ASSIGNMENT – 3.3

Name: K.Rishitha

Batch: 23

HT No: 2303A51680

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)

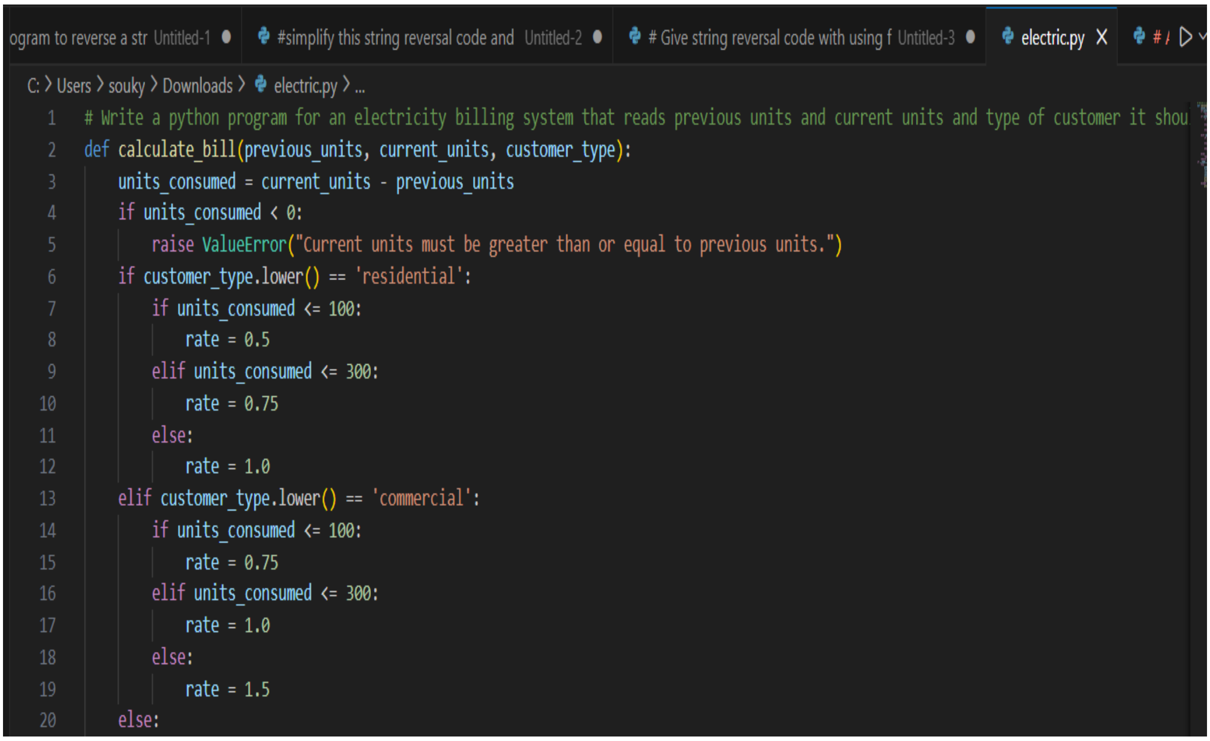
Expected Output

• Correct input reading

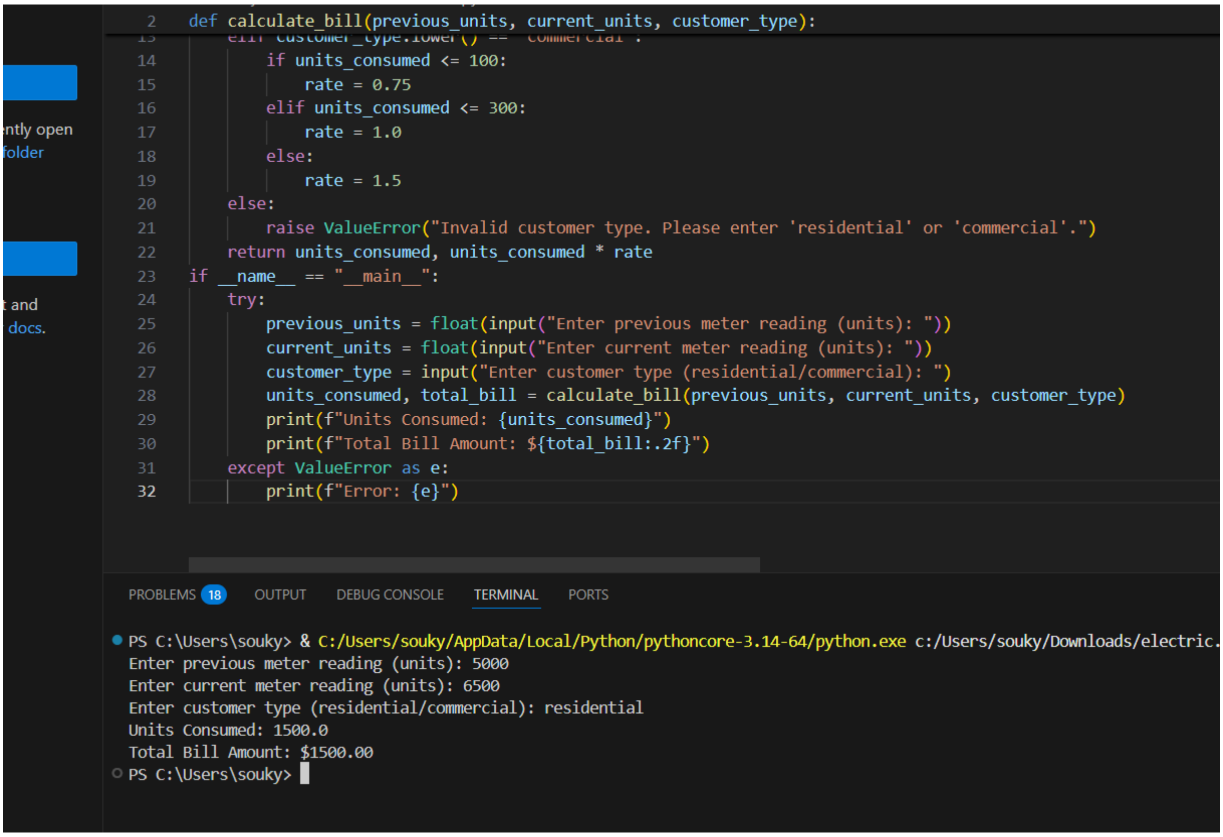
• Units consumed calculation

• Screenshot showing AI-generated code

• Sample input and output


ogram to reverse a str Untitled-1
#simplify this string reversal code and Untitled-2
# Give string reversal code with using f Untitled-3
electric.py
C: > Users > souky > Downloads > electric.py > ...

```python
# Write a python program for an electricity billing system that reads previous units and current units and type of customer it shou
def calculate_bill(previous_units, current_units, customer_type):
    units_consumed = current_units - previous_units
    if units_consumed < 0:
        raise ValueError("Current units must be greater than or equal to previous units.")
    if customer_type.lower() == 'residential':
        if units_consumed <= 100:
            rate = 0.5
        elif units_consumed <= 300:
            rate = 0.75
        else:
            rate = 1.0
    elif customer_type.lower() == 'commercial':
        if units_consumed <= 100:
            rate = 0.75
        elif units_consumed <= 300:
            rate = 1.0
        else:
            rate = 1.5
    else:
```



```python
def calculate_bill(previous_units, current_units, customer_type):
    elif customer_type.lower() == 'commercial':
