Name : Ruhi Bisen
Class : 1st year
Batch :CE-2-C
Roll no: 120
PRN: B25CE1120

EXPERIMENT-2

C PROGRAM FOR CALCULATING ELECTRIC BILL

1.Research

In real-world applications, many systems operate on the principle of **slab-based categorization**, where values are grouped into ranges to determine outcomes. Two common examples are the **grading system in colleges** and **electricity billing systems**. In colleges, students are assigned grades based on the range of marks they achieve, ensuring fair classification of academic performance. Similarly, in electricity billing, consumers are charged according to slabs of energy consumption, with lower rates for basic usage and higher rates for excessive consumption. This parallel makes the electricity billing problem an ideal case study for C programming, as it allows the use of **conditional control structures** to model real-world scenarios in a simple, logical way.

The topic of an **Electricity Billing System** was chosen because it represents a practical and widely used application of slab-based classification, which is also the foundation of the **grading system in colleges**. In real life, most state electricity

boards in India, calculate charges using a **slab or telescopic tariff system**, where the cost per unit increases as consumption crosses specific thresholds. This model is designed to promote fair billing and encourage efficient use of electricity, ensuring affordability for low-consumption households while discouraging wastage. Similarly, in academics, students' performance is categorized into grades (A, B, C, etc.) based on percentage ranges, ensuring fairness and easy evaluation. The **common principle of range-based classification** between college grading and electricity billing makes this topic highly relevant for C programming practice, as it allows the use of conditional statements, piecewise logic, and modular programming to simulate real-world processes. Furthermore, slab-based billing not only affects millions of households but also serves as a clear and educational example for programming assignments, which is why it was selected for this project.

REFERENCES

1. https://www.seriindia.org/grading-system.html
2. https://www.nobse.in/GrandingDivisionSystem.aspx
3. https://www.thehindu.com/news/national/andhra-pradesh/ap-power-tariff-entire-consumption-will-not-be-charged-under-single-slab/article65274871.ece
4.

2.ANALYSE

The **Electricity Billing System** is a practical application of **slab-based computation**, where outcomes are determined by predefined consumption ranges. From a computational perspective, the **input parameters** include the

customer's identification details (ID and name) along with the total number of electricity units consumed. The **processing stage** applies tariff slabs in a stepwise manner: for example, the first 100 units may be charged at a concessional rate, the next 200 units at a moderate rate, and all units above 300 at a premium rate. This progressive model ensures that essential consumption remains affordable while higher consumption is billed at increased rates, thereby promoting conservation and fairness.

In terms of programming logic, this problem directly translates into the use of **conditional control structures** such as `if-else` or `switch-case` statements, which determine the applicable slab based on units consumed. For scalability, **loops** can be incorporated to handle billing for multiple customers within a single execution cycle. To achieve modularity and clarity, **functions** may be designed for distinct tasks such as input handling, slab-wise calculation, and formatted output display. The **output** is a structured bill containing customer information, units consumed, and the computed payable amount.

The analysis demonstrates that this topic not only reflects a socially relevant process but also provides an excellent programming exercise. It encapsulates key principles of structured programming—**decision-making, modular design, and real-world simulation**—making it an academically meaningful and technically robust choice for a C programming project.

3.IDEATE

The design of the **Electricity Billing System** begins with the recognition that real-world billing processes must be both **accurate** and **transparent**. The initial implementation can focus on a **single-customer model**, where the program accepts essential inputs—customer ID, name, and units consumed—and computes the bill based on slab rates using conditional logic. To enhance clarity and maintainability, the system can be modularized into functions such as

`inputData()`, `calculateBill()`, and `displayBill()`. This modular approach aligns with the principles of structured programming and ensures that the program remains scalable.

Beyond the basic model, the system can evolve into a **multi-customer billing solution** using arrays or structures to store and process multiple records in a single execution cycle. Introducing **file handling** allows permanent storage of bills and retrieval of customer history, bridging the gap between classroom learning and real-world billing applications. To simulate actual practices, the program may incorporate **fixed service charges**, **tax percentages**, or **subsidy provisions** for specific user categories.

Looking further, the project has potential for **future expansion** into advanced domains. A graphical or menu-driven interface could make the system user-friendly, while integration with **databases** would allow handling of larger datasets in real time. With minor modifications, the same logic could be extended to mobile or web applications, reflecting how utility companies manage billing at scale. Thus, the ideation not only outlines a feasible solution within the scope of a C programming project but also positions the topic as a foundation for developing more sophisticated, real-world billing software.

4. BUILD

Here is a c program build to execute the above program in the most appropriate way:-

```
#include <stdio.h>

int main() {

    int customerID, units;
```

```
 char customerName[50];
 float bill;
 printf("Enter Customer ID: ");
 scanf("%d", &customerID);
printf("Enter Customer Name: ");
 scanf("%s", customerName);

 printf("Enter Units Consumed: ");
 scanf("%d", &units);
 if (units <= 100) {
    bill = units * 1.50;
 }
 else if (units <= 300) {
    bill = (100 * 1.50) + (units - 100) * 2.50;
 }
 else {
    bill = (100 * 1.50) + (200 * 2.50) + (units - 300) * 4.00;
 }
 printf("\n----- Electricity Bill -----\n");
 printf("Customer ID   : %d\n", customerID);
```

```
    printf("Customer Name : %s\n", customerName);
    printf("Units Consumed: %d\n", units);
    printf("Total Bill    : Rs. %.2f\n", bill);
    printf("----------------------------\n");
 return 0;
}
```

OUTPUT:

5. TESTING

The program was tested before being implemented here is result:

Error check:

Enter Customer ID: 55

Enter Customer Name: XYZ

Enter Units Consumed: ABC

----- Electricity Bill -----

Customer ID   : 55

Customer Name : XYZ

Units Consumed: 0

Total Bill   : Rs. 0.00

When alphabets were used in place of numbers in place of units the result showed bill as 0.

ACCURATE OUTPUTS:

Case 1:

When input amount is less than 100 units :-

Enter Customer ID: 21

Enter Customer Name: xyz

Enter Units Consumed: 99

----- Electricity Bill -----

Customer ID : 21

Customer Name : xyz

Units Consumed: 99

Total Bill : Rs. 148.50

Case 2:

When input amount is less than 300 units:-

Enter Customer ID: 21

Enter Customer Name: abc

Enter Units Consumed: 234

----- Electricity Bill -----

Customer ID : 21

Customer Name : abc

Units Consumed: 234

Total Bill : Rs. 485.00

Case 3:

When input amount is more than 300 units:-

Enter Customer ID: 21

Enter Customer Name: xyz

Enter Units Consumed: 301

----- Electricity Bill -----

Customer ID : 21

Customer Name : xyz

Units Consumed: 301

Total Bill : Rs. 654.00

---

6. IMPLEMENTATION

The project is published on github for easy access:

https://github.com/ruheebisen37/Calculating-electric-bill-using-c-programming

7.CONCLUSION

The Electricity Billing System program effectively applies slab-based tariff logic using conditional structures in C. It delivers accurate results, demonstrates structured programming principles, and provides a scalable foundation for future enhancements such as multi-customer handling and database integration.

THANK YOU!!