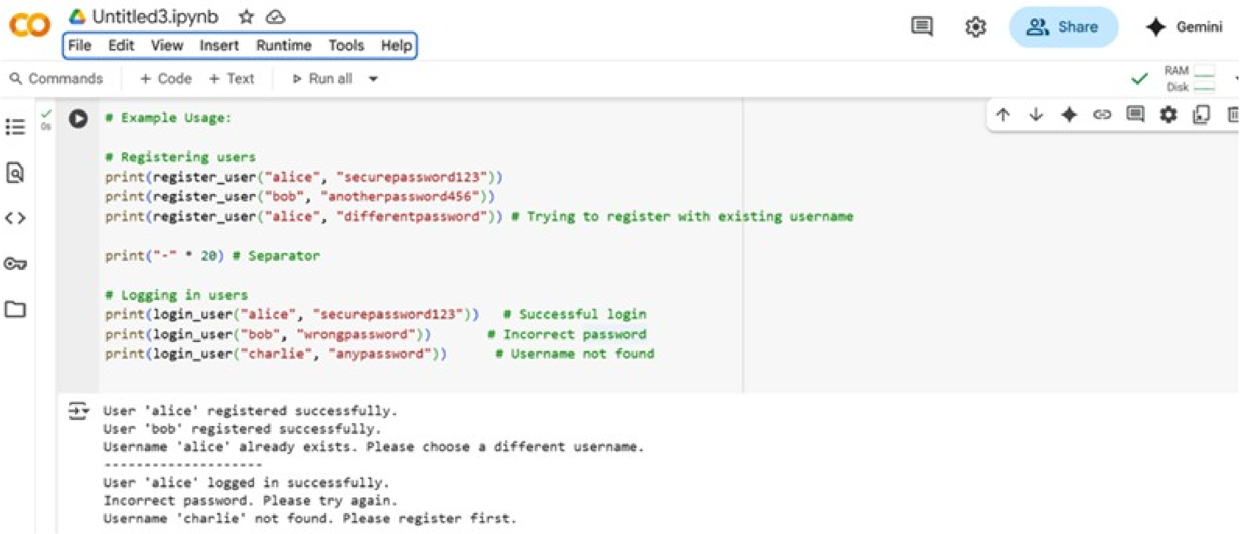


```
def login_user(username, password):
  """
  Checks if the provided login details (username and password) are correct.

  Args:
    username: The username to check.
    password: The password to check.

  Returns:
    A message indicating whether login was successful or if the username or password was incorrect.
  """
  if username in registered_users:
    if registered_users[username] == password:
      return f"User '{username}' logged in successfully."
    else:
      return "Incorrect password. Please try again."
  else:
    return f"Username '{username}' not found. Please register first."

# Example Usage:

# Registering users
print(register_user("alice", "securepassword123"))
```


Untitled3.ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands
+ Code
+ Text
Run all
RAM
Disk
Start coding or generate with AI.

```
# Example Usage:

# Registering users
print(register_user("alice", "securepassword123"))
print(register_user("bob", "anotherpassword456"))
print(register_user("alice", "differentpassword")) # Trying to register with existing username

print("-" * 20) # Separator

# Logging in users
print(login_user("alice", "securepassword123"))   # Successful login
print(login_user("bob", "wrongpassword"))       # Incorrect password
print(login_user("charlie", "anypassword"))      # Username not found
```

Output-



```
# Example Usage:

# Registering users
print(register_user("alice", "securepassword123"))
print(register_user("bob", "anotherpassword456"))
print(register_user("alice", "differentpassword")) # Trying to register with existing username

print("-" * 20) # Separator

# Logging in users
print(login_user("alice", "securepassword123"))   # Successful login
print(login_user("bob", "wrongpassword"))       # Incorrect password
print(login_user("charlie", "anypassword"))      # Username not found
```

```
User 'alice' registered successfully.
User 'bob' registered successfully.
Username 'alice' already exists. Please choose a different username.
--------------------
User 'alice' logged in successfully.
Incorrect password. Please try again.
Username 'charlie' not found. Please register first.
```