        if units_consumed <= 100:
            rate = 0.75
        elif units_consumed <= 300:
            rate = 1.0
        else:
            rate = 1.5
    else:
        raise ValueError("Invalid customer type. Please enter 'residential' or 'commercial'.")
    return units_consumed, units_consumed * rate
if __name__ == "__main__":
    try:
        previous_units = float(input("Enter previous meter reading (units): "))
        current_units = float(input("Enter current meter reading (units): "))
        customer_type = input("Enter customer type (residential/commercial): ")
        units_consumed, total_bill = calculate_bill(previous_units, current_units, customer_type)
        print(f"Units Consumed: {units_consumed}")
        print(f"Total Bill Amount: ${total_bill:.2f}")
    except ValueError as e:
        print(f"Error: {e}")
```

```
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-64/python.exe c:/Users/souky/Downloads/electric.
Enter previous meter reading (units): 5000
Enter current meter reading (units): 6500
Enter customer type (residential/commercial): residential
Units Consumed: 1500.0
Total Bill Amount: $1500.00
PS C:\Users\souky>
```

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

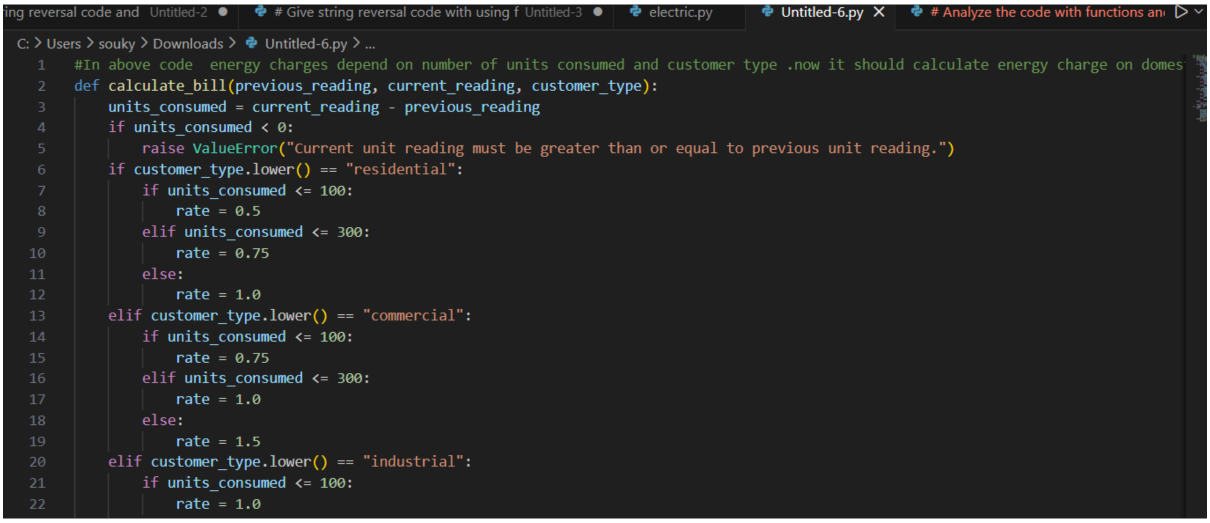
- Improve readability using AI prompts such as:

o “Simplify energy charge calculation logic”

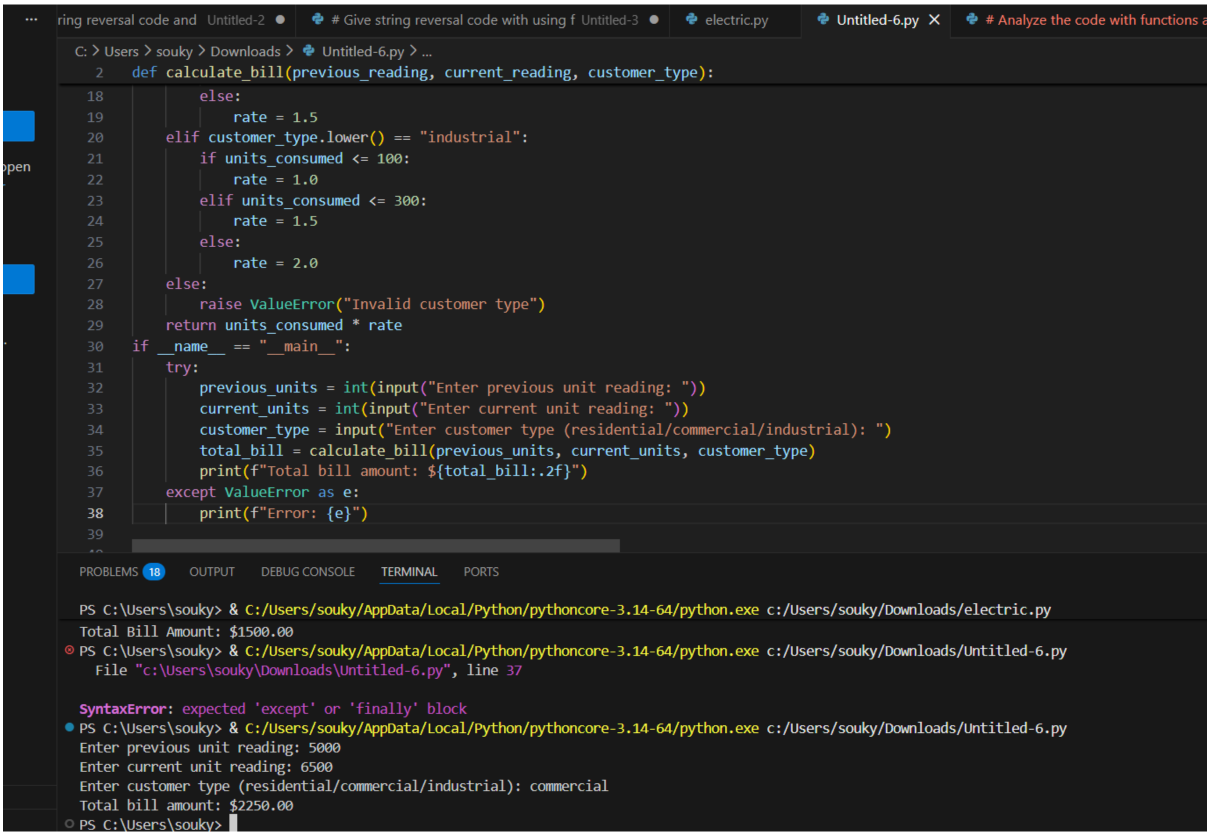
o “Optimize conditional statements”

