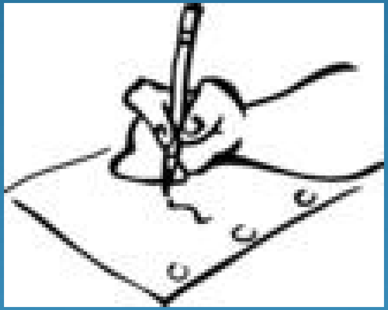
Lecture 2
Operators & Expressions



Definition

“An operator is a symbol (+,-,*,/) that directs the computer to perform certain mathematical or logical manipulations and is usually used to manipulate data and variables”

Ex: a+b

Operators in C

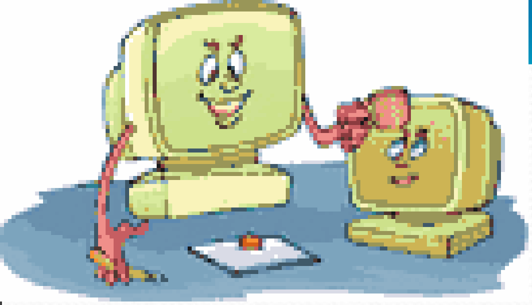
1. Arithmetic operators
2. Relational operators
3. Logical operators
4. Assignment operators
5. Increment and decrement operators
6. Conditional operators
7. Bitwise operators
8. Special operators

Arithmetic operators

Operator	example	Meaning
+	a + b	Addition –unary
-	a – b	Subtraction- unary
*	a * b	Multiplication
/	a / b	Division
%	a % b	Modulo division- remainder

Relational Operators

Operator	Meaning
<	Is less than
<=	Is less than or equal to
>	Is greater than
>=	Is greater than or equal to
==	Equal to
!=	Not equal to



Logical Operators

Operator	Meaning
&&	Logical AND
||	Logical OR
!	Logical NOT

Logical expression or a compound relational expression-

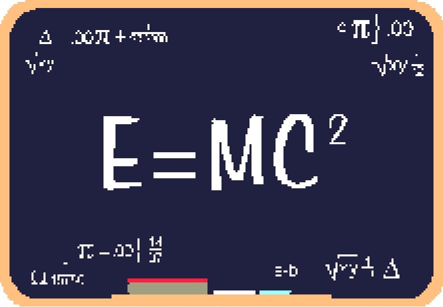
An expression that combines two or more relational expressions

Ex: if (a==b && b==c)

Truth Table

a	b	Value of the expression	
		a && b	a || b
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Assignment operators



Syntax:

v op = exp;

Where v = variable,

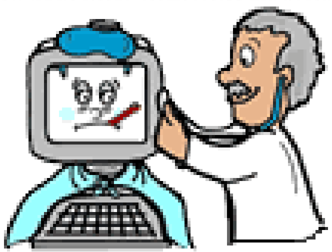
op = shorthand assignment operator

exp = expression

Ex: x=x+3

x+=3

Shorthand Assignment operators



Simple assignment operator	Shorthand operator
a = a+1	a + =1
a = a-1	a - =1
a = a* (m+n)	a * = m+n
a = a / (m+n)	a / = m+n
a = a %b	a %=b

Increment & Decrement Operators

C supports 2 useful operators namely

1. Increment ++
2. Decrement – operators

The ++ operator adds a value 1 to the operand

The – operator subtracts 1 from the operand

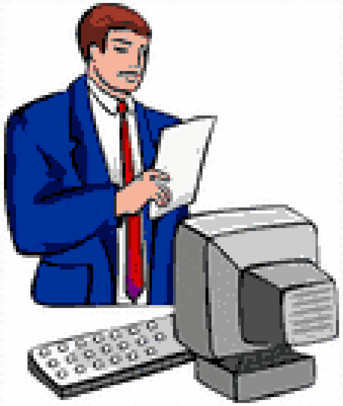
++a or a++

--a or a--

Rules for ++ & -- operators

1. These require variables as their operands
2. When postfix either ++ or – is used with the variable in a given expression, the expression is evaluated first and then it is incremented or decremented by one
3. When prefix either ++ or – is used with the variable in a given expression, it is incremented or decremented by one first and then the expression is evaluated with the new value

Examples for ++ & -- operators



Let the value of a =5 and b=++a then

a = b =6

Let the value of a = 5 and b=a++ then

a =5 but b=6

i.e.:

1. a prefix operator first adds 1 to the operand and then the result is assigned to the variable on the left
2. a postfix operator first assigns the value to the variable on left and then increments the operand.

Conditional operators

Syntax:

exp1 ? exp2 : exp3

Where exp1,exp2 and exp3 are expressions

Working of the ? Operator:

Exp1 is evaluated first, if it is nonzero(1/true) then the expression2 is evaluated and this becomes the value of the expression,

If exp1 is false(0/zero) exp3 is evaluated and its value becomes the value of the expression

Ex: m=2;

```
n=3
r=(m>n) ? m : n;
```

Bitwise operators

These operators allow manipulation of data at the bit level

Operator	Meaning
&	Bitwise AND
|	Bitwise OR
^	Bitwise exclusive OR
<<	Shift left
>>	Shift right

Special operators

1. Comma operator (,)
2. sizeof operator – sizeof( )
3. Pointer operators – (& and *)
4. Member selection operators – (. and ->)

Arithmetic Expressions

Algebraic expression	C expression
axb-c	a*b-c
(m+n)(x+y)	(m+n)*(x+y)
$\left[\frac{ab}{c}\right]$	a*b/c
$3x^2+2x+1$	3*x*x+2*x+1
$\frac{a}{b}$	a/b
S= $\frac{a+b+c}{2}$	S=(a+b+c)/2

Arithmetic Expressions

Algebraic expression	C expression
area= $\sqrt{s(s-a)(s-b)(s-c)}$	area=sqrt(s*(s-a)*(s-b)*(s-c))
Sin $\left(\frac{b}{\sqrt{a^2+b^2}}\right)$	sin(b/sqrt(a*a+b*b))
$\tau_1 = \sqrt{\left\{\frac{\sigma_x-\sigma_y}{2}\right\} + \tau xy^2}$	tow1=sqrt((rowx-rowy)/2+tow*x*y*y)
$\tau_1 = \sqrt{\left\{\frac{\sigma_x-\sigma_y}{2}\right\}^2 + \tau xy^2}$	**tow1=sqrt(pow((rowx-rowy)/2,2)+tow*x*y*y)**
$y = \frac{\alpha+\beta}{\sin\theta} + \lvert x\rvert$	**y=(alpha+beta)/sin(theta*3.1416/180)+abs(x)**

Precedence of operators

BODMAS RULE-

Brackets of Division Multiplication Addition Subtraction

Brackