AI Assisted Coding

ASSIGNMENT 3.3

Name: K.Rupaas shyni

HT No: 2303A52417

Batch: 32

Lab 3: Application for TGNPDCL - Electricity Bill Generation Using Python & AI Tools

Lab Objectives

• To design a real-world electricity billing application using Python

• To use AI-assisted coding tools for logic generation and optimization

• To understand conditional logic and arithmetic operations

• To generate structured billing output similar to utility bills

Question 1:

Task 1: AI-Generated Logic for Reading Consumer Details

Scenario:

An electricity billing system must collect accurate consumer data.

Task Description:

Use an AI tool (GitHub Copilot / Gemini) to generate a Python program that:

• Reads:

Previous Units (PU)

Current Units (CU)

Type of Customer

• Calculates units consumed

• Implements logic directly in the main program (no functions)

Expected Output:

• Correct input reading

• Units consumed calculation

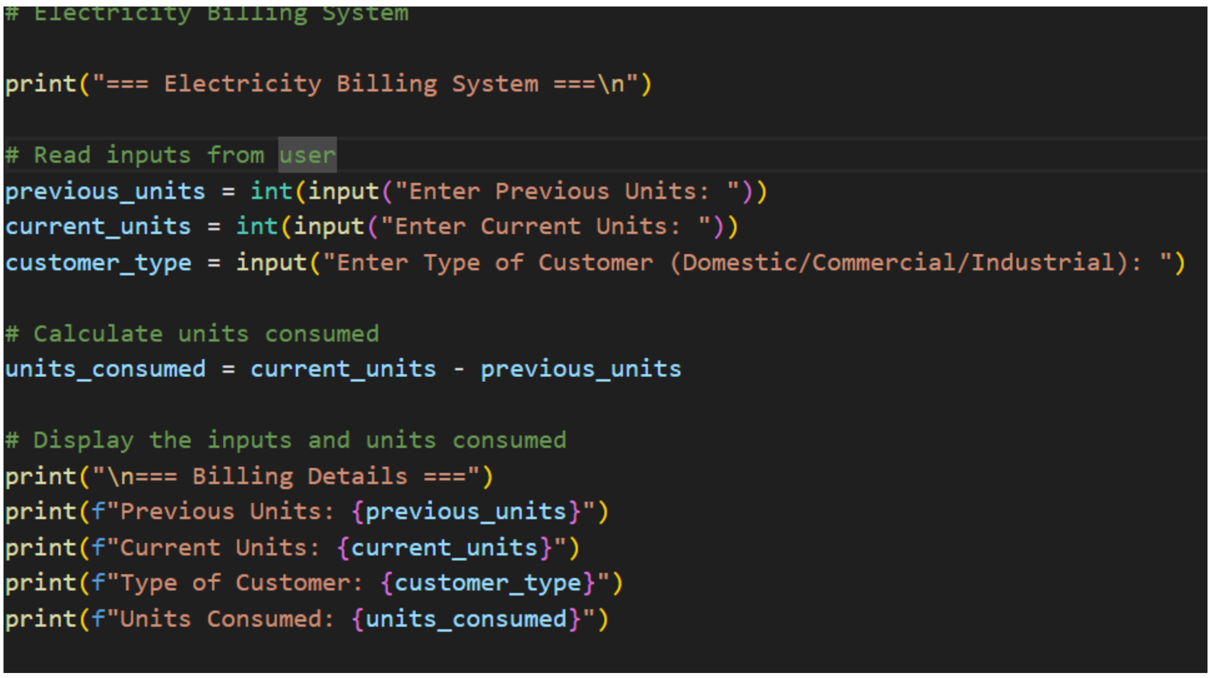
• Screenshot showing AI-generated code

• Sample input and output

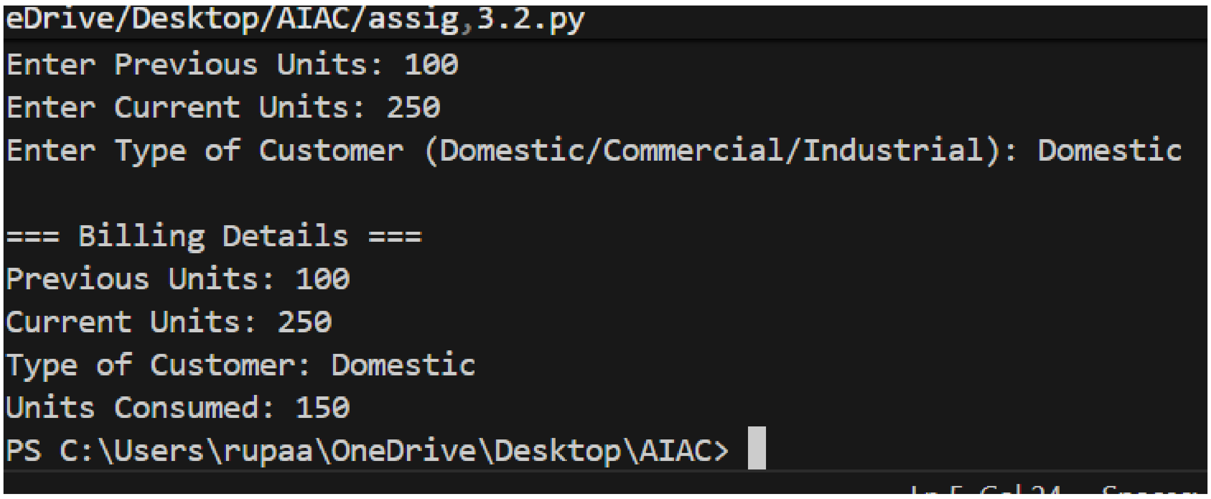
Prompt:

Write a Python program for an electricity billing system that reads Previous Units, Current Units, and Type of Customer from the user, calculates units consumed as (Current Units - Previous Units), implements all logic directly in the main program without using functions, and prints the inputs and units consumed.

