ASSIGNMENT_3-3

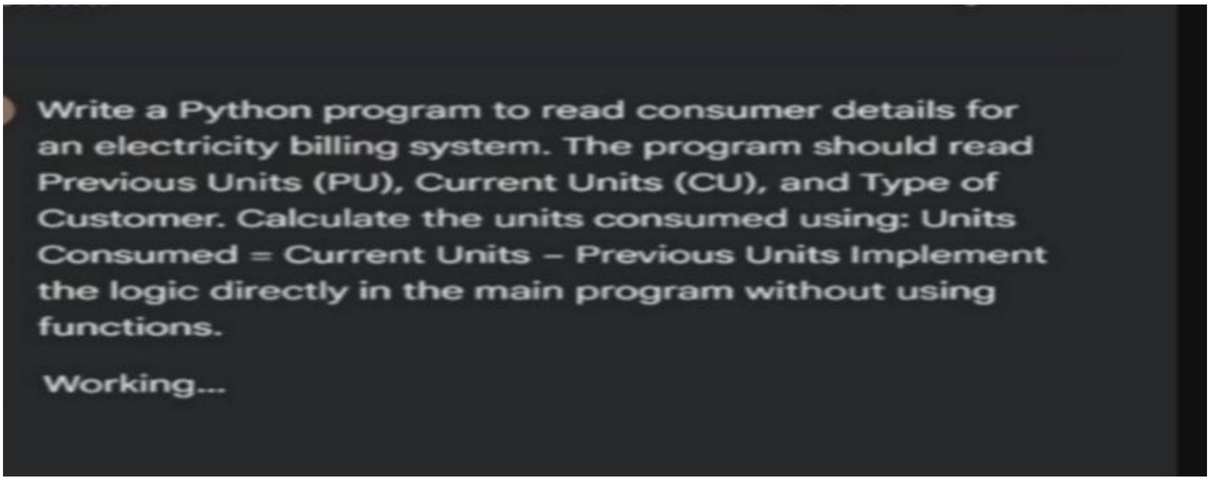
Name:Ch.Jyothika

Ht.no:2303A51280

Batch:05

Task 1: AI-Generated Logic for Reading Consumer Details.

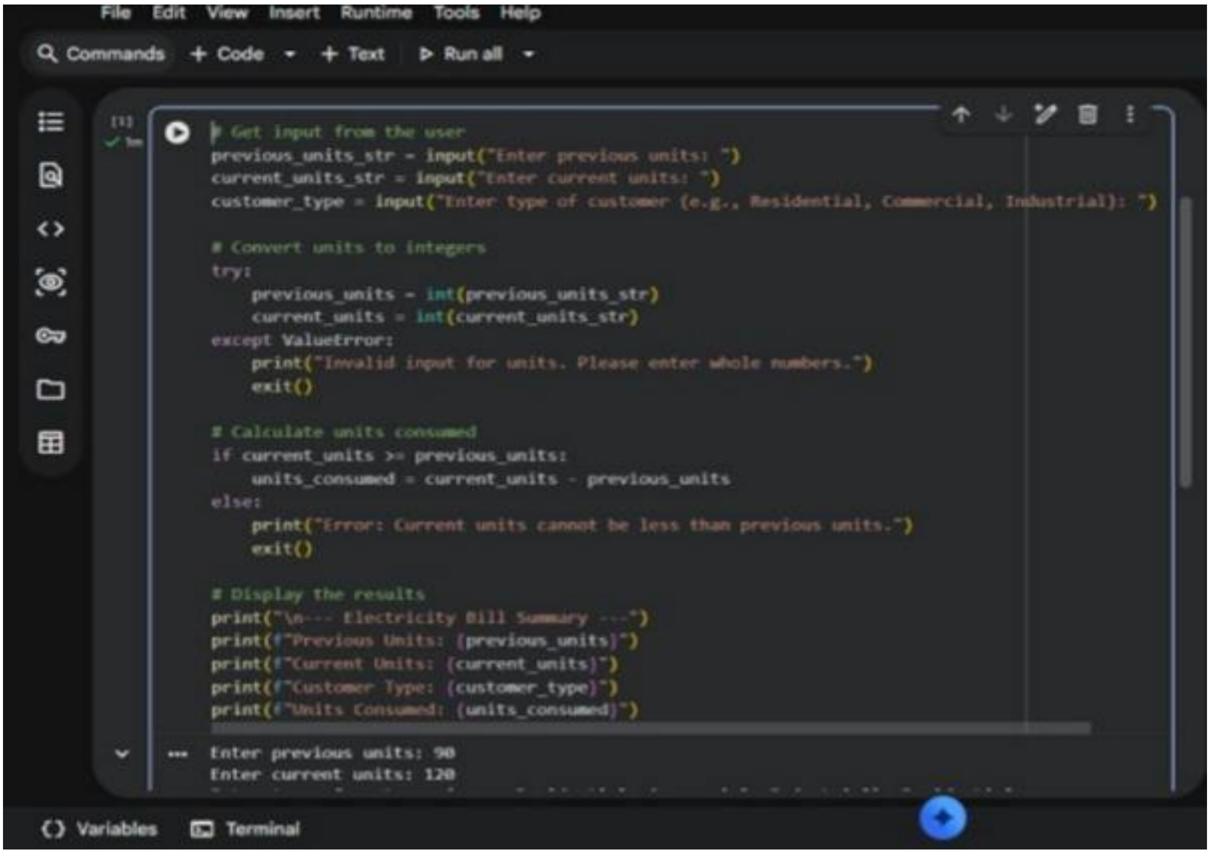
PROMPT:



Write a Python program to read consumer details for an electricity billing system. The program should read Previous Units (PU), Current Units (CU), and Type of Customer. Calculate the units consumed using: Units Consumed = Current Units – Previous Units Implement the logic directly in the main program without using functions.

Working...

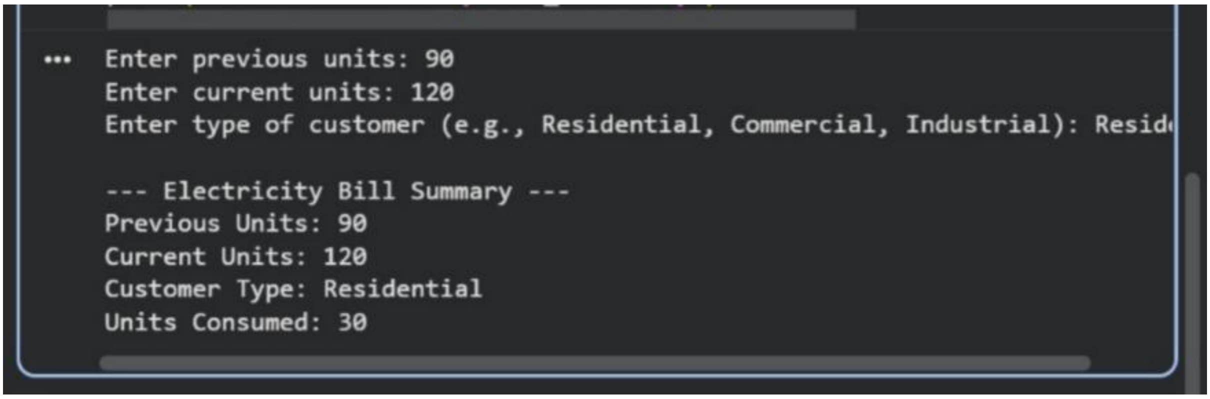
CODE:



```python
# Get input from the user
previous_units_str = input("Enter previous units: ")
current_units_str = input("Enter current units: ")
customer_type = input("Enter type of customer (e.g., Residential, Commercial, Industrial): ")

# Convert units to integers
try:
    previous_units = int(previous_units_str)
    current_units = int(current_units_str)
except ValueError:
    print("Invalid input for units. Please enter whole numbers.")
    exit()

# Calculate units consumed
if current_units >= previous_units:
    units_consumed = current_units - previous_units
else:
    print("Error: Current units cannot be less than previous units.")
    exit()

# Display the results
print("\n--- Electricity Bill Summary ---")
print(f"Previous Units: {previous_units}")
print(f"Current Units: {current_units}")
print(f"Customer Type: {customer_type}")
print(f"Units Consumed: {units_consumed}")
```

```
Enter previous units: 90
Enter current units: 120
```

