

## AI Assisted Coding-Lab 3

Shivani Pambidi

2303A52172

Batch – 41

### Question 1:

**TASK 1** : AI-Generated Logic for Reading Consumer Details

**PROMPT:**Generate sample electricity billing input data for 10 customers.For each customer, include Customer ID, Previous Units, Current Units, and Customer Type (Domestic or Commercial).Make sure Current Units are greater than Previous Units.Display the data in a clear table or list format suitable for input to a Python electricity billing program.

**CODE:**

```
customers = [
    ("C001", "Domestic", 120, 180),
    ("C002", "Commercial", 200, 260),
    ("C003", "Domestic", 90, 140),
    ("C004", "Commercial", 300, 380),
    ("C005", "Domestic", 150, 210),
    ("C006", "Commercial", 400, 470),
    ("C007", "Domestic", 220, 290),
    ("C008", "Domestic", 180, 240),
    ("C009", "Commercial", 260, 330),
    ("C010", "Domestic", 100, 160)
]
print("===== TGNPDCL ELECTRICITY BILL =====\n")
customer_number = 1
for customer in customers:
    print("Customer", customer_number)
    customer_id = customer[0]
    customer_type = customer[1]
    previous_units = customer[2]
    current_units = customer[3]
    units_used = current_units - previous_units
    print("Customer ID :", customer_id)
    print("Customer Type :", customer_type)
    print("Previous Units:", previous_units)
    print("Current Units :", current_units)
```

```

current_units = customer[3]
units_used = current_units - previous_units
print("Customer ID :", customer_id)
print("Customer Type :", customer_type)
print("Previous Units:", previous_units)
print("Current Units :", current_units)
print("Units Used      :", units_used)
print("=====\\n")
customer_number += 1

```

## OUTPUT:

```

PS C:\Users\pambi\OneDrive\Documents\Desktop\AI-Assited-Coding> & C:\Users\pambi\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/pambi/OneDrive/Documents/Desktop/AI-Assited-Coding/Lab 3.py"
===== TGNPDCL ELECTRICITY BILL =====

Customer 1
Customer ID   : C001
Customer Type : Domestic
Previous Units: 120
Current Units : 180
Units Used    : 60
=====

Customer 2
Customer ID   : C002
Customer Type : Commercial
Previous Units: 200
Current Units : 260
Units Used    : 60
=====

Customer 3
Customer ID   : C003
Customer Type : Domestic
Previous Units: 90
Current Units : 140
Units Used    : 50
=====

Customer 4
Customer ID   : C004

```

## JUSTIFICATION:

Customer details are taken as input to identify each consumer. Previous and current units help track electricity usage. Units consumed are calculated using a simple formula. Correct input ensures accurate billing. This task forms the base for all further calculations.

## Question 2:

### TASK 2: Energy Charges Calculation Based on Units Consumed

#### PROMPT:

Write a Python program to calculate Energy Charges (EC) based on units consumed. Use conditional statements for Domestic, Commercial, and Industrial customers with different rates. Display customer ID, type, units consumed, and energy charges. Keep logic in the main program and optimize for readability.

#### CODE:

```
Lab 3.py > ... C:\Users\pambi\OneDrive\Documents\Desktop\AI-Assited-Coding\Lab 2.py
35 customers = [
36     ("C001", "Domestic", 120, 180),
37     ("C002", "Commercial", 200, 260),
38     ("C003", "Industrial", 90, 150),
39     ("C004", "Commercial", 300, 380),
40     ("C005", "Domestic", 150, 210),
41     ("C006", "Industrial", 400, 470),
42     ("C007", "Domestic", 220, 290),
43     ("C008", "Domestic", 180, 240),
44     ("C009", "Commercial", 260, 330),
45     ("C010", "Domestic", 100, 160)
46 ]
47 print("===== TGNPDCL ELECTRICITY BILL WITH ENERGY CHARGES =====\n")
48 customer_number = 1
49 for customer in customers:
50     customer_id = customer[0]
51     customer_type = customer[1]
52     previous_units = customer[2]
53     current_units = customer[3]
54     units_used = current_units - previous_units
55     if customer_type == "Domestic":
56         # Example rate: Rs. 5 per unit for Domestic
57         ec = units_used * 5
58     elif customer_type == "Commercial":
59         # Example rate: Rs. 8 per unit for Commercial
60         ec = units_used * 8
61     elif customer_type == "Industrial":
62         # Example rate: Rs. 10 per unit for Industrial
63         ec = units_used * 10
64     else:
65         ec = 0 # default if customer type is invalid
```

```

        ec = units_used * 10
    else:
        ec = 0 # default if customer type is invalid
    # Print bill details
    print("Customer", customer_number)
    print("Customer ID :", customer_id)
    print("Customer Type :", customer_type)
    print("Previous Units:", previous_units)
    print("Current Units :", current_units)
    print("Units Used      :", units_used)
    print("Energy Charges: Rs.", ec)
    print("=====\n")
    customer_number += 1

