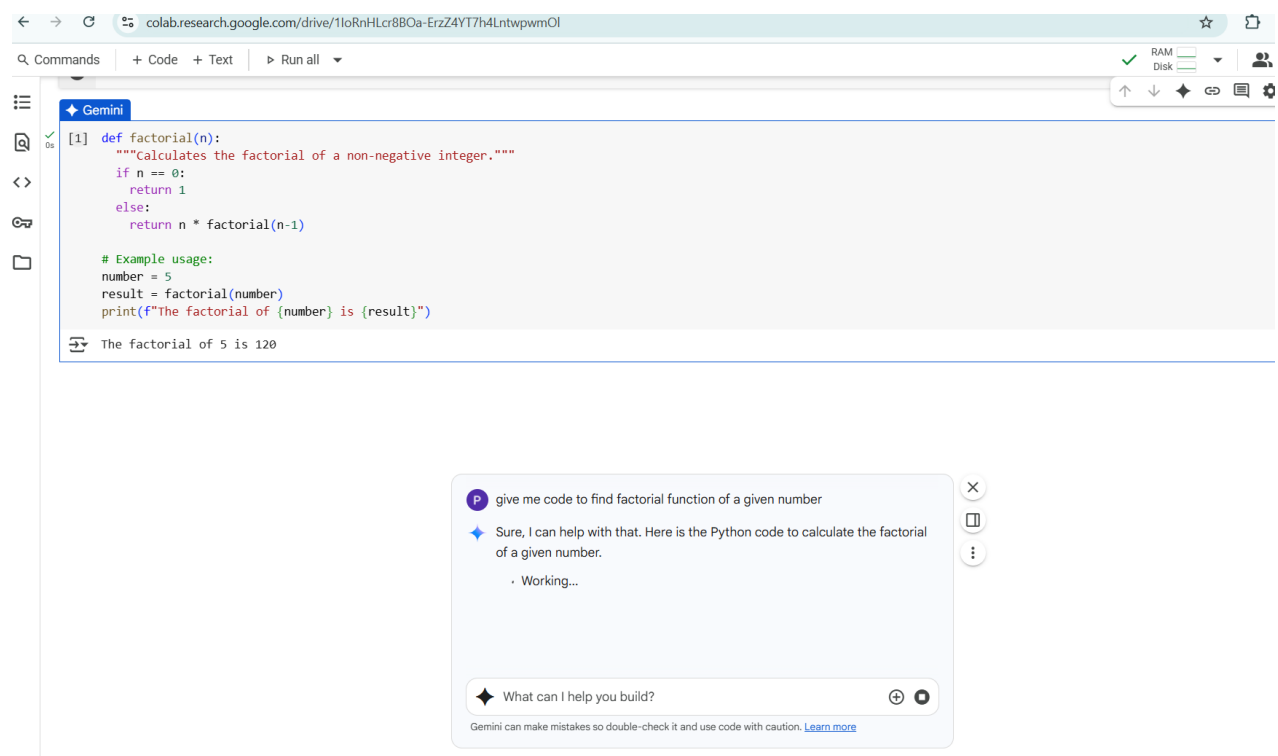


ASSIGNMENT – 3.3

NAME : Thota Akanksha
BATCH : 01
ROLL NO : 2403A510D8
COURSE : AI ASSISTED CODING

TASK 1:

PROMPT 1:



The screenshot displays a Google Colab notebook interface. The top bar shows the URL `colab.research.google.com/drive/1oRnHLCr8BOa-ErzZ4YT7h4LntwpwmOI`. The notebook contains a Python function `def factorial(n):` with a docstring, an `if` statement for `n == 0`, and a recursive call. Below the function, an example usage is shown with `number = 5`, `result = factorial(number)`, and `print(f"The factorial of {number} is {result}")`. The output of the cell is `The factorial of 5 is 120`. On the right side, a Gemini chat interface is open. The user prompt is `give me code to find factorial function of a given number`. The Gemini response is `Sure, I can help with that. Here is the Python code to calculate the factorial of a given number.` followed by a `Working...` status. The chat interface also includes a search bar with the text `What can I help you build?` and a disclaimer at the bottom: `Gemini can make mistakes so double-check it and use code with caution. Learn more`.

PROMPT 2:

✓

0s

[2]

```
def factorial_loop(n):  
    """Calculates the factorial of a non-negative integer using a loop."""  
    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_loop(number)  
print(f"The factorial of {number} using a loop is {result}")
```

The factorial of 5 using a loop is 120

P

give me code to create a factorial program using a loop example usage take 5

◆

Okay, here is a version of the factorial function that uses a loop, along with an example using the number 5.

I've provided two different ways to calculate the factorial of a number. Would you like to:

- Compare the performance of the two methods?

◆

What can I help you build?