OUTPUT:



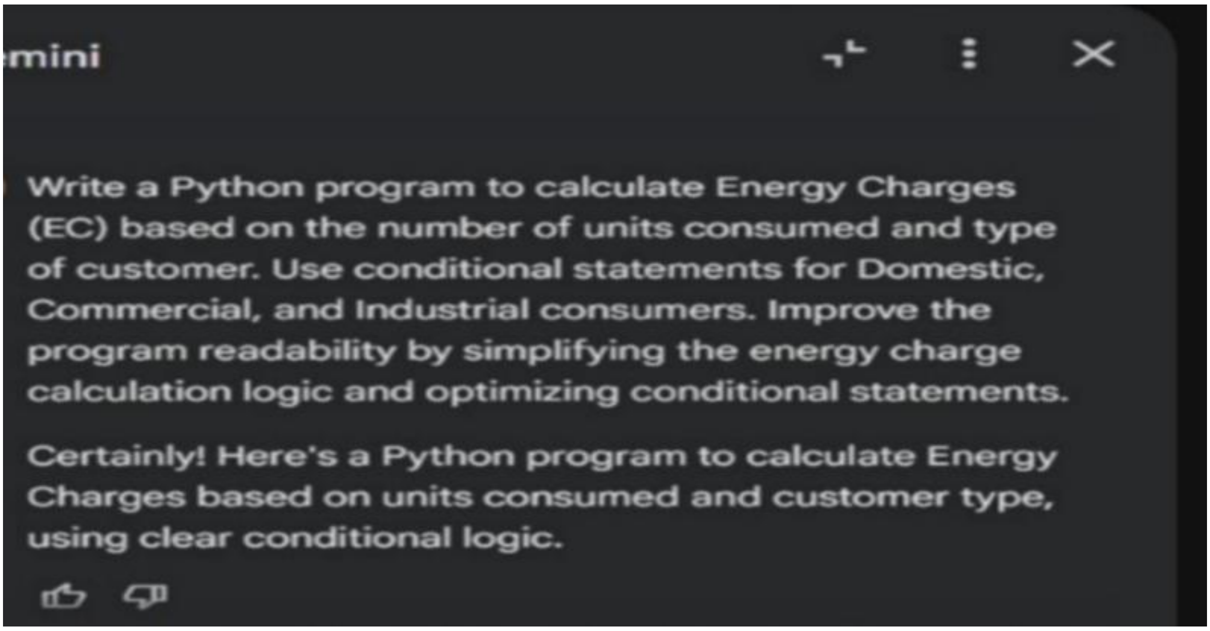
```
Enter previous units: 90
Enter current units: 120
Enter type of customer (e.g., Residential, Commercial, Industrial): Resid

--- Electricity Bill Summary ---
Previous Units: 90
Current Units: 120
Customer Type: Residential
Units Consumed: 30
```

EXPLANATION:

- The program first reads the previous meter reading.
- It then reads the current meter reading.
- The type of customer (Domestic / Commercial / Industrial) is taken as input.
- Units consumed are calculated by subtracting previous units from current units.
- Finally, the program displays the customer type and total units consumed.

<u>Task 2:</u> Energy Charges Calculation Based on Units Consumed.

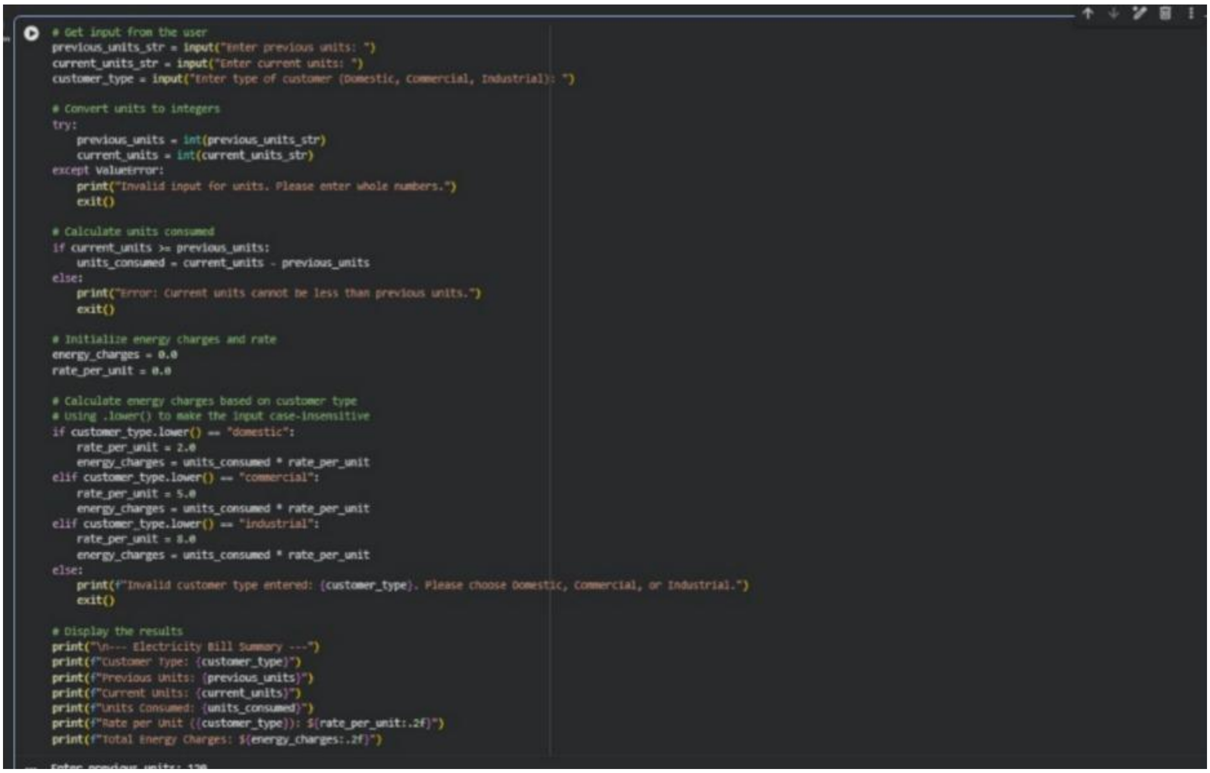
<u>PROMPT:</u>



Write a Python program to calculate Energy Charges (EC) based on the number of units consumed and type of customer. Use conditional statements for Domestic, Commercial, and Industrial consumers. Improve the program readability by simplifying the energy charge calculation logic and optimizing conditional statements.