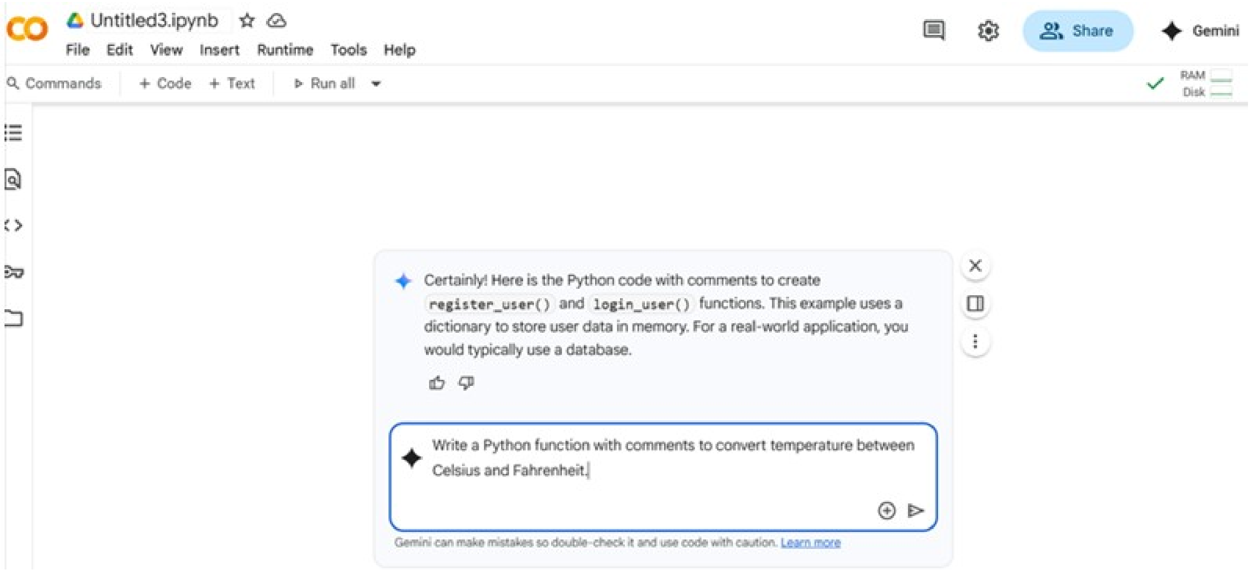
Task Description-5:

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

Prompt-5

- Write a function with comments to convert temperature between Celsius and Fahrenheit.

Question-

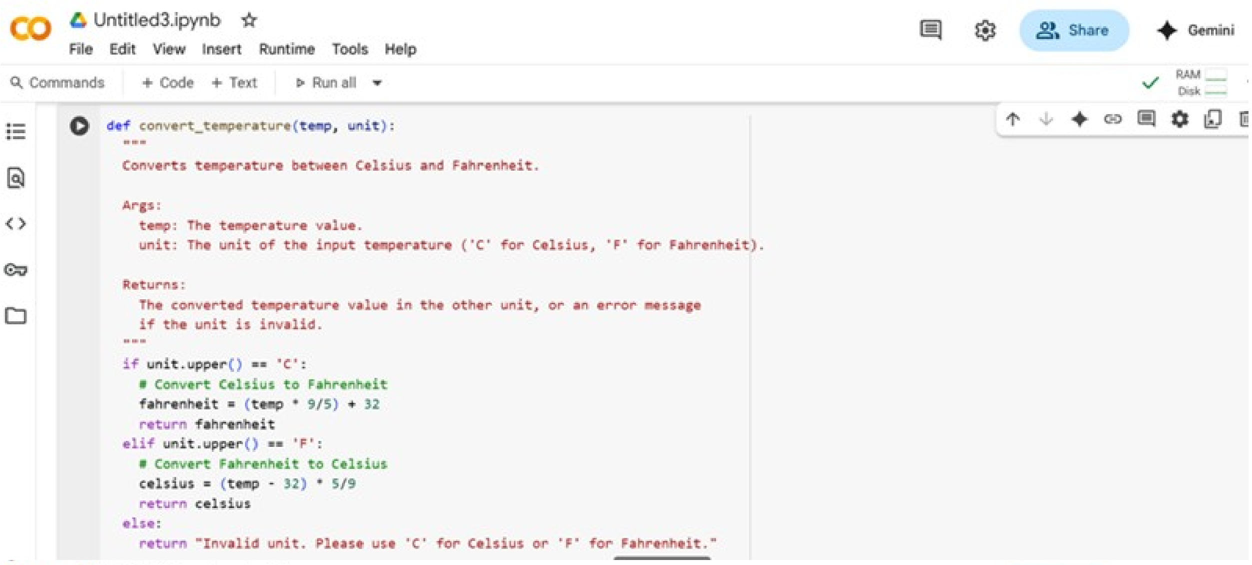


Certainly! Here is the Python code with comments to create `register_user()` and `login_user()` functions. This example uses a dictionary to store user data in memory. For a real-world application, you would typically use a database.

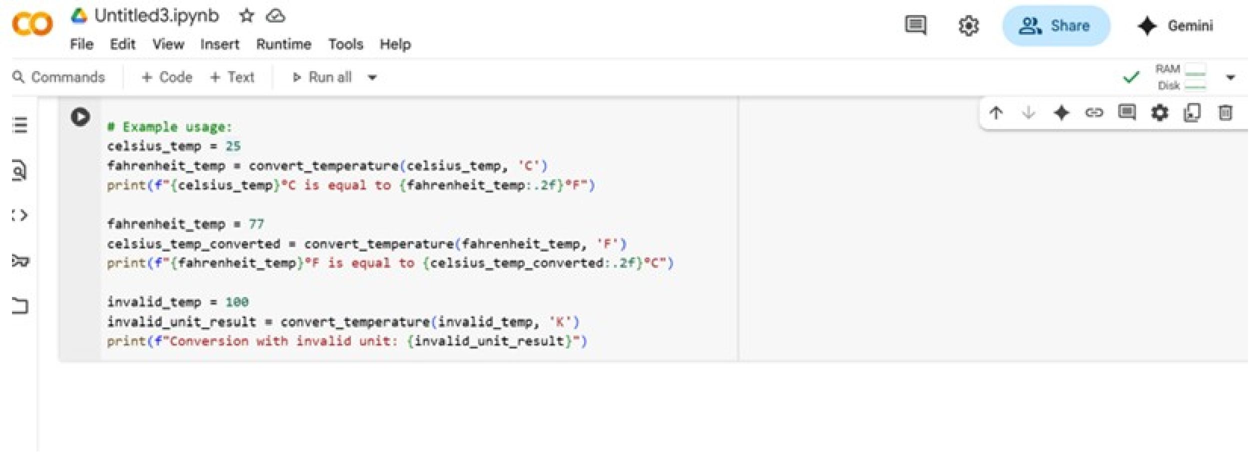
Write a Python function with comments to convert temperature between Celsius and Fahrenheit.

