Operators and expressions

CHAPTER - 5

Operators

- An Operator is a symbol used for certain type of manipulations, logical or mathematical.
- The data on which the operators act are called operands.

Types of operators

- Unary
- Binary
- Ternary
- Special

Unary operator

- Operator that operate on a single operand are known as unary operators.

  Unary plus(+)

  Unary minus(-)

  Increment (++)

  Decrement (--)

  Logical Not (!)

  Address of (&)

Binary operators

- Arithmetic operators
- Relational Operators
- Logical Operators
- Assignment operators
- Bitwise operators

Arithmetic operators

- C supports five arithmetic operators. These operators are used for arithmetic calculations

(E-x) A = 10 , B=20

Operator	Description	Example
+	Adds two operands.	A + B = 30
−	Subtracts second operand from the first.	A − B = -10
*	Multiplies both operands.	A * B = 200
/	Divides numerator by de-numerator.	B / A = 2
%	Modulus Operator and remainder of after an integer division.	B % A = 0

Relational operators

RELATIONAL OPERATOR:

- Relational Operators are used to compare two quantities and take certain decision depending on their relation.

  If the specified relation is true it returns one.

  If the specified relation is false it returns zero.

OPERATOR	MEANING
<	Is less than
<=	Is less than or equal to
>	Is greater than
>=	Is greater than or equal to
==	Is equal to
!=	Is not equal to

Operator	Description	Example
&&	Called Logical AND operator. If both the operands are non-zero, then the condition becomes true.	(A && B) is false.
||	Called Logical OR Operator. If any of the two operands is non-zero, then the condition becomes true.	(A || B) is true.
!	Called Logical NOT Operator. It is used to reverse the logical state of its operand. If a condition is true, then Logical NOT operator will make it false.	!(A && B) is true.

Logical operators

- It is used to combine two or more logical expressions.

Bitwise shift operator

- The shift operators are used to push the bit patterns to the right or left by the number of bits given by their right operand.

The right shift operator is of the form

expression >> n

Where n is the number of bit patterns to be shifted right

E-x y >> 2 y=33

- Left shift operator

  expression << n

  where n is the number of bit positions to be shifted left

Special operators

Operator	Description
,	Comma operator
sizeof	Size in bytes
&	Address of operand
*	Accessing the value at that address
.	Dot operator
→	arrow operator

Comma operator

- The comma operator is used as separators.

- E-x
- variable = exp1,exp2,…expn

sizeof operator

- The sizeof operator returns the size in bytes of the operand on the right.

The syntax is

sizeof(n)

Where n can be a variable, a constant or a datatype.

Expressions

Valid expressions are a combination of operators ,constants and variables.

- Constant
- Arithmetic
- Relational
- Logical
- Pointer
- Bitwise

Evaluation of expression

- To evaluate the expression ,the given algebraic expression must be converted to a C expression.

E-x

Mathematical expression	c expression
ab+cd	a*b+c*d

Precedence of Arithmetic operators

- An arithmetic expression is evaluated according to the rule of precedence of operators.

Precedence level	operators	Associativity
1	( )	Left to right
2	unary	Right to left
3	* , / , %	left to right
4	+ , -	Left to right

Type Conversion

- Converting from one datatype to another datatype.

Two types of type conversions are

- Implicit type conversion
- Explicit type conversion.

Implicit type conversion

- When the operands are of different types, the lower type is converted to higher type automatically and the resultant value is of higher type.

Operands	Conversion	Result
float, double	Converts float to double	double
float , int	Converts int to float	float
int , long	converts int to long	long

Explicit conversion (Type casting)

- A user can convert data explicitly by type casting. The general form is

(data type) expression

E-x

n1 = (int)25.5