AI Assisted Coding

ASSIGNMENT 3.3

Name: M.JAYANTH
HT No: 2303A52485
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:

```
# Electricity Billing System

print("=" * 50)
print("ELECTRICITY BILLING SYSTEM")
print("=" * 50)

# Input previous and current units
pu = int(input("Enter Previous Units (PU): "))
cu = int(input("Enter Current Units (CU): "))
customer_type = input("Enter Customer Type (Domestic/Commercial/Industrial): ").lower()

# Calculate units consumed
units_consumed = cu - pu

# Determine rate based on customer type
if customer_type == "domestic":
    rate_per_unit = 5.0
elif customer_type == "commercial":
    rate_per_unit = 7.5
elif customer_type == "industrial":
    rate_per_unit = 4.0
else:
    rate_per_unit = 5.0
    print("Invalid customer type. Using default rate.")
```

```python
# Calculate bill amount
bill_amount = units_consumed * rate_per_unit

# Apply surcharge if applicable
surcharge = 0
if units_consumed > 500:
    surcharge = bill_amount * 0.10

total_amount = bill_amount + surcharge

# Display billing details
print("\n" + "=" * 50)
print("BILLING DETAILS")
print("-" * 50)
print(f"Previous Units (PU)     : {pu}")
print(f"Current Units (CU)      : {cu}")
print(f"Units Consumed          : {units_consumed}")
print(f"Customer Type           : {customer_type.capitalize()}")
print(f"Rate per Unit           : ₹ {rate_per_unit}")
print(f"Bill Amount             : ₹ {bill_amount:.2f}")
print(f"Surcharge (10%)         : ₹ {surcharge:.2f}")
print("-" * 50)
print(f"Total Amount Due        : ₹ {total_amount:.2f}")
print("=" * 50)
```

Output(O/P)

```
PS D:\AI - AC\AIAC2485> ^C
PS D:\AI - AC\AIAC2485>
PS D:\AI - AC\AIAC2485>  d:; cd 'd:\AI - AC\AIAC2485'; & 'c:\Users\mummi\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\mum
extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '63228' '--' 'd:\AI - AC\AIAC2485\DAY3.3.py'
==================================================
ELECTRICITY BILLING SYSTEM
==================================================
Enter Previous Units (PU): 123
Enter Current Units (CU): 3444
Enter Customer Type (Domestic/Commercial/Industrial): INDUSTRIAL

==================================================
BILLING DETAILS
==================================================
Previous Units (PU)     : 123
Current Units (CU)      : 3444
Units Consumed          : 3321
Customer Type           : Industrial
Rate per Unit           : ₹ 4.0
Bill Amount             : ₹ 13284.00
Surcharge (10%)         : ₹ 1328.40
--------------------------------------------------
Total Amount Due        : ₹ 14612.40
==================================================
PS D:\AI - AC\AIAC2485>
```

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:

Write a Python program for an electricity billing system that reads Previous Units (PU), Current Units (CU), and Customer Type (Domestic, Commercial, or Industrial), calculates the units consumed by subtracting PU from CU, and then determines the Energy Charges (EC) using conditional statements where Domestic customers are charged Rs. 5 per unit, Commercial customers Rs. 8 per unit, and Industrial customers Rs. 10 per unit; the program should display all details including PU, CU, Customer Type, Units Consumed, and EC clearly, with the logic implemented directly in the main program without functions, and the code should be optimized for readability and simplicity.

Code:

```
# Electricity Billing System

print("=" * 50)
print("ELECTRICITY BILLING SYSTEM")
print("=" * 50)

# Input previous and current units
pu = int(input("Enter Previous Units (PU): "))
cu = int(input("Enter Current Units (CU): "))
customer_type = input("Enter Customer Type (Domestic/Commercial/Industrial): ").lower()

# Calculate units consumed
units_consumed = cu - pu

# Determine rate based on customer type
if customer_type == "domestic":
    rate_per_unit = 5
elif customer_type == "commercial":
    rate_per_unit = 8
elif customer_type == "industrial":
    rate_per_unit = 10
else:
    rate_per_unit = 5
    print("Invalid customer type. Using default rate (Domestic).")

# Calculate energy charges
energy_charges = units_consumed * rate_per_unit

# Display billing details
print("\n" + "=" * 50)
print("BILLING DETAILS")
print("=" * 50)
print(f"Previous Units (PU)     : {pu}")
print(f"Current Units (CU)      : {cu}")
print(f"Units Consumed          : {units_consumed}")
print(f"Customer Type           : {customer_type.capitalize()}")
print(f"Rate per Unit           : ₹ {rate_per_unit}")
print(f"Energy Charges (EC)     : ₹ {energy_charges:.2f}")
print("=" * 50)
```