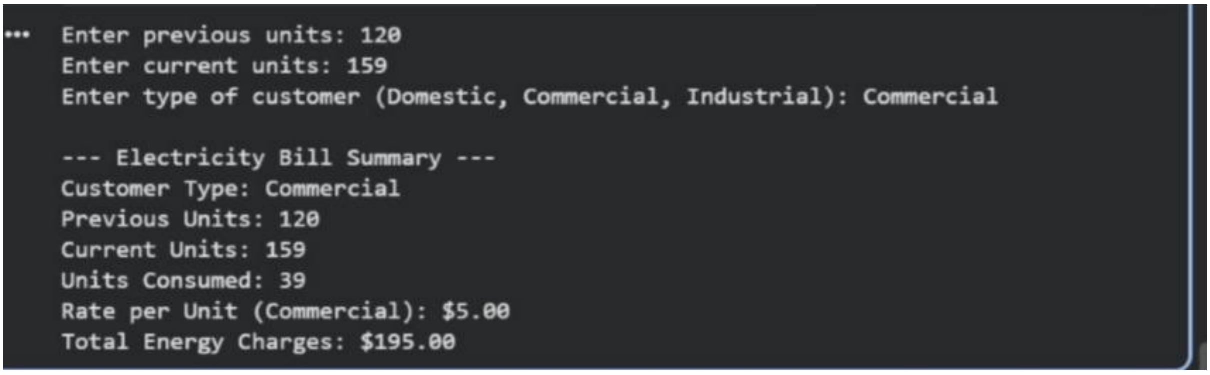
Certainly! Here's a Python program to calculate Energy Charges based on units consumed and customer type, using clear conditional logic.

<u>CODE:</u>



```
# Get input from the user
previous_units_str = input("Enter previous units: ")
current_units_str = input("Enter current units: ")
customer_type = input("Enter type of customer (Domestic, Commercial, Industrial): ")

# Convert units to integers
try:
    previous_units = int(previous_units_str)
    current_units = int(current_units_str)
except ValueError:
    print("Invalid input for units. Please enter whole numbers.")
    exit()

# Calculate units consumed
if current_units >= previous_units:
    units_consumed = current_units - previous_units
else:
    print("Error: Current units cannot be less than previous units.")
    exit()

# Initialize energy charges and rate
energy_charges = 0.0
rate_per_unit = 0.0

# Calculate energy charges based on customer type
# Using .lower() to make the input case-insensitive
if customer_type.lower() == "domestic":
    rate_per_unit = 2.0
    energy_charges = units_consumed * rate_per_unit
elif customer_type.lower() == "commercial":
    rate_per_unit = 5.0
    energy_charges = units_consumed * rate_per_unit
elif customer_type.lower() == "industrial":
    rate_per_unit = 8.0
    energy_charges = units_consumed * rate_per_unit
else:
    print(f"Invalid customer type entered: {customer_type}. Please choose Domestic, Commercial, or Industrial.")
    exit()

# Display the results
print("\n--- Electricity Bill Summary ---")
print(f"Customer Type: {customer_type}")
print(f"Previous Units: {previous_units}")
print(f"Current Units: {current_units}")
print(f"Units Consumed: {units_consumed}")
print(f"Rate per Unit ({customer_type}): ${rate_per_unit:.2f}")
print(f"Total Energy Charges: ${energy_charges:.2f}")
```

<u>OUTPUT:</u>



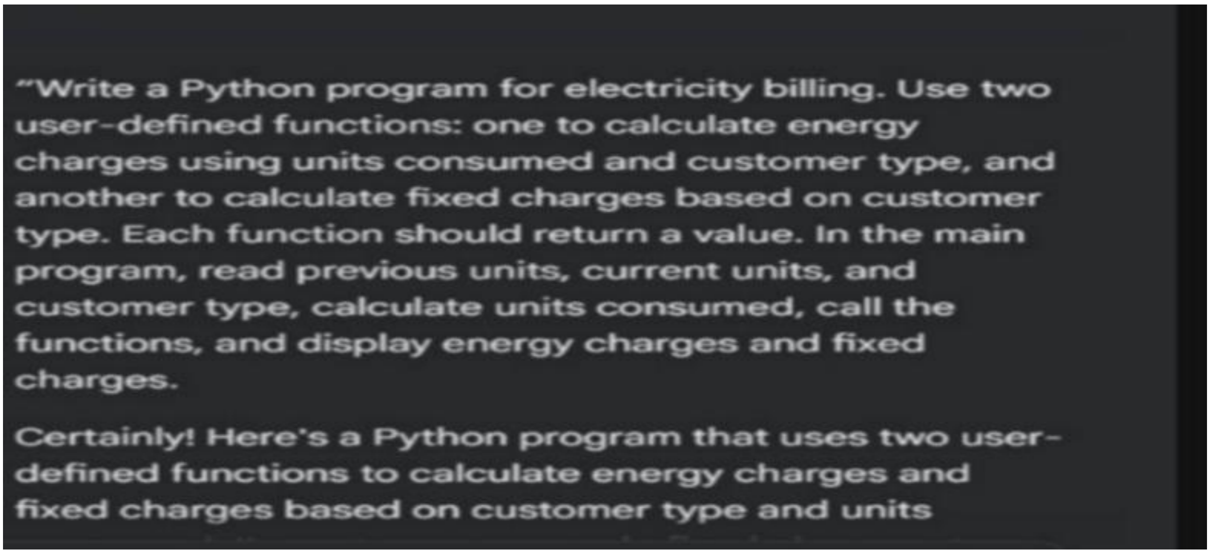
```
Enter previous units: 120
Enter current units: 159
Enter type of customer (Domestic, Commercial, Industrial): Commercial

--- Electricity Bill Summary ---
Customer Type: Commercial
Previous Units: 120
Current Units: 159
Units Consumed: 39
Rate per Unit (Commercial): $5.00
Total Energy Charges: $195.00
```

EXPLANATION

- The program takes previous meter reading and current meter reading as input.
- It also takes the type of customer (Domestic, Commercial, or Industrial).
- The program calculates units consumed by subtracting previous units from current units.
- Using if–else conditions, it selects the rate based on customer type.
- Energy charges are calculated by multiplying units consumed × rate.
- Finally, the program displays the customer type, units consumed, and energy charges.
- All logic is written inside the main method without using functions.
- The program is simple and easy to understand for beginners.

Task 3: Modular Design Using AI Assistance (Using Functions).