Code:



```python
# Electricity Billing System

print("=== Electricity Billing System ===\n")

# Read inputs from user
previous_units = int(input("Enter Previous Units: "))
current_units = int(input("Enter Current Units: "))
customer_type = input("Enter Type of Customer (Domestic/Commercial/Industrial): ")

# Calculate units consumed
units_consumed = current_units - previous_units

# Display the inputs and units consumed
print("\n=== Billing Details ===")
print(f"Previous Units: {previous_units}")
print(f"Current Units: {current_units}")
print(f"Type of Customer: {customer_type}")
print(f"Units Consumed: {units_consumed}")
```



```
eDrive/Desktop/AIAC/assig,3.2.py
Enter Previous Units: 100
Enter Current Units: 250
Enter Type of Customer (Domestic/Commercial/Industrial): Domestic

=== Billing Details ===
Previous Units: 100
Current Units: 250
Type of Customer: Domestic
Units Consumed: 150
PS C:\Users\rupaa\OneDrive\Desktop\AIAC>
```

Explanation:

This task uses an AI tool to generate a Python program that reads consumer details such as Previous Units, Current Units, and Customer Type. The program calculates the units consumed by subtracting Previous Units from Current Units. All logic is written directly in the main program without using functions, ensuring simplicity and clarity. The output correctly displays the entered values and the calculated units consumed, along with sample input and output for verification

Question 2:

Task 2: Energy Charges Calculation Based on Units Consumed Scenario:

Energy charges depend on the number of units consumed and customer type.

Task Description:

Review the AI-generated code from Task 1 and extend it to:

• Calculate Energy Charges (EC)

• Use conditional statements based on:

Domestic

Commercial

Industrial consumers

• Improve readability using AI prompts such as:

“Simplify energy charge calculation logic”

“Optimize conditional statements”

Expected Output

• Correct EC calculation

• Clear conditional logic

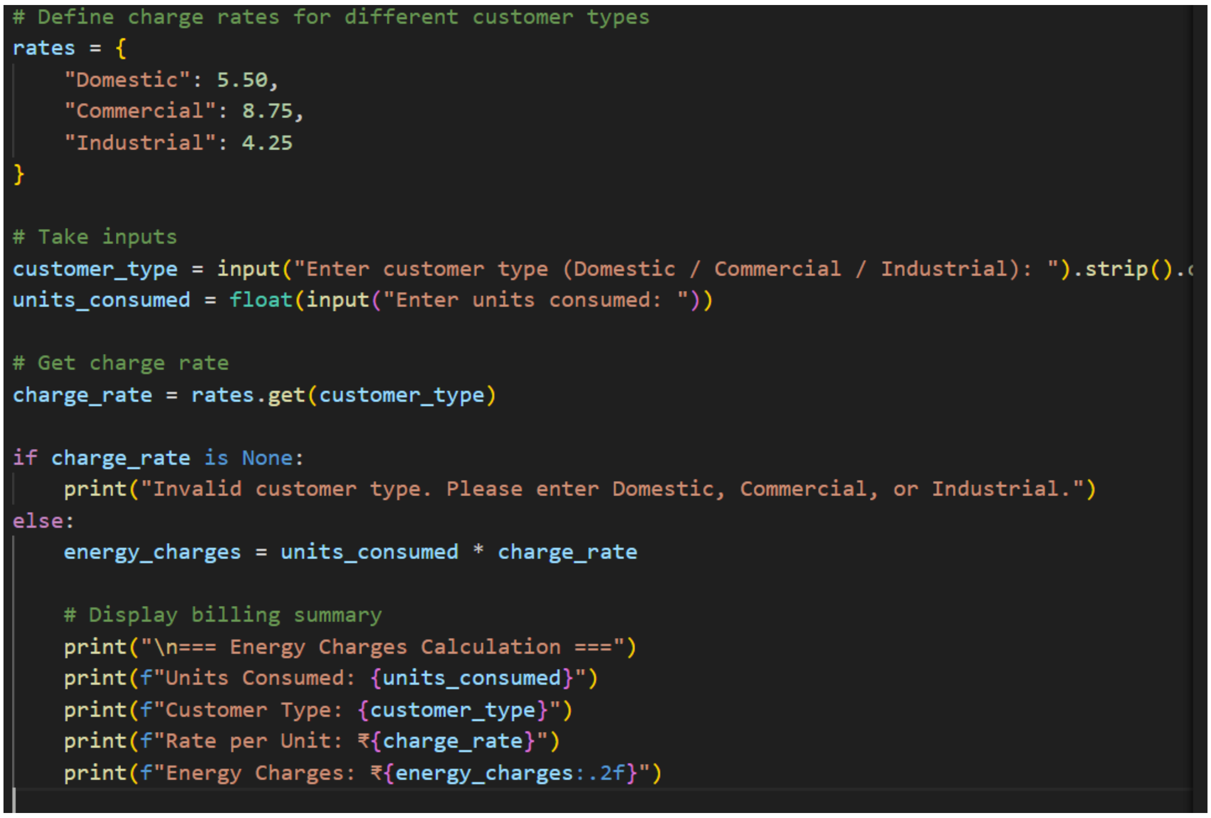
• Original and improved versions (optional)

• Sample execution results

Prompt:

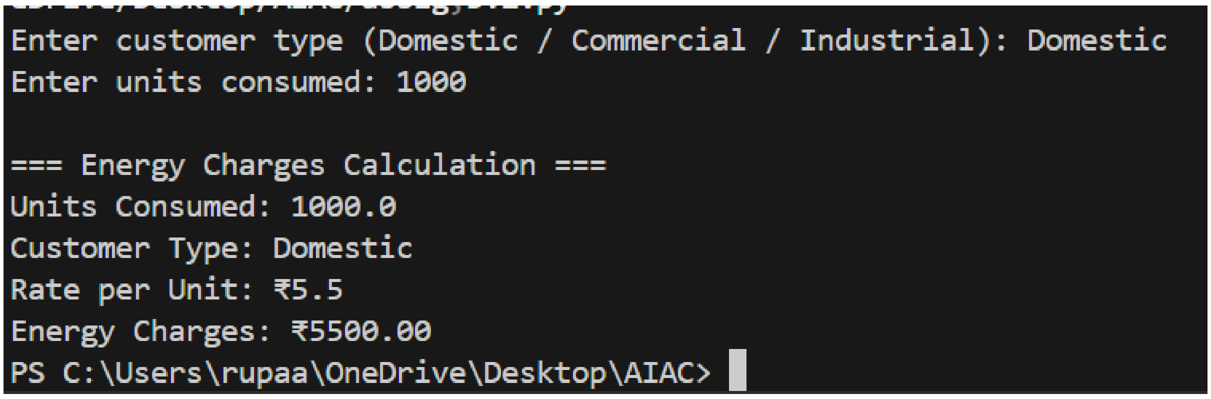
Review the existing Python electricity billing program and extend it to calculate Energy Charges based on units consumed and customer type. Use conditional statements to handle Domestic, Commercial, and Industrial consumers with different charge rates. Improve code readability by simplifying the energy charge calculation logic and optimizing conditional statements. Print units consumed, customer type, and calculated Energy Charges.

