**AI Assistant Coding**

**Assignment - 3.3**

**Salupala Naveen Yadav**

**2303A52230**

**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:**

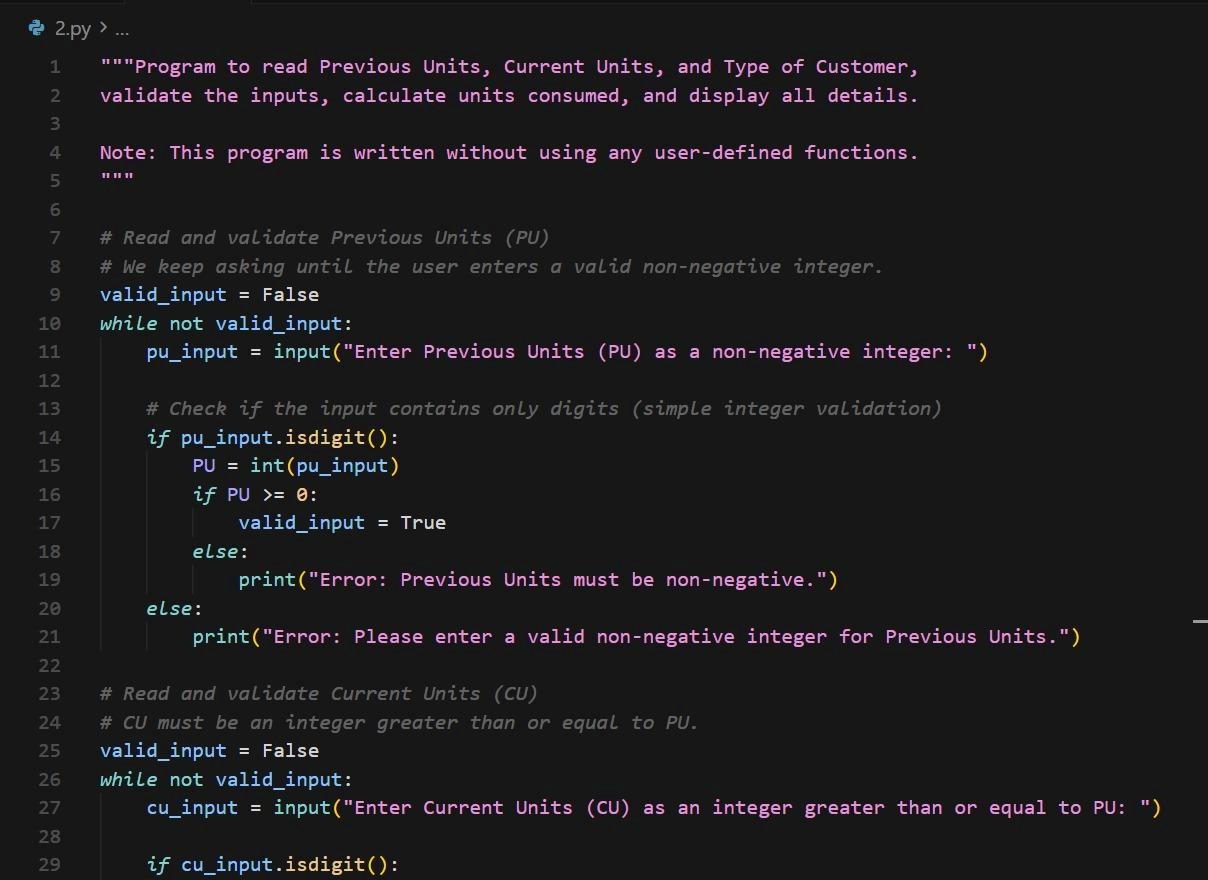
**o Previous Units (PU) o Current Units (CU) o Type of Customer**

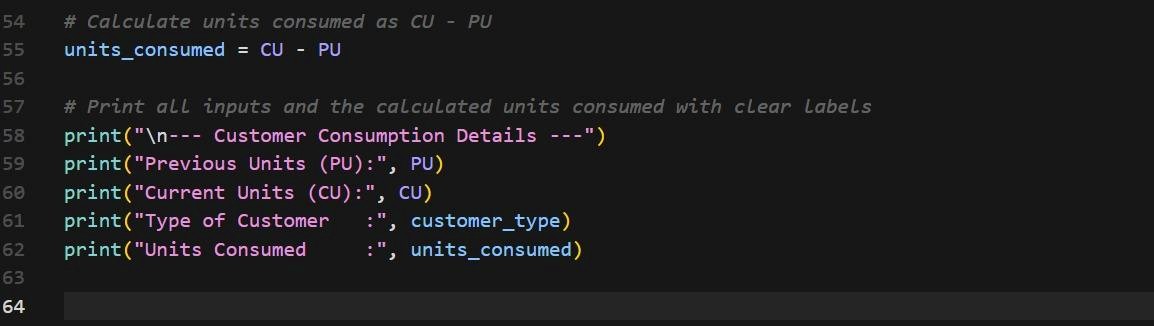
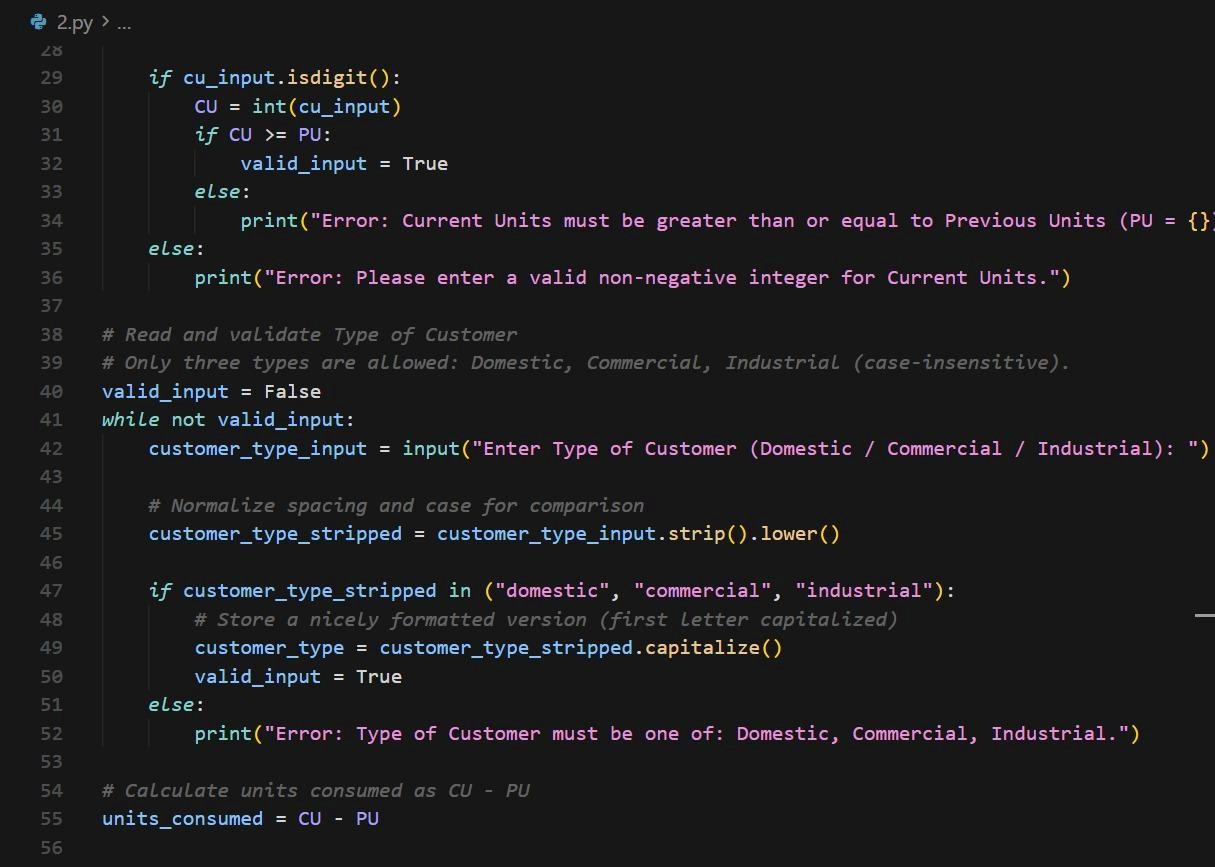
* **Calculates units consumed**
* **Implements logic directly in the main program (no functions)**

**Prompt:**

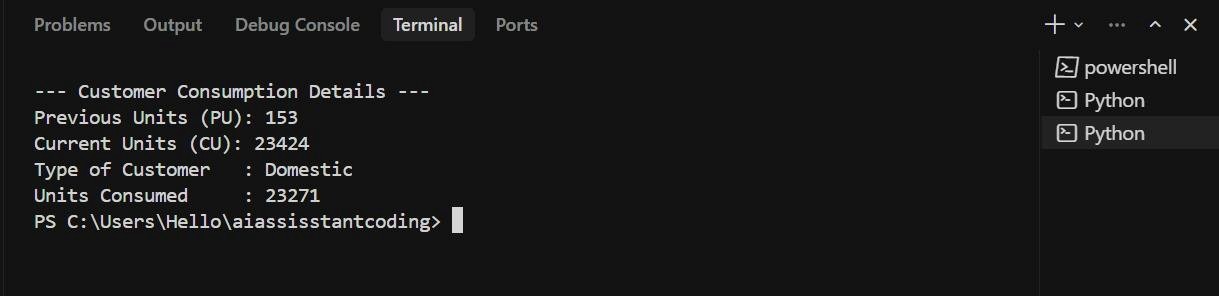
Generate a Python program (without using functions) to read: - Previous Units (PU) - Current Units (CU) - Type of customer (Domestic / Commercial / Industrial) Validate inputs and calculate units consumed as: units\_consumed = CU - PU Print all inputs and the calculated units consumed with clear labels. Add comments explaining each step.

**Code:**





**Output:**



**Justification:**

This code is designed to accurately read essential consumer inputs such as previous units, current units, and customer type. Input validation ensures logical correctness by preventing invalid meter readings. The units consumed calculation reflects real-world electricity meter behavior. Conditional checks improve data integrity and reliability. Implementing logic directly in the main program helps beginners understand sequential execution clearly.

**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:**

**o Domestic o Commercial o Industrial consumers**

**• Improve readability using AI prompts such as: o “Simplify energy charge calculation logic” o “Optimize conditional statements”**

**Prompt:**

Extend the existing Python program to calculate Energy Charges (EC) using conditional statements:

Domestic: - First 100 units: ₹1.5/unit - Above 100 units: ₹2.5/unit

Commercial: - Flat rate ₹4.0/unit

Industrial: - Flat rate ₹6.0/unit

Use if-elif-else statements and print the calculated EC.

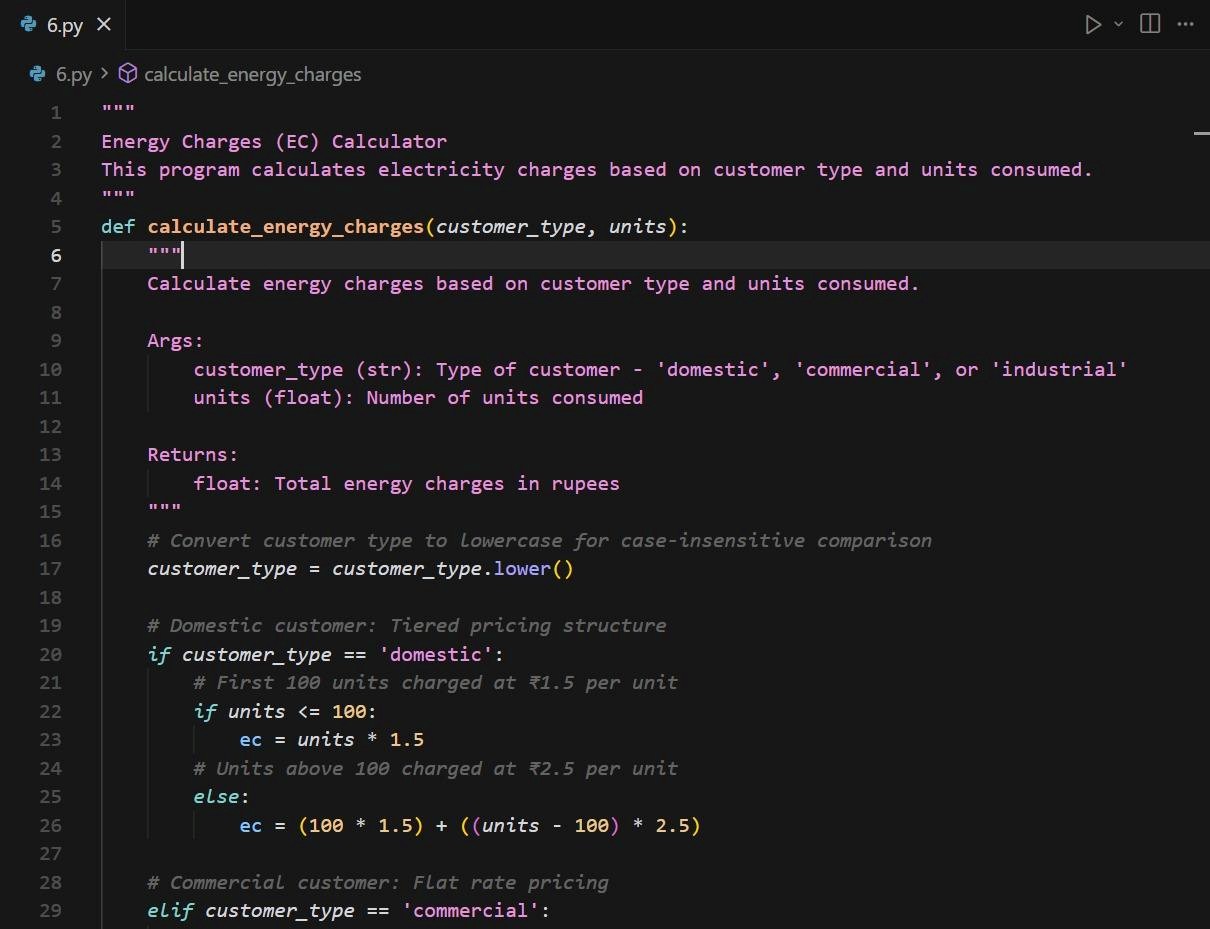
Add meaningful comments.

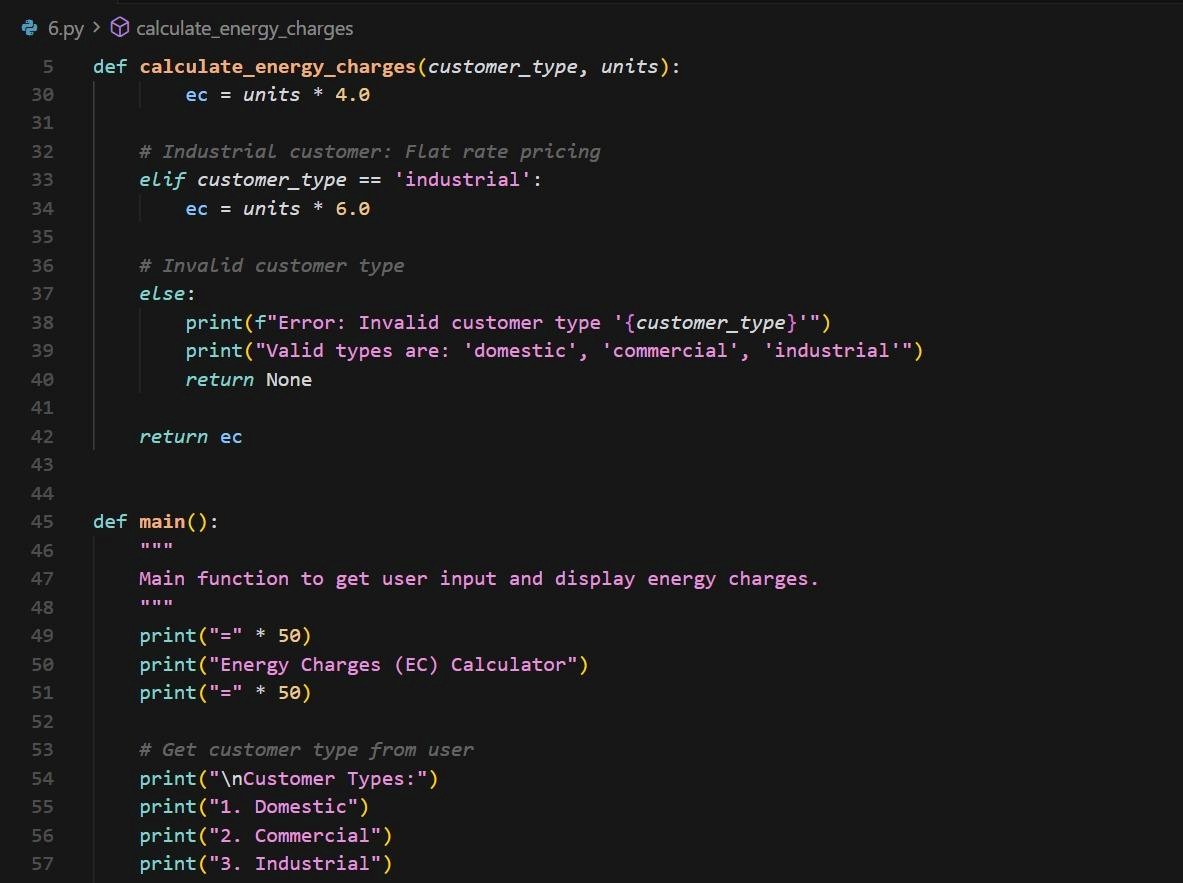
For logic optimization:

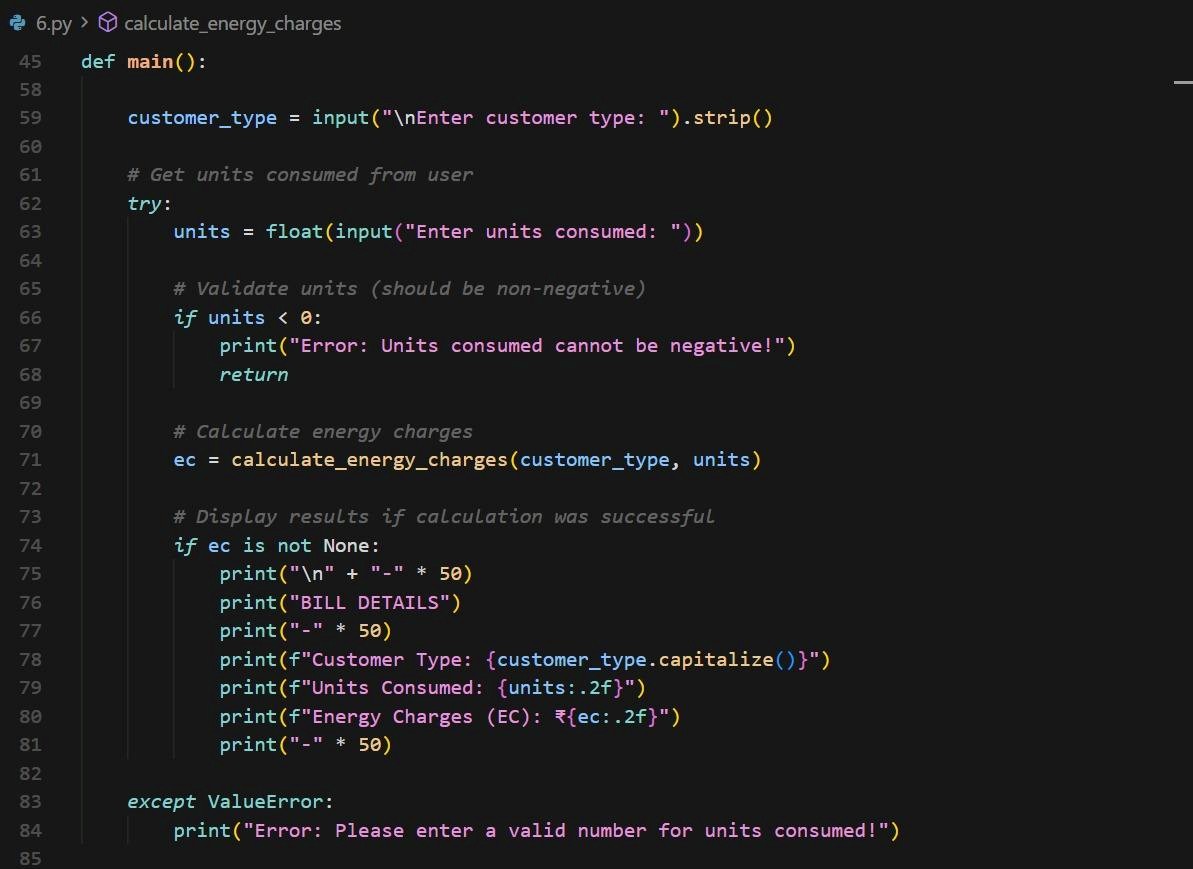
Simplify and optimize the energy charge calculation logic to improve readability.

Ensure the conditional structure is clean and easy for students to understand.

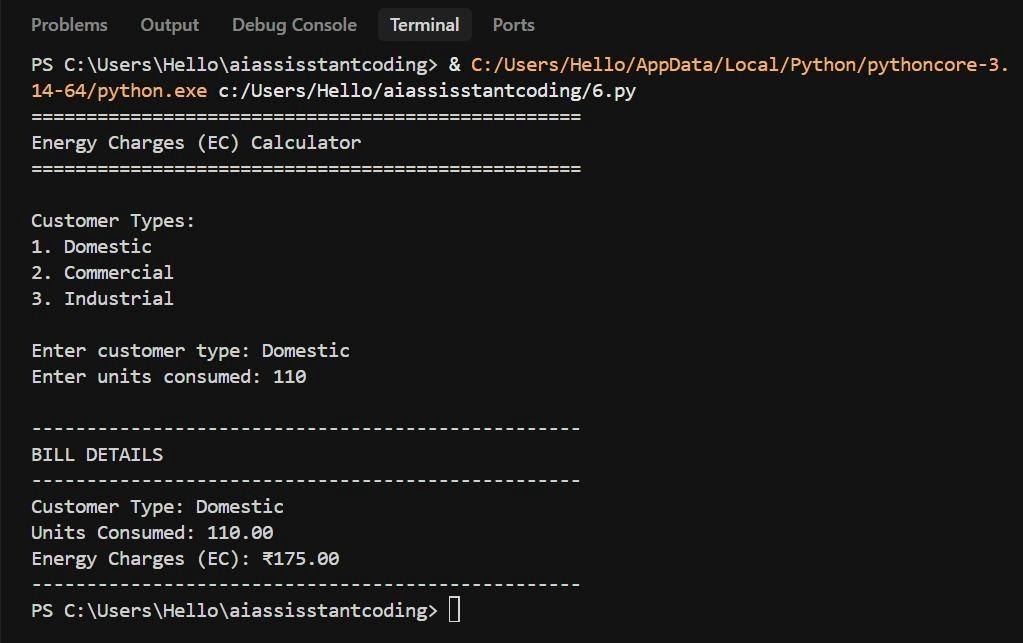
**Code:**







**Output:**



**Justification:**

This task applies tariff-based billing logic using conditional statements to differentiate customer categories. The use of if–elif–else structures ensures correct energy charge computation for domestic, commercial, and industrial consumers. Slab-wise calculation mirrors real electricity board billing practices. Optimized conditional logic improves readability and maintainability. The approach enhances understanding of decision-making constructs in Python.

**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:**

**o Calculate Energy Charges o Calculate Fixed Charges • Returns calculated values**

**• Includes meaningful comments**

**Prompt:**

(Function for Energy Charges)

Rewrite the program using user-defined functions.

Create a function named calculate\_energy\_charges(units, customer\_type)

that returns the energy charges based on tariff rules.

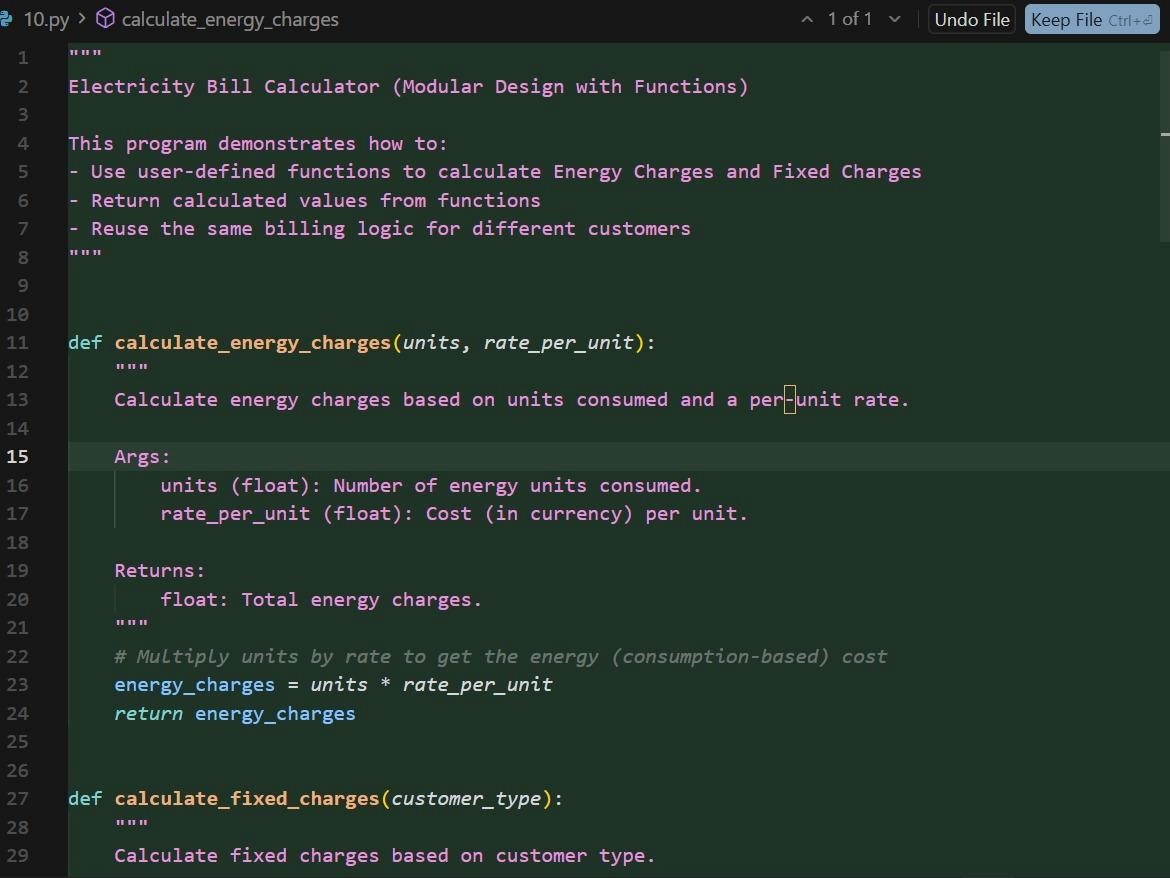
Call the function from the main program. Add proper comments and sample output printing. Add Fixed Charges Function) Add another user-defined function named calculate\_fixed\_charges(customer\_type) with the following logic: Domestic: ₹50

