

ASSIGNMENT-3.3

2303A51546

BATCH-10

❖ TASK-1:

PROMPT:

Write a Python program to calculate the electricity bill using previous and current meter readings.

The program should ask for customer type (Domestic, Commercial, or Industrial).

Calculate units consumed and apply different rates based on unit slabs and customer type.

Finally, display units consumed and total bill amount, and show an error for invalid input.

CODE:

```
previous_units = float(input("Enter Previous Units (PU): "))
current_units = float(input("Enter Current Units (CU): "))
customer_type = input("Enter Type of Customer (Domestic/Commercial): ").strip().lower()
# Calculate units consumed
units_consumed = current_units - previous_units
# Initialize bill amount
bill_amount = 0.0
# Calculate bill based on customer type
if customer_type == "domestic":
    if units_consumed <= 100:
        bill_amount = units_consumed * 1.5
    elif units_consumed <= 300:
        bill_amount = (100 * 1.5) + (units_consumed - 100) * 2.5
    else:
        bill_amount = (100 * 1.5) + (200 * 2.5) + (units_consumed - 300) * 4.0
elif customer_type == "commercial":
    if units_consumed <= 100:
        bill_amount = units_consumed * 2.0
    elif units_consumed <= 300:
        bill_amount = (100 * 2.0) + (units_consumed - 100) * 4.0
    else:
        bill_amount = (100 * 2.0) + (200 * 4.0) + (units_consumed - 300) * 6.0
else:
    print("Invalid customer type. Please enter 'Domestic' or 'Commercial'.")
```

```

exit()
# Display the bill amount
print(f"Units Consumed: {units_consumed}")
print(f"Bill Amount: ${bill_amount:.2f}")

```

OUTPUT:

```

day3.py
13 # calculate units consumed
14 units_consumed = current_units - previous_units
15 # initialize bill amount
16 bill_amount = 0.0
17 # Calculate bill based on customer type
18 if customer_type == "domestic":
19     if units_consumed <= 100:
20         bill_amount = units_consumed * 1.5
21     elif units_consumed <= 300:
22         bill_amount = (100 * 1.5) + (units_consumed - 100) * 2.5
23     else:
24         bill_amount = (100 * 1.5) + (200 * 2.5) + (units_consumed - 300) * 4.0
25 elif customer_type == "commercial":
26     if units_consumed <= 100:
27         bill_amount = units_consumed * 2.0
28     elif units_consumed <= 300:
29         bill_amount = (100 * 2.0) + (units_consumed - 100) * 4.0
30     else:
31         bill_amount = (100 * 2.0) + (200 * 4.0) + (units_consumed - 300) * 6.0
32 else:
33     print("Invalid customer type. Please enter 'Domestic' or 'Commercial'.")
34     exit()
35 # display the bill amount
36 print(f"Units Consumed: {units_consumed}")
37 print(f"Bill Amount: ${bill_amount:.2f}")
38
39
40

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GIT LENS

```

Enter Previous Units (PU): 45
Enter Current Units (CU): 50
Enter Type of Customer (Domestic/Commercial): DOMESTIC
Units Consumed: 5.0
Bill Amount: $7.50
PS C:\Users\sriva\OneDrive\Desktop\AI Assisted>

```

Ln 22, Col 65 Spaces: 4 UTF-8 CRLF Python 3.13.7 (venv) 9-0 Go Live

EXPLANATION:

This task focuses on collecting correct consumer details required for electricity billing.

The program reads previous units, current units, and consumer type from the user.

Units consumed are calculated using meter readings, which is the base for billing.

This step ensures accurate input handling for further calculations.

❖ TASK-2

PROMPT:

Create a Python program that calculates electricity charges depending on how many units a customer used and their category (Domestic, Commercial, Industrial).

Use conditional statements to apply different tariff rates.

Print the total bill clearly and explain how the calculation works.

CODE:

```
previous_units = float(input("Enter Previous Units (PU): "))
```

```

current_units = float(input("Enter Current Units (CU): "))
customer_type = input("Enter Type of Customer
(Domestic/Commercial/Industrial): ").strip().lower()
# Calculate units consumed
units_consumed = current_units - previous_units
# Initialize bill amount
bill_amount = 0.0
# Calculate bill based on customer type
if customer_type == "domestic":
    if units_consumed <= 100:
        bill_amount = units_consumed * 1.5
    elif units_consumed <= 300:
        bill_amount = (100 * 1.5) + (units_consumed - 100) * 2.5
    else:
        bill_amount = (100 * 1.5) + (200 * 2.5) + (units_consumed - 300) * 4.0
elif customer_type == "commercial":
    if units_consumed <= 100:
        bill_amount = units_consumed * 2.0
    elif units_consumed <= 300:
        bill_amount = (100 * 2.0) + (units_consumed - 100) * 4.0
    else:
        bill_amount = (100 * 2.0) + (200 * 4.0) + (units_consumed - 300) * 6.0
elif customer_type == "industrial":
    if units_consumed <= 100:
        bill_amount = units_consumed * 2.5
    elif units_consumed <= 300:
        bill_amount = (100 * 2.5) + (units_consumed - 100) * 5.0
    else:
        bill_amount = (100 * 2.5) + (200 * 5.0) + (units_consumed - 300) * 7.0
else:
    print("Invalid customer type. Please enter 'Domestic', 'Commercial', or
'Industrial'.")
exit()
# Display the bill amount
print(f"Units Consumed: {units_consumed}")
print(f"Bill Amount: ${bill_amount:.2f}")

```

OUTPUT:

```
64 units_consumed = current_units - previous_units
65 # Initialize bill amount
66 bill_amount = 0.0
67 # calculate bill based on customer type
68 if customer_type == "domestic":
69     if units_consumed <= 100:
70         bill_amount = units_consumed * 1.5
71     elif units_consumed <= 300:
72         bill_amount = (100 * 1.5) + (units_consumed - 100) * 2.5
73     else:
74         bill_amount = (100 * 1.5) + (200 * 2.5) + (units_consumed - 300) * 4.0
75 elif customer_type == "commercial":
76     if units_consumed <= 100:
77         bill_amount = units_consumed * 2.0
78     elif units_consumed <= 300:
79         bill_amount = (100 * 2.0) + (units_consumed - 100) * 4.0
80     else:
81         bill_amount = (100 * 2.0) + (200 * 4.0) + (units_consumed - 300) * 6.0
82 elif customer_type == "industrial":
83     if units_consumed <= 100:
84         bill_amount = units_consumed * 2.5
85     elif units_consumed <= 300:
86         bill_amount = (100 * 2.5) + (units_consumed - 100) * 5.0
87     else:
88         bill_amount = (100 * 2.5) + (200 * 5.0) + (units_consumed - 300) * 7.0
89 else:
90     print("Invalid customer type. Please enter 'Domestic', 'Commercial', or 'Industrial'.")
91     exit()
92 # Display the bill amount

Bill Amount: $15.00
Enter Previous Units (PU): 35
Enter Current Units (CU): 55
Enter Type of Customer (Domestic/Commercial/Industrial): industrial
Units Consumed: 20.0
Bill Amount: $50.00
```

EXPLANATION:

In this task, energy charges are calculated based on units consumed and consumer type.

Conditional logic is used to apply different tariff rates for domestic, commercial, and industrial users.

❖ TASK-3:

PROMPT:

Write a Python electricity billing program using functions.

Create separate functions to calculate Energy Charges and Fixed Charges based on customer type.

Take inputs, compute units consumed, and display all charges with proper labels.

Add comments, sample input/output, and at least three test cases.

CODE:

```
def calculate_energy_charges(units_consumed, customer_type):
    if customer_type == "domestic":
        if units_consumed <= 100:
            return units_consumed * 1.5
        elif units_consumed <= 300:
            return (100 * 1.5) + (units_consumed - 100) * 2.5
        else:
            return (100 * 1.5) + (200 * 2.5) + (units_consumed - 300) * 4.0
    elif customer_type == "commercial":
        if units_consumed <= 100:
            return units_consumed * 2.0
        elif units_consumed <= 300:
```

```

        return (100 * 2.0) + (units_consumed - 100) * 4.0
    else:
        return (100 * 2.0) + (200 * 4.0) + (units_consumed - 300) * 6.0
elif customer_type == "industrial":
    if units_consumed <= 100:
        return units_consumed * 2.5
    elif units_consumed <= 300:
        return (100 * 2.5) + (units_consumed - 100) * 5.0
    else:
        return (100 * 2.5) + (200 * 5.0) + (units_consumed - 300) * 7.0
else:
    return None

```

Function to calculate Fixed Charges