Gemini can make mistakes so double-check it and use code with caution. Learn more

Code-

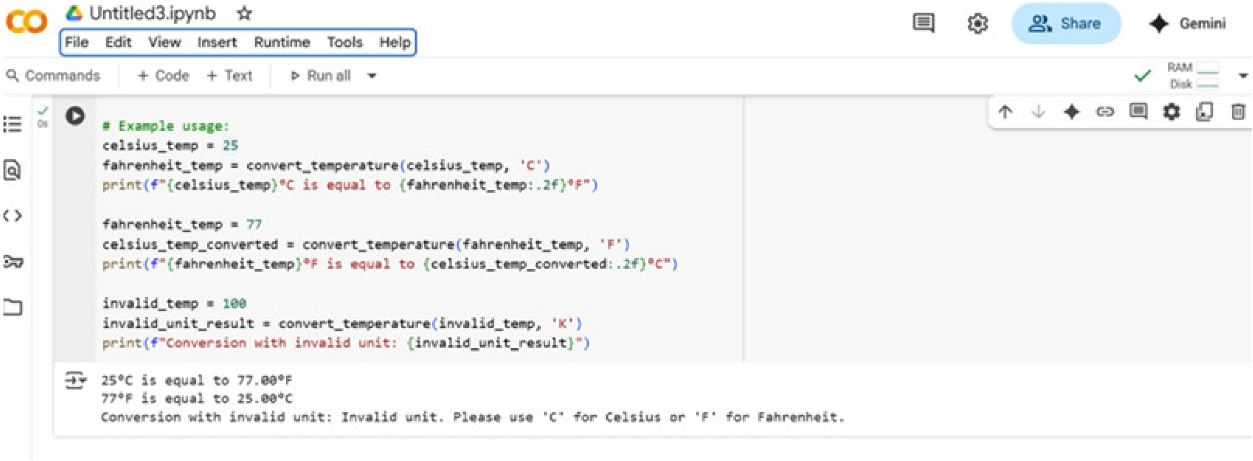


```
def convert_temperature(temp, unit):
  """
  Converts temperature between Celsius and Fahrenheit.

  Args:
    temp: The temperature value.
    unit: The unit of the input temperature ('C' for Celsius, 'F' for Fahrenheit).

  Returns:
    The converted temperature value in the other unit, or an error message
    if the unit is invalid.
  """
  if unit.upper() == 'C':
    # Convert Celsius to Fahrenheit
    fahrenheit = (temp * 9/5) + 32
    return fahrenheit
  elif unit.upper() == 'F':
    # Convert Fahrenheit to Celsius
    celsius = (temp - 32) * 5/9
    return celsius
  else:
    return "Invalid unit. Please use 'C' for Celsius or 'F' for Fahrenheit."
```



```
# Example usage:
celsius_temp = 25
fahrenheit_temp = convert_temperature(celsius_temp, 'C')
print(f"{celsius_temp}°C is equal to {fahrenheit_temp:.2f}°F")

fahrenheit_temp = 77
celsius_temp_converted = convert_temperature(fahrenheit_temp, 'F')
print(f"{fahrenheit_temp}°F is equal to {celsius_temp_converted:.2f}°C")

invalid_temp = 100
invalid_unit_result = convert_temperature(invalid_temp, 'K')
print(f"Conversion with invalid unit: {invalid_unit_result}")
```

Output-



```
# Example usage:
celsius_temp = 25
fahrenheit_temp = convert_temperature(celsius_temp, 'C')
print(f"{celsius_temp}°C is equal to {fahrenheit_temp:.2f}°F")

fahrenheit_temp = 77
celsius_temp_converted = convert_temperature(fahrenheit_temp, 'F')
print(f"{fahrenheit_temp}°F is equal to {celsius_temp_converted:.2f}°C")

invalid_temp = 100
invalid_unit_result = convert_temperature(invalid_temp, 'K')
print(f"Conversion with invalid unit: {invalid_unit_result}")
```

```
25°C is equal to 77.00°F
77°F is equal to 25.00°C
Conversion with invalid unit: Invalid unit. Please use 'C' for Celsius or 'F' for Fahrenheit.
```