Expected Output

- Correct EC calculation
- Clear conditional logic
- Original and improved versions


ing reversal code and Untitled-2
# Give string reversal code with using f Untitled-3
electric.py
Untitled-6.py
# Analyze the code with functions an
C: > Users > souky > Downloads > Untitled-6.py > ...

```python
#In above code  energy charges depend on number of units consumed and customer type .now it should calculate energy charge on domes
def calculate_bill(previous_reading, current_reading, customer_type):
    units_consumed = current_reading - previous_reading
    if units_consumed < 0:
        raise ValueError("Current unit reading must be greater than or equal to previous unit reading.")
    if customer_type.lower() == "residential":
        if units_consumed <= 100:
            rate = 0.5
        elif units_consumed <= 300:
            rate = 0.75
        else:
            rate = 1.0
    elif customer_type.lower() == "commercial":
        if units_consumed <= 100:
            rate = 0.75
        elif units_consumed <= 300:
            rate = 1.0
        else:
            rate = 1.5
    elif customer_type.lower() == "industrial":
        if units_consumed <= 100:
            rate = 1.0
```


ring reversal code and Untitled-2
# Give string reversal code with using f Untitled-3
electric.py
Untitled-6.py
# Analyze the code with functions a
C: > Users > souky > Downloads > Untitled-6.py > ...
pen
PROBLEMS 18
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS

```python
def calculate_bill(previous_reading, current_reading, customer_type):
        else:
            rate = 1.5
    elif customer_type.lower() == "industrial":
        if units_consumed <= 100:
            rate = 1.0
        elif units_consumed <= 300:
            rate = 1.5
        else:
            rate = 2.0
    else:
        raise ValueError("Invalid customer type")
    return units_consumed * rate
if __name__ == "__main__":
    try:
        previous_units = int(input("Enter previous unit reading: "))
        current_units = int(input("Enter current unit reading: "))
        customer_type = input("Enter customer type (residential/commercial/industrial): ")
        total_bill = calculate_bill(previous_units, current_units, customer_type)
        print(f"Total bill amount: ${total_bill:.2f}")
    except ValueError as e:
        print(f"Error: {e}")
```

```
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-64/python.exe c:/Users/souky/Downloads/electric.py
Total Bill Amount: $1500.00
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-64/python.exe c:/Users/souky/Downloads/Untitled-6.py
  File "c:\Users\souky\Downloads\Untitled-6.py", line 37

SyntaxError: expected 'except' or 'finally' block
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-64/python.exe c:/Users/souky/Downloads/Untitled-6.py
Enter previous unit reading: 5000
Enter current unit reading: 6500
Enter customer type (residential/commercial/industrial): commercial
Total bill amount: $2250.00
PS C:\Users\souky>
```

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:

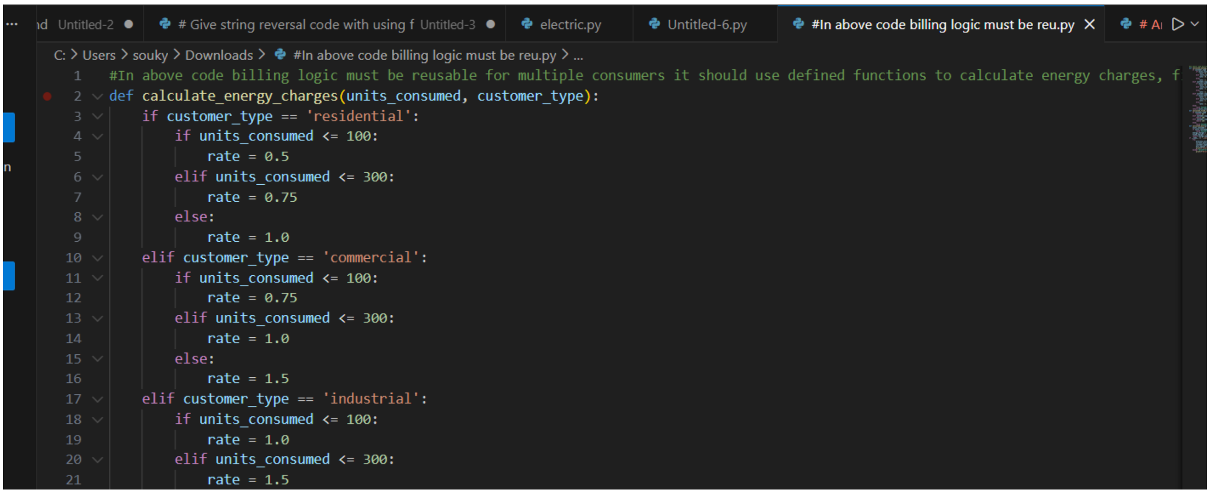
o Calculate Energy Charges

o Calculate Fixed Charges

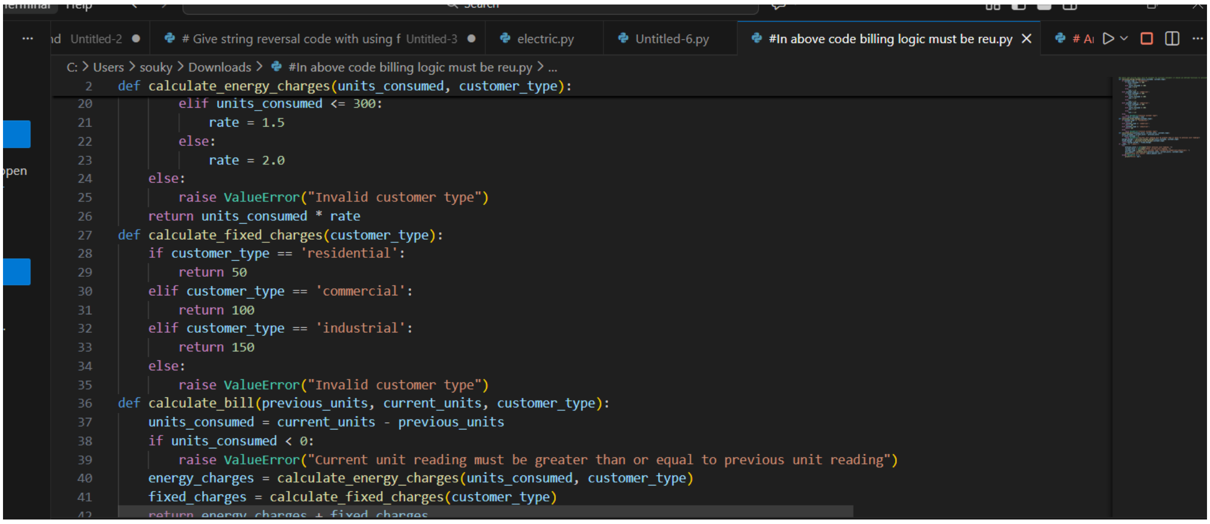
- Returns calculated values
- Includes meaningful comments