⊕ ▶

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

Variables

Terminal

◆

✓ 1:53 PM Python 3

PROMPT 3:

✓

0s

[3]

```
import math  
  
number = 5  
result = math.factorial(number)  
print(f"The factorial of {number} using math.factorial() is {result}")
```

The factorial of 5 using math.factorial() is 120

P

give me code to find factorial function using the built in math module take example as 5

◆

You can indeed use Python's built-in `math` module to calculate factorials. Here's how to do it for the number 5:

I've shown you three different ways to calculate factorials in Python: using recursion, a loop, and the built-in `math` module.

What would you like to do next? We could:

◆

What can I help you build?

⊕ ▶

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

Variables

Terminal

◆

✓ 2:06 PM Python 3

TASK 2:

The screenshot shows a code editor with a Python function `sort_descending()` that takes a list of integers as input and sorts them in descending order. The function uses a `while True` loop to prompt the user for input, converts the input to a list of integers, and sorts it using `numbers.sort(reverse=True)`. The output shows the sorted list: `[99, 76, 55, 34, 22, 11]`.

The AI assistant (Gemini) provides the following response:

give me code to write a function that sorts a list of integers in descending order by taking input from the user

Certainly, I can provide code to create a function that sorts a list of integers in descending order based on user input.

I hope that function is helpful! Would you like to:

- Modify the function to sort in ascending order?
- Add error handling for non-integer input?
- Explore different sorting algorithms?

TASK 3:

The screenshot shows a code editor with a Python function `calculate_power_bill(consumption_kwh)` that calculates the power bill based on consumption in kWh. The function uses a `while True` loop to prompt the user for input, calculates the bill amount based on consumption tiers, and displays the final bill.

The AI assistant (Gemini) provides the following response:

generate python code to calculate power bill

I can help you with that. To calculate a power bill, we'll need to consider a few factors. Here is a plan to generate the Python code for it.

- **Define a function to calculate the bill:** Create a function that takes the electricity consumption (in kilowatt-hours) as input.
- **Implement billing logic:** Inside the function, apply the billing logic based on different consumption tiers and their respective rates.
- **Add user input:** Include code to prompt the user for their electricity consumption.
- **Calculate and display the bill:** Call the calculation function with the user's input and display the final calculated power bill.
- **Finish task:** Provide the complete Python code for calculating the power bill.

considering this factors give me code

Okay, here is the Python code to calculate the power bill based on the factors we discussed in the plan. This code includes a function to calculate the bill based on consumption tiers and takes user input.

I hope the power bill calculation code is useful! Is there anything else you'd like to do with it? We could:

- Add more billing tiers or different rate structures.
- Include fixed charges or taxes in the calculation.
- Save the bill calculation results to a file.
- Work on a different coding problem.

TASK 4:

P generate python code using if and else for login user and registraion user

◆ I can definitely help you with that. Here is a plan to create Python code using if/else statements for user login and registration.

- **Simulate a user database:** Create a simple way to store usernames and passwords, perhaps using a dictionary.
- **Implement registration logic:**
 - Prompt the user for a new username and password.
 - Check if the username already exists in the database.
 - If the username is unique, add the new user to the database.
 - If the username exists, inform the user.
- **Implement login logic:**
 - Prompt the user for their username and password.
 - Check if the username exists in the database.
 - If the username exists, verify the password.
 - If the username and password match, grant access.
 - If the username doesn't exist or the password is incorrect, inform the user.
- **Add a main program loop:** Create a loop that allows the user to choose between registration and login.
- **Finish task:** Provide the complete Python code for the login and registration system.



```
# Simulate a user database using a dictionary
user_database = {}

def register_user():
    """Registers a new user."""
    username = input("Enter a new username: ")
    if username in user_database:
        print("Username already exists. Please choose a different username.")
    else:
        password = input("Enter a password: ")
        user_database[username] = password
        print("Registration successful!")

def login_user():
    """Logs in an existing user."""
    username = input("Enter your username: ")
    if username in user_database:
        password = input("Enter your password: ")
        if user_database[username] == password:
            print("Login successful!")
        else:
            print("Incorrect password.")
    else:
        print("Username not found.")

# Main program loop
while True:
    print("\nChoose an option:")
    print("1. Register")
    print("2. Login")
    print("3. Exit")

    choice = input("Enter your choice: ")

    if choice == '1':
        register_user()
    elif choice == '2':
        login_user()
    elif choice == '3':
        break
```

Variables Terminal

Gemini X

and passwords, perhaps using a dictionary.

- **Implement registration logic:**
 - Prompt the user for a new username and password.
 - Check if the username already exists in the database.
 - If the username is unique, add the new user to the database.
 - If the username exists, inform the user.
- **Implement login logic:**
 - Prompt the user for their username and password.
 - Check if the username exists in the database.
 - If the username exists, verify the password.
 - If the username and password match, grant access.
 - If the username doesn't exist or the password is incorrect, inform the user.
- **Add a main program loop:** Create a loop that allows the user to choose between registration and login.
- **Finish task:** Provide the complete Python code for the login and registration system.

P using this plan give me code

◆ Great! Here is the Python code that implements the login and registration system based on the plan we created. It uses a dictionary to simulate a user database and if/else statements for the login and registration logic.