Code:



```python
# Define charge rates for different customer types
rates = {
    "Domestic": 5.50,
    "Commercial": 8.75,
    "Industrial": 4.25
}

# Take inputs
customer_type = input("Enter customer type (Domestic / Commercial / Industrial): ").strip().c
units_consumed = float(input("Enter units consumed: "))

# Get charge rate
charge_rate = rates.get(customer_type)

if charge_rate is None:
    print("Invalid customer type. Please enter Domestic, Commercial, or Industrial.")
else:
    energy_charges = units_consumed * charge_rate

    # Display billing summary
    print("\n=== Energy Charges Calculation ===")
    print(f"Units Consumed: {units_consumed}")
    print(f"Customer Type: {customer_type}")
    print(f"Rate per Unit: ₹{charge_rate}")
    print(f"Energy Charges: ₹{energy_charges:.2f}")
```

Output:



```
Enter customer type (Domestic / Commercial / Industrial): Domestic
Enter units consumed: 1000

=== Energy Charges Calculation ===
Units Consumed: 1000.0
Customer Type: Domestic
Rate per Unit: ₹5.5
Energy Charges: ₹5500.00
PS C:\Users\rupaa\OneDrive\Desktop\AIAC>
```

Explanation:

In this task, the program generated in Task 1 is extended to calculate Energy Charges based on units consumed and the type of customer. Conditional statements are used to apply different charge rates for Domestic, Commercial, and Industrial consumers. AI assistance helps simplify the calculation logic and optimize conditions, improving readability. The program correctly computes and displays Energy Charges with sample execution results for verification.

Question 3:

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

Billing logic must be reusable for multiple consumers.

Task Description:

Use AI assistance to generate a Python program that:

• Uses user-defined functions to:

Calculate Energy Charges

Calculate Fixed Charges

• Returns calculated values

• Includes meaningful comments

Expected Output:

• Function-based Python program

• Correct EC and FC values

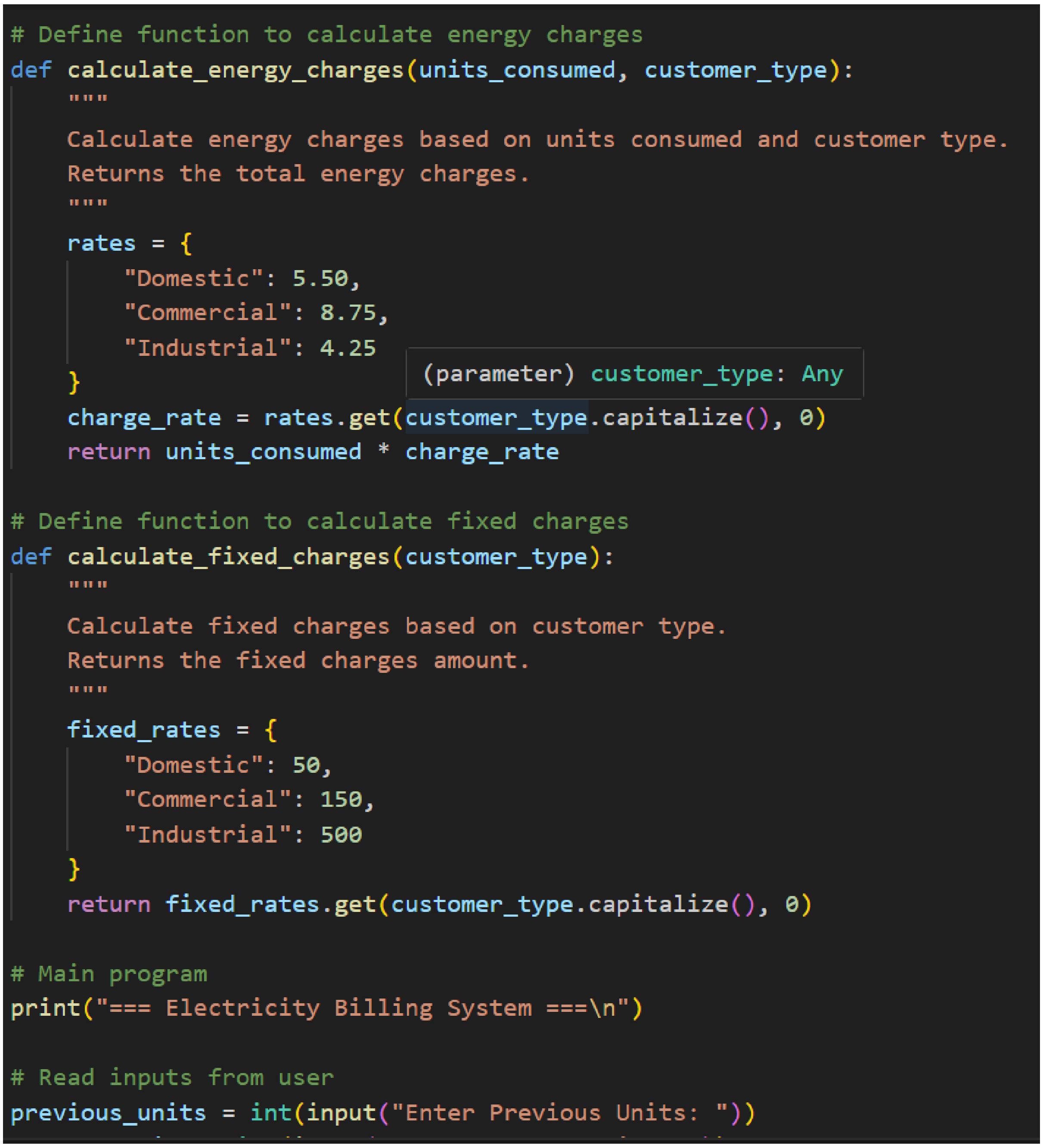
• Screenshots of AI-assisted function generation