```

def calculate_fixed_charges(customer_type):
    if customer_type == "domestic":
        return 50.0
    elif customer_type == "commercial":
        return 100.0
    elif customer_type == "industrial":
        return 150.0
    else:
        return None

```

Main program

```

previous_units = float(input("Enter Previous Units (PU): "))
current_units = float(input("Enter Current Units (CU): "))
customer_type = input("Enter Type of Customer
(Domestic/Commercial/Industrial): ").strip().lower()

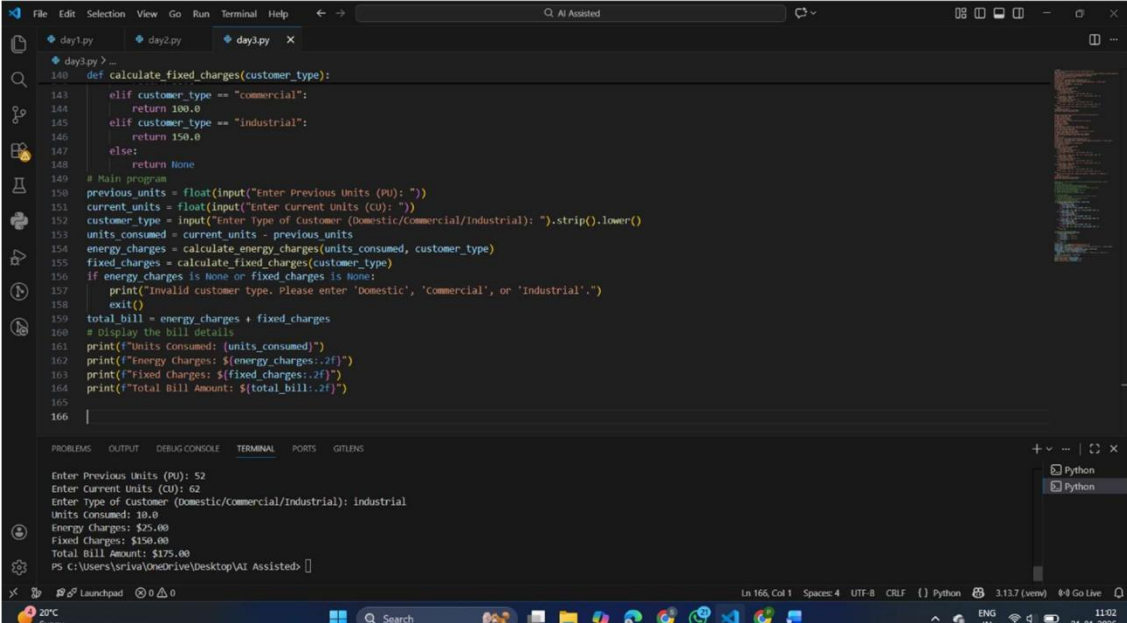
```

```

units_consumed = current_units - previous_units
energy_charges = calculate_energy_charges(units_consumed,
customer_type)
fixed_charges = calculate_fixed_charges(customer_type)
if energy_charges is None or fixed_charges is None:
    print("Invalid customer type. Please enter 'Domestic', 'Commercial', or
'Industrial'.")
    exit()
total_bill = energy_charges + fixed_charges
# Display the bill details
print(f"Units Consumed: {units_consumed}")
print(f"Energy Charges: ${energy_charges:.2f}")
print(f"Fixed Charges: ${fixed_charges:.2f}")
print(f"Total Bill Amount: ${total_bill:.2f}")

```

OUTPUT:



The screenshot shows a code editor with a Python script and its execution output in the terminal. The code defines two functions, `calculate_energy_charges` and `calculate_fixed_charges`, and a main program that takes user input, calculates the bill, and displays the details.