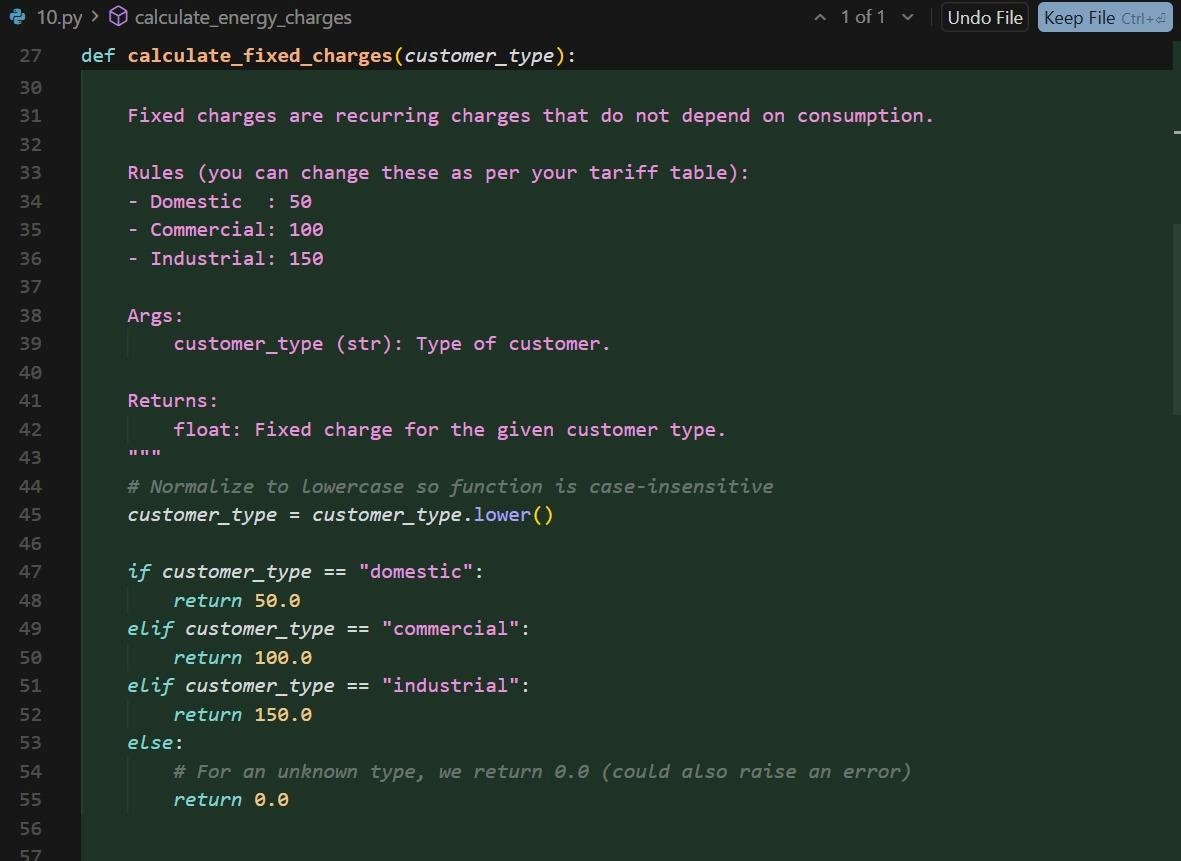
Commercial: ₹100

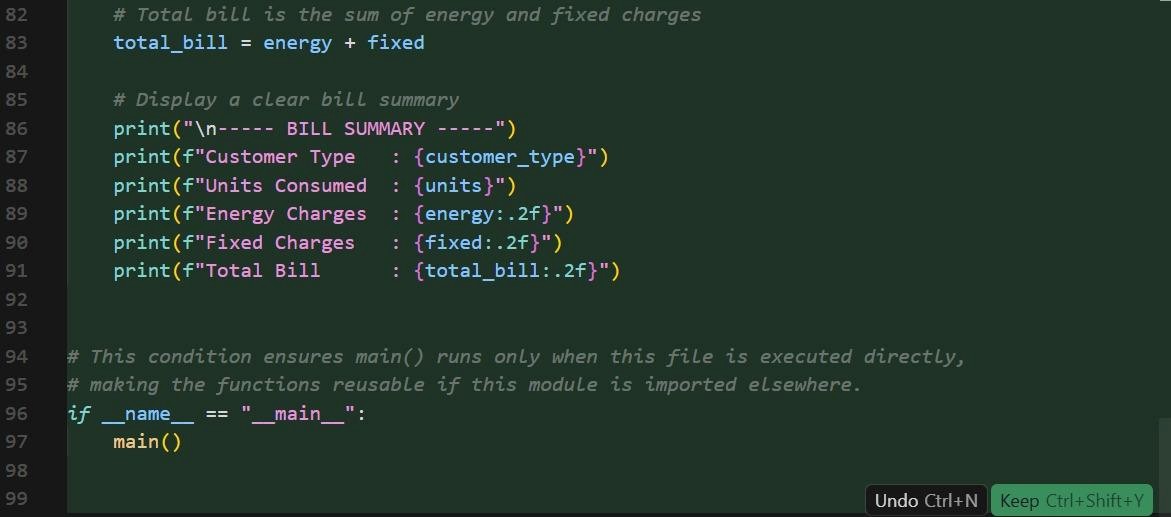
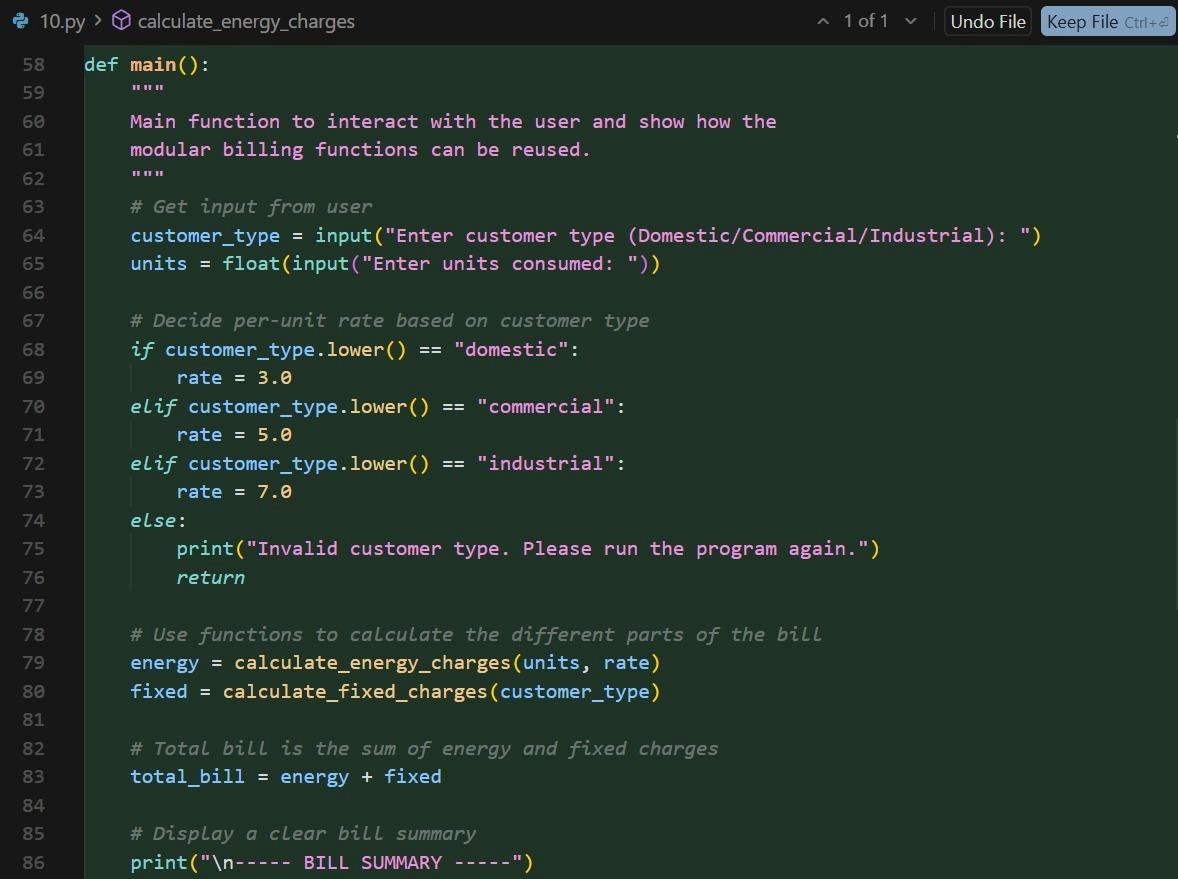
Industrial: ₹150

Return the fixed charges and display them in the main program. Include comments explaining function usage.

**Code:**







**Output:**



**Justification:**

The program is modularized using user-defined functions to promote code reusability and clarity. Separate functions for energy and fixed charge calculations reduce redundancy. This design allows easy extension for future tariff changes.

Clear function naming improves program readability.

Modular structure reflects professional software development practices.

**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”**

**Prompt:**

Add Extra Charges

Extend the function-based electricity billing program to calculate:

- Customer Charges (CC) = ₹30 for all consumers - Electricity Duty (ED) = 5% of Energy

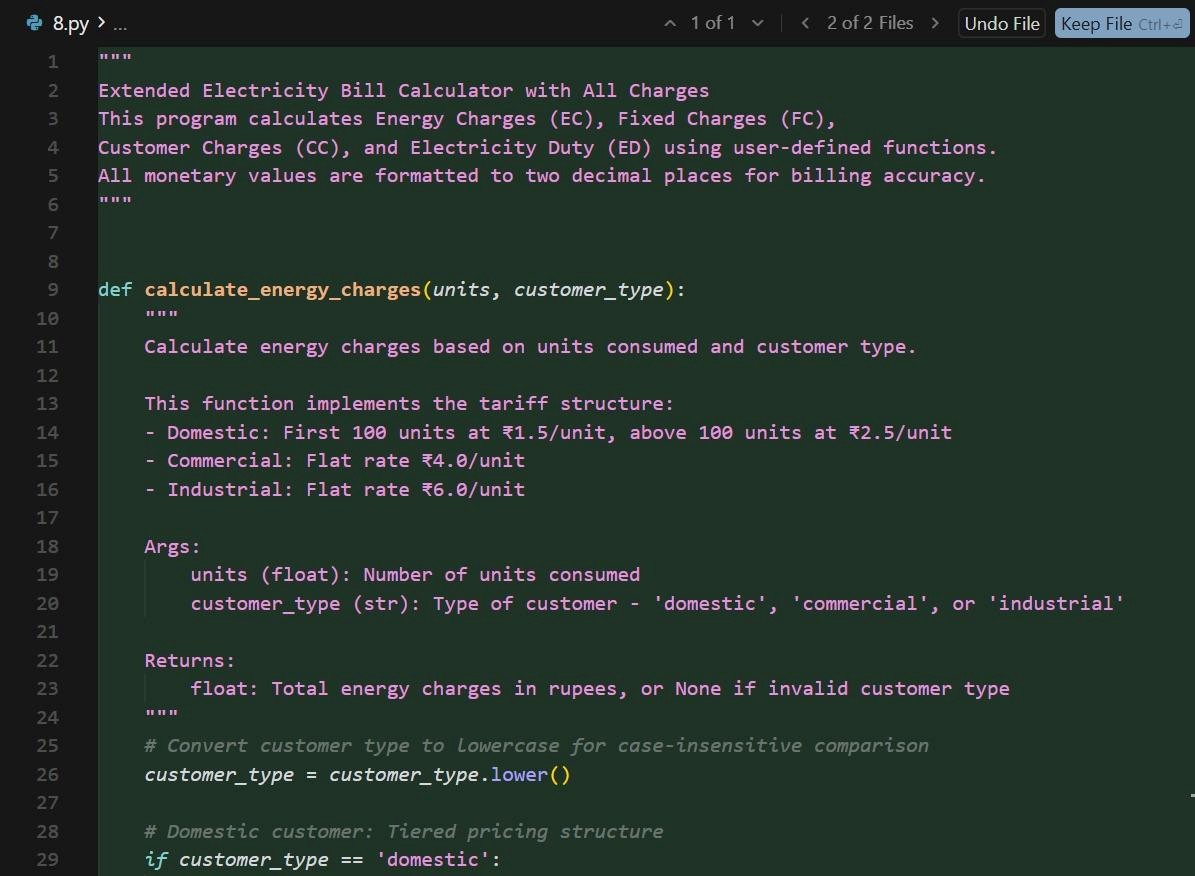
Charges (EC)

Print EC, FC, CC, and ED separately with proper formatting.

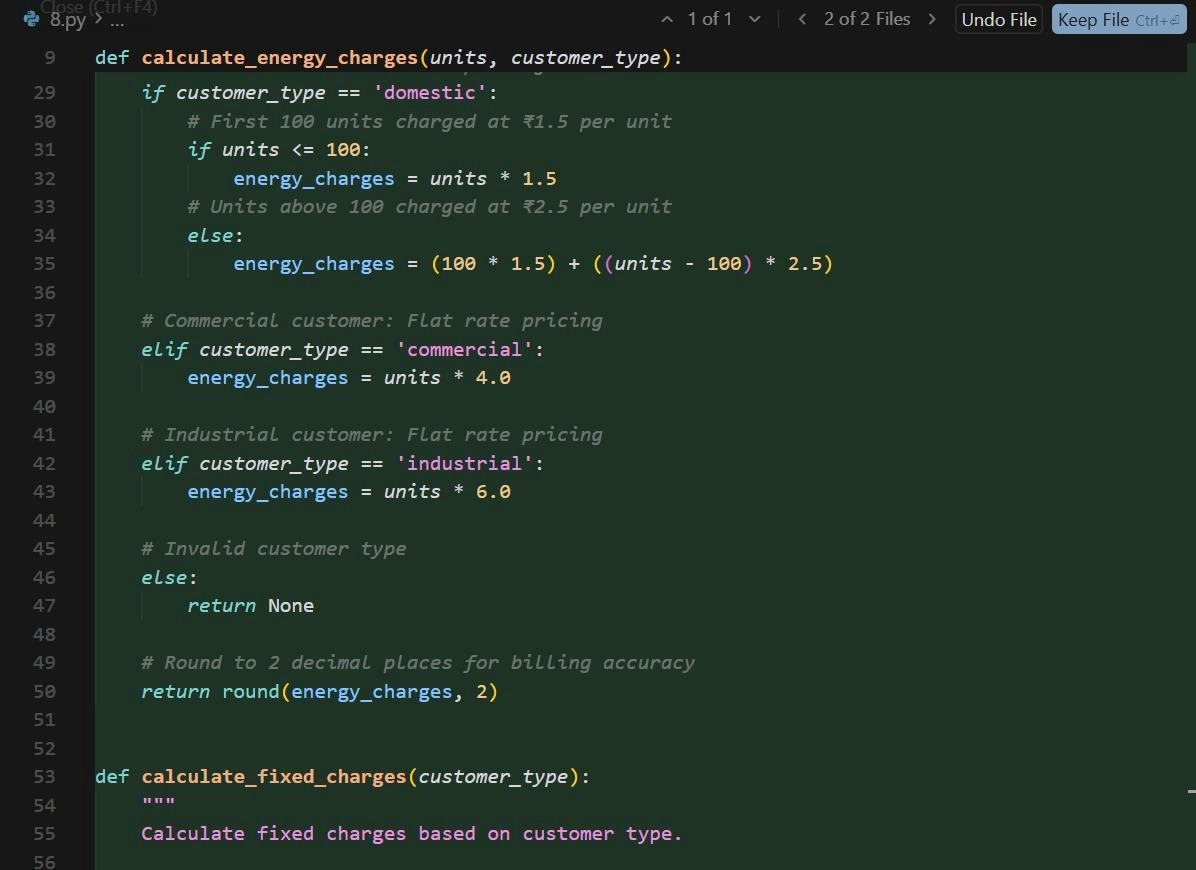
Add comments for billing accuracy.