```

## OUTPUT:

```

PS C:\Users\pambi\OneDrive\Documents\Desktop\AI-Assited-Coding> ^C
PS C:\Users\pambi\OneDrive\Documents\Desktop\AI-Assited-Coding> & C:\Users\pambi\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/pambi/OneDrive/Documents/Desktop/AI-Assited-Coding/Lab 3.py"
===== TGNPDCL ELECTRICITY BILL WITH ENERGY CHARGES =====

Customer 1
Customer ID   : C001
Customer Type : Domestic
Previous Units: 120
Current Units : 180
Units Used    : 60
Energy Charges: Rs. 300
=====

Customer 2
Customer ID   : C002
Customer Type : Commercial
Previous Units: 200
Current Units : 260
Units Used    : 60
Energy Charges: Rs. 480
=====

Customer 3
Customer ID   : C003
Customer Type : Industrial
Previous Units: 90
Current Units : 150
Units Used    : 60
Energy Charges: Rs. 600
=====

Customer 4
Customer ID   : C004

```

## JUSTIFICATION:

This task calculates energy charges based on the number of units consumed. Different customer types are charged at different rates to ensure fair billing. Conditional statements are used to apply the correct tariff for each customer. This logic makes the program closer to real electricity billing systems. Energy charges form the major part of the total bill. This task helps understand how electricity costs are calculated in real life.

### Question 3:

#### TASK 3: Modular Design Using AI Assistance (Using Functions)

##### PROMPT:

Create a Python program using functions to calculate units consumed, energy charges, and fixed charges for Domestic, Commercial, and Industrial customers, then display customer details and total charges with comments.

##### CODE:

```
Lab 3.py > C:\Users\pambi\OneDrive\Documents\Desktop\AI-Assited-Coding\Lab 2.py
81 # Function to calculate units consumed
82 def calculate_units(current_units, previous_units):
83     """
84     Returns units consumed for a customer.
85     """
86     return current_units - previous_units
87 # Function to calculate energy charges based on customer type
88 def calculate_energy_charges(units, customer_type):
89     """
90     Returns energy charges (EC) based on customer type.
91     Example rates: Domestic=5/unit, Commercial=8/unit, Industrial=10/unit
92     """
93     if customer_type == "Domestic":
94         return units * 5
95     elif customer_type == "Commercial":
96         return units * 8
97     elif customer_type == "Industrial":
98         return units * 10
99     else:
100         return 0
101 # Function to calculate fixed charges based on customer type
102 def calculate_fixed_charges(customer_type):
103     """
104     Returns fixed charges (FC) based on customer type.
105     Example FC: Domestic=50, Commercial=100, Industrial=200
106     """
107     if customer_type == "Domestic":
108         return 50
109     elif customer_type == "Commercial":
110         return 100
111     elif customer_type == "Industrial":
112         return 200
113     else:
114         return 0
115 # Sample customer data: (Customer ID, Customer Type, Previous Units, Current Units)
116 customers = []
```



```

114         return 0
115     customers = [
116         ("C001", "Domestic", 120, 180),
117         ("C002", "Commercial", 200, 260),
118         ("C003", "Industrial", 90, 150),
119         ("C004", "Commercial", 300, 380),
120         ("C005", "Domestic", 150, 210),
121         ("C006", "Industrial", 400, 470),
122         ("C007", "Domestic", 220, 290),
123         ("C008", "Domestic", 180, 240),
124         ("C009", "Commercial", 260, 330),
125         ("C010", "Domestic", 100, 160)
126     ]
127     print("===== TGNPDCL FUNCTION-BASED ELECTRICITY BILL =====\n")
128     customer_number = 1
129     for customer in customers:
130         customer_id = customer[0]
131         customer_type = customer[1]
132         previous_units = customer[2]
133         current_units = customer[3]
134         units_used = calculate_units(current_units, previous_units)
135         ec = calculate_energy_charges(units_used, customer_type)
136         fc = calculate_fixed_charges(customer_type)
137         total = ec + fc
138         print(f"Customer {customer_number}")
139         print("Customer ID   :", customer_id)
140         print("Customer Type :", customer_type)
141         print("Previous Units:", previous_units)
142         print("Current Units  :", current_units)
143         print("Units Used    :", units_used)
144         print("Energy Charges: Rs.", ec)
145         print("Fixed Charges  : Rs.", fc)
146         print("Total Charges : Rs.", total)
147         print("===== \n")
148         customer_number += 1