```

140 def calculate_fixed_charges(customer_type):
141     if customer_type == "commercial":
142         return 100.0
143     elif customer_type == "industrial":
144         return 150.0
145     else:
146         return None
147
148 # Main program
149 previous_units = float(input("Enter Previous Units (PU): "))
150 current_units = float(input("Enter current Units (CU): "))
151 customer_type = input("Enter Type of Customer (Domestic/commercial/Industrial): ").strip().lower()
152 units_consumed = current_units - previous_units
153 energy_charges = calculate_energy_charges(units_consumed, customer_type)
154 fixed_charges = calculate_fixed_charges(customer_type)
155 if energy_charges is None or fixed_charges is None:
156     print("Invalid customer type. Please enter 'Domestic', 'Commercial', or 'Industrial'.")
157     exit()
158 total_bill = energy_charges + fixed_charges
159 # Display the bill details
160 print(f"Units Consumed: {units_consumed}")
161 print(f"Energy Charges: ${energy_charges:.2f}")
162 print(f"Fixed Charges: ${fixed_charges:.2f}")
163 print(f"Total Bill Amount: ${total_bill:.2f}")
164
165
166

```

Terminal Output:

```

Enter Previous Units (PU): 52
Enter Current Units (CU): 62
Enter Type of Customer (Domestic/commercial/Industrial): Industrial
Units Consumed: 10.0
Energy Charges: $25.00
Fixed Charges: $150.00
Total Bill Amount: $175.00
PS C:\Users\sriya\OneDrive\Desktop\AI Assisted >

```

EXPLANATION:

introduces modular programming using user-defined methods. Separate methods are used to calculate energy charges and fixed charges. This makes the program reusable and easier to maintain. Modular design improves code structure and readability.

❖ TASK-4:

PROMPT:

Extend the electricity billing program by adding more charges like Fixed Charges and Electricity Duty. Electricity Duty should be calculated as a percentage of Energy Charges.

Display each charge separately and make the output clear and easy to read.

Ensure all calculations are correct.

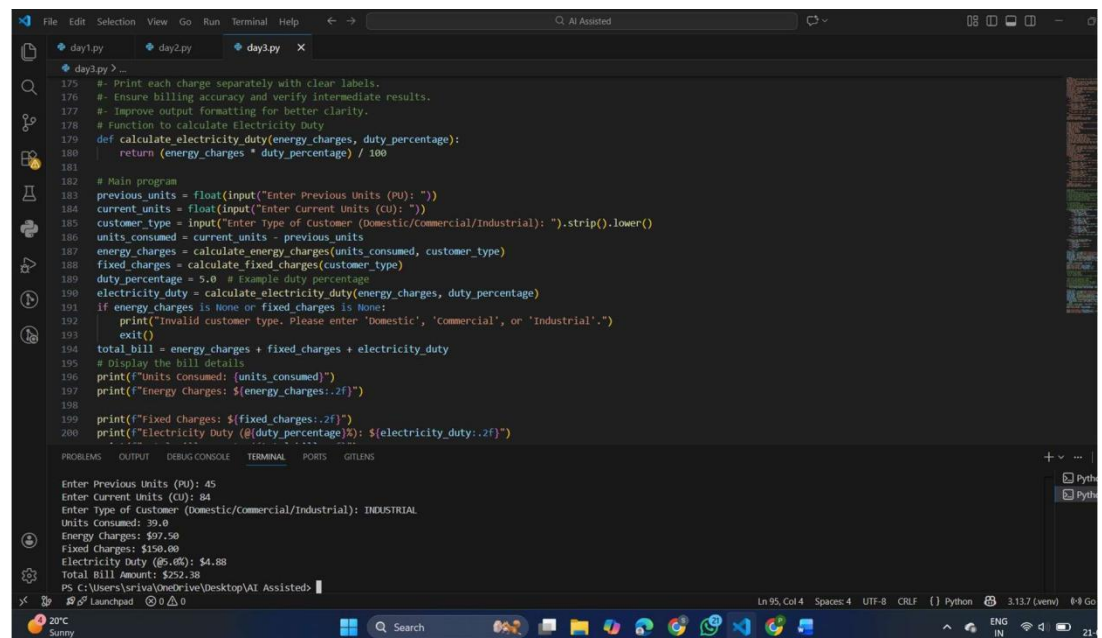
CODE:

```
def calculate_electricity_duty(energy_charges, duty_percentage):
    return (energy_charges * duty_percentage) / 100