• Test cases with outputs

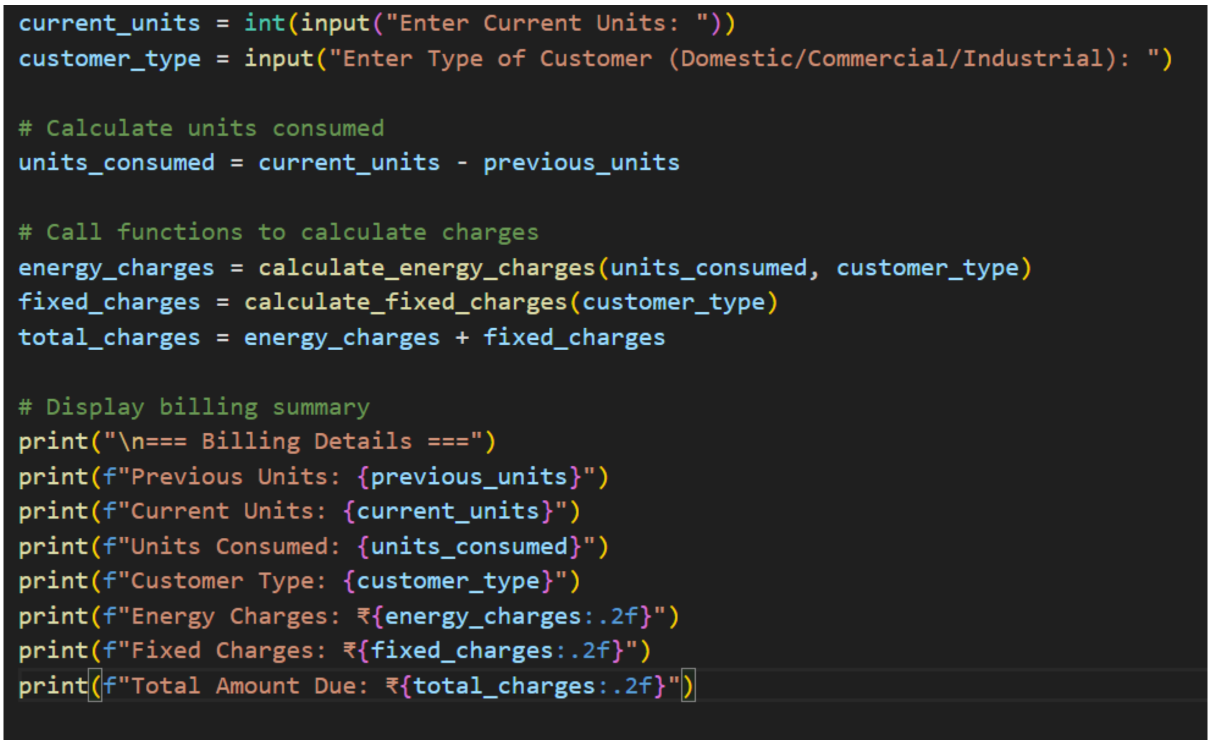
Prompt:

Generate a Python electricity billing program using modular design. Create user-defined functions to calculate Energy Charges and Fixed Charges based on units consumed and customer type. Each function should return the calculated value and include meaningful comments. In the main program, read inputs, call the functions, and display the results.

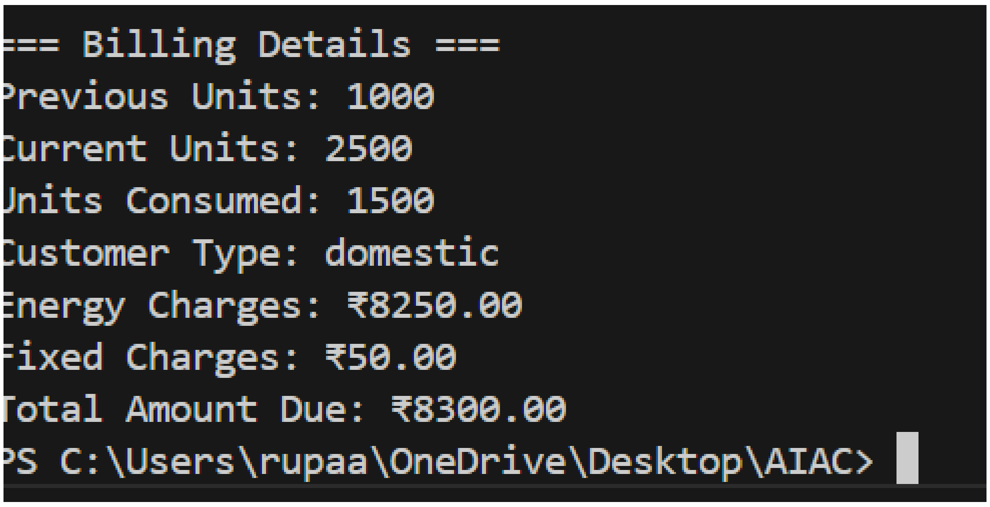
Code:



```
# Define function to calculate energy charges
def calculate_energy_charges(units_consumed, customer_type):
    """
    Calculate energy charges based on units consumed and customer type.
    Returns the total energy charges.
    """
    rates = {
        "Domestic": 5.50,
        "Commercial": 8.75,
        "Industrial": 4.25          (parameter) customer_type: Any
    }
    charge_rate = rates.get(customer_type.capitalize(), 0)
    return units_consumed * charge_rate

# Define function to calculate fixed charges
def calculate_fixed_charges(customer_type):
    """
    Calculate fixed charges based on customer type.
    Returns the fixed charges amount.
    """
    fixed_rates = {
        "Domestic": 50,
        "Commercial": 150,
        "Industrial": 500
    }
    return fixed_rates.get(customer_type.capitalize(), 0)

# Main program
print("=== Electricity Billing System ===\n")

# Read inputs from user
previous_units = int(input("Enter Previous Units: "))
```



```python
current_units = int(input("Enter Current Units: "))
customer_type = input("Enter Type of Customer (Domestic/Commercial/Industrial): ")

# Calculate units consumed
units_consumed = current_units - previous_units

# Call functions to calculate charges
energy_charges = calculate_energy_charges(units_consumed, customer_type)
fixed_charges = calculate_fixed_charges(customer_type)
total_charges = energy_charges + fixed_charges

# Display billing summary
print("\n=== Billing Details ===")
print(f"Previous Units: {previous_units}")
print(f"Current Units: {current_units}")
print(f"Units Consumed: {units_consumed}")
print(f"Customer Type: {customer_type}")
print(f"Energy Charges: ₹{energy_charges:.2f}")
print(f"Fixed Charges: ₹{fixed_charges:.2f}")
print(f"Total Amount Due: ₹{total_charges:.2f}")
```



```
== Billing Details ===
revious Units: 1000
urrent Units: 2500
nits Consumed: 1500
ustomer Type: domestic
nergy Charges: ₹8250.00
ixed Charges: ₹50.00
otal Amount Due: ₹8300.00
S C:\Users\rupaa\OneDrive\Desktop\AIAC>
```

Question 4:

Task 4: Calculation of Additional Charges

Scenario:

Electricity bills include multiple additional charges.