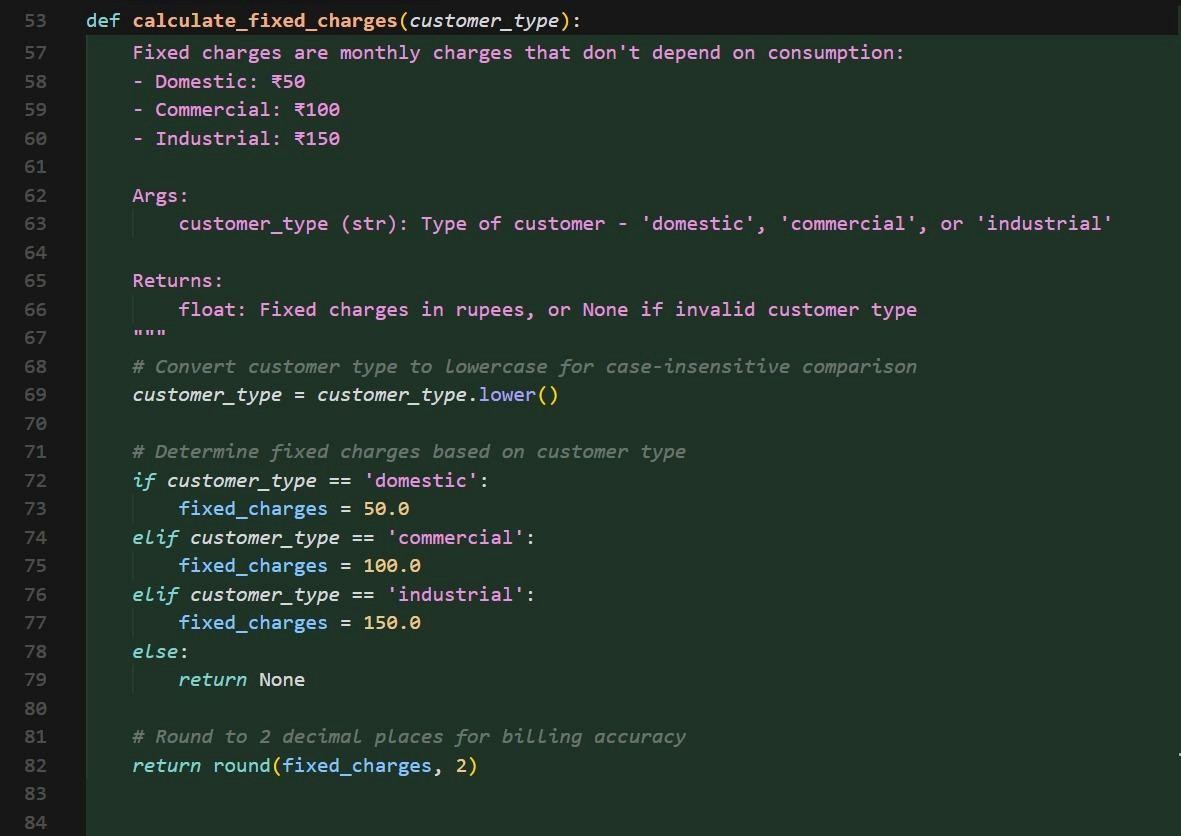
Improve billing accuracy by formatting all monetary values to two decimal places.

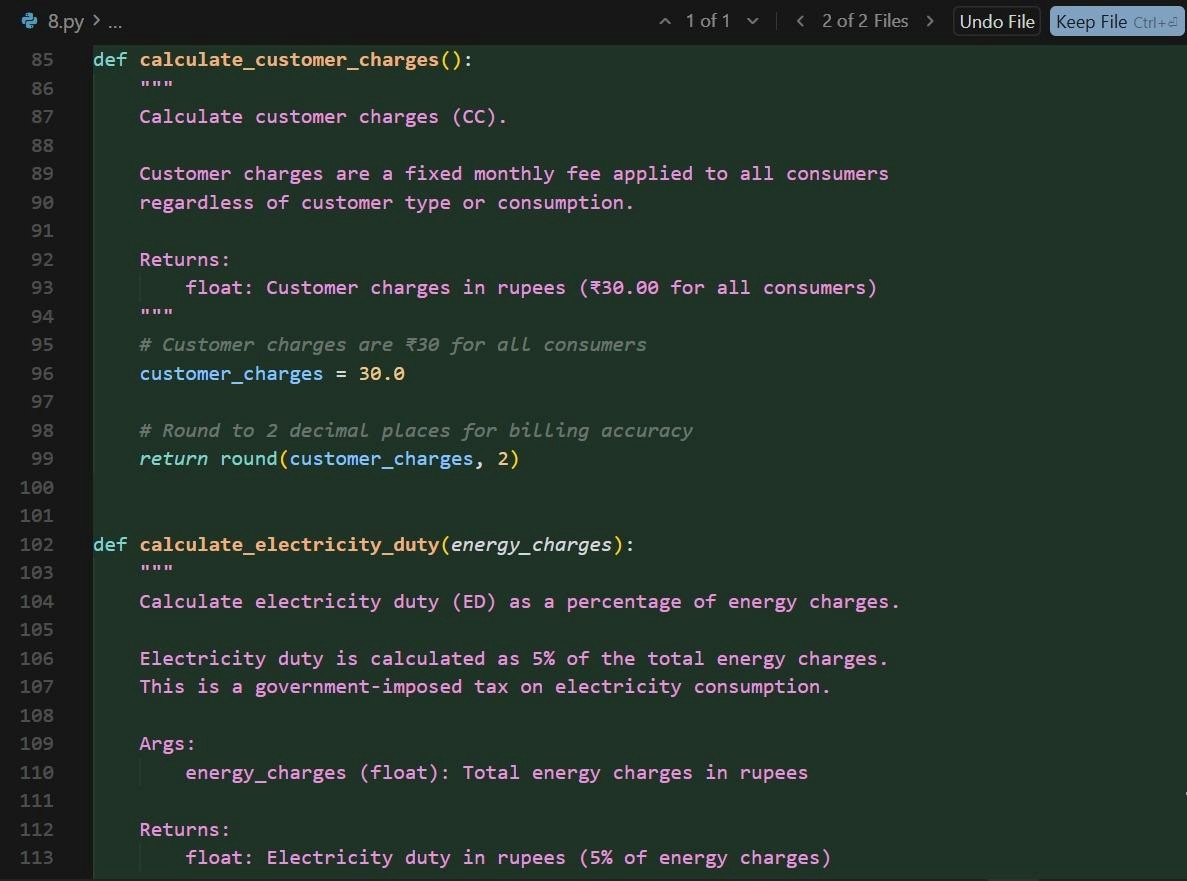
Ensure calculations are clear and correct.