# Main program
previous_units = float(input("Enter Previous Units (PU): "))
current_units = float(input("Enter Current Units (CU): "))
customer_type = input("Enter Type of Customer
(Domestic/Commercial/Industrial): ").strip().lower()
units_consumed = current_units - previous_units
energy_charges = calculate_energy_charges(units_consumed,
customer_type)
fixed_charges = calculate_fixed_charges(customer_type)
duty_percentage = 5.0 # Example duty percentage
electricity_duty = calculate_electricity_duty(energy_charges,
duty_percentage)
if energy_charges is None or fixed_charges is None:
    print("Invalid customer type. Please enter 'Domestic', 'Commercial',
or 'Industrial'.")
    exit()
total_bill = energy_charges + fixed_charges + electricity_duty
# Display the bill details
print(f"Units Consumed: {units_consumed}")
print(f"Energy Charges: ${energy_charges:.2f}")

print(f"Fixed Charges: ${fixed_charges:.2f}")
print(f"Electricity Duty (@{duty_percentage}%):
${electricity_duty:.2f}")
print(f"Total Bill Amount: ${total_bill:.2f}")
```

OUTPUT:



```
175 # Print each charge separately with clear labels.
176 # Ensure billing accuracy and verify intermediate results.
177 # Improve output formatting for better clarity.
178 # Function to calculate Electricity Duty
179 def calculate_electricity_duty(energy_charges, duty_percentage):
180     return (energy_charges * duty_percentage) / 100
181
182 # Main program
183 previous_units = float(input("Enter Previous Units (PU): "))
184 current_units = float(input("Enter Current Units (CU): "))
185 customer_type = input("Enter Type of Customer (Domestic/Commercial/Industrial): ").strip().lower()
186 units_consumed = current_units - previous_units
187 energy_charges = calculate_energy_charges(units_consumed, customer_type)
188 fixed_charges = calculate_fixed_charges(customer_type)
189 duty_percentage = 5.0 # Example duty percentage
190 electricity_duty = calculate_electricity_duty(energy_charges, duty_percentage)
191 if energy_charges is None or fixed_charges is None:
192     print("Invalid customer type. Please enter 'Domestic', 'Commercial', or 'Industrial'.")
193     exit()
194 total_bill = energy_charges + fixed_charges + electricity_duty
195 # Display the bill details
196 print(f"Units Consumed: {units_consumed}")
197 print(f"Energy Charges: ${energy_charges:.2f}")
198
199 print(f"Fixed Charges: ${fixed_charges:.2f}")
200 print(f"Electricity Duty (@{duty_percentage}%): ${electricity_duty:.2f}")
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

Enter Previous Units (PU): 45
Enter Current Units (CU): 84
Enter Type of Customer (Domestic/Commercial/Industrial): INDUSTRIAL
Units Consumed: 39.0
Energy Charges: \$92.50
Fixed Charges: \$150.00
Electricity Duty (@5.0%): \$4.88
Total Bill Amount: \$252.38

EXPLANATION:

This task extends billing by adding additional charges like fixed charges, customer charges, and electricity duty.

Electricity duty is calculated as a percentage of energy charges.

Printing individual charges helps verify calculation accuracy.

This step makes the bill more realistic and detailed.

❖ TASK-5:

PROMPT:

Create a complete electricity billing system in Python.

Calculate Energy Charges (EC), Fixed Charges (FC), Customer Charges (CC), and Electricity Duty (ED).

Find the total bill using all these values.

Display the final result in a neat format that looks like a real electricity bill.

CODE:

```
def calculate_customer_charges(customer_type):  
    if customer_type == "domestic":  
        return 20.0
```