```

## OUTPUT:

```

PS C:\Users\pambi\OneDrive\Documents\Desktop\AI-Assited-Coding> ^C
PS C:\Users\pambi\OneDrive\Documents\Desktop\AI-Assited-Coding> & C:\Users\pambi\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/pambi/OneDrive/Documents/Desktop/AI-Assited-Coding/Lab 3.py"
===== TGNPDCL FUNCTION-BASED ELECTRICITY BILL =====

Customer 1
Customer ID   : C001
Customer Type : Domestic
Previous Units: 120
Current Units : 180
Units Used    : 60
Energy Charges: Rs. 300
Fixed Charges  : Rs. 50
Total Charges : Rs. 350
=====

Customer 2
Customer ID   : C002
Customer Type : Commercial
Previous Units: 200
Current Units : 260
Units Used    : 60
Energy Charges: Rs. 480
Fixed Charges  : Rs. 100
Total Charges : Rs. 580
=====

Customer 3
Customer ID   : C003
Customer Type : Industrial
Previous Units: 90
Current Units : 150

```

## JUSTIFICATION:

In this task, functions are used to make the program modular and reusable. Separate functions are created to calculate energy charges and fixed charges. This improves code readability and makes debugging easier. Modular design reduces repetition and errors in the program. It also allows handling multiple customers efficiently. This approach makes future modifications simple and manageable.

## Question 4:

### TASK 4: Calculation of Additional Charges

#### PROMPT:

*Write a Python program using functions to calculate electricity bills with Energy Charges (EC), Fixed Charges (FC), Customer Charges (CC), and Electricity Duty (ED). Include units consumed formula, return values from functions, and display all charges and total in a clear format.*

#### CODE:

```
Lab 3.py > ...
153 def calculate_units(current_units, previous_units):
154     """Returns units consumed for a customer."""
155     return current_units - previous_units
156 def calculate_energy_charges(units, customer_type):
157     """Returns energy charges (EC) based on customer type."""
158     rates = {"Domestic": 5, "Commercial": 8, "Industrial": 10}
159     return units * rates.get(customer_type, 0)
160 def calculate_fixed_charges(customer_type):
161     """Returns fixed charges (FC) based on customer type."""
162     fc_rates = {"Domestic": 50, "Commercial": 100, "Industrial": 200}
163     return fc_rates.get(customer_type, 0)
164 def calculate_customer_charges():
165     """Returns flat customer charges (CC)."""
166     return 25 # as percentage of EC
167 def calculate_electricity_duty(ec):
168     """Returns electricity duty (ED) as 5% of energy charges."""
169     return ec * 0.05
170 customers = [
171     ("C001", "Domestic", 120, 180),
172     ("C002", "Commercial", 200, 260),
173     ("C003", "Industrial", 90, 150),
174     ("C004", "Commercial", 300, 380),
175     ("C005", "Domestic", 150, 210),
176     ("C006", "Industrial", 400, 470),
177     ("C007", "Domestic", 220, 290),
178     ("C008", "Domestic", 180, 240),
179     ("C009", "Commercial", 260, 330),
180     ("C010", "Domestic", 100, 160)
181 ]
182 print("===== TGNPDCL ELECTRICITY BILL WITH ADDITIONAL CHARGES =====\n")
183 customer_number = 1
```

```

        ("C010", "Domestic", 100, 160)
    ]
    print("===== TGNPDCL ELECTRICITY BILL WITH ADDITIONAL CHARGES =====\n")
    customer_number = 1
    for customer in customers:
        customer_id = customer[0]
        customer_type = customer[1]
        previous_units = customer[2]
        current_units = customer[3]
        units_used = calculate_units(current_units, previous_units)
        ec = calculate_energy_charges(units_used, customer_type)
        fc = calculate_fixed_charges(customer_type)
        cc = calculate_customer_charges()
        ed = calculate_electricity_duty(ec)
        total = ec + fc + cc + ed
        print(f"Customer {customer_number}")
        print("Customer ID   :", customer_id)
        print("Customer Type :", customer_type)
        print("Previous Units:", previous_units)
        print("Current Units :", current_units)
        print("Units Used    :", units_used)
        print("Energy Charges: Rs.", ec)
        print("Fixed Charges : Rs.", fc)
        print("Customer Charges: Rs.", cc)
        print("Electricity Duty: Rs.", round(ed,2))
        print("Total Charges : Rs.", round(total,2))
        print("=====\\n")
        customer_number += 1