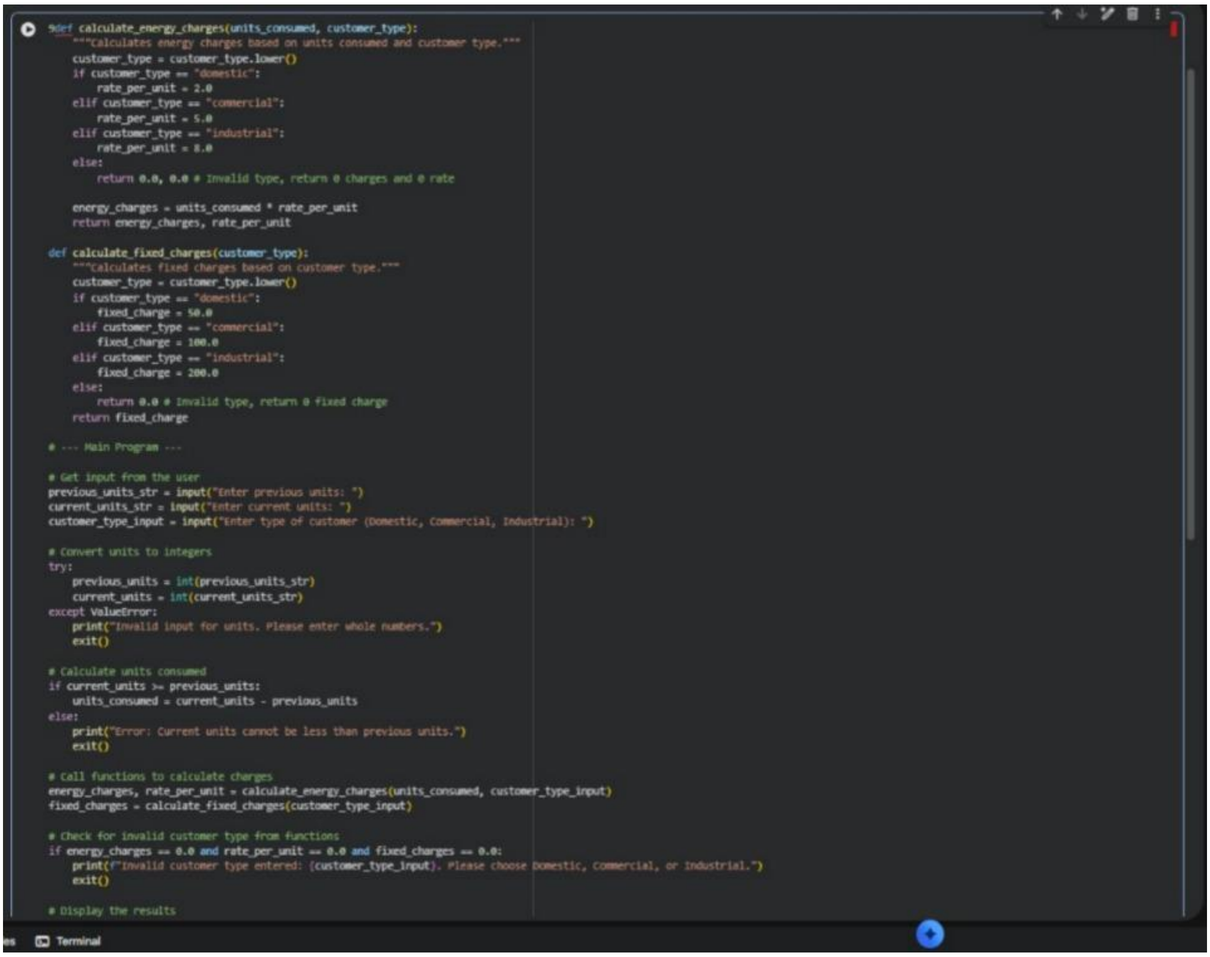
PROMPT:



"Write a Python program for electricity billing. Use two user-defined functions: one to calculate energy charges using units consumed and customer type, and another to calculate fixed charges based on customer type. Each function should return a value. In the main program, read previous units, current units, and customer type, calculate units consumed, call the functions, and display energy charges and fixed charges.

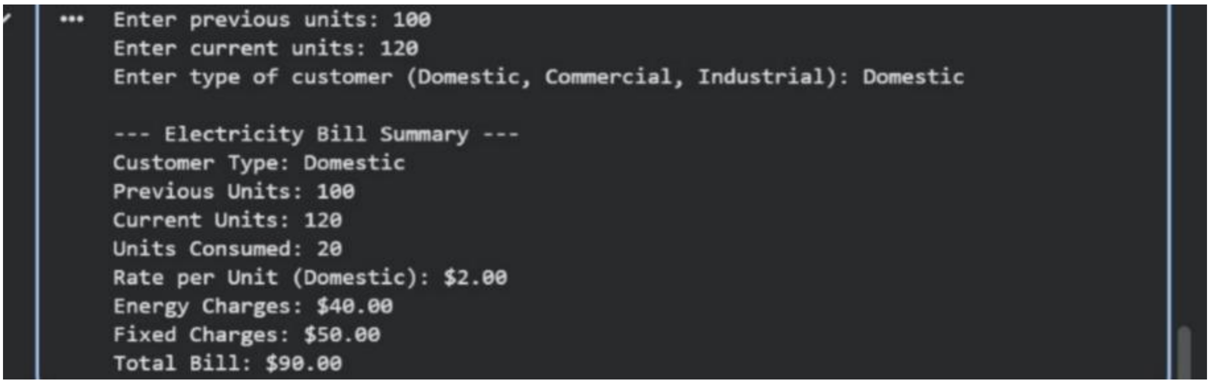
Certainly! Here's a Python program that uses two user-defined functions to calculate energy charges and fixed charges based on customer type and units

CODE:



```python
def calculate_energy_charges(units_consumed, customer_type):
    """Calculates energy charges based on units consumed and customer type."""
    customer_type = customer_type.lower()
    if customer_type == "domestic":
        rate_per_unit = 2.0
    elif customer_type == "commercial":
        rate_per_unit = 5.0
    elif customer_type == "industrial":
        rate_per_unit = 8.0
    else:
        return 0.0, 0.0 # Invalid type, return 0 charges and 0 rate

    energy_charges = units_consumed * rate_per_unit
    return energy_charges, rate_per_unit

def calculate_fixed_charges(customer_type):
    """Calculates fixed charges based on customer type."""
    customer_type = customer_type.lower()
    if customer_type == "domestic":
        fixed_charge = 50.0
    elif customer_type == "commercial":
        fixed_charge = 100.0
    elif customer_type == "industrial":
        fixed_charge = 200.0
    else:
        return 0.0 # Invalid type, return 0 fixed charge
    return fixed_charge

# --- Main Program ---

# Get input from the user
previous_units_str = input("Enter previous units: ")
current_units_str = input("Enter current units: ")
customer_type_input = input("Enter type of customer (Domestic, Commercial, Industrial): ")

# Convert units to integers
try:
    previous_units = int(previous_units_str)
    current_units = int(current_units_str)
except ValueError:
    print("Invalid input for units. Please enter whole numbers.")
    exit()

# Calculate units consumed
if current_units >= previous_units:
    units_consumed = current_units - previous_units
else:
    print("Error: Current units cannot be less than previous units.")
    exit()

# Call functions to calculate charges
energy_charges, rate_per_unit = calculate_energy_charges(units_consumed, customer_type_input)
fixed_charges = calculate_fixed_charges(customer_type_input)

# Check for invalid customer type from functions
if energy_charges == 0.0 and rate_per_unit == 0.0 and fixed_charges == 0.0:
    print(f"Invalid customer type entered: {customer_type_input}. Please choose Domestic, Commercial, or Industrial.")
    exit()

# Display the results
```