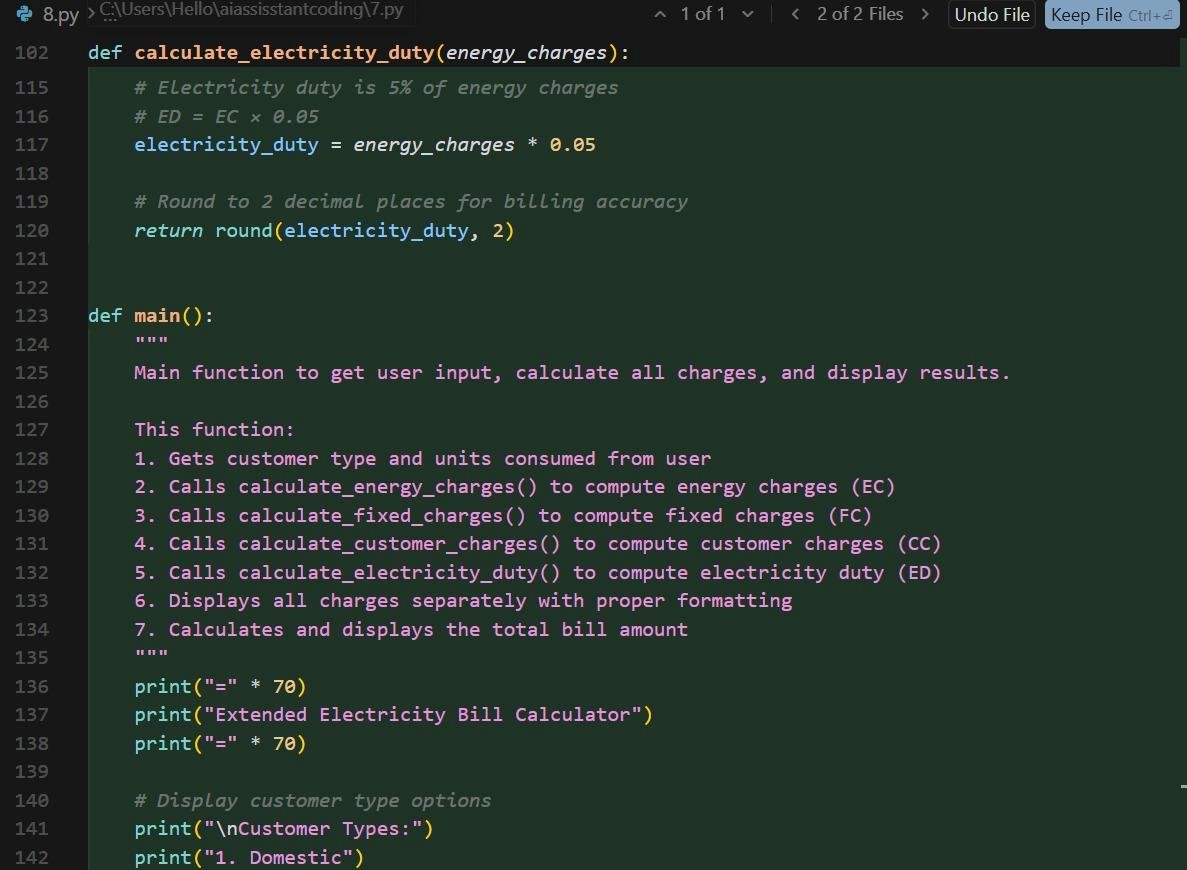


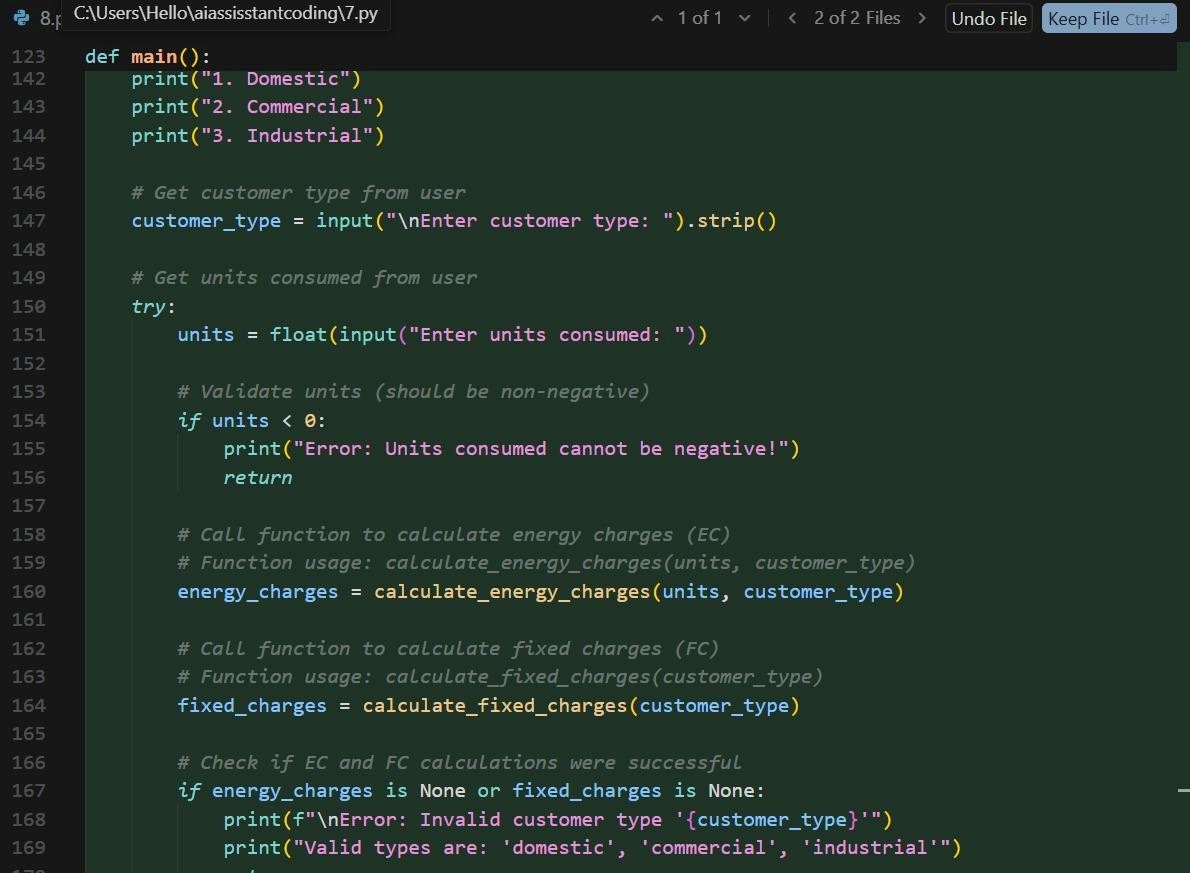
**Code:**

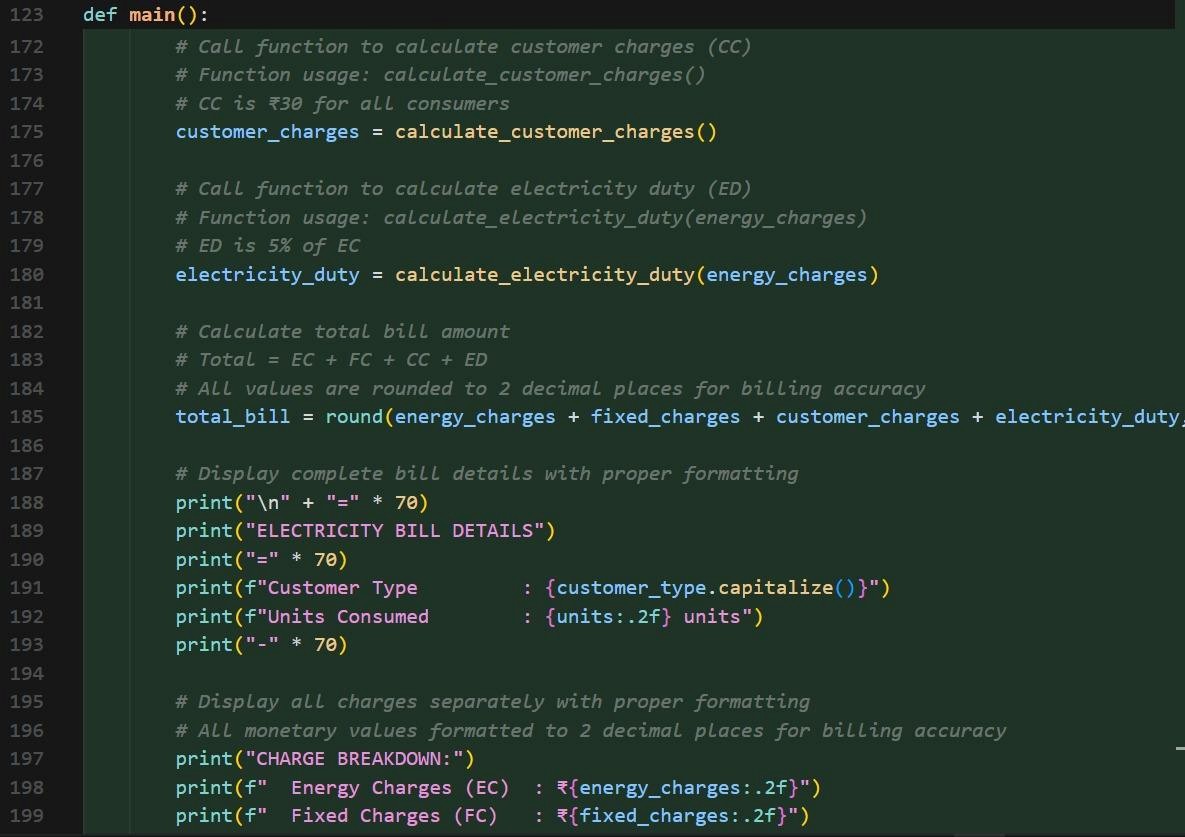


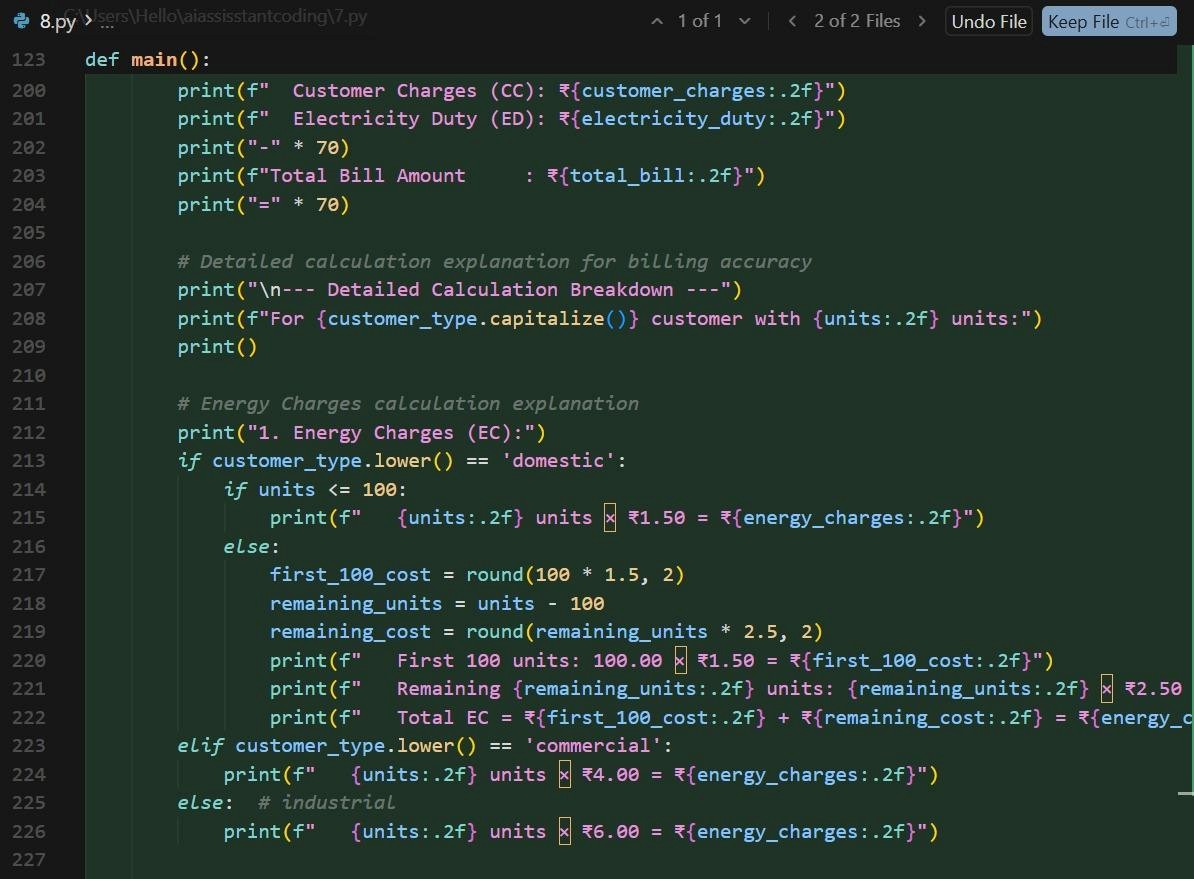


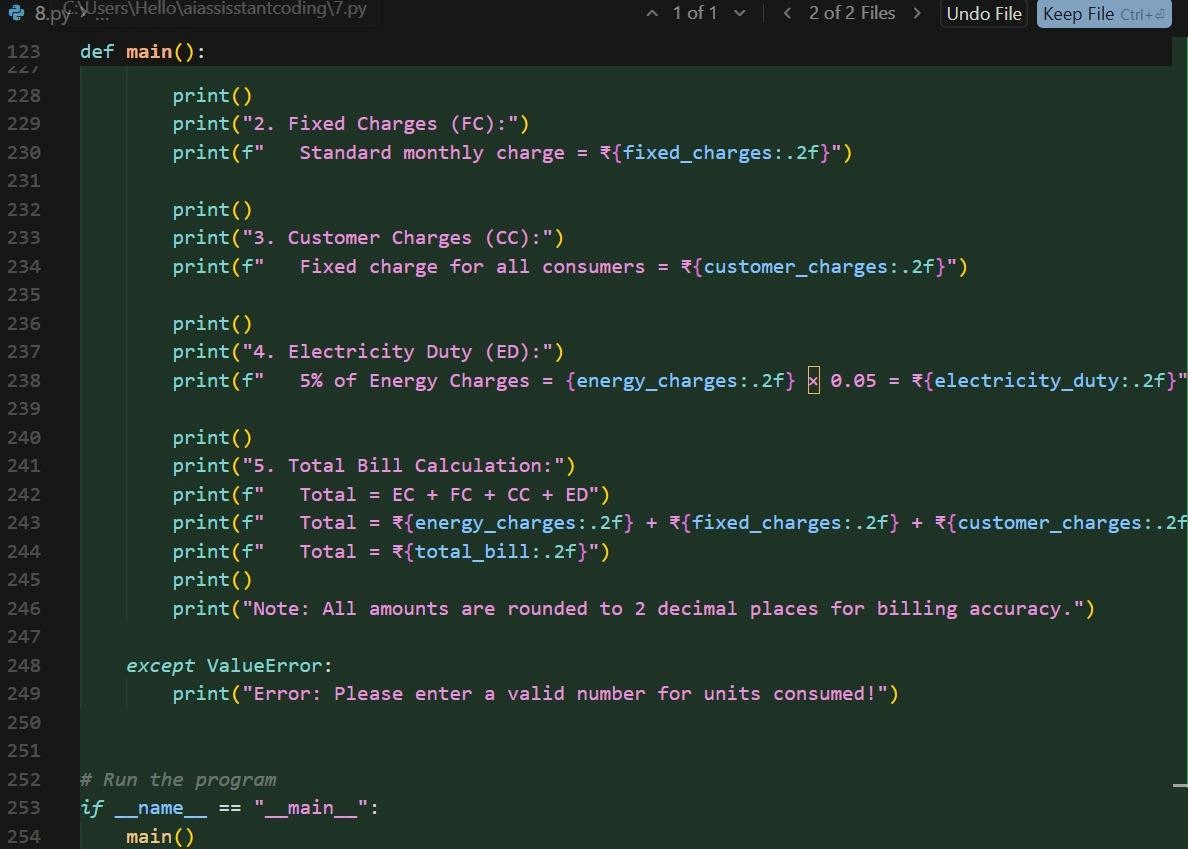




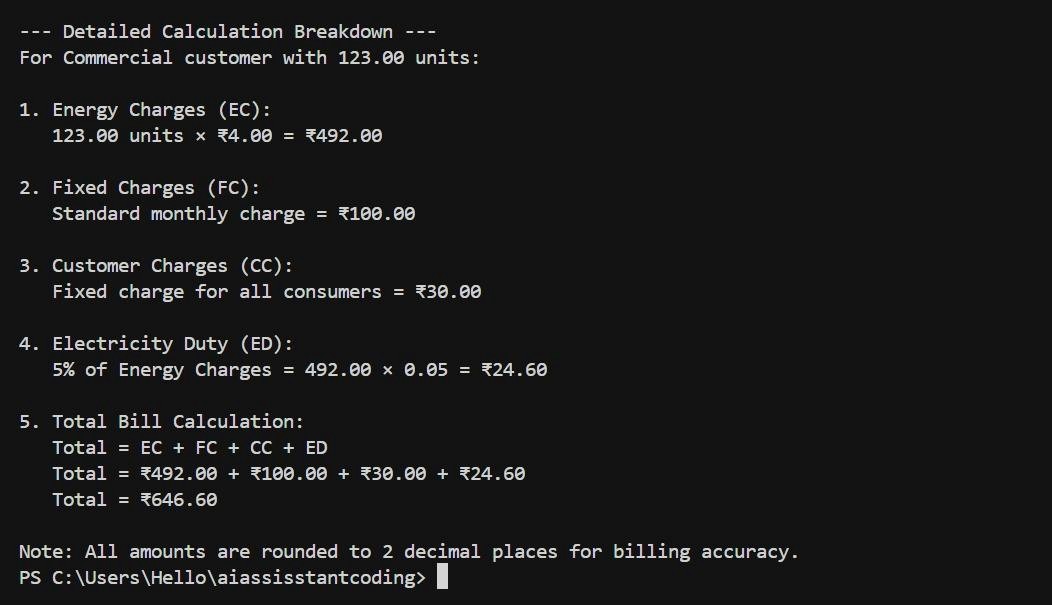
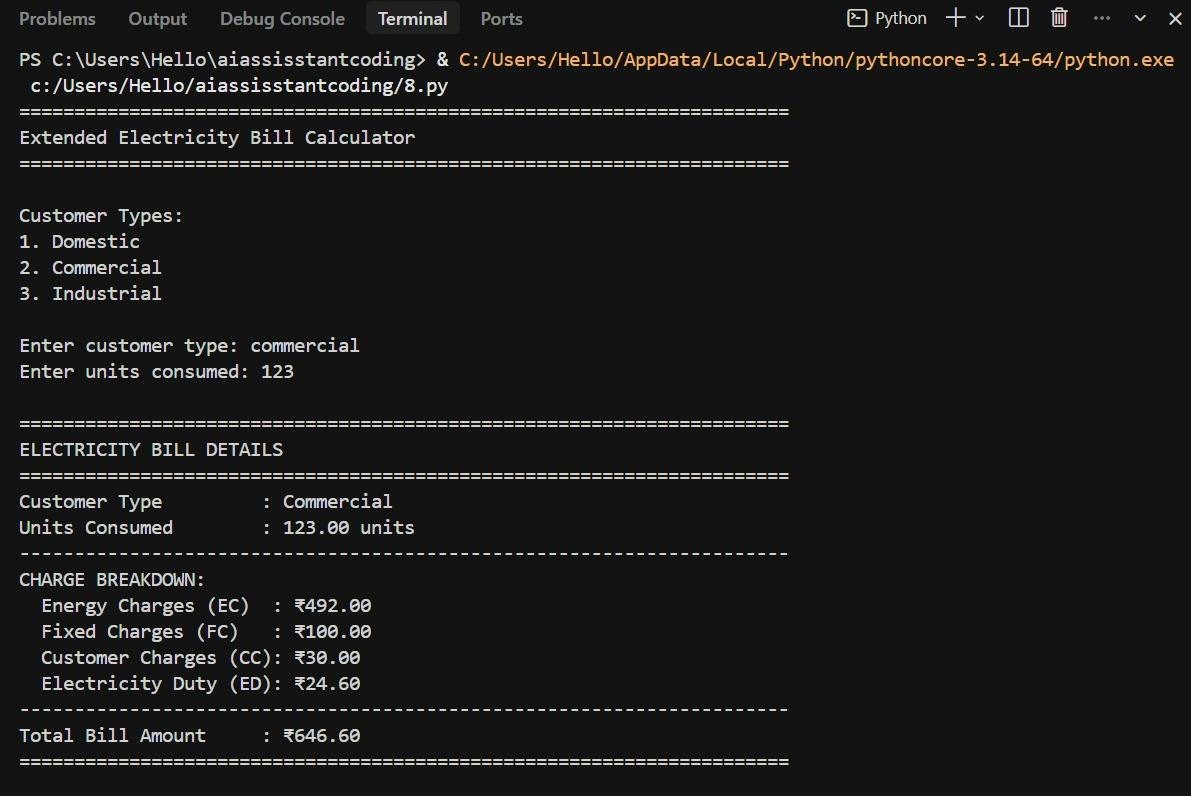








**Output:**



**Justification:**

Additional charges such as fixed charges, customer charges, and electricity duty are calculated separately to improve transparency. Percentage-based duty calculation ensures billing accuracy. Explicit printing of each charge enhances verification and debugging. The design aligns with actual utility billing standards. This structured approach simplifies understanding of multi-component billing systems.