Expected Output

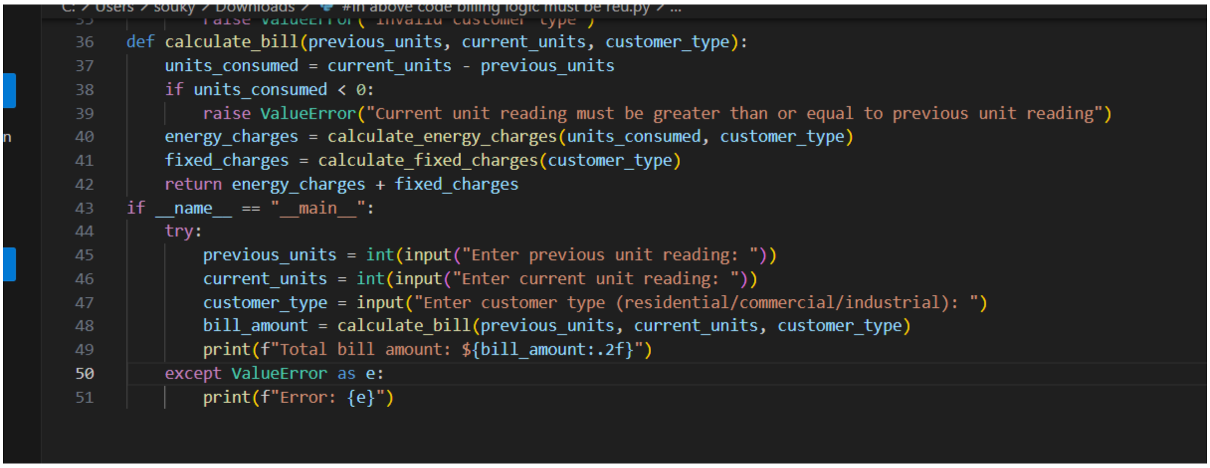
- Function-based Python program
- Correct EC and FC values
- Screenshots of AI-assisted function generation
- Test cases with outputs


Untitled-2
# Give string reversal code with using f Untitled-3
electric.py
Untitled-6.py
#In above code billing logic must be reu.py
# AI
C: > Users > souky > Downloads > #In above code billing logic must be reu.py > ...

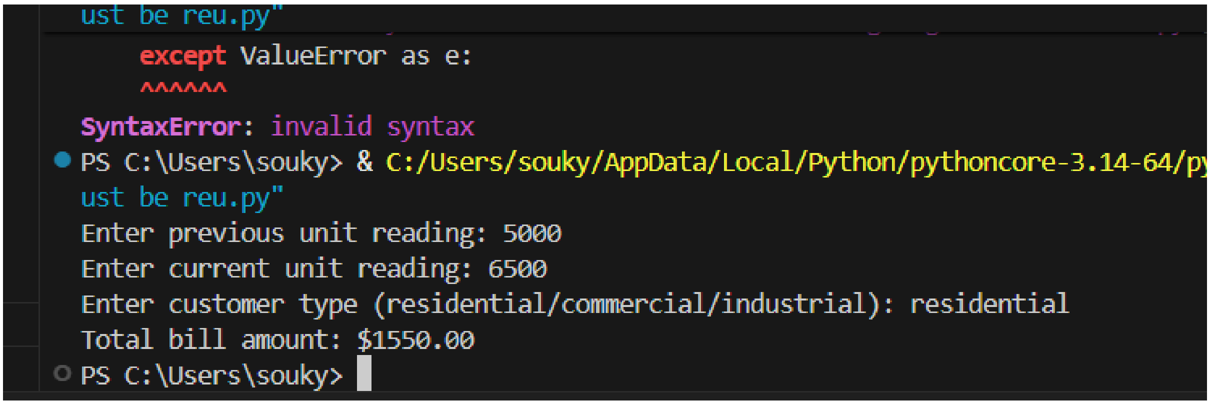
```python
#In above code billing logic must be reusable for multiple consumers it should use defined functions to calculate energy charges, f
def calculate_energy_charges(units_consumed, customer_type):
    if customer_type == 'residential':
        if units_consumed <= 100:
            rate = 0.5
        elif units_consumed <= 300:
            rate = 0.75
        else:
            rate = 1.0
    elif customer_type == 'commercial':
        if units_consumed <= 100:
            rate = 0.75
        elif units_consumed <= 300:
            rate = 1.0
        else:
            rate = 1.5
    elif customer_type == 'industrial':
        if units_consumed <= 100:
            rate = 1.0
        elif units_consumed <= 300:
            rate = 1.5
```


Untitled-2
# Give string reversal code with using f Untitled-3
electric.py
Untitled-6.py
#In above code billing logic must be reu.py
# AI
C: > Users > souky > Downloads > #In above code billing logic must be reu.py > ...

```python
def calculate_energy_charges(units_consumed, customer_type):
        elif units_consumed <= 300:
            rate = 1.5
        else:
            rate = 2.0
    else:
        raise ValueError("Invalid customer type")
    return units_consumed * rate
def calculate_fixed_charges(customer_type):
    if customer_type == 'residential':
        return 50
    elif customer_type == 'commercial':
        return 100
    elif customer_type == 'industrial':
        return 150
    else:
        raise ValueError("Invalid customer type")
def calculate_bill(previous_units, current_units, customer_type):
    units_consumed = current_units - previous_units
    if units_consumed < 0:
        raise ValueError("Current unit reading must be greater than or equal to previous unit reading")
    energy_charges = calculate_energy_charges(units_consumed, customer_type)
    fixed_charges = calculate_fixed_charges(customer_type)
    return energy_charges + fixed_charges
```


C: > Users > souky > Downloads > #In above code billing logic must be reu.py > ...

```python
        raise ValueError("Invalid customer type")
def calculate_bill(previous_units, current_units, customer_type):
    units_consumed = current_units - previous_units
    if units_consumed < 0:
        raise ValueError("Current unit reading must be greater than or equal to previous unit reading")
    energy_charges = calculate_energy_charges(units_consumed, customer_type)
    fixed_charges = calculate_fixed_charges(customer_type)
    return energy_charges + fixed_charges
if __name__ == "__main__":
    try:
        previous_units = int(input("Enter previous unit reading: "))
        current_units = int(input("Enter current unit reading: "))
        customer_type = input("Enter customer type (residential/commercial/industrial): ")
        bill_amount = calculate_bill(previous_units, current_units, customer_type)
        print(f"Total bill amount: ${bill_amount:.2f}")
    except ValueError as e:
        print(f"Error: {e}")
```



```
ust be reu.py"
    except ValueError as e:
    ^^^^^^

SyntaxError: invalid syntax
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-64/py
ust be reu.py"
Enter previous unit reading: 5000
Enter current unit reading: 6500
Enter customer type (residential/commercial/industrial): residential
Total bill amount: $1550.00
PS C:\Users\souky> 
```

Task 4: Calculation of Additional Charges

Scenario

Electricity bills include multiple additional charges.