OUTPUT:



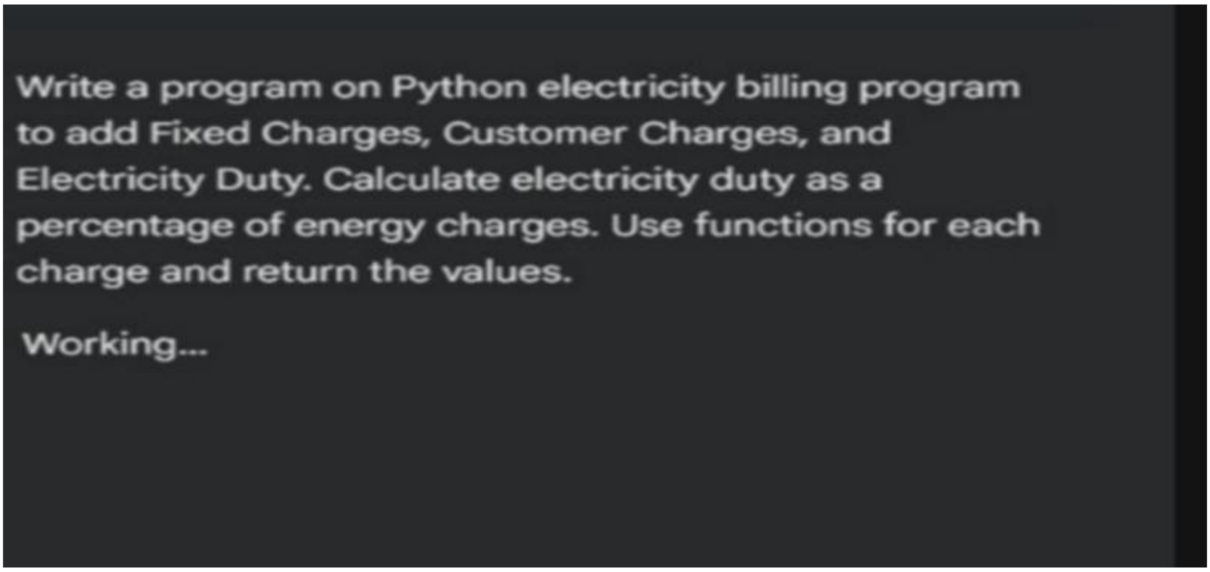
```
Enter previous units: 100
Enter current units: 120
Enter type of customer (Domestic, Commercial, Industrial): Domestic

--- Electricity Bill Summary ---
Customer Type: Domestic
Previous Units: 100
Current Units: 120
Units Consumed: 20
Rate per Unit (Domestic): $2.00
Energy Charges: $40.00
Fixed Charges: $50.00
Total Bill: $90.00
```

EXPLANATION:

- The program calculates an electricity bill.
- It uses functions so the same logic can be used again.
- One function calculates energy charges.
- Another function calculates fixed charges.
- Functions return the calculated values.
- The main program reads previous units, current units, and customer type.
- It finds units consumed.
- It calls the functions to get charges.
- Finally, it displays energy charges and fixed charges.

Task 4: Calculation of Additional Charges

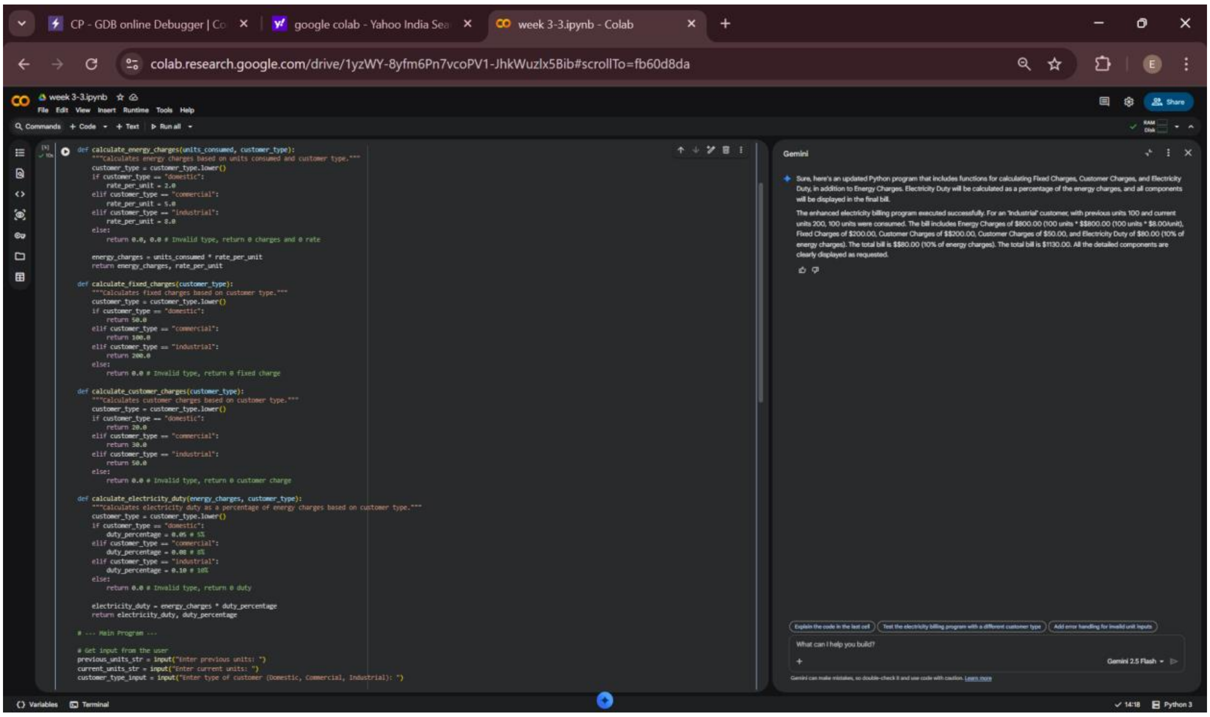
PROMPT:



Write a program on Python electricity billing program to add Fixed Charges, Customer Charges, and Electricity Duty. Calculate electricity duty as a percentage of energy charges. Use functions for each charge and return the values.

Working...