**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:**

**o Energy Charges (EC) o Fixed Charges (FC) o Customer Charges (CC) o Electricity Duty (ED) o Total Bill Amount**

**• Analyze the program based on:**

* **Accuracy o Readability**
* **Real-world applicability**

**Prompt:**

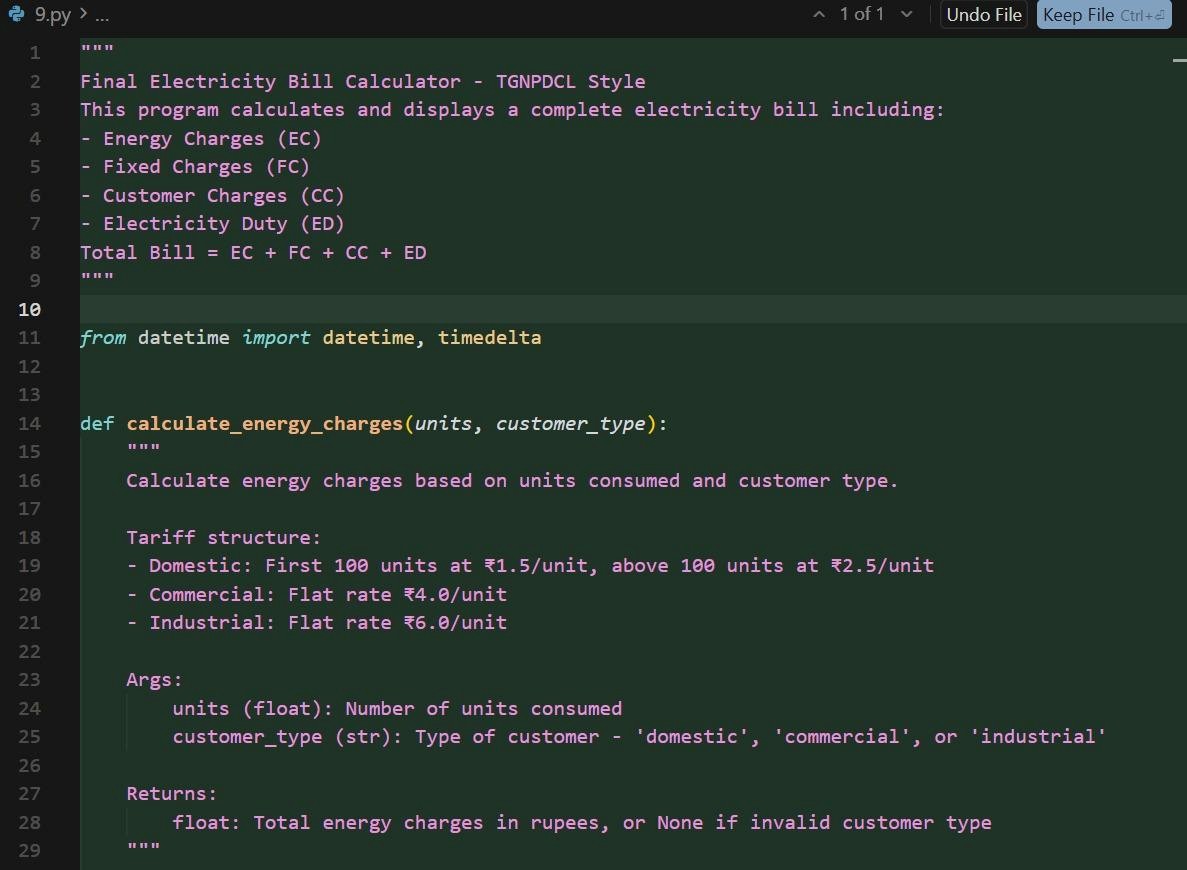
Final Bill Calculation

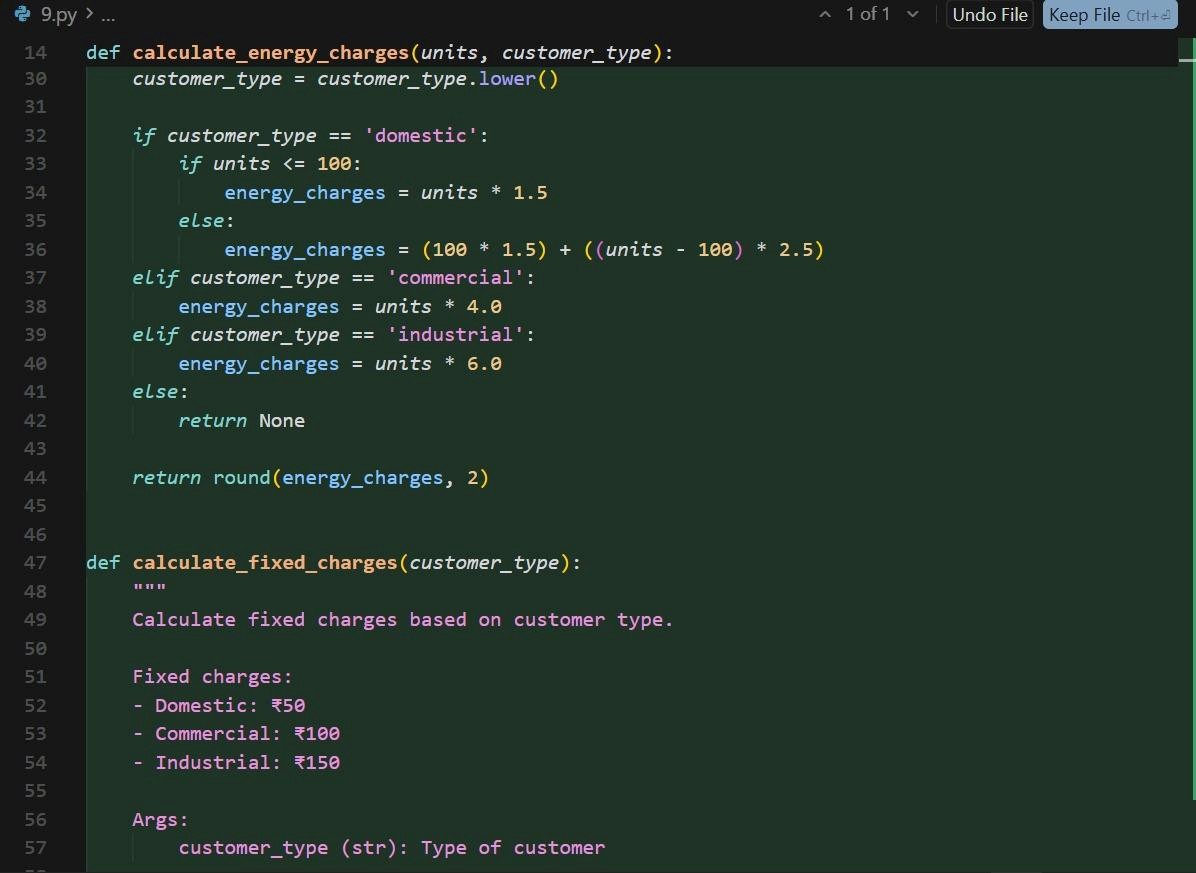
Generate the final electricity bill by calculating:

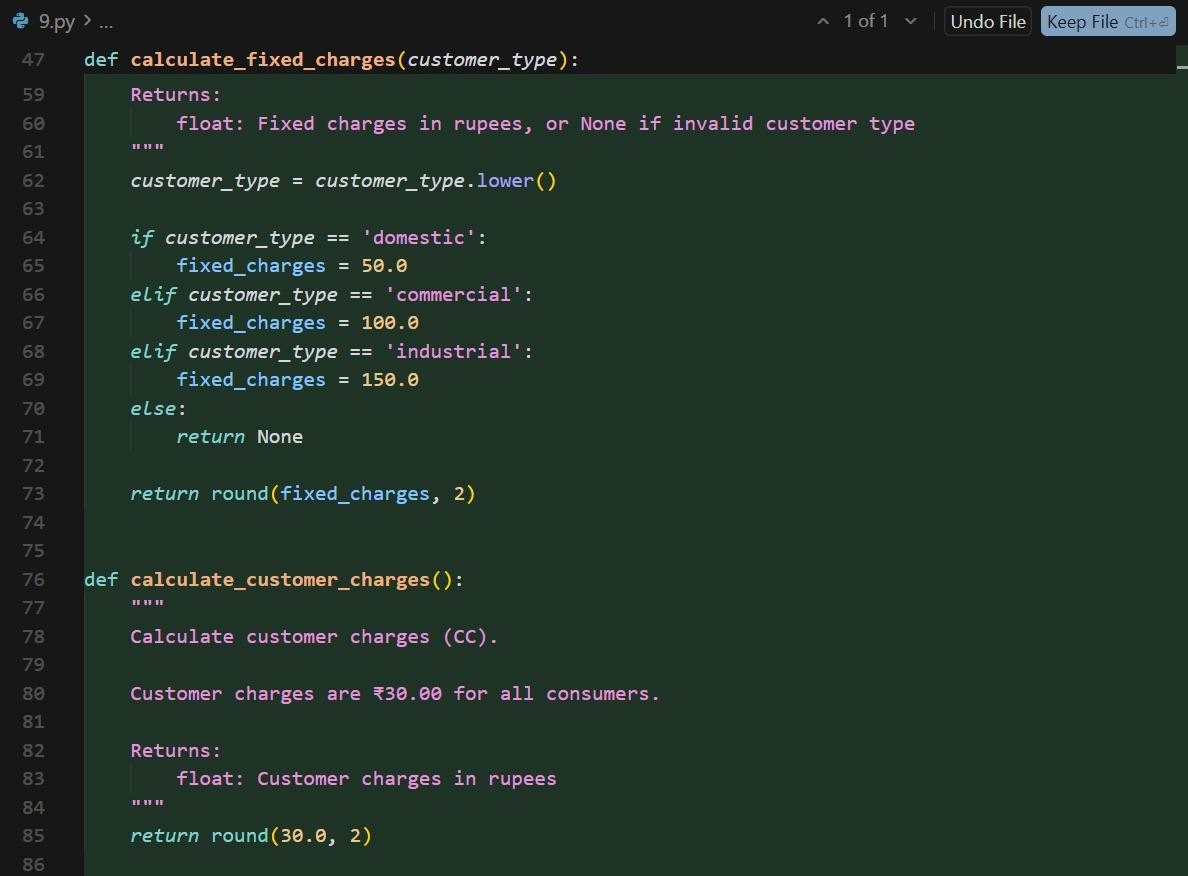
Total Bill = Energy Charges + Fixed Charges + Customer Charges + Electricity Duty Display a neatly formatted electricity bill output similar to a real TGNPDCL bill.

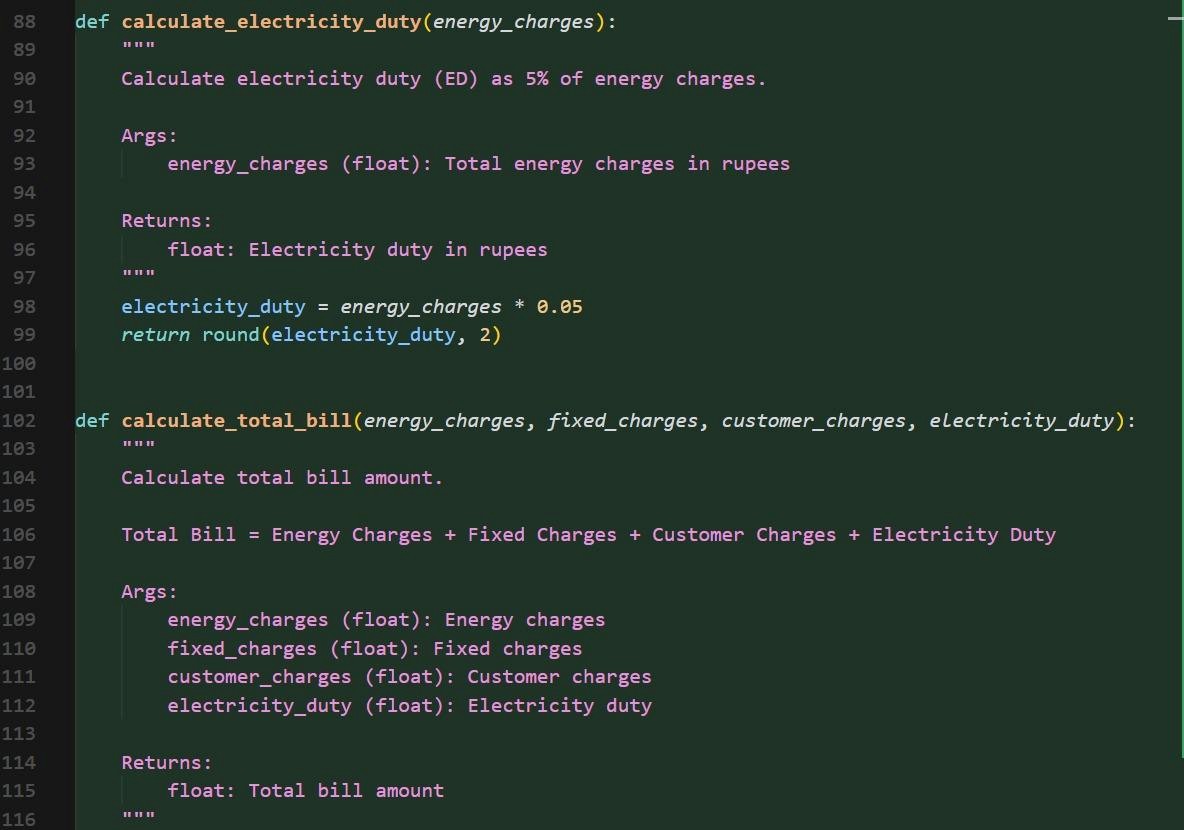
Use clear headings and labels. Analysis Paragraph for Report) Write a short analysis paragraph evaluating the electricity billing program based on: - Accuracy - Readability - Real-world applicability Keep the explanation suitable for a laboratory record.