Output:

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:

Write a Python program for an electricity billing system that uses user-defined functions to make the billing logic reusable. The program should read Previous Units (PU), Current Units (CU), and Customer Type (Domestic, Commercial, or Industrial), calculate the units consumed by subtracting PU from CU, and then use one function to calculate Energy Charges (EC) based on customer type with rates of Rs. 5 per unit for Domestic, Rs. 8 per unit for Commercial, and Rs. 10 per unit for Industrial, and another function to calculate Fixed Charges (FC) with values of Rs. 50 for Domestic, Rs. 100 for Commercial, and Rs. 200 for Industrial. Both functions should return their calculated values, and the program should display PU, CU, Customer Type, Units Consumed, EC, FC, and the Total Bill Amount clearly, with meaningful comments included for readability.

Code:

```
def calculate_energy_charges(units, customer_type):
    """Calculate energy charges based on customer type and units consumed"""
    rates = {
        "domestic": 5,
        "commercial": 8,
        "industrial": 10
    }
    rate = rates.get(customer_type.lower(), 5)
    return units * rate

def calculate_fixed_charges(customer_type):
    """Calculate fixed charges based on customer type"""
    charges = {
        "domestic": 50,
        "commercial": 100,
        "industrial": 200
    }
    return charges.get(customer_type.lower(), 50)

# Main program
print("=" * 50)
print("ELECTRICITY BILLING SYSTEM")
print("=" * 50)

# Input from user
pu = int(input("Enter Previous Units (PU): "))
cu = int(input("Enter Current Units (CU): "))
customer_type = input("Enter Customer Type (Domestic/Commercial/Industrial): ")

# Calculate units consumed
units_consumed = cu - pu

# Calculate charges using functions
energy_charges = calculate_energy_charges(units_consumed, customer_type)
fixed_charges = calculate_fixed_charges(customer_type)

# Calculate total bill
total_bill = energy_charges + fixed_charges

# Display billing details
print("\n" + "=" * 50)
print("BILLING DETAILS")
print("=" * 50)
print(f"Previous Units (PU)     : {pu}")
print(f"Current Units (CU)      : {cu}")
print(f"Units Consumed          : {units_consumed}")
print(f"Customer Type           : {customer_type.capitalize()}")
print(f"Energy Charges (EC)     : ₹ {energy_charges:.2f}")
print(f"Fixed Charges (FC)      : ₹ {fixed_charges:.2f}")
print("-" * 50)
print(f"Total Bill Amount       : ₹ {total_bill:.2f}")
print("=" * 50)
```

Output:

Explanation:

This task applies modular design by using user-defined functions to calculate Energy Charges and Fixed Charges. Each function performs a specific calculation and returns the result, making the billing logic reusable for multiple consumers. Meaningful comments improve code clarity and understanding. The main program reads inputs, calls the functions, and displays correct EC and FC values, verified using a test case.

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:

Write a Python program for an electricity billing system that extends previous logic to include additional charges. The program should read Previous Units (PU), Current Units (CU), and Customer Type (Domestic, Commercial, or Industrial), calculate the units consumed, and then use functions to calculate Energy Charges (EC), Fixed Charges (FC), Customer Charges (CC), and Electricity Duty (ED) as a percentage of EC (for example, 10%). Each function should return its calculated value, and the program should display PU, CU, Customer Type, Units Consumed, EC, FC, CC, ED, and the Total Bill Amount clearly. The output must be well-structured, include meaningful comments for readability, and verify intermediate results for accuracy.