```

## OUTPUT:

```

PS C:\Users\pambi\OneDrive\Documents\Desktop\AI-Assited-Coding> & C:\Users\pambi\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/pambi/OneDrive/Documen
ts/Desktop/AI-Assited-Coding/Lab 3.py"
===== TGNPDCL ELECTRICITY BILL WITH ADDITIONAL CHARGES =====

Customer 1
Customer ID   : C001
Customer Type : Domestic
Previous Units: 120
Current Units : 180
Units Used    : 60
Energy Charges: Rs. 300
Fixed Charges : Rs. 50
Customer Charges: Rs. 25
Electricity Duty: Rs. 15.0
Total Charges : Rs. 390.0
=====

Customer 2
Customer ID   : C002
Customer Type : Commercial
Previous Units: 200
Current Units : 260
Units Used    : 60
Energy Charges: Rs. 480
Fixed Charges : Rs. 100
Customer Charges: Rs. 25
Electricity Duty: Rs. 24.0
Total Charges : Rs. 629.0
=====

Customer 3
Customer ID   : C003
Customer Type : Industrial
Previous Units: 90
Current Units : 150
Units Used    : 60
Energy Charges: Rs. 600
Fixed Charges : Rs. 200
Customer Charges: Rs. 25

```



## JUSTIFICATION:

This task adds additional charges such as fixed charges, customer charges, and electricity duty to the bill. Electricity duty is calculated as a percentage of energy charges. Including these charges ensures accurate and complete billing. This task follows real electricity board billing rules. It increases the reliability of the bill generated. It also improves transparency by showing individual charges clearly.

## Question 5:

### TASK 5: Final Bill Generation and Output Analysis

#### PROMPT:

*Write a Python program using functions to generate the final electricity bill. Calculate Units Consumed, Energy Charges (EC), Fixed Charges (FC), Customer Charges (CC), Electricity Duty (ED), and Total Bill. Display all values neatly with customer details and provide clear, readable output.*