Task Description

Extend the program to calculate:

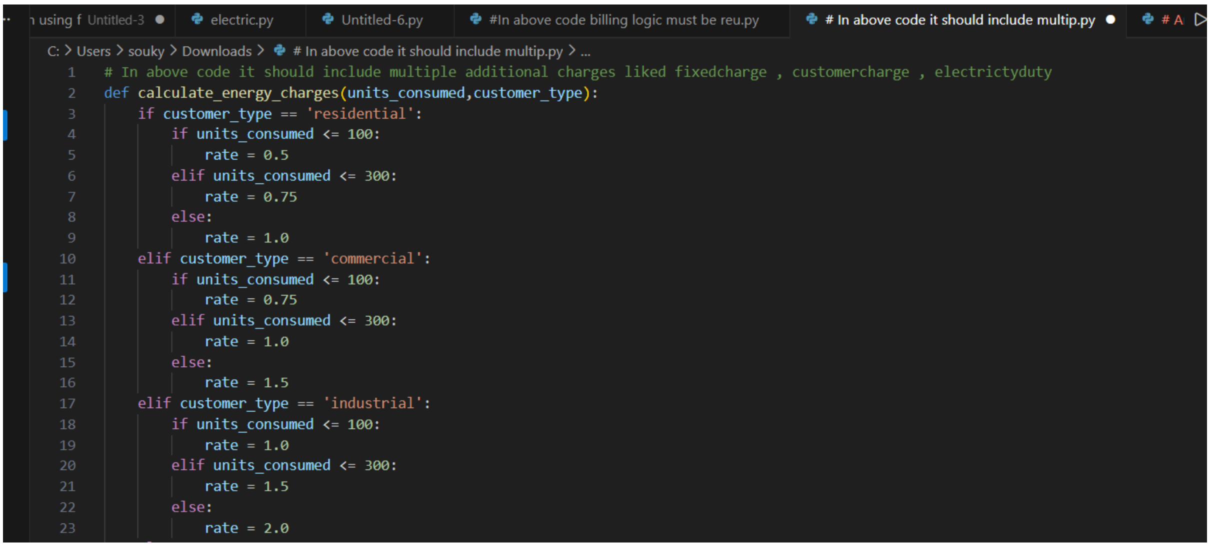
- FC – Fixed Charges
- CC – Customer Charges
- ED – Electricity Duty (percentage of EC)

Use AI prompts like:

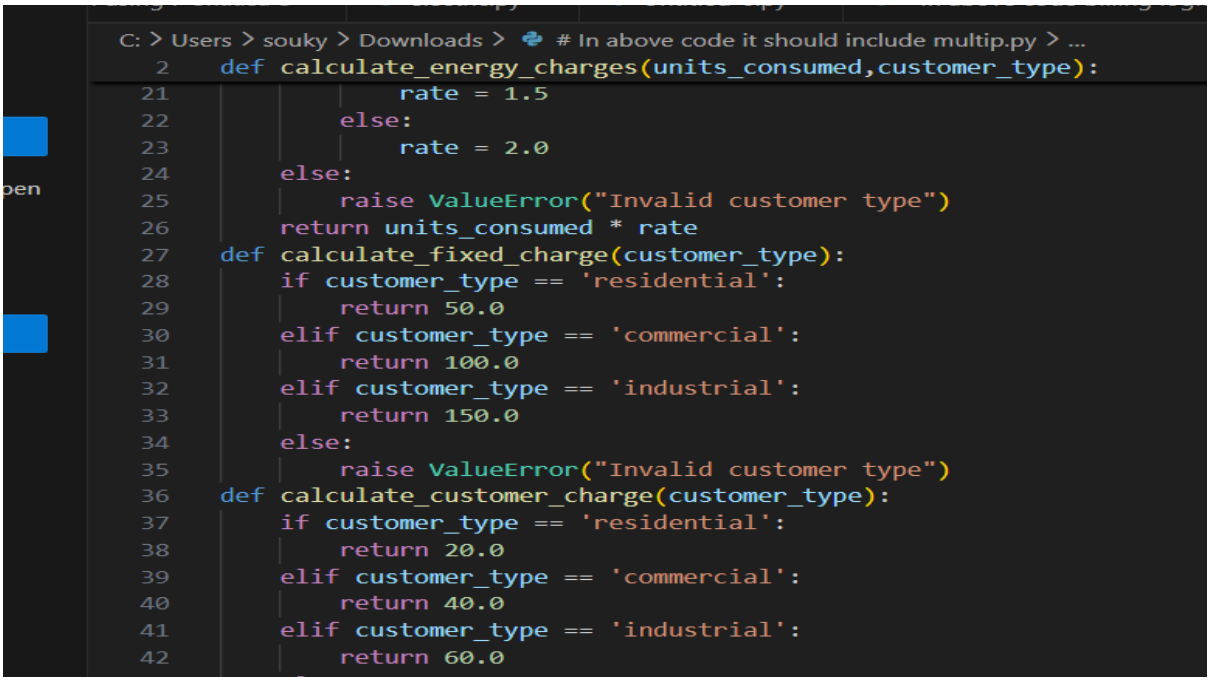
- “Add electricity duty calculation”
- “Improve billing accuracy”

Expected Output

- Individual charge values printed
- Correct duty calculation
- Well-structured output
- Verified intermediate results