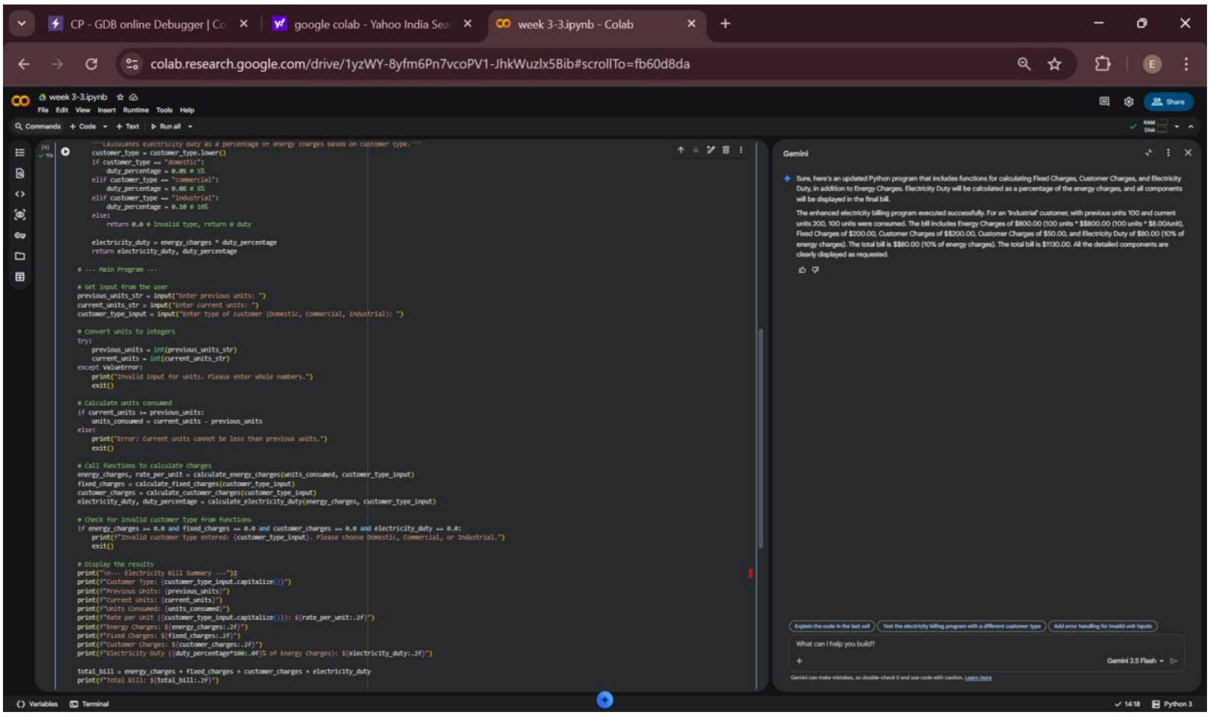
CODE:


CP - GDB online Debugger | Co
google colab - Yahoo India Sea
week 3-3.ipynb - Colab
colab.research.google.com/drive/1yzWY-8yfm6Pn7vcoPV1-JhkWuzlx5Bib#scrollTo=fb60d8da
week 3-3.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text ▷ Run all
Share
RAM
Disk
Gemini
Explain the code in the last cell
Test the electricity billing program with a different customer type
Add error handling for invalid unit inputs
What can I help you build?
Gemini 2.5 Flash
Gemini can make mistakes, so double-check it and use code with caution. Learn more
Variables
Terminal
14:18
Python 3

```python
def calculate_energy_charges(units_consumed, customer_type):
    """Calculates energy charges based on units consumed and customer type."""
    customer_type = customer_type.lower()
    if customer_type == "domestic":
        rate_per_unit = 2.0
    elif customer_type == "commercial":
        rate_per_unit = 5.0
    elif customer_type == "industrial":
        rate_per_unit = 8.0
    else:
        return 0.0, 0.0 # Invalid type, return 0 charges and 0 rate

    energy_charges = units_consumed * rate_per_unit
    return energy_charges, rate_per_unit

def calculate_fixed_charges(customer_type):
    """Calculates fixed charges based on customer type."""
    customer_type = customer_type.lower()
    if customer_type == "domestic":
        return 50.0
    elif customer_type == "commercial":
        return 100.0
    elif customer_type == "industrial":
        return 200.0
    else:
        return 0.0 # Invalid type, return 0 fixed charge

def calculate_customer_charges(customer_type):
    """Calculates customer charges based on customer type."""
    customer_type = customer_type.lower()
    if customer_type == "domestic":
        return 20.0
    elif customer_type == "commercial":
        return 30.0
    elif customer_type == "industrial":
        return 50.0
    else:
        return 0.0 # Invalid type, return 0 customer charge

def calculate_electricity_duty(energy_charges, customer_type):
    """Calculates electricity duty as a percentage of energy charges based on customer type."""
    customer_type = customer_type.lower()
    if customer_type == "domestic":
        duty_percentage = 0.05 # 5%
    elif customer_type == "commercial":
        duty_percentage = 0.08 # 8%
    elif customer_type == "industrial":
        duty_percentage = 0.10 # 10%
    else:
        return 0.0 # Invalid type, return 0 duty

    electricity_duty = energy_charges * duty_percentage
    return electricity_duty, duty_percentage

# --- Main Program ---

# Get input from the user
previous_units_str = input("Enter previous units: ")
current_units_str = input("Enter current units: ")
customer_type_input = input("Enter type of customer (Domestic, Commercial, Industrial): ")
```

Sure, here's an updated Python program that includes functions for calculating Fixed Charges, Customer Charges, and Electricity Duty, in addition to Energy Charges. Electricity Duty will be calculated as a percentage of the energy charges, and all components will be displayed in the final bill.

The enhanced electricity billing program executed successfully. For an 'Industrial' customer, with previous units 100 and current units 200, 100 units were consumed. The bill includes Energy Charges of $800.00 (100 units * $$800.00 (100 units * $8.00/unit), Fixed Charges of $200.00, Customer Charges of $$200.00, Customer Charges of $50.00, and Electricity Duty of $80.00 (10% of energy charges). The total bill is $$80.00 (10% of energy charges). The total bill is $1130.00. All the detailed components are clearly displayed as requested.

