Experiment No. 2
Smart Billing Calculator for Grocery Store using Operators in C

Aim: Design a C program to assist a grocery store manager with daily billing tasks. The application should compute the total cost of items, apply discounts, and compare prices using various C operators—arithmetic, relational, logical, assignment, unary, conditional, and comma operators—to streamline routine calculations efficiently.

Learning Outcomes:

- Apply various C operators (arithmetic, relational, logical, assignment, unary, conditional, and comma) in solving computational problems.
- Perform calculations, comparisons, and decision-making effectively using operators.
- Optimize operations and improve efficiency through appropriate use of operators.
- Strengthen structured problem-solving and logical thinking skills.
- Enhance overall programming proficiency by implementing operators in real-world scenarios.

Theory: Operators in C are special symbols that perform specific operations on variables and values. They are the building blocks of expressions and allow tasks like arithmetic, comparisons, and logic processing.

Types, Syntax & Examples

1. **Arithmetic Operators : Perform basic mathematical operations.**
   Syntax: **a + b, a - b, a * b, a / b, a % b**
   Example:
   ```
   int a = 10, b = 3;
   printf("%d", a + b); // Output: 13
   ```

2. **Relational Operators : Compare two values and return true/false.**
   Syntax: **a > b, a < b, a == b, a != b, a >= b, a <= b**
   Example:
   ```
   if(a > b) printf("a is greater");
   ```

3. **Logical Operators: Combine multiple conditions**
   Syntax: **&& (AND), || (OR), ! (NOT)**
   Example:
   ```
   if(a > 0 && b > 0) printf("Both positive");
   ```

4. **Assignment Operators: Assign values to variables (with shortcuts)**
   Syntax: **=, +=, -=, *=, /=, %=**
   Example:
   ```
   a += 5; // same as a = a + 5
   ```

5. **Unary Operators: Operate on a single operand (increment, decrement, negation**
   Syntax: **++a, a++, --a, a--, -a**
   Example:
   int x = 5;
   printf("%d", ++x); // Output: 6

6. **Conditional (Ternary) Operator: Short-hand for decision making**
   Syntax: **(condition) ? expression1 : expression2;**
   Example:
   int max = (a > b) ? a : b;

7. **Comma Operator: Evaluate multiple expressions, returns last value**
   Syntax: **(expr1, expr2, expr3, ...)**
   Example:
   int y = (a = 5, a + 10); // y = 15

Program Code:

Students are expected to implement this experiment by writing the complete C program for the Grocery billing system, making use of the various C operators.

Output:

Students are expected to execute the program and provide two sample outputs for the same code to demonstrate its functionality with different input values.

Conclusion: This experiment demonstrated how C operators can be applied to a real-world grocery billing system for tasks like totals, discounts, and price comparisons. It strengthened students' understanding of operators and demonstrated their practical use in simplifying business operations.