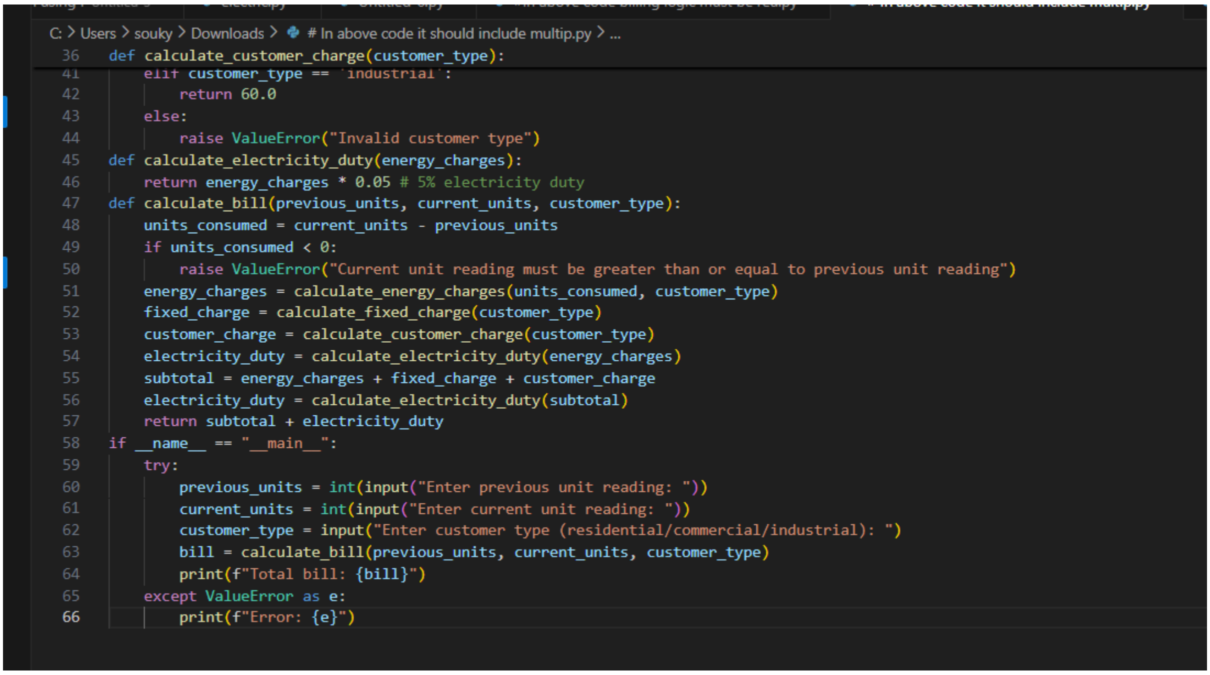

n using f Untitled-3 ●
electric.py
Untitled-6.py
#In above code billing logic must be reu.py
# In above code it should include multip.py ●
# A
C: > Users > souky > Downloads > # In above code it should include multip.py > ...

```
# In above code it should include multiple additional charges liked fixedcharge , customercharge , electrictyduty
def calculate_energy_charges(units_consumed,customer_type):
    if customer_type == 'residential':
        if units_consumed <= 100:
            rate = 0.5
        elif units_consumed <= 300:
            rate = 0.75
        else:
            rate = 1.0
    elif customer_type == 'commercial':
        if units_consumed <= 100:
            rate = 0.75
        elif units_consumed <= 300:
            rate = 1.0
        else:
            rate = 1.5
    elif customer_type == 'industrial':
        if units_consumed <= 100:
            rate = 1.0
        elif units_consumed <= 300:
            rate = 1.5
        else:
            rate = 2.0
```



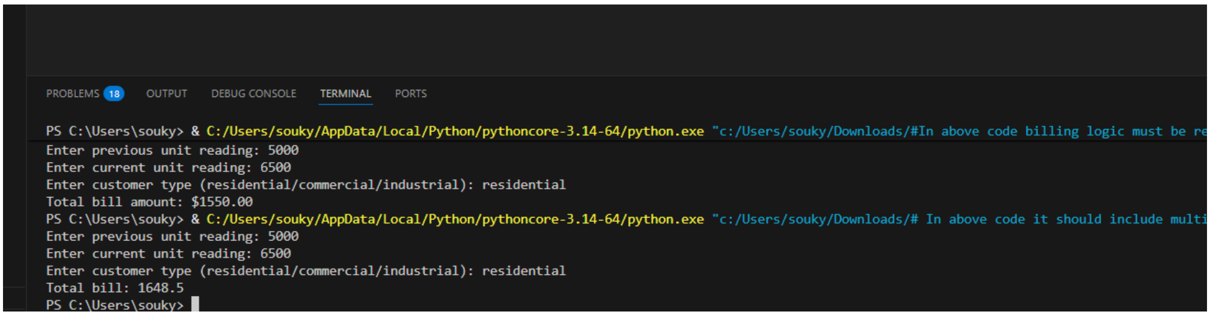
C: > Users > souky > Downloads > # In above code it should include multip.py > ...

```python
def calculate_energy_charges(units_consumed,customer_type):
            rate = 1.5
        else:
            rate = 2.0
    else:
        raise ValueError("Invalid customer type")
    return units_consumed * rate
def calculate_fixed_charge(customer_type):
    if customer_type == 'residential':
        return 50.0
    elif customer_type == 'commercial':
        return 100.0
    elif customer_type == 'industrial':
        return 150.0
    else:
        raise ValueError("Invalid customer type")
def calculate_customer_charge(customer_type):
    if customer_type == 'residential':
        return 20.0
    elif customer_type == 'commercial':
        return 40.0
    elif customer_type == 'industrial':
        return 60.0
```



C: > Users > souky > Downloads > # In above code it should include multip.py > ...

```python
def calculate_customer_charge(customer_type):
    elif customer_type == 'industrial':
        return 60.0
    else:
        raise ValueError("Invalid customer type")
def calculate_electricity_duty(energy_charges):
    return energy_charges * 0.05 # 5% electricity duty
def calculate_bill(previous_units, current_units, customer_type):
    units_consumed = current_units - previous_units
    if units_consumed < 0:
        raise ValueError("Current unit reading must be greater than or equal to previous unit reading")
    energy_charges = calculate_energy_charges(units_consumed, customer_type)
    fixed_charge = calculate_fixed_charge(customer_type)
    customer_charge = calculate_customer_charge(customer_type)
    electricity_duty = calculate_electricity_duty(energy_charges)
    subtotal = energy_charges + fixed_charge + customer_charge
    electricity_duty = calculate_electricity_duty(subtotal)
    return subtotal + electricity_duty
if __name__ == "__main__":
    try:
        previous_units = int(input("Enter previous unit reading: "))
        current_units = int(input("Enter current unit reading: "))
        customer_type = input("Enter customer type (residential/commercial/industrial): ")
        bill = calculate_bill(previous_units, current_units, customer_type)
        print(f"Total bill: {bill}")
    except ValueError as e:
        print(f"Error: {e}")
```



```
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/Users/souky/Downloads/#In above code billing logic must be re
Enter previous unit reading: 5000
Enter current unit reading: 6500
Enter customer type (residential/commercial/industrial): residential
Total bill amount: $1550.00
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/Users/souky/Downloads/# In above code it should include multi
Enter previous unit reading: 5000
Enter current unit reading: 6500
Enter customer type (residential/commercial/industrial): residential
Total bill: 1648.5
PS C:\Users\souky>
```

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:

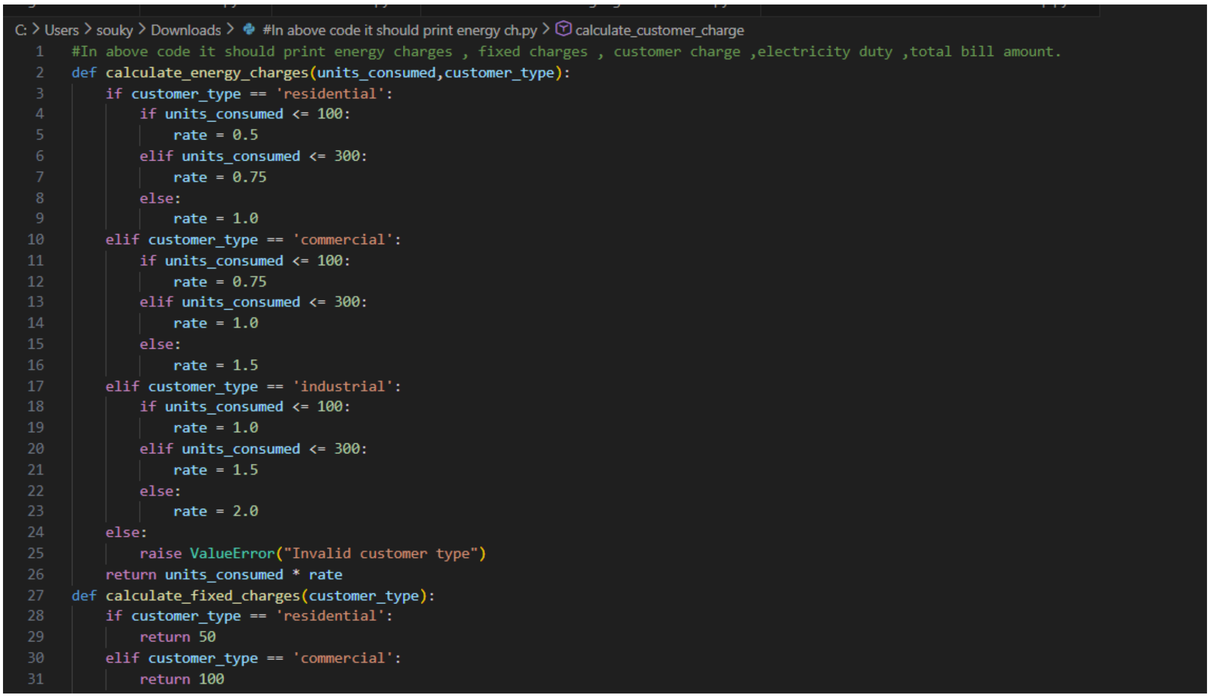
o Accuracy

o Readability

o Real-world applicability

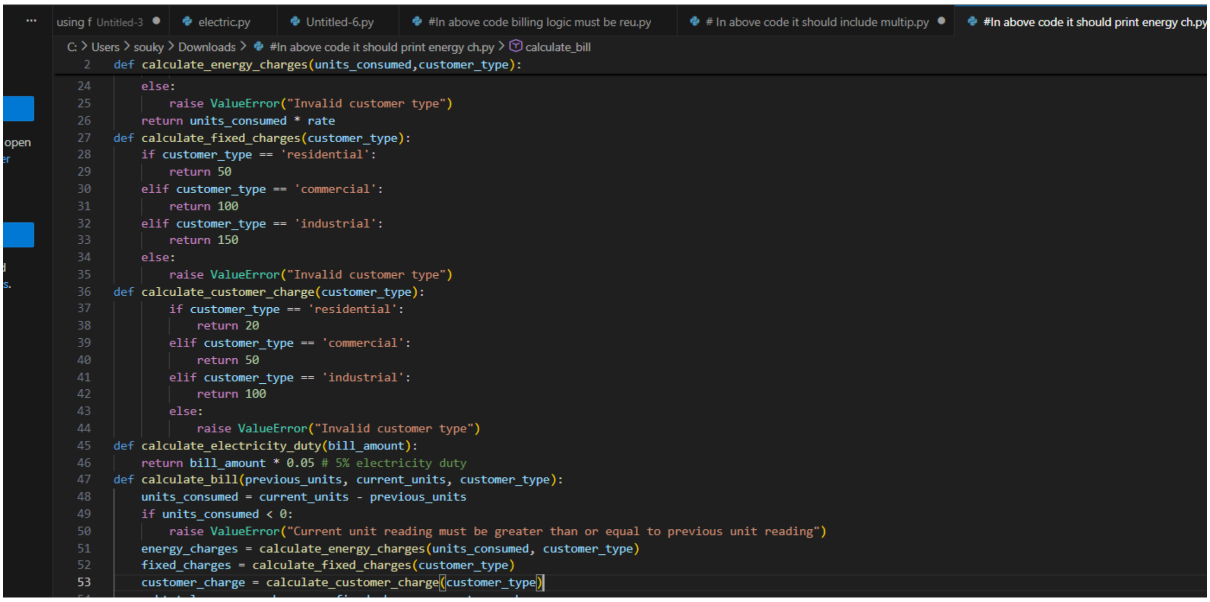
Expected Output

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

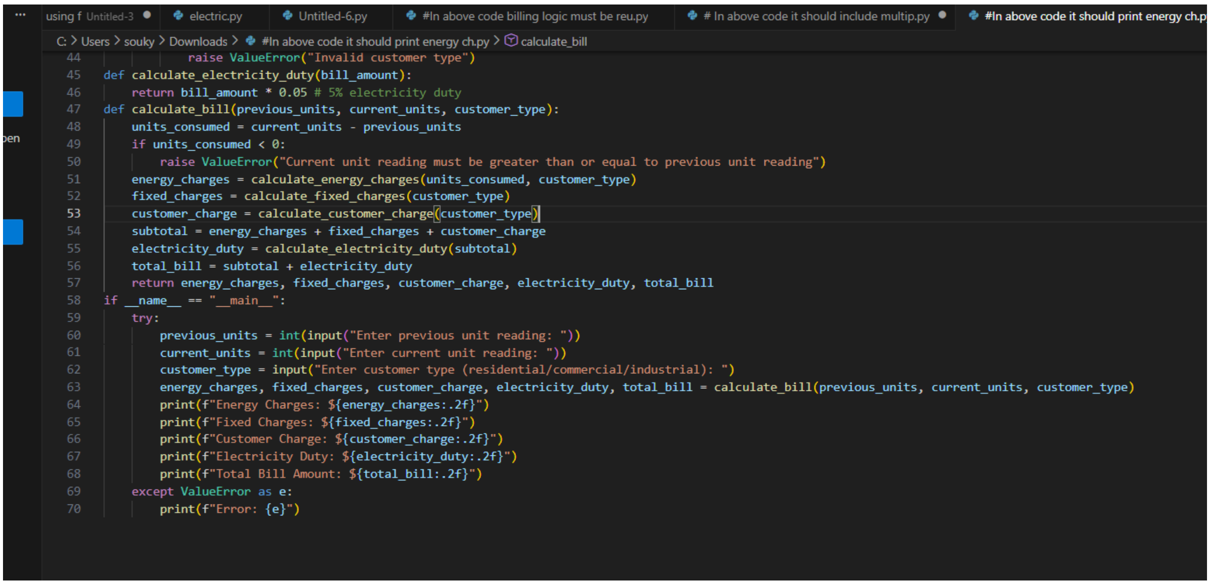


C: > Users > souky > Downloads > #In above code it should print energy ch.py > calculate_customer_charge