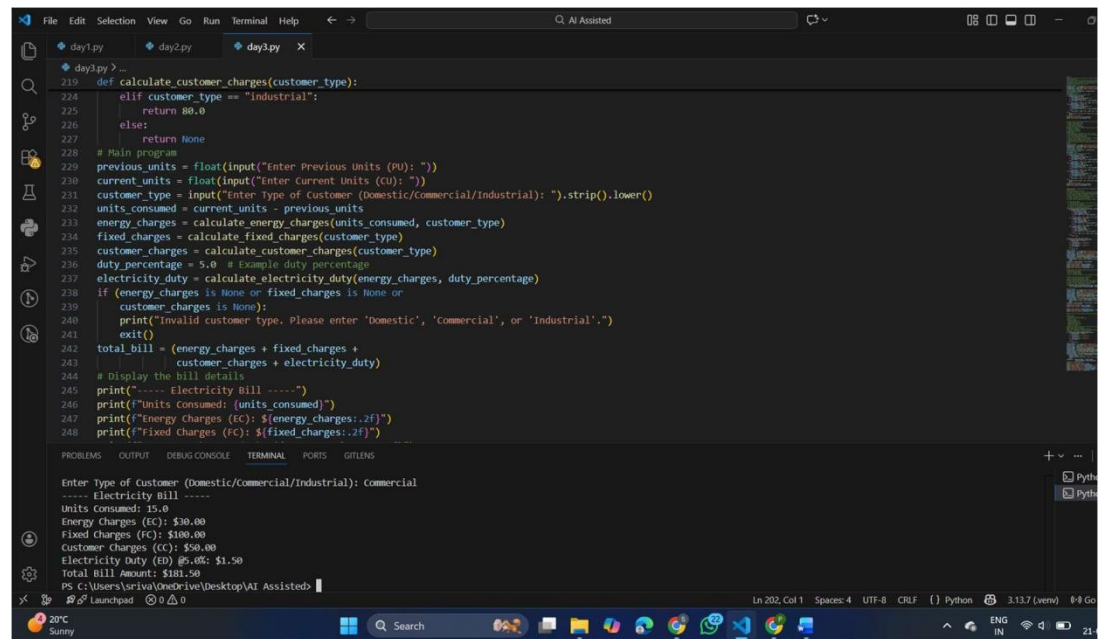


```

elif customer_type == "commercial":
    return 50.0
elif customer_type == "industrial":
    return 80.0
else:
    return None
# Main program
previous_units = float(input("Enter Previous Units (PU): "))
current_units = float(input("Enter Current Units (CU): "))
customer_type = input("Enter Type of Customer
(Domestic/Commercial/Industrial): ").strip().lower()
units_consumed = current_units - previous_units
energy_charges = calculate_energy_charges(units_consumed,
customer_type)
fixed_charges = calculate_fixed_charges(customer_type)
customer_charges = calculate_customer_charges(customer_type)
duty_percentage = 5.0 # Example duty percentage
electricity_duty = calculate_electricity_duty(energy_charges,
duty_percentage)
if (energy_charges is None or fixed_charges is None or
    customer_charges is None):
    print("Invalid customer type. Please enter 'Domestic', 'Commercial',
or 'Industrial'.")
    exit()
total_bill = (energy_charges + fixed_charges +
              customer_charges + electricity_duty)
# Display the bill details
print("----- Electricity Bill ---")
print(f"Units Consumed: {units_consumed}")
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) @{duty_percentage}%:
${electricity_duty:.2f}")
print(f"Total Bill Amount: ${total_bill:.2f}")

```

OUTPUT:



```
File Edit Selection View Go Run Terminal Help
day1.py day2.py day3.py X

def calculate_customer_charges(customer_type):
    if customer_type == "Industrial":
        return 80.0
    else:
        return None
# Main program
previous_units = float(input("Enter Previous Units (PU): "))
current_units = float(input("Enter Current Units (CU): "))
customer_type = input("Enter Type of Customer (Domestic/Commercial/Industrial): ").strip().lower()
units_consumed = current_units - previous_units
energy_charges = calculate_energy_charges(units_consumed, customer_type)
fixed_charges = calculate_fixed_charges(customer_type)
customer_charges = calculate_customer_charges(customer_type)
duty_percentage = 5.0 # Example duty percentage
electricity_duty = calculate_electricity_duty(energy_charges, duty_percentage)
if (energy_charges is None or fixed_charges is None or
    customer_charges is None):
    print("Invalid customer type. Please enter 'Domestic', 'Commercial', or 'Industrial'.")
    exit()
total_bill = (energy_charges + fixed_charges +
              customer_charges + electricity_duty)
# Display the bill details
print("----- Electricity Bill -----")
print(f"Units Consumed: {units_consumed}")
print(f"Energy Charges (EC): ${energy_charges:.2f}")
print(f"Fixed Charges (FC): ${fixed_charges:.2f}")

Enter Type of Customer (Domestic/Commercial/Industrial): Commercial
----- Electricity Bill -----
Units Consumed: 15.0
Energy Charges (EC): $30.00
Fixed Charges (FC): $100.00
Customer Charges (CC): $50.00
Electricity Duty (ED) @ 5.0%: $1.50
Total Bill Amount: $181.50
PS C:\Users\sriya\OneDrive\Desktop\AI Assisted>
```

EXPLANATION:

Task 5 generates the final electricity bill by combining all charge components.

The total bill amount is calculated by adding EC, FC, CC, and ED.

The output is displayed in a neat, bill-like format for clarity.

This task demonstrates a complete, real-world electricity billing application