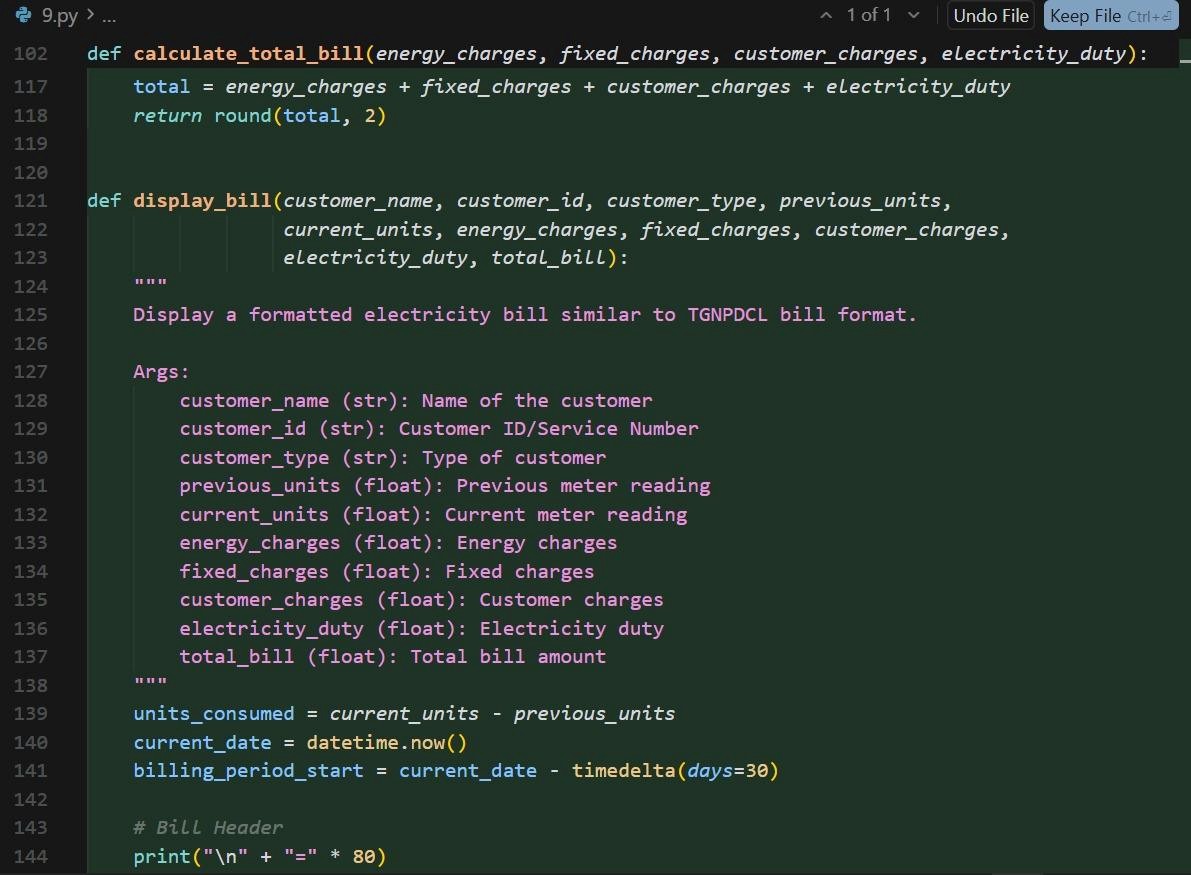
**Code:**

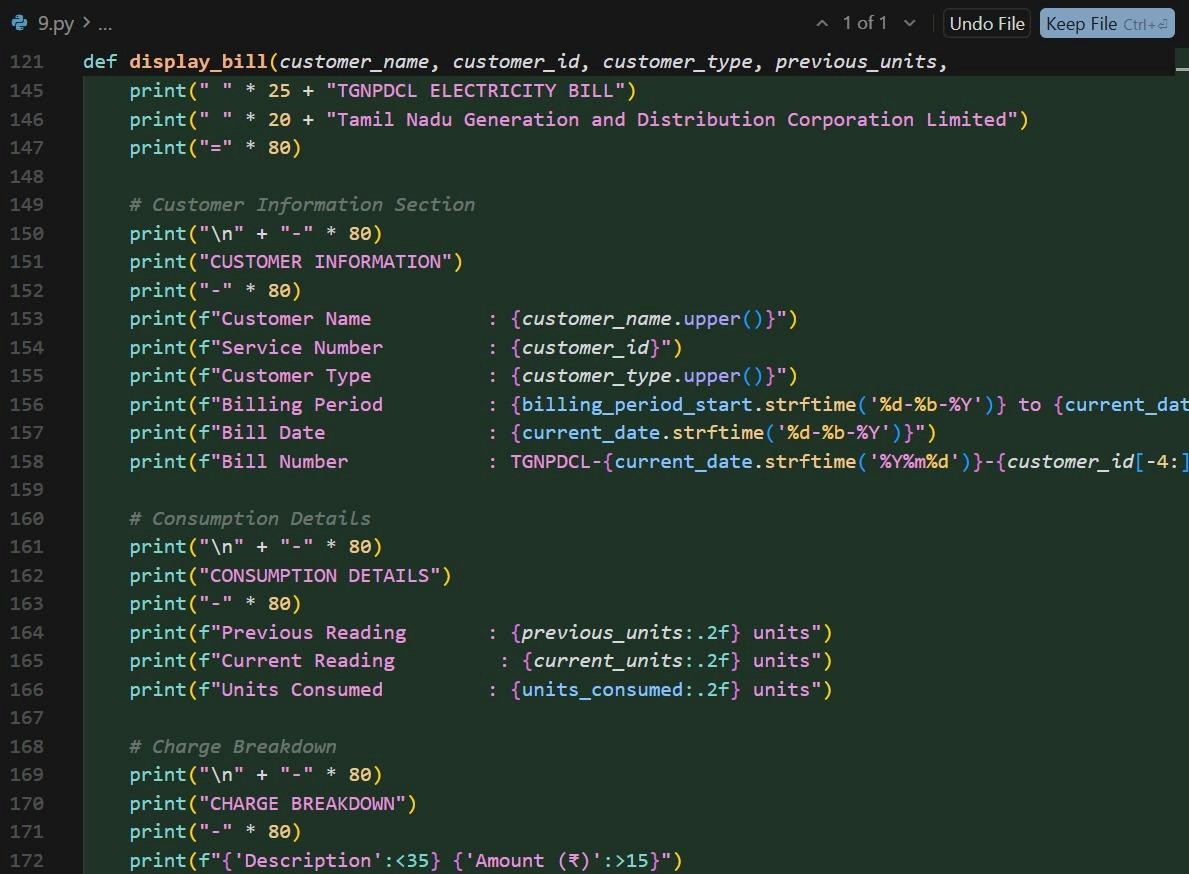


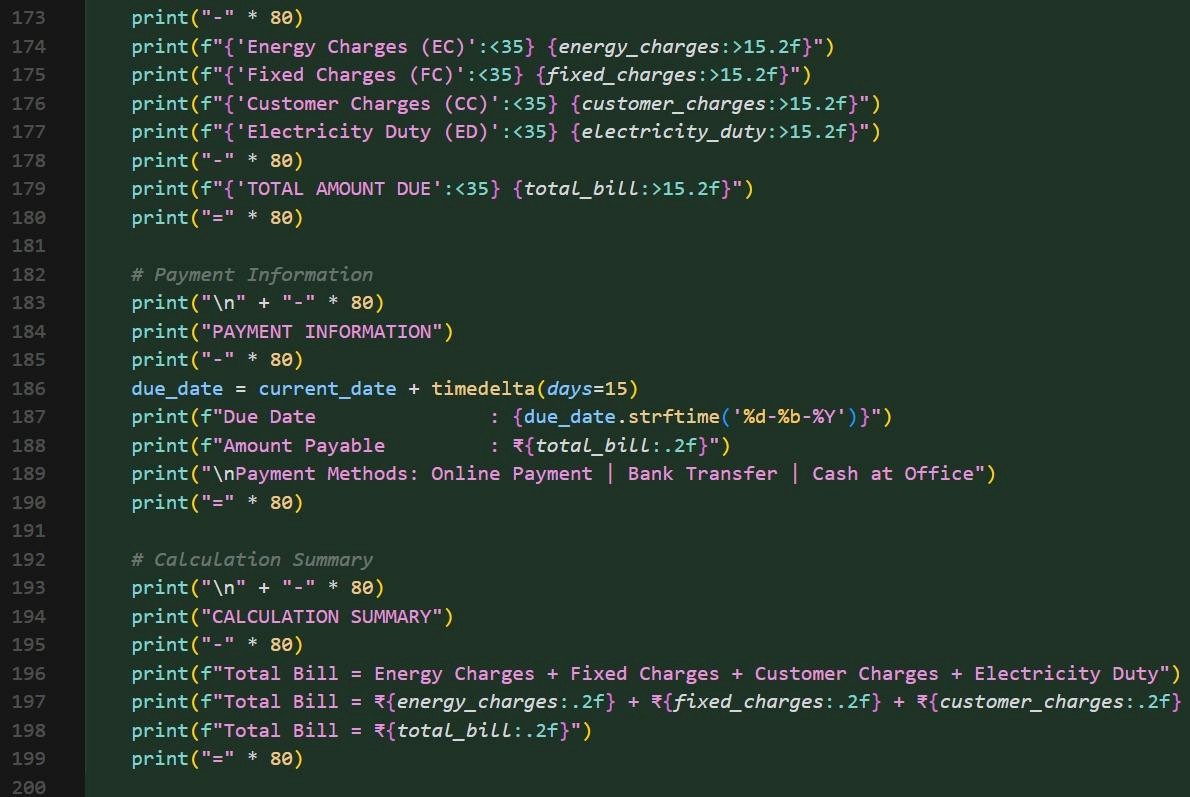


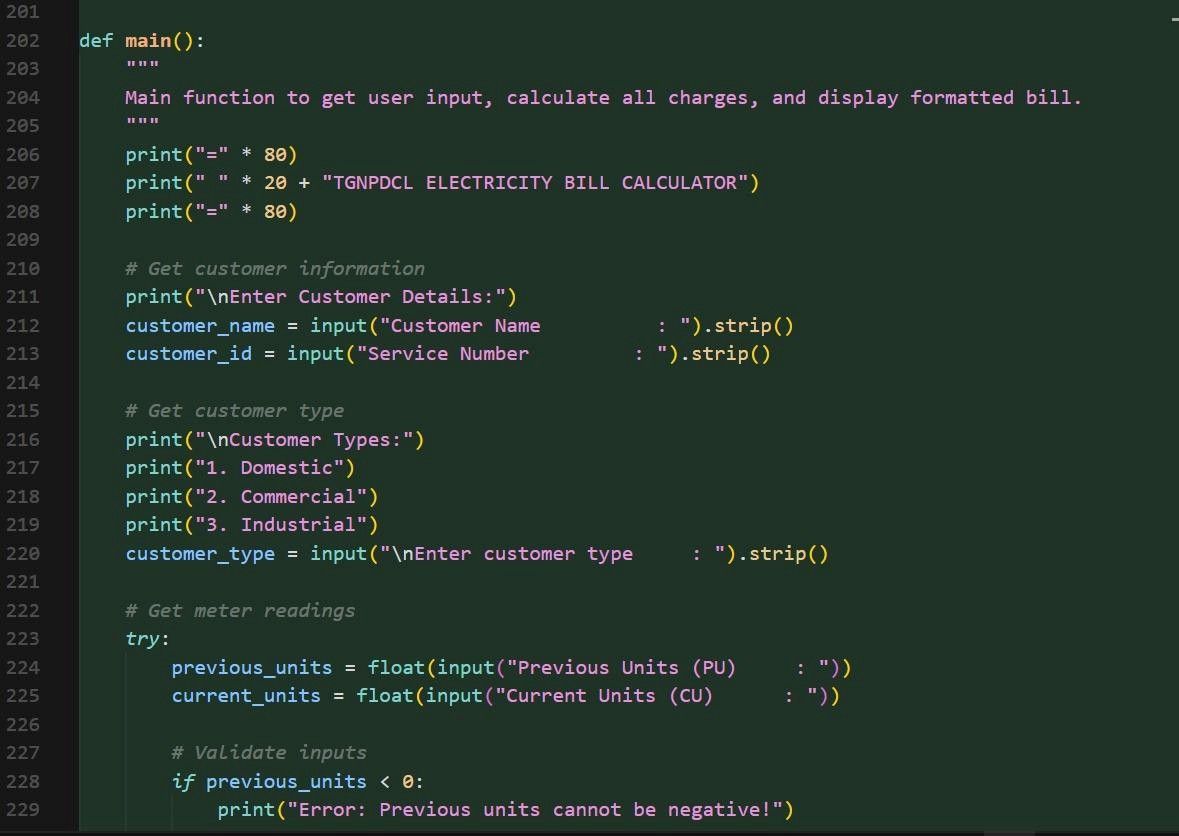


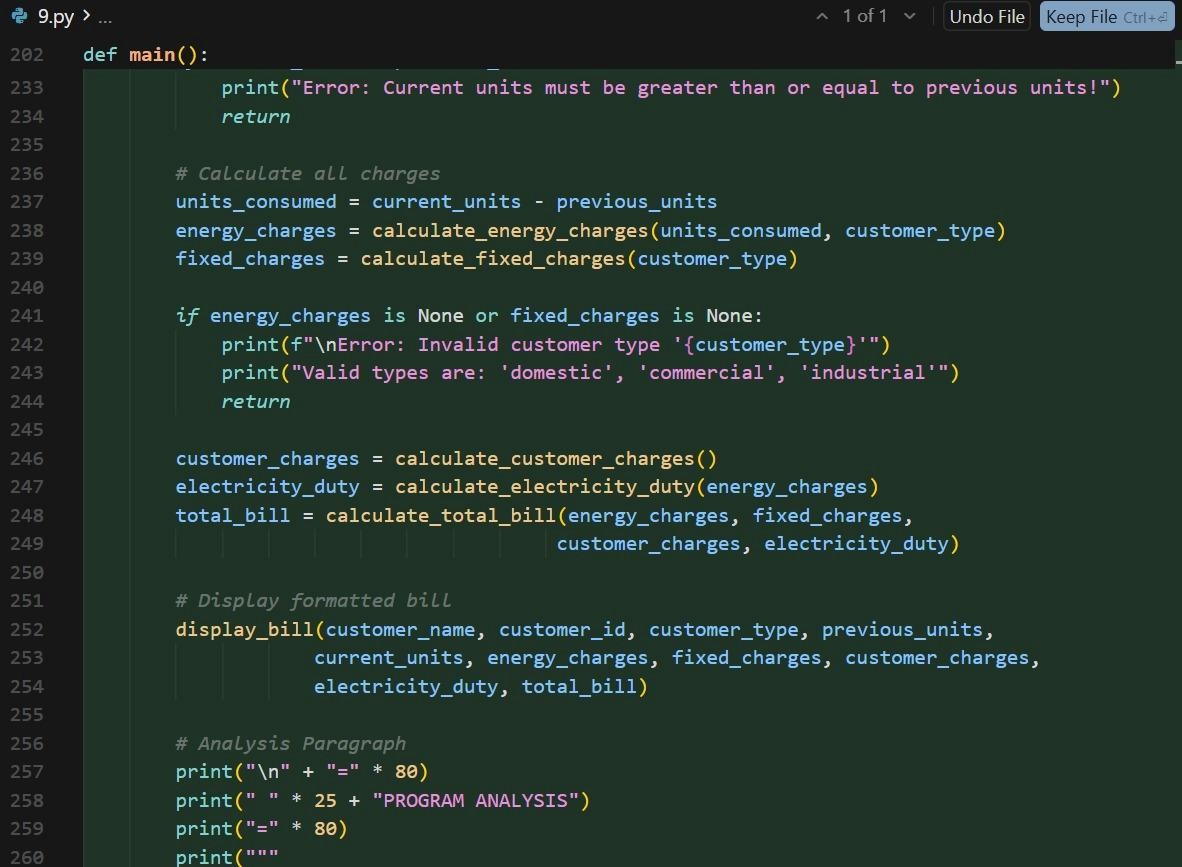


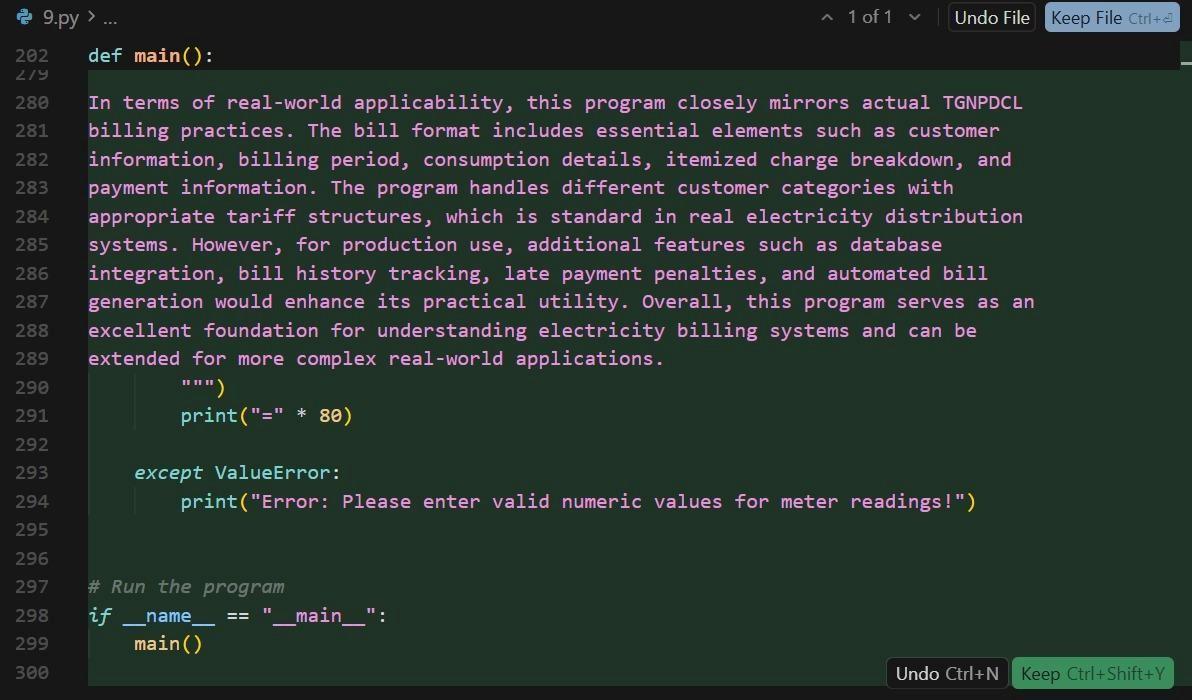
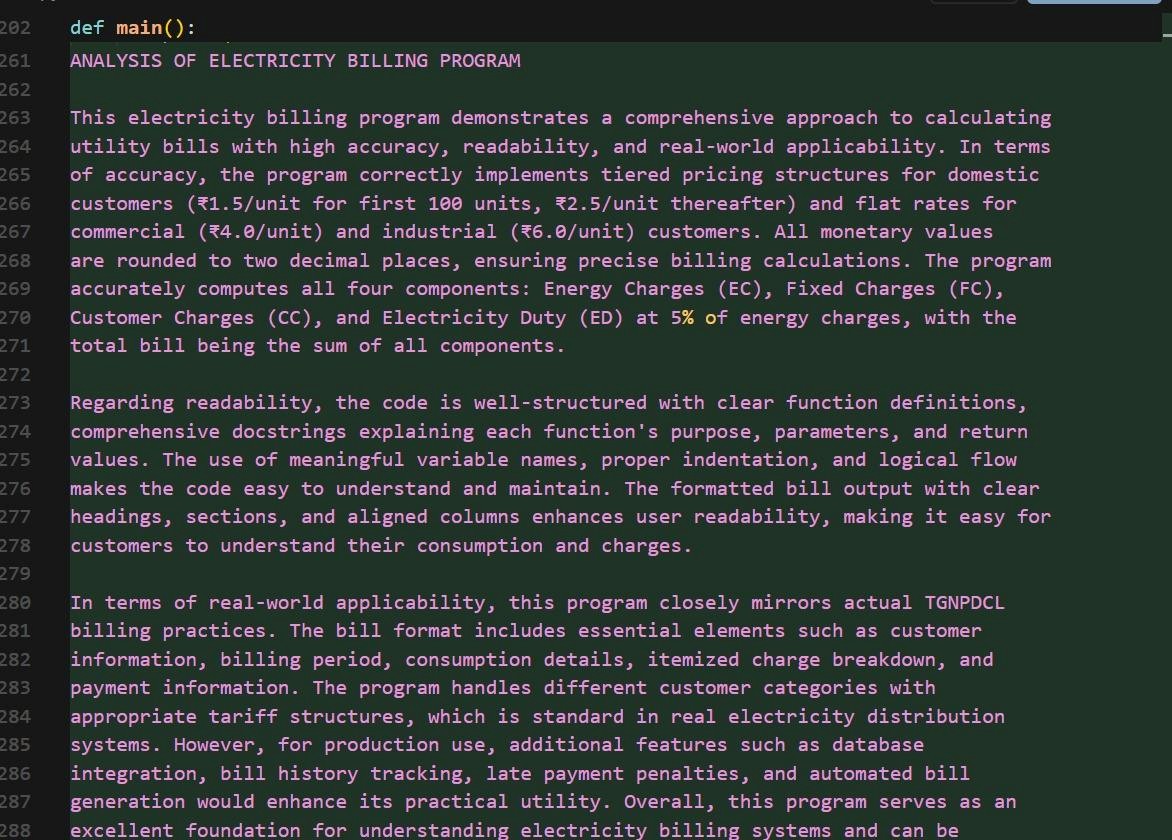




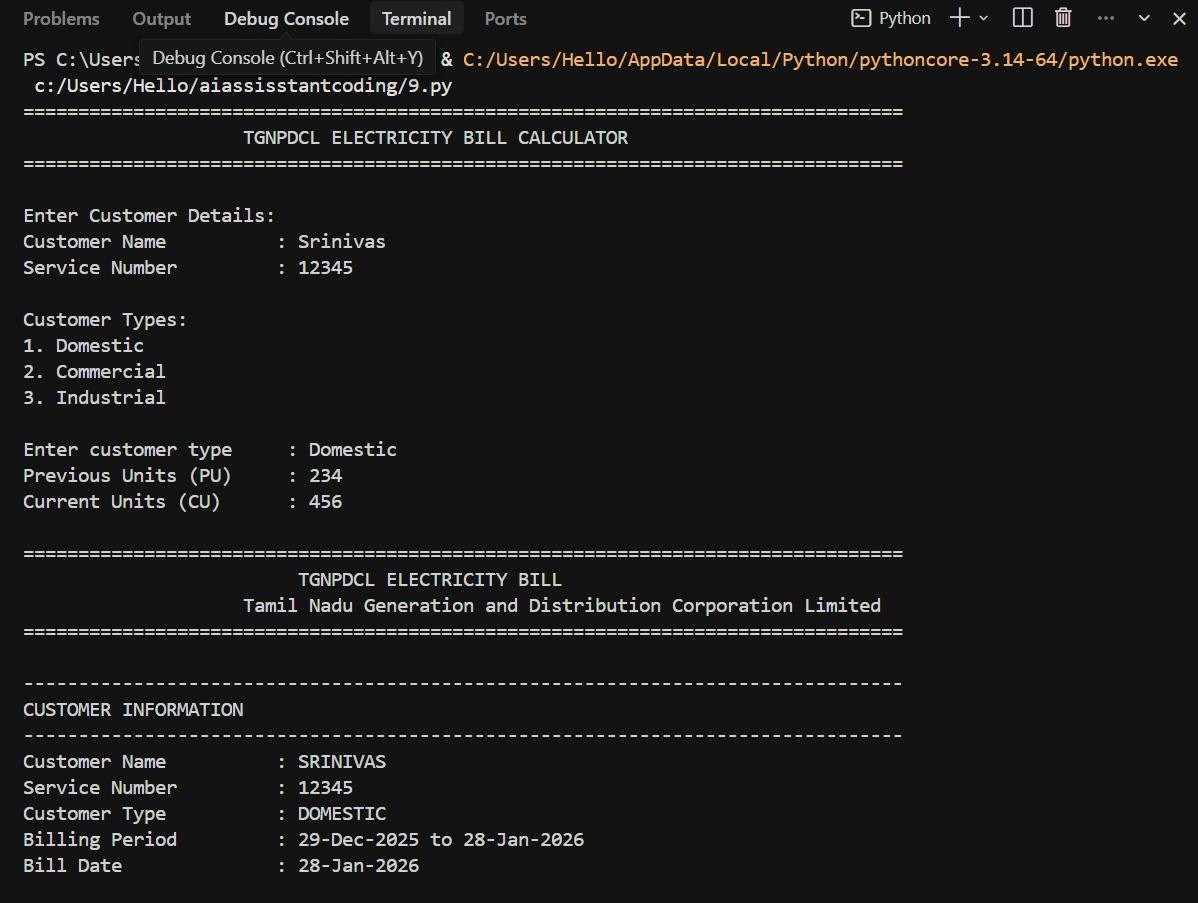


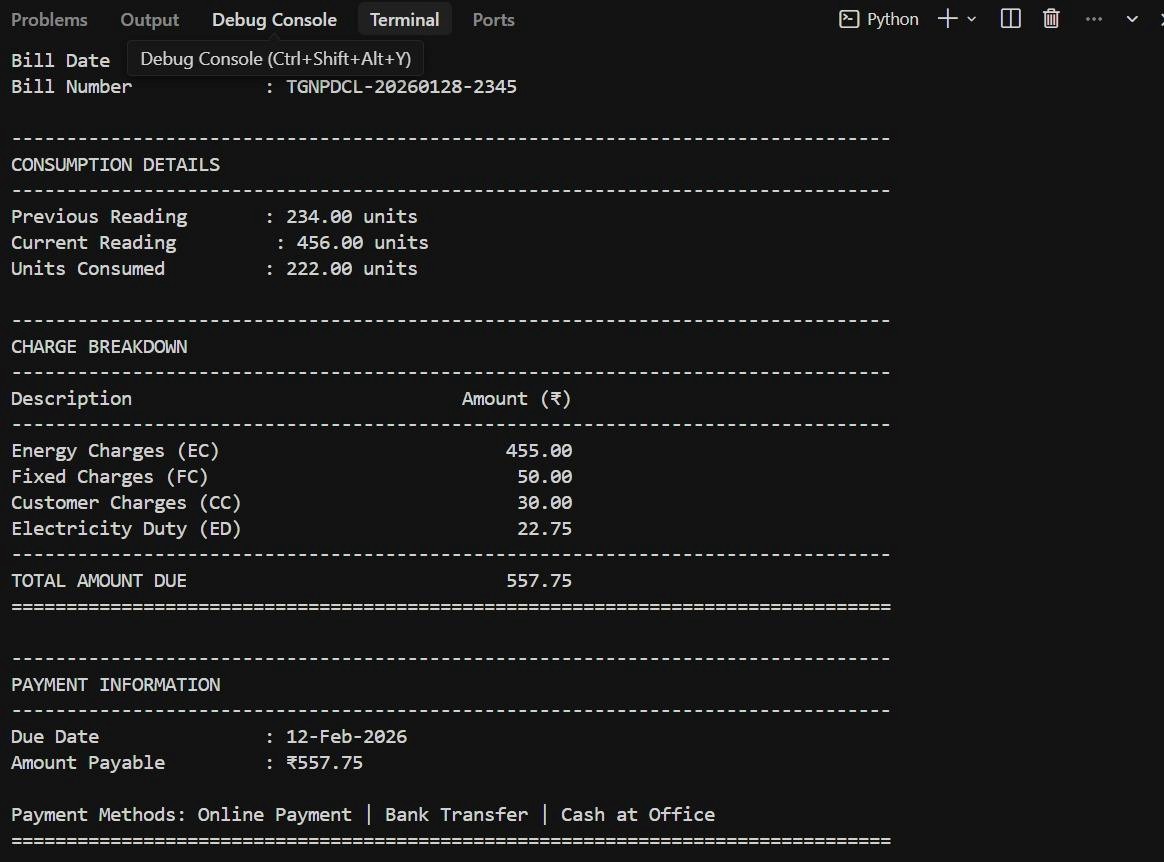


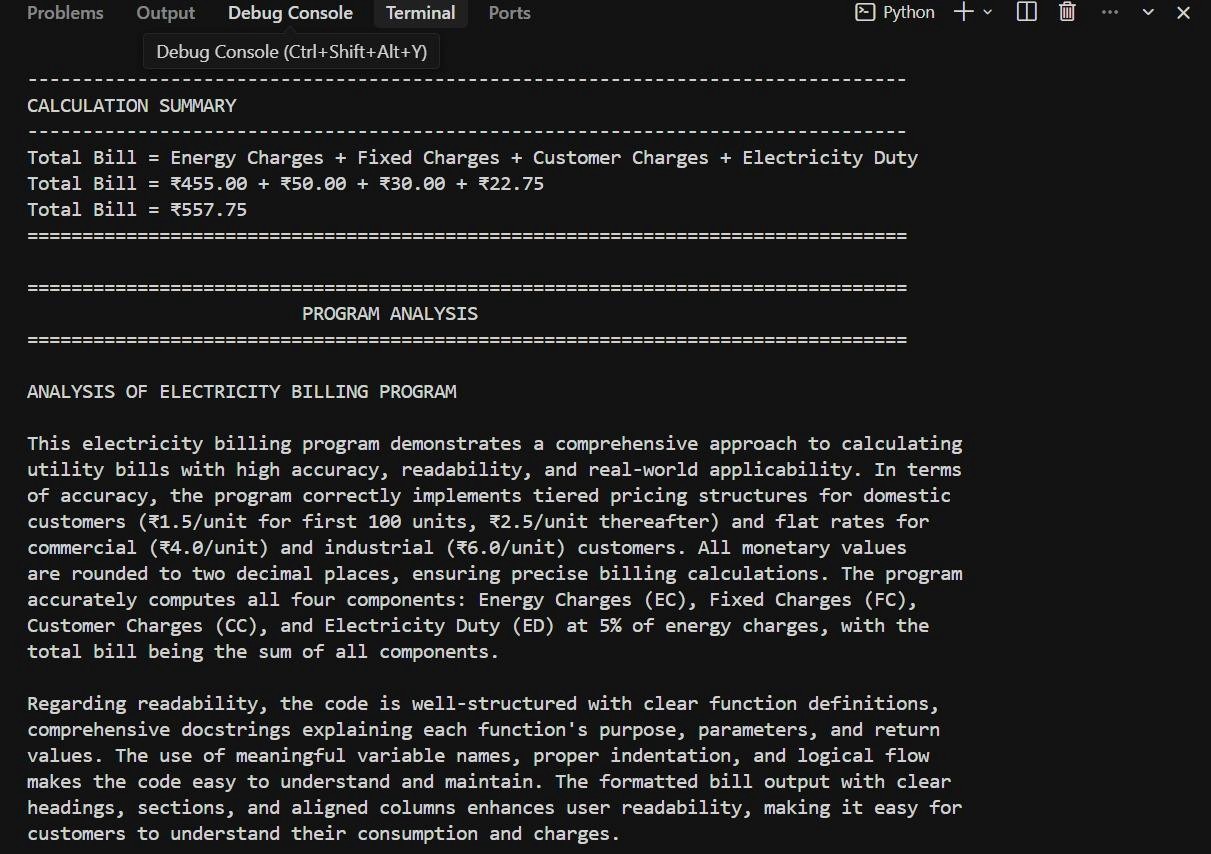




**Output:**







**Justification:**

The final task integrates all calculated components into a complete electricity bill. Clear formatting improves user interpretation and professionalism of output. Accurate aggregation ensures reliable total bill computation. The program reflects a real-time electricity billing application. Overall, it demonstrates practical applicability, correctness, and structured coding practice.