Code:

```
def calculate_energy_charges(units, customer_type):
    """Calculate energy charges based on customer type and units consumed."""
    rates = {
        "domestic": 5,
        "commercial": 8,
        "industrial": 10
    }
    rate = rates.get(customer_type.lower(), 5)
    return units * rate

def calculate_fixed_charges(customer_type):
    """Calculate fixed charges based on customer type."""
    charges = {
        "domestic": 50,
        "commercial": 100,
        "industrial": 200
    }
    return charges.get(customer_type.lower(), 50)

def calculate_customer_charges(customer_type):
    """Calculate customer charges based on customer type."""
    charges = {
        "domestic": 20,
        "commercial": 30,
        "industrial": 50
    }
    return charges.get(customer_type.lower(), 20)

def calculate_electricity_duty(energy_charges):
    """Calculate electricity duty as a percentage of energy charges."""
    return energy_charges * 0.10  # 10% of energy charges
```

```python
# Main program
print("=" * 50)
print("ELECTRICITY BILLING SYSTEM")
print("=" * 50)

# Input from user
pu = int(input("Enter Previous Units (PU): "))
cu = int(input("Enter Current Units (CU): "))
customer_type = input("Enter Customer Type (Domestic/Commercial/Industrial): ")

# Calculate units consumed
units_consumed = cu - pu

# Calculate charges using functions
energy_charges = calculate_energy_charges(units_consumed, customer_type)
fixed_charges = calculate_fixed_charges(customer_type)
customer_charges = calculate_customer_charges(customer_type)
electricity_duty = calculate_electricity_duty(energy_charges)

# Calculate total bill
total_bill = energy_charges + fixed_charges + customer_charges + electricity_duty

# Display billing details
print("\n" + "=" * 50)
print("BILLING DETAILS")
print("=" * 50)
print(f"Previous Units (PU)     : {pu}")
print(f"Current Units (CU)      : {cu}")
print(f"Units Consumed          : {units_consumed}")
print(f"Customer Type           : {customer_type.capitalize()}")
print(f"Energy Charges (EC)     : ₹ {energy_charges:.2f}")
print(f"Fixed Charges (FC)      : ₹ {fixed_charges:.2f}")
print(f"Customer Charges (CC)   : ₹ {customer_charges:.2f}")
print(f"Electricity Duty (ED)   : ₹ {electricity_duty:.2f}")
print("-" * 50)
print(f"Total Bill Amount       : ₹ {total_bill:.2f}")
print("=" * 50)
```

Output:

```
PS D:\AI - AC\AIAC2485> d:; cd 'd:\AI - AC\AIAC2485'; & 'c:\Users\mami\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\mami\.vscode\
extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '50553' '--' 'd:\AI - AC\AIAC2485\DAY3.3.py'
==================================================
ELECTRICITY BILLING SYSTEM
==================================================
Enter Previous Units (PU): 1200
Enter Current Units (CU): 1300
Enter Customer Type (Domestic/Commercial/Industrial): industrial

==================================================
BILLING DETAILS
==================================================
Previous Units (PU)     : 1200
Current Units (CU)      : 1300
Units Consumed          : 100
Customer Type           : Industrial
Energy Charges (EC)     : ₹ 1000.00
Fixed Charges (FC)      : ₹ 200.00
Customer Charges (CC)   : ₹ 50.00
Electricity Duty (ED)   : ₹ 100.00
--------------------------------------------------
Total Bill Amount       : ₹ 1350.00
==================================================
PS D:\AI - AC\AIAC2485>
```

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:
Write a Python program for an electricity billing system that generates the final bill output. The program should read Previous Units (PU), Current Units (CU), and Customer Type (Domestic, Commercial, or Industrial), calculate the units consumed, and then use functions to compute Energy Charges (EC), Fixed Charges (FC), Customer Charges (CC), and Electricity Duty (ED) as a percentage of EC. The program must calculate the Total Bill as EC + FC + CC + ED and display all values clearly, including PU, CU, Customer Type, Units Consumed, EC, FC, CC, ED, and the Total Bill Amount in a neatly formatted output. Include meaningful comments for readability, ensure accuracy of calculations, and structure the program so it reflects real-world applicability. Provide sample input/output and a short analysis paragraph discussing accuracy, readability, and practical use.