Task Description:

Extend the program to calculate:

• FC – Fixed Charges

• CC – Customer Charges

• ED – Electricity Duty (percentage of EC)

Use AI prompts like:

• “Add electricity duty calculation”

• “Improve billing accuracy”

Expected Output

• Individual charge values printed

• Correct duty calculation

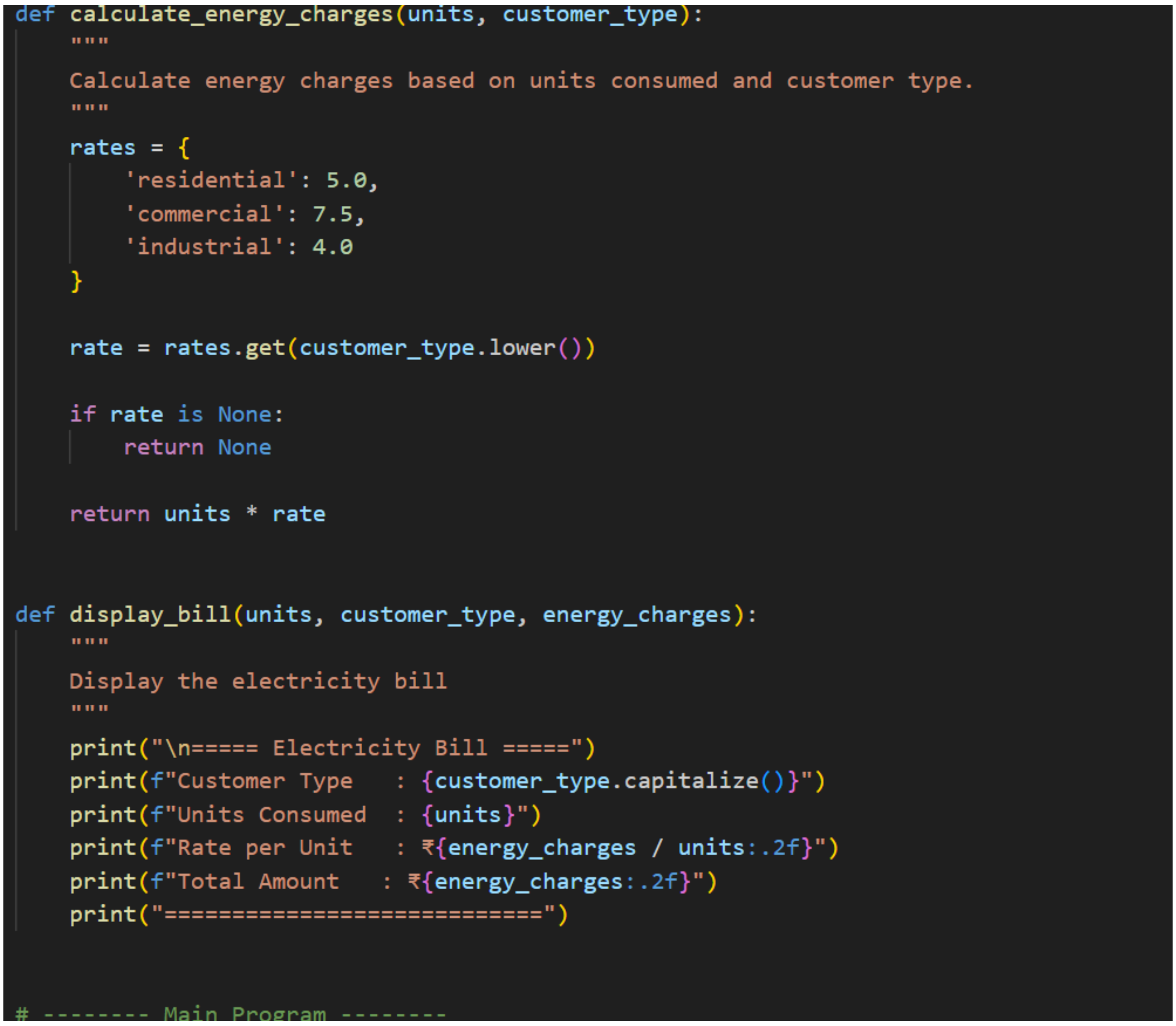
• Well-structured output

• Verified intermediate results

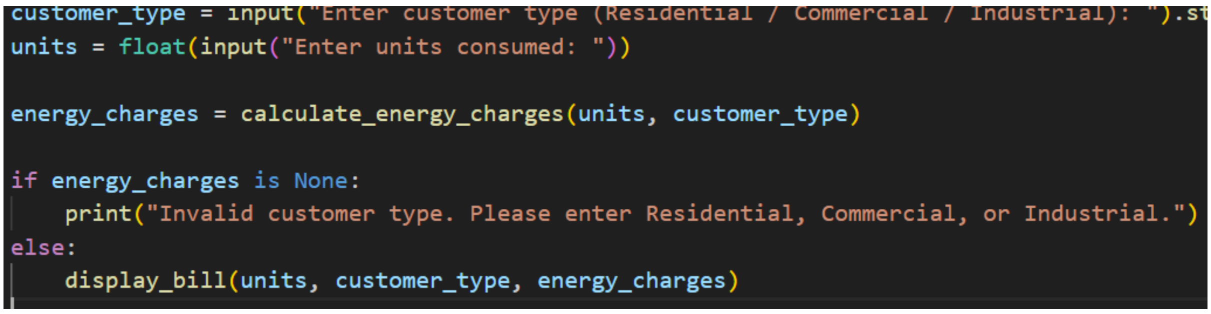
Prompt:

Extend the existing Python electricity billing program to calculate additional charges, including Fixed Charges (FC), Customer Charges (CC), and Electricity Duty (ED as a percentage of Energy Charges). Print each charge separately along with intermediate calculation results. Ensure accurate duty calculation, improved billing accuracy, and well-structured, readable output

Code:

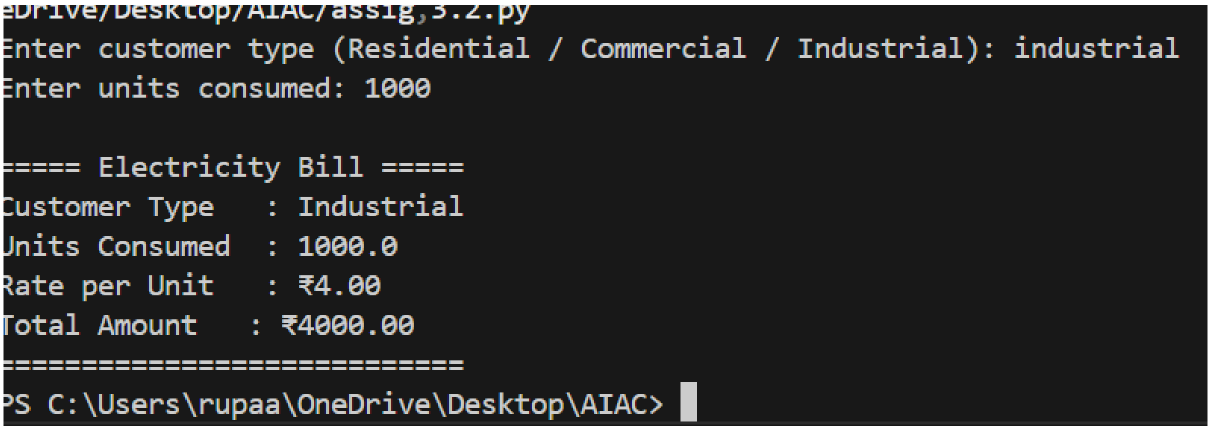


```python
def calculate_energy_charges(units, customer_type):
    """
    Calculate energy charges based on units consumed and customer type.
    """
    rates = {
        'residential': 5.0,
        'commercial': 7.5,
        'industrial': 4.0
    }

    rate = rates.get(customer_type.lower())

    if rate is None:
        return None

    return units * rate


def display_bill(units, customer_type, energy_charges):
    """
    Display the electricity bill
    """
    print("\n===== Electricity Bill =====")
    print(f"Customer Type    : {customer_type.capitalize()}")
    print(f"Units Consumed   : {units}")
    print(f"Rate per Unit    : ₹{energy_charges / units:.2f}")
    print(f"Total Amount    : ₹{energy_charges:.2f}")
    print("============================")


# -------- Main Program --------
```



```python
customer_type = input("Enter customer type (Residential / Commercial / Industrial): ").st
units = float(input("Enter units consumed: "))

energy_charges = calculate_energy_charges(units, customer_type)

if energy_charges is None:
    print("Invalid customer type. Please enter Residential, Commercial, or Industrial.")
else:
    display_bill(units, customer_type, energy_charges)
```

Output:



```
eDrive/Desktop/AIAC/assig,3.2.py
Enter customer type (Residential / Commercial / Industrial): industrial
Enter units consumed: 1000

===== Electricity Bill =====
Customer Type    : Industrial
Units Consumed   : 1000.0
Rate per Unit    : ₹4.00
Total Amount    : ₹4000.00
============================
PS C:\Users\rupaa\OneDrive\Desktop\AIAC>
```

Explanation:

In this task, the electricity billing program is extended to calculate additional charges such as Fixed Charges, Customer Charges, and Electricity Duty, which is computed as a percentage of Energy Charges. AI assistance helps improve billing accuracy and output structure. The program prints each charge separately and verifies intermediate results, ensuring correctness and clarity in the bill calculation.

Question 5:

Task 5: Final Bill Generation and Output Analysis

Scenario:

The final electricity bill must present all values clearly.

Task Description:

Develop the final Python application to:

• Calculate total bill:

• Total Bill = EC + FC + CC + ED

• Display:

Energy Charges (EC)

Fixed Charges (FC)

Customer Charges (CC)

Electricity Duty (ED)

Total Bill Amount

• Analyze the program based on:

Accuracy

Readability

Real-world applicability

Expected Output

• Complete electricity bill output

• Neatly formatted display

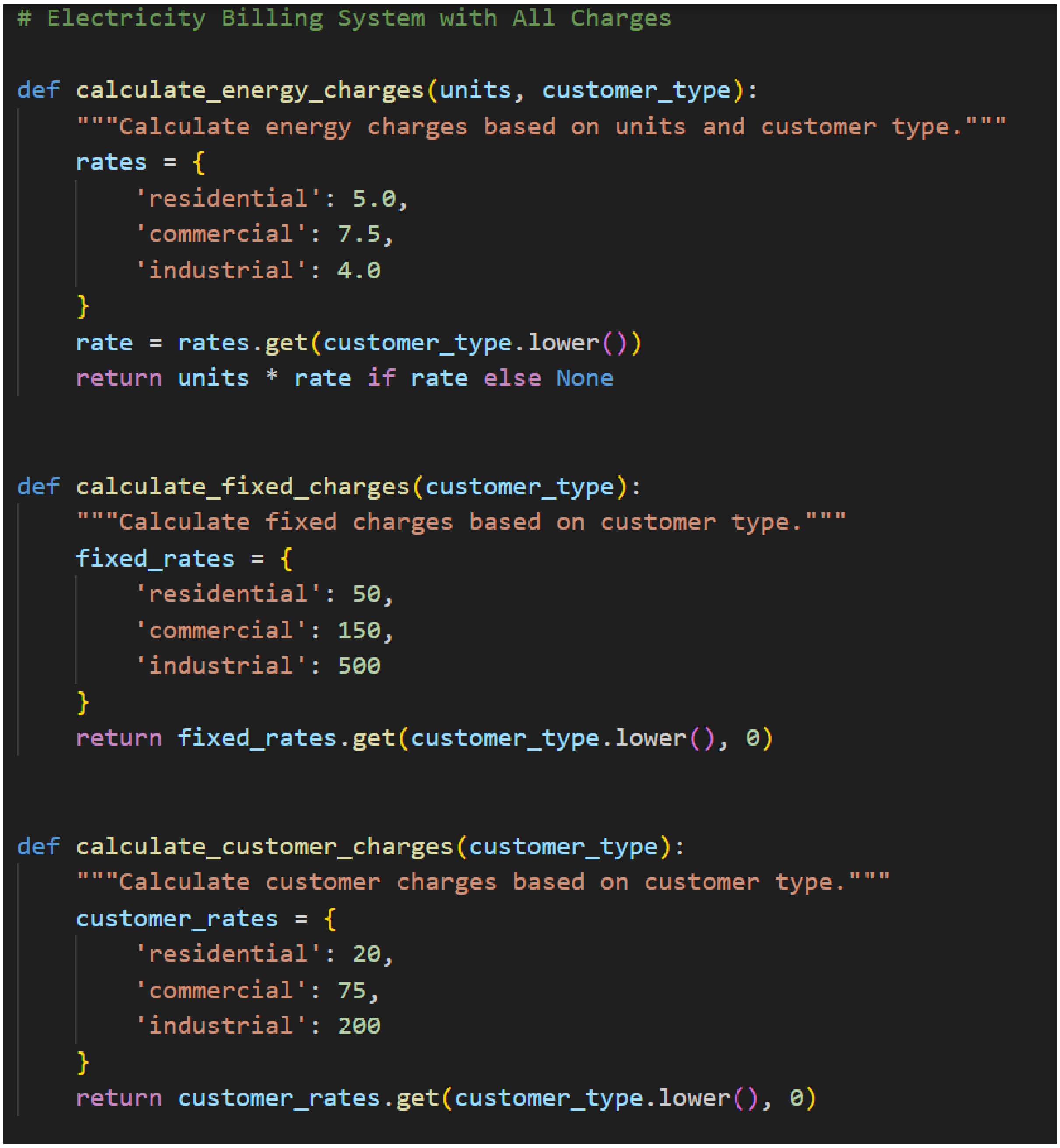
• Sample input/output

• Short analysis paragraph

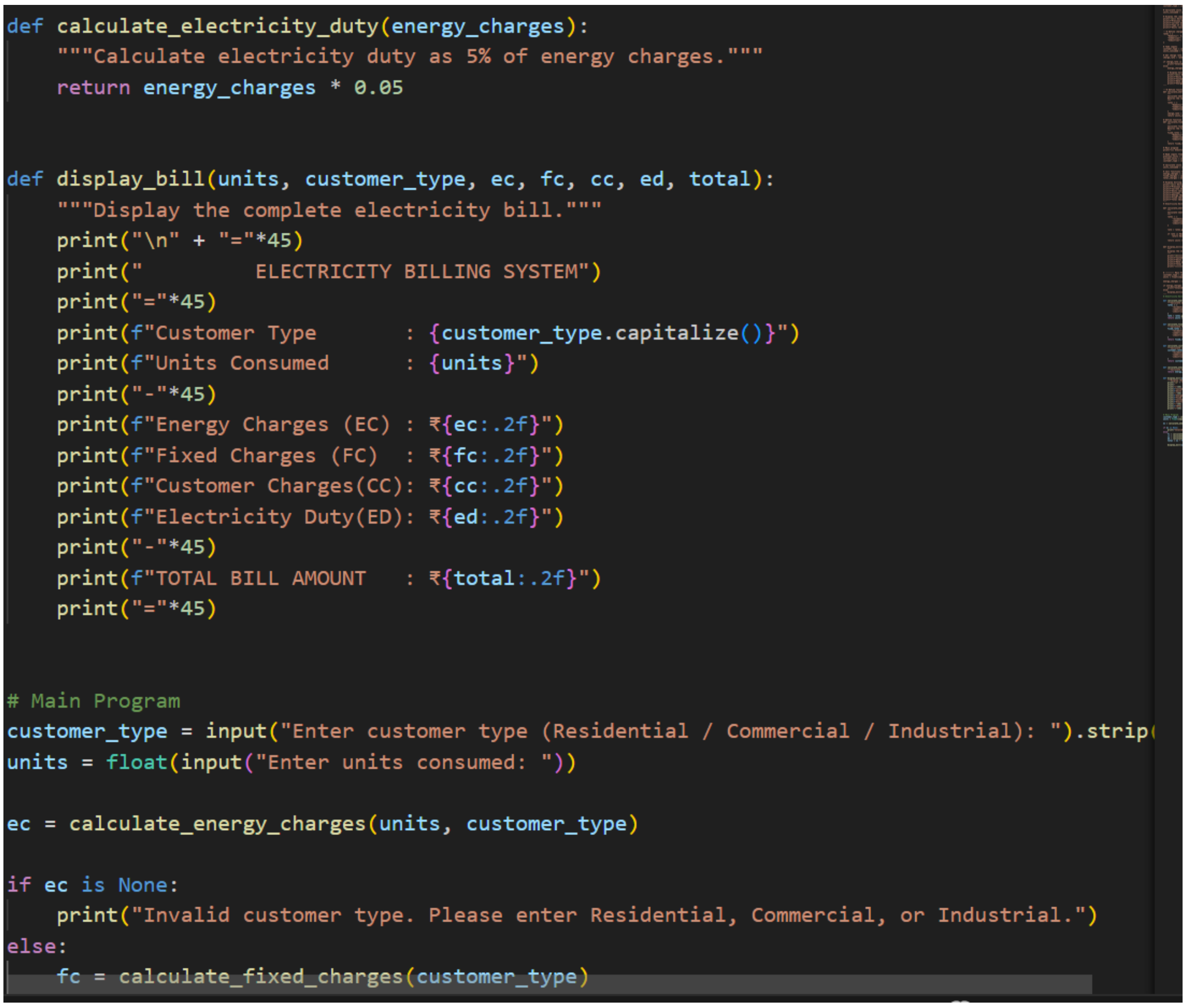
Prompt:

Develop the final Python electricity billing application that calculates Energy Charges, Fixed Charges, Customer Charges, and Electricity Duty, then computes the Total Bill as EC + FC + CC + ED. Display each charge clearly along with the total bill amount in a neatly formatted output.