#### CODE:

```
def calculate_units(current_units, previous_units):
    """Returns units consumed for a customer."""
    return current_units - previous_units

def calculate_energy_charges(units, customer_type):
    """Returns energy charges (EC) based on customer type."""
    rates = {"Domestic": 5, "Commercial": 8, "Industrial": 10}
    return units * rates.get(customer_type, 0)

def calculate_fixed_charges(customer_type):
    """Returns fixed charges (FC) based on customer type."""
    fc_rates = {"Domestic": 50, "Commercial": 100, "Industrial": 200}
    return fc_rates.get(customer_type, 0)

def calculate_customer_charges():
    """Returns flat customer charges (CC)."""
    return 25

def calculate_electricity_duty(ec):
    """Returns electricity duty (ED) as 5% of EC."""
    return ec * 0.05

customers = [
    ("C001", "Domestic", 120, 180),
    ("C002", "Commercial", 200, 260),
    ("C003", "Industrial", 90, 150),
    ("C004", "Commercial", 300, 380),
    ("C005", "Domestic", 150, 210),
    ("C006", "Industrial", 400, 470),
    ("C007", "Domestic", 220, 290),
    ("C008", "Domestic", 180, 240),
    ("C009", "Commercial", 260, 330),
    ("C010", "Domestic", 100, 160)
]

print("===== TGNPDCL FINAL ELECTRICITY BILL =====\n")
customer_number = 1
for customer in customers:
```

```

for customer in customers:
    customer_id = customer[0]
    customer_type = customer[1]
    previous_units = customer[2]
    current_units = customer[3]
    units_used = calculate_units(current_units, previous_units)
    # Charges calculation
    ec = calculate_energy_charges(units_used, customer_type)
    fc = calculate_fixed_charges(customer_type)
    cc = calculate_customer_charges()
    ed = calculate_electricity_duty(ec)
    total_bill = ec + fc + cc + ed
    print(f"Customer {customer_number}")
    print("Customer ID      :", customer_id)
    print("Customer Type     :", customer_type)
    print("Previous Units    :", previous_units)
    print("Current Units     :", current_units)
    print("Units Consumed    :", units_used)
    print("Energy Charges   : Rs.", ec)
    print("Fixed Charges    : Rs.", fc)
    print("Customer Charges : Rs.", cc)
    print("Electricity Duty : Rs.", round(ed, 2))
    print("Total Bill Amount: Rs.", round(total_bill, 2))
    print("=====\n")
    customer_number += 1
print("Program Analysis:")
print("Accuracy: Charges are calculated using correct formulas for EC, FC, CC, and ED.")
print("Readability: Functions modularize the logic and comments improve clarity.")
print("Real-world applicability: Output format matches standard electricity bills and is scalable for multiple customers.")

```

## OUTPUT:

```

Problems  Output  Debug Console  Terminal  Ports
PS C:\Users\pambi\OneDrive\Documents\Desktop\AI-Assited-Coding> & C:\Users\pambi\AppData\Local\Programs\Python\Python313\python.exe C:\Users\pambi\OneDrive\Documents\Desktop\AI-Assited-Coding\Lab 3.py
===== TGNPCL FINAL ELECTRICITY BILL =====

Customer 1
Customer ID      : C001
Customer Type    : Domestic
Previous Units   : 120
Current Units    : 180
Units Consumed   : 60
Energy Charges   : Rs. 300
Fixed Charges    : Rs. 50
Customer Charges : Rs. 25
Electricity Duty : Rs. 15.0
Total Bill Amount: Rs. 390.0
=====

Customer 2
Customer ID      : C002
Customer Type    : Commercial
Previous Units   : 200
Current Units    : 260
Units Consumed   : 60
Energy Charges   : Rs. 480
Fixed Charges    : Rs. 100
Customer Charges : Rs. 25
Electricity Duty : Rs. 24.0
Total Bill Amount: Rs. 629.0
=====

Customer 3
Customer ID      : C003
Customer Type    : Industrial
Previous Units   : 90

```

```
Problems Output Debug Console Terminal Ports
Customer 9
Customer ID      : C009
Customer Type    : Commercial
Previous Units   : 260
Current Units    : 330
Units Consumed   : 70
Energy Charges   : Rs. 560
Fixed Charges    : Rs. 100
Customer Charges : Rs. 25
Electricity Duty : Rs. 28.0
Total Bill Amount: Rs. 713.0
=====

Customer 10
Customer ID      : C010
Customer Type    : Domestic
Previous Units   : 100
Current Units    : 160
Units Consumed   : 60
Energy Charges   : Rs. 300
Fixed Charges    : Rs. 50
Customer Charges : Rs. 25
Electricity Duty : Rs. 15.0
Total Bill Amount: Rs. 390.0
=====

Program Analysis:
Accuracy: Charges are calculated using correct formulas for EC, FC, CC, and ED.
Readability: Functions modularize the logic and comments improve clarity.
Real-world applicability: Output format matches standard electricity bills and is scalable for multiple customers.
PS C:\Users\pambi\OneDrive\Documents\Desktop\AI-Assited-Coding> |
```

## JUSTIFICATION:

In this task, all charges are added together to calculate the total bill amount. The output is displayed in a neat and clear format for easy understanding. Showing each charge separately helps customers know how the bill is formed. The program produces accurate and reliable billing results. This task completes the electricity billing application. It also helps in analyzing and verifying the correctness of calculations.