Working...

◆ What can I help you build?

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

Execution (1m 17s) Python 3

```
if user_database[username] == password:
    print("Login successful!")
else:
    print("Incorrect password.")
else:
    print("Username not found.")

# Main program loop
while True:
    print("\nChoose an option:")
    print("1. Register")
    print("2. Login")
    print("3. Exit")

    choice = input("Enter your choice: ")

    if choice == '1':
        register_user()
    elif choice == '2':
        login_user()
    elif choice == '3':
        print("Exiting program.")
        break
    else:
        print("Invalid choice. Please try again.")

...
Choose an option:
1. Register
2. Login
3. Exit
Enter your choice: login
Invalid choice. Please try again.

Choose an option:
1. Register
2. Login
3. Exit
Enter your choice: exit
Invalid choice. Please try again.
```

Gemini X

and passwords, perhaps using a dictionary.

- **Implement registration logic:**
 - Prompt the user for a new username and password.
 - Check if the username already exists in the database.
 - If the username is unique, add the new user to the database.
 - If the username exists, inform the user.
- **Implement login logic:**
 - Prompt the user for their username and password.
 - Check if the username exists in the database.
 - If the username exists, verify the password.
 - If the username and password match, grant access.
 - If the username doesn't exist or the password is incorrect, inform the user.
- **Add a main program loop:** Create a loop that allows the user to choose between registration and login.
- **Finish task:** Provide the complete Python code for the login and registration system.

using this plan give me code

Great! Here is the Python code that implements the login and registration system based on the plan we created. It uses a dictionary to simulate a user database and if/else statements for the login and registration logic.

Working...

What can I help you build?

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

Executing (2m 1s) Python 3

TASK 5:

PROMPT 1 :

Here are the formulas for converting between Fahrenheit and Celsius:

Fahrenheit to Celsius: $C = (F - 32) \times \frac{5}{9}$

Celsius to Fahrenheit: $F = (C \times \frac{9}{5}) + 32$

Gemini

```
def fahrenheit_to_celsius(fahrenheit):
    """Converts Fahrenheit to Celsius."""
    celsius = (fahrenheit - 32) * 5/9
    return celsius

def celsius_to_fahrenheit(celsius):
    """Converts Celsius to Fahrenheit."""
    fahrenheit = (celsius * 9/5) + 32
    return fahrenheit

# Example usage:
fahrenheit_temp = 68
celsius_temp = fahrenheit_to_celsius(fahrenheit_temp)
print(f"{fahrenheit_temp}°F is equal to {celsius_temp:.2f}°C")

celsius_temp_2 = 20
fahrenheit_temp_2 = celsius_to_fahrenheit(celsius_temp_2)
print(f"{celsius_temp_2}°C is equal to {fahrenheit_temp_2:.2f}°F")
```

Gemini X

give me code to convert temperature

Sure, I can provide you with code to convert temperatures between Fahrenheit and Celsius. I'll also include the formulas for your reference.

PROMPT 2:

```
def celsius_to_fahrenheit(celsius):
    """Converts Celsius to Fahrenheit."""
    fahrenheit = (celsius * 9/5) + 32
    return fahrenheit

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

25°C is equal to 77.00°F

give me code for a python function that converts celsius to fahrenheit

I can provide you with that. Here is the function to convert Celsius to Fahrenheit:

I hope that was helpful! Do you want to try converting Fahrenheit to Celsius, or perhaps explore other types of unit conversions?

PROMPT 3:

```
def fahrenheit_to_celsius(fahrenheit):
    """Converts Fahrenheit to Celsius."""
    celsius = (fahrenheit - 32) * 5/9
    return celsius

def celsius_to_fahrenheit(celsius):
    """Converts Celsius to Fahrenheit."""
    fahrenheit = (celsius * 9/5) + 32
    return fahrenheit

while True:
    try:
        temperature = float(input("Enter the temperature value: "))
        unit = input("Enter the unit of temperature (C for Celsius, F for Fahrenheit): ").upper()

        if unit == 'C':
            fahrenheit_result = celsius_to_fahrenheit(temperature)
            print(f"{temperature}°C is equal to {fahrenheit_result:.2f}°F")
            break
        elif unit == 'F':
            celsius_result = fahrenheit_to_celsius(temperature)
            print(f"{temperature}°F is equal to {celsius_result:.2f}°C")
            break
        else:
            print("Invalid unit. Please enter 'C' or 'F'.")
    except ValueError:
        print("Invalid input. Please enter a numerical value for temperature.")
```

Enter the temperature value: 25
Enter the unit of temperature (C for Celsius, F for Fahrenheit): C
25.0°C is equal to 77.00°F

give me code that converts temperature from celsius to fahrenheit and fahrenheit to celsius. take input from user and return the converted value

Certainly! Here is the code that takes temperature input from the user and converts it between Celsius and Fahrenheit.

Great! The temperature conversion code is ready. Would you like to explore other unit conversions, like weight or distance? Or perhaps you'd like to add more error handling to the current code?