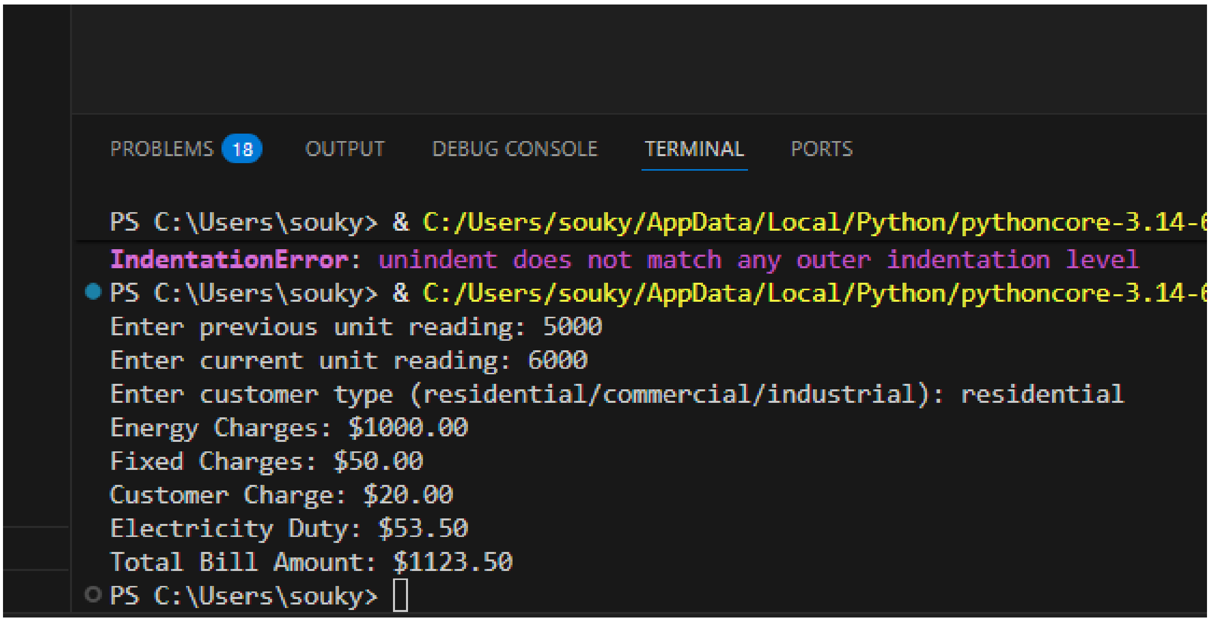
```python
#In above code it should print energy charges , fixed charges , customer charge ,electricity duty ,total bill amount.
def calculate_energy_charges(units_consumed,customer_type):
    if customer_type == 'residential':
        if units_consumed <= 100:
            rate = 0.5
        elif units_consumed <= 300:
            rate = 0.75
        else:
            rate = 1.0
    elif customer_type == 'commercial':
        if units_consumed <= 100:
            rate = 0.75
        elif units_consumed <= 300:
            rate = 1.0
        else:
            rate = 1.5
    elif customer_type == 'industrial':
        if units_consumed <= 100:
            rate = 1.0
        elif units_consumed <= 300:
            rate = 1.5
        else:
            rate = 2.0
    else:
        raise ValueError("Invalid customer type")
    return units_consumed * rate
def calculate_fixed_charges(customer_type):
    if customer_type == 'residential':
        return 50
    elif customer_type == 'commercial':
        return 100
```


using f Untitled-3
electric.py
Untitled-6.py
#In above code billing logic must be reu.py
# In above code it should include multip.py
#In above code it should print energy ch.py
C: > Users > souky > Downloads > #In above code it should print energy ch.py > calculate_bill

```python
def calculate_energy_charges(units_consumed,customer_type):
    else:
        raise ValueError("Invalid customer type")
    return units_consumed * rate
def calculate_fixed_charges(customer_type):
    if customer_type == 'residential':
        return 50
    elif customer_type == 'commercial':
        return 100
    elif customer_type == 'industrial':
        return 150
    else:
        raise ValueError("Invalid customer type")
def calculate_customer_charge(customer_type):
        if customer_type == 'residential':
            return 20
        elif customer_type == 'commercial':
            return 50
        elif customer_type == 'industrial':
            return 100
        else:
            raise ValueError("Invalid customer type")
def calculate_electricity_duty(bill_amount):
    return bill_amount * 0.05 # 5% electricity duty
def calculate_bill(previous_units, current_units, customer_type):
    units_consumed = current_units - previous_units
    if units_consumed < 0:
        raise ValueError("Current unit reading must be greater than or equal to previous unit reading")
    energy_charges = calculate_energy_charges(units_consumed, customer_type)
    fixed_charges = calculate_fixed_charges(customer_type)
    customer_charge = calculate_customer_charge(customer_type)
```


using f Untitled-3
electric.py
Untitled-6.py
#In above code billing logic must be reu.py
# In above code it should include multip.py
#In above code it should print energy ch.py
C: > Users > souky > Downloads > #In above code it should print energy ch.py > calculate_bill

```python
            raise ValueError("Invalid customer type")
def calculate_electricity_duty(bill_amount):
    return bill_amount * 0.05 # 5% electricity duty
def calculate_bill(previous_units, current_units, customer_type):
    units_consumed = current_units - previous_units
    if units_consumed < 0:
        raise ValueError("Current unit reading must be greater than or equal to previous unit reading")
    energy_charges = calculate_energy_charges(units_consumed, customer_type)
    fixed_charges = calculate_fixed_charges(customer_type)
    customer_charge = calculate_customer_charge(customer_type)
    subtotal = energy_charges + fixed_charges + customer_charge
    electricity_duty = calculate_electricity_duty(subtotal)
    total_bill = subtotal + electricity_duty
    return energy_charges, fixed_charges, customer_charge, electricity_duty, total_bill
if __name__ == "__main__":
    try:
        previous_units = int(input("Enter previous unit reading: "))
        current_units = int(input("Enter current unit reading: "))
        customer_type = input("Enter customer type (residential/commercial/industrial): ")
        energy_charges, fixed_charges, customer_charge, electricity_duty, total_bill = calculate_bill(previous_units, current_units, customer_type)
        print(f"Energy Charges: ${energy_charges:.2f}")
        print(f"Fixed Charges: ${fixed_charges:.2f}")
        print(f"Customer Charge: ${customer_charge:.2f}")
        print(f"Electricity Duty: ${electricity_duty:.2f}")
        print(f"Total Bill Amount: ${total_bill:.2f}")
    except ValueError as e:
        print(f"Error: {e}")
```



```
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-
IndentationError: unindent does not match any outer indentation level
PS C:\Users\souky> & C:/Users/souky/AppData/Local/Python/pythoncore-3.14-
Enter previous unit reading: 5000
Enter current unit reading: 6000
Enter customer type (residential/commercial/industrial): residential
Energy Charges: $1000.00
Fixed Charges: $50.00
Customer Charge: $20.00
Electricity Duty: $53.50
Total Bill Amount: $1123.50
PS C:\Users\souky> 
```