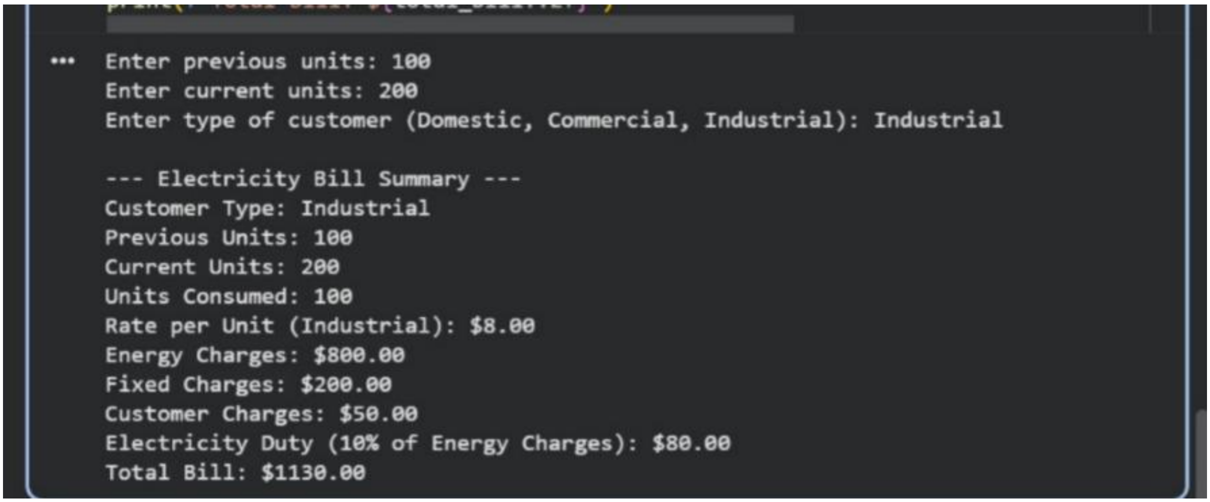

CP - GDB online Debugger | Co
google colab - Yahoo India Sea
week 3-3.ipynb - Colab
colab.research.google.com/drive/1yzWY-8yfm6Pn7vcoPV1-JhkWuzlx5Bib#scrollTo=fb60d8da
week 3-3.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text ▷ Run all
Share
RAM
Disk
Gemini
Explain the code in the last cell
Test the electricity billing program with a different customer type
Add error handling for invalid unit inputs
What can I help you build?
Gemini 2.5 Flash
Gemini can make mistakes, so double-check it and use code with caution. Learn more
Variables
Terminal
14:18
Python 3

```python
    """Calculates electricity duty as a percentage of energy charges based on customer type."""
    customer_type = customer_type.lower()
    if customer_type == "domestic":
        duty_percentage = 0.05 # 5%
    elif customer_type == "commercial":
        duty_percentage = 0.08 # 8%
    elif customer_type == "industrial":
        duty_percentage = 0.10 # 10%
    else:
        return 0.0 # Invalid type, return 0 duty

    electricity_duty = energy_charges * duty_percentage
    return electricity_duty, duty_percentage

# --- Main Program ---

# Get input from the user
previous_units_str = input("Enter previous units: ")
current_units_str = input("Enter current units: ")
customer_type_input = input("Enter type of customer (Domestic, Commercial, Industrial): ")

# Convert units to integers
try:
    previous_units = int(previous_units_str)
    current_units = int(current_units_str)
except ValueError:
    print("Invalid input for units. Please enter whole numbers.")
    exit()

# Calculate units consumed
if current_units >= previous_units:
    units_consumed = current_units - previous_units
else:
    print("Error: Current units cannot be less than previous units.")
    exit()

# Call functions to calculate charges
energy_charges, rate_per_unit = calculate_energy_charges(units_consumed, customer_type_input)
fixed_charges = calculate_fixed_charges(customer_type_input)
customer_charges = calculate_customer_charges(customer_type_input)
electricity_duty, duty_percentage = calculate_electricity_duty(energy_charges, customer_type_input)

# Check for invalid customer type from functions
if energy_charges == 0.0 and fixed_charges == 0.0 and customer_charges == 0.0 and electricity_duty == 0.0:
    print(f"Invalid customer type entered: {customer_type_input}. Please choose Domestic, Commercial, or Industrial.")
    exit()

# Display the results
print("\n--- Electricity Bill Summary ---")2
print(f"Customer Type: {customer_type_input.capitalize()}")
print(f"Previous Units: {previous_units}")
print(f"Current Units: {current_units}")
print(f"Units Consumed: {units_consumed}")
print(f"Rate per Unit ({customer_type_input.capitalize()}): ${rate_per_unit:.2f}")
print(f"Energy Charges: ${energy_charges:.2f}")
print(f"Fixed Charges: ${fixed_charges:.2f}")
print(f"Customer Charges: ${customer_charges:.2f}")
print(f"Electricity Duty ({duty_percentage*100:.0f}% of Energy Charges): ${electricity_duty:.2f}")

total_bill = energy_charges + fixed_charges + customer_charges + electricity_duty
print(f"Total Bill: ${total_bill:.2f}")
```

Sure, here's an updated Python program that includes functions for calculating Fixed Charges, Customer Charges, and Electricity Duty, in addition to Energy Charges. Electricity Duty will be calculated as a percentage of the energy charges, and all components will be displayed in the final bill.

The enhanced electricity billing program executed successfully. For an 'Industrial' customer, with previous units 100 and current units 200, 100 units were consumed. The bill includes Energy Charges of $800.00 (100 units * $$800.00 (100 units * $8.00/unit), Fixed Charges of $200.00, Customer Charges of $$200.00, Customer Charges of $50.00, and Electricity Duty of $80.00 (10% of energy charges). The total bill is $$80.00 (10% of energy charges). The total bill is $1130.00. All the detailed components are clearly displayed as requested.

OUTPUT:



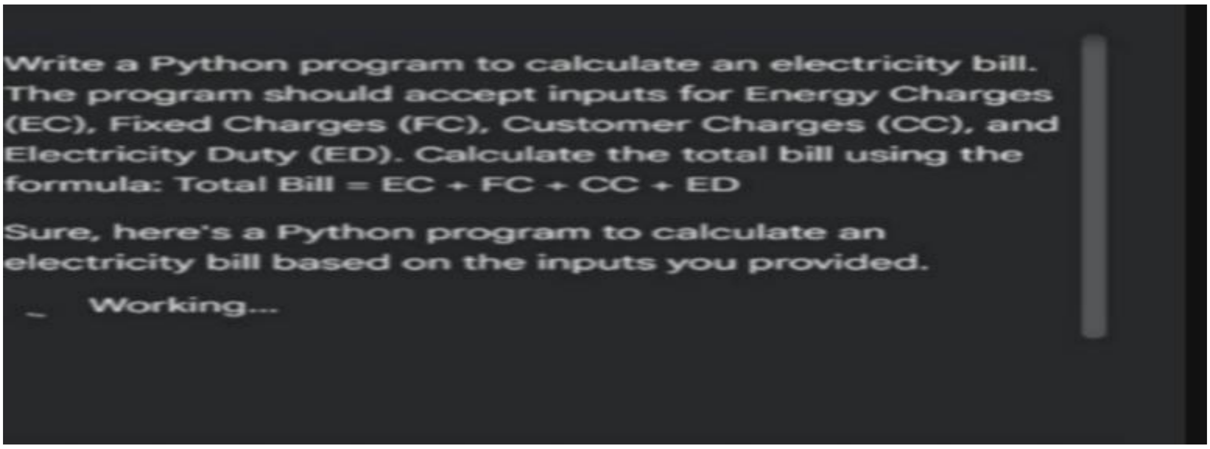
```
Enter previous units: 100
Enter current units: 200
Enter type of customer (Domestic, Commercial, Industrial): Industrial

--- Electricity Bill Summary ---
Customer Type: Industrial
Previous Units: 100
Current Units: 200
Units Consumed: 100
Rate per Unit (Industrial): $8.00
Energy Charges: $800.00
Fixed Charges: $200.00
Customer Charges: $50.00
Electricity Duty (10% of Energy Charges): $80.00
Total Bill: $1130.00
```

EXPLANATION

- The program calculates an electricity bill.
- It uses functions to calculate different charges so the logic can be reused.
- There is a function for Energy Charges (EC), based on units used and customer type.
- Another function calculates Fixed Charges (FC) and another for Customer Charges (CC).
- Electricity Duty (ED) is calculated as a percentage of Energy Charges.
- The program first reads previous units, current units, and customer type.
- It finds units consumed and calls the functions to get all charges.
- Finally, it prints all charges separately in a clear and readable way.
- The program is simple, reusable, and easy to understand.