Code:

```
# Electricity Billing System - Final Bill Generation
# This program calculates Energy Charges (EC), Fixed Charges (FC),
# Customer Charges (CC), and Electricity Duty (ED), then generates
# the final bill output in a neatly formatted way.

# Function to calculate Energy Charges based on customer type
def calculate_energy_charges(units_consumed, customer_type):
    """
    Calculate Energy Charges (EC) based on customer type.
    Rates:
    - Domestic: Rs. 5 per unit
    - Commercial: Rs. 8 per unit
    - Industrial: Rs. 10 per unit
    """
    rates = {"domestic": 5, "commercial": 8, "industrial": 10}
    return units_consumed * rates.get(customer_type.lower(), 0)

# Function to calculate Fixed Charges
def calculate_fixed_charges(customer_type):
    """
    Calculate Fixed Charges (FC) based on customer type.
    Rates:
    - Domestic: Rs. 50
    - Commercial: Rs. 100
    - Industrial: Rs. 200
    """
    fixed_rates = {"domestic": 50, "commercial": 100, "industrial": 200}
    return fixed_rates.get(customer_type.lower(), 0)


# Function to calculate Fixed Charges
def calculate_fixed_charges(customer_type):
    """
    Calculate Fixed Charges (FC) based on customer type.
    Rates:
    - Domestic: Rs. 50
    - Commercial: Rs. 100
    - Industrial: Rs. 200
    """
    fixed_rates = {"domestic": 50, "commercial": 100, "industrial": 200}
    return fixed_rates.get(customer_type.lower(), 0)

# Function to calculate Customer Charges
def calculate_customer_charges(customer_type):
    """
    Calculate Customer Charges (CC) based on customer type.
    Rates:
    - Domestic: Rs. 30
    - Commercial: Rs. 60
    - Industrial: Rs. 120
    """
    cc_rates = {"domestic": 30, "commercial": 60, "industrial": 120}
    return cc_rates.get(customer_type.lower(), 0)

# Function to calculate Electricity Duty
def calculate_electricity_duty(ec):
    """
    Calculate Electricity Duty (ED) as a percentage of EC.
    Example: 10% of EC
    """
    duty_rate = 0.10  # 10%
    return ec * duty_rate
```

```python
# ---------------- Main Program ----------------
# Read consumer details
previous_units = int(input("Enter Previous Units (PU): "))
current_units = int(input("Enter Current Units (CU): "))
customer_type = input("Enter Type of Customer (Domestic/Commercial/Industrial): ")

# Calculate units consumed
units_consumed = current_units - previous_units

# Calculate charges using functions
ec = calculate_energy_charges(units_consumed, customer_type)
fc = calculate_fixed_charges(customer_type)
cc = calculate_customer_charges(customer_type)
ed = calculate_electricity_duty(ec)

# Calculate total bill
total_bill = ec + fc + cc + ed

# Display results in a neatly formatted output
print("\n--- Final Electricity Bill ---")
print(f"Previous Units (PU): {previous_units}")
print(f"Current Units (CU): {current_units}")
print(f"Customer Type: {customer_type.capitalize()}")
print(f"Units Consumed: {units_consumed}")
print(f"Energy Charges (EC): Rs. {ec}")
print(f"Fixed Charges (FC): Rs. {fc}")
print(f"Customer Charges (CC): Rs. {cc}")
print(f"Electricity Duty (ED): Rs. {ed:.2f}")
print(f"Total Bill Amount: Rs. {total_bill:.2f}")
```

Output:

```
PS D:\AI - AC\AIAC2485>  d:; cd 'd:\AI - AC\AIAC2485'; & 'c:\Users\mummi\AppData\Local\Programs\Python\Pyth
extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '60880' '--' 'd:\AI - AC\AI
Enter Previous Units (PU): 1000
Enter Current Units (CU): 1300
Enter Type of Customer (Domestic/Commercial/Industrial): industrial

--- Final Electricity Bill ---
Previous Units (PU): 1000
Current Units (CU): 1300
Customer Type: Industrial
Units Consumed: 300
Energy Charges (EC): Rs. 3000
Fixed Charges (FC): Rs. 200
Customer Charges (CC): Rs. 120
Electricity Duty (ED): Rs. 300.00
Total Bill Amount: Rs. 3620.00
PS D:\AI - AC\AIAC2485>
```

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.