

2303A51679

Batch -23

Assinment-3.3

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

Prompt1 : Write a python program for an electricity billing system that reads previous units and current units and type of customer it should calculate units consumed. It should implements logic directly in main program and the energy charges depend on number of units consumed and customer type

```
# Task 1: Reading 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): ")
units_consumed = current_units - previous_units
print(f"Previous Units : {previous_units}")
print(f"Current Units : {current_units}")
print(f"Customer Type : {customer_type}")
print(f"Units Consumed : {units_consumed}")
```

```
lab-3"
Enter Previous Units (PU): 40
Enter Current Units (CU): 30
Enter Type of Customer (Domestic/Commercial/Industrial): commercial
Previous Units : 40
Current Units : 30
```

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

Prompt2: In above code energy charges depend on number of units consumed and customer type .now it should calculate energy charge on domestic commercial industrial consumers

```
# Task 2: Energy Charges Calculation
# Original version: direct conditional logic
if customer_type.lower() == "domestic":
    if units_consumed <= 100:
        ec = units_consumed * 1.5
    elif units_consumed <= 200:
        ec = 100 * 1.5 + (units_consumed - 100) * 2.5
    elif units_consumed <= 300:
        ec = 100 * 1.5 + 100 * 2.5 + (units_consumed - 200) * 4
    else:
        ec = 100 * 1.5 + 100 * 2.5 + 100 * 4 + (units_consumed - 300) * 6
elif customer_type.lower() == "commercial":
    ec = units_consumed * 8
elif customer_type.lower() == "industrial":
    ec = units_consumed * 10
else:
    ec = 0
    print("Invalid customer type entered!")
print(f"Energy Charges (EC): ₹{ec:.2f}")
```

```
Enter Type of Customer (Domestic/Commercial/Industrial): commercial
Previous Units : 40
Current Units  : 30
Customer Type  : commercial
Units Consumed : -10
```

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

Prompt3: In above code billing logic must be reusable for multiple consumers it should use defined functions to calculate energy charges, fixed charges and it return calculated value.

```

# Task 3: Modular Design (Functions)
def calculate_energy_charges(units, ctype):
    """Calculate EC based on customer type and units consumed."""
    if ctype.lower() == "domestic":
        if units <= 100:
            return units * 1.5
        elif units <= 200:
            return 100 * 1.5 + (units - 100) * 2.5
        elif units <= 300:
            return 100 * 1.5 + 100 * 2.5 + (units - 200) * 4
        else:
            return 100 * 1.5 + 100 * 2.5 + 100 * 4 + (units - 300) * 6
    elif ctype.lower() == "commercial":
        return units * 8
    elif ctype.lower() == "industrial":
        return units * 10
    else:
        return 0
def calculate_fixed_charges(ctype):
    """Return fixed charges depending on customer type."""
    if ctype.lower() == "domestic":
        return 50
    elif ctype.lower() == "commercial":
        return 100
    elif ctype.lower() == "industrial":
        return 150
    else:
        return 0
# Test modular functions
ec = calculate_energy_charges(units_consumed, customer_type)
fc = calculate_fixed_charges(customer_type)
print(f"Energy Charges (EC): ₹{ec:.2f}")
print(f"Fixed Charges (FC): ₹{fc:.2f}")

```

```

Energy Charges (EC): ₹-80.00
Energy Charges (EC): ₹-80.00
Fixed Charges (FC): ₹100.00
Fixed Charges (FC): ₹100.00

```

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: In above code it should include multiple additional charges liked fixedcharge , customercharge , electricityduty.

```
# Task 4: Additional Charges
# Customer Charges (CC) - flat ₹30
cc = 30
# Electricity Duty (ED) - 5% of EC
ed = 0.05 * ec

print(f"Fixed Charges (FC): ₹{fc:.2f}")
print(f"Customer Charges (CC): ₹{cc:.2f}")
print(f"Electricity Duty (ED): ₹{ed:.2f}")
```

```
Customer Charges (CC): ₹30.00
Electricity Duty (ED): ₹-4.00
```

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

- o Readability
- o Real-world applicability

Expected Output

- Complete electricity bill output
- Neatly formatted display
- Sample input/output
- Short analysis paragraph

Prompt: #In above code it should print energy charges , fixed charges , customer charge ,electricity duty ,total bill amount.

```
# Task 5: Final Bill Generation
total_bill = ec + fc + cc + ed
print(" Electricity Bill")
print(f"Customer Type : {customer_type}")
print(f"Units Consumed : {units_consumed}")
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(f"Total Bill Amount : ₹{total_bill:.2f}")
```

```
TGNPDCL Electricity Bill
Customer Charges (CC): ₹30.00
Electricity Duty (ED): ₹-4.00
TGNPDCL Electricity Bill
Customer Type : commercial
Units Consumed : -10
TGNPDCL Electricity Bill
Customer Type : commercial
Units Consumed : -10
Customer Type : commercial
Units Consumed : -10
Units Consumed : -10
Energy Charges (EC): ₹-80.00
Fixed Charges (FC): ₹100.00
Energy Charges (EC): ₹-80.00
Fixed Charges (FC): ₹100.00
Customer Charges(CC): ₹30.00
Fixed Charges (FC): ₹100.00
Customer Charges(CC): ₹30.00
Customer Charges(CC): ₹30.00
Electricity Duty(ED): ₹-4.00
Total Bill Amount : ₹46.00
```