Task 5: Final Bill Generation and Output Analysis

PROMPT:

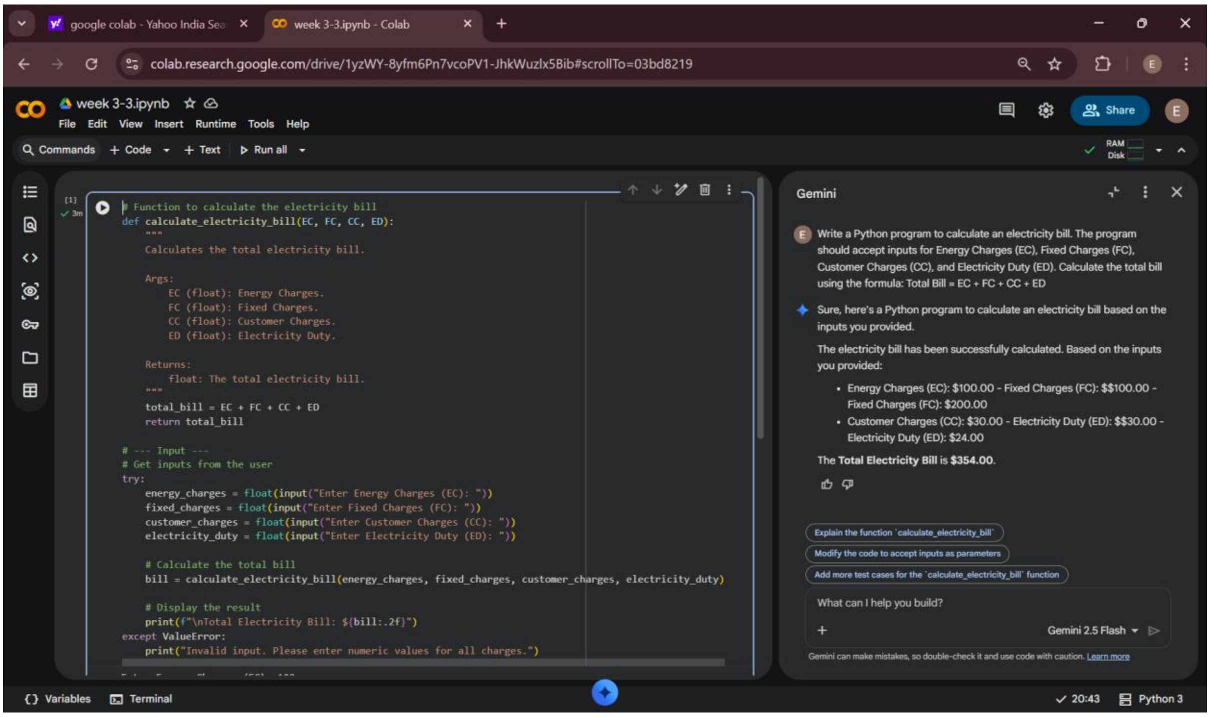


Write a Python program to calculate an electricity bill. The program should accept inputs for Energy Charges (EC), Fixed Charges (FC), Customer Charges (CC), and Electricity Duty (ED). Calculate the total bill using the formula: Total Bill = EC + FC + CC + ED

Sure, here's a Python program to calculate an electricity bill based on the inputs you provided.

Working...

CODE:



google colab - Yahoo India Sea
week 3-3.ipynb - Colab
colab.research.google.com/drive/1yzWY-8yfm6Pn7vcoPV1-JhkWuzlx5Bib#scrollTo=03bd8219
week 3-3.ipynb
File Edit View Insert Runtime Tools Help
Commands
+ Code
+ Text
Run all
Share
RAM
Disk
[1]
3m
Variables
Terminal
20:43
Python 3

```
# Function to calculate the electricity bill
def calculate_electricity_bill(EC, FC, CC, ED):
    """
    Calculates the total electricity bill.

    Args:
        EC (float): Energy Charges.
        FC (float): Fixed Charges.
        CC (float): Customer Charges.
        ED (float): Electricity Duty.

    Returns:
        float: The total electricity bill.
    """
    total_bill = EC + FC + CC + ED
    return total_bill

# --- Input ---
# Get inputs from the user
try:
    energy_charges = float(input("Enter Energy Charges (EC): "))
    fixed_charges = float(input("Enter Fixed Charges (FC): "))
    customer_charges = float(input("Enter Customer Charges (CC): "))
    electricity_duty = float(input("Enter Electricity Duty (ED): "))

    # Calculate the total bill
    bill = calculate_electricity_bill(energy_charges, fixed_charges, customer_charges, electricity_duty)

    # Display the result
    print(f"\nTotal Electricity Bill: ${bill:.2f}")
except ValueError:
    print("Invalid input. Please enter numeric values for all charges.")
```

Gemini

Write a Python program to calculate an electricity bill. The program should accept inputs for Energy Charges (EC), Fixed Charges (FC), Customer Charges (CC), and Electricity Duty (ED). Calculate the total bill using the formula: Total Bill = EC + FC + CC + ED

Sure, here's a Python program to calculate an electricity bill based on the inputs you provided.

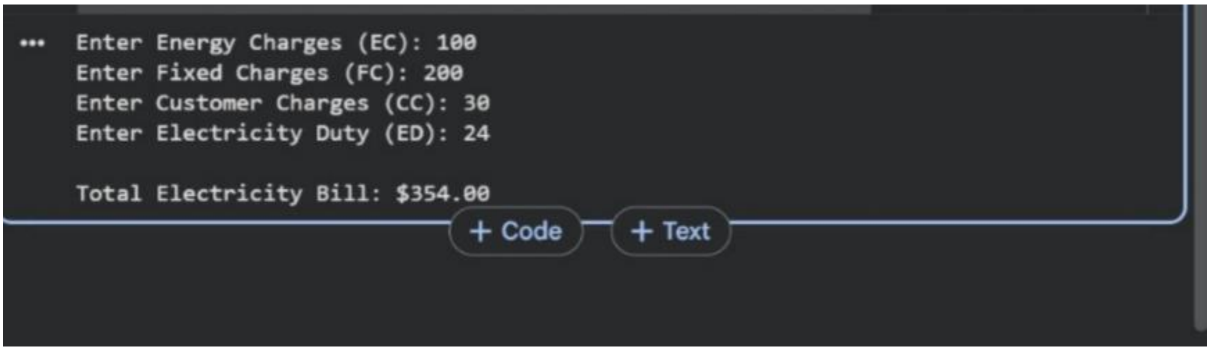
The electricity bill has been successfully calculated. Based on the inputs you provided:

- Energy Charges (EC): $100.00 - Fixed Charges (FC): $$100.00 - Fixed Charges (FC): $200.00
- Customer Charges (CC): $30.00 - Electricity Duty (ED): $$30.00 - Electricity Duty (ED): $24.00

The **Total Electricity Bill** is **$354.00**.

Explain the function `calculate_electricity_bill`
Modify the code to accept inputs as parameters
Add more test cases for the `calculate_electricity_bill` function
What can I help you build?
Gemini 2.5 Flash
Gemini can make mistakes, so double-check it and use code with caution. Learn more

OUTPUT:



```
Enter Energy Charges (EC): 100
Enter Fixed Charges (FC): 200
Enter Customer Charges (CC): 30
Enter Electricity Duty (ED): 24

Total Electricity Bill: $354.00
```

EXPLANATION:

- Takes user input for Energy Charges (EC), Fixed Charges (FC), Customer Charges (CC), and Electricity Duty (ED).
- Stores each value in a separate variable for clarity.
- Calculates the total electricity bill by adding all charges.
- Uses the formula: Total Bill = EC + FC + CC + ED.
- Displays each charge clearly.
- Displays the final total bill amount.
- Uses simple logic and readable variable names.
- Represents a real-world electricity billing process.