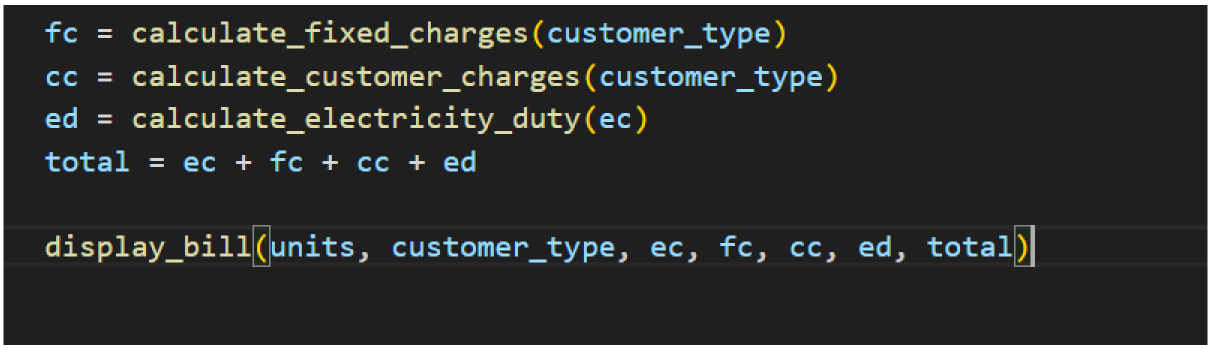
Code:



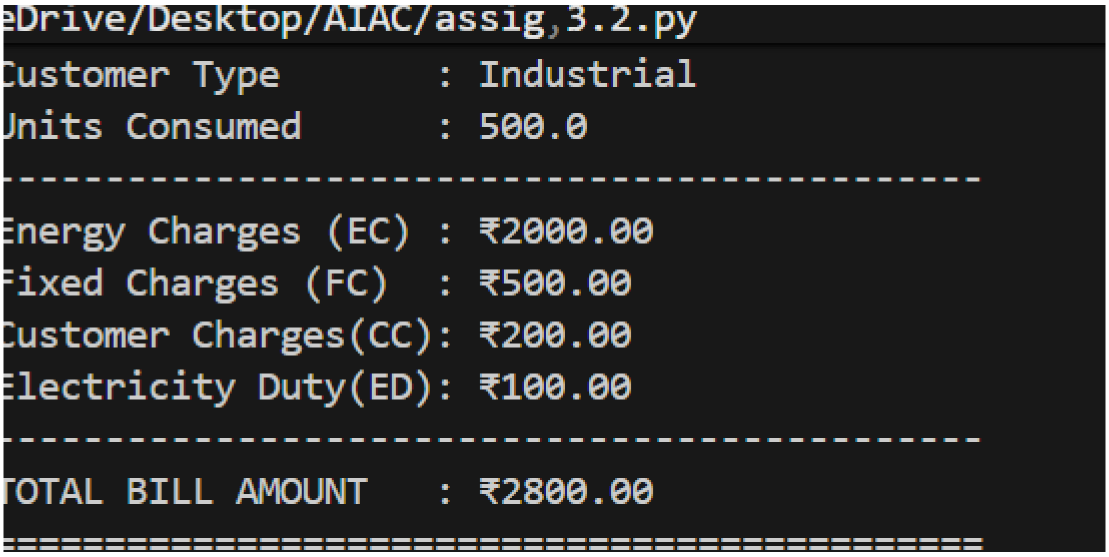
```python
# Electricity Billing System with All Charges

def calculate_energy_charges(units, customer_type):
    """Calculate energy charges based on units and customer type."""
    rates = {
        'residential': 5.0,
        'commercial': 7.5,
        'industrial': 4.0
    }
    rate = rates.get(customer_type.lower())
    return units * rate if rate else None


def calculate_fixed_charges(customer_type):
    """Calculate fixed charges based on customer type."""
    fixed_rates = {
        'residential': 50,
        'commercial': 150,
        'industrial': 500
    }
    return fixed_rates.get(customer_type.lower(), 0)


def calculate_customer_charges(customer_type):
    """Calculate customer charges based on customer type."""
    customer_rates = {
        'residential': 20,
        'commercial': 75,
        'industrial': 200
    }
    return customer_rates.get(customer_type.lower(), 0)
```



```
def calculate_electricity_duty(energy_charges):
    """Calculate electricity duty as 5% of energy charges."""
    return energy_charges * 0.05


def display_bill(units, customer_type, ec, fc, cc, ed, total):
    """Display the complete electricity bill."""
    print("\n" + "="*45)
    print("          ELECTRICITY BILLING SYSTEM")
    print("="*45)
    print(f"Customer Type       : {customer_type.capitalize()}")
    print(f"Units Consumed      : {units}")
    print("-"*45)
    print(f"Energy Charges (EC) : ₹{ec:.2f}")
    print(f"Fixed Charges (FC)  : ₹{fc:.2f}")
    print(f"Customer Charges(CC): ₹{cc:.2f}")
    print(f"Electricity Duty(ED): ₹{ed:.2f}")
    print("-"*45)
    print(f"TOTAL BILL AMOUNT   : ₹{total:.2f}")
    print("="*45)


# Main Program
customer_type = input("Enter customer type (Residential / Commercial / Industrial): ").strip(
units = float(input("Enter units consumed: "))

ec = calculate_energy_charges(units, customer_type)

if ec is None:
    print("Invalid customer type. Please enter Residential, Commercial, or Industrial.")
else:
    fc = calculate_fixed_charges(customer_type)
```



```
    fc = calculate_fixed_charges(customer_type)
    cc = calculate_customer_charges(customer_type)
    ed = calculate_electricity_duty(ec)
    total = ec + fc + cc + ed

    display_bill(units, customer_type, ec, fc, cc, ed, total)
```



```
eDrive/Desktop/AIAC/assig,3.2.py
Customer Type        : Industrial
Units Consumed       : 500.0
--------------------------------------------
Energy Charges (EC) : ₹2000.00
Fixed Charges (FC)  : ₹500.00
Customer Charges(CC): ₹200.00
Electricity Duty(ED): ₹100.00
--------------------------------------------
TOTAL BILL AMOUNT   : ₹2800.00
============================================
```

Explanation:

The program accurately calculates all components of the electricity bill, including Energy Charges, Fixed Charges, Customer Charges, and Electricity Duty, and correctly computes the total bill amount. The code is readable due to clear variable names, structured calculations, and formatted output. This billing logic closely matches real-world electricity billing systems, making the program practical and easy to extend for different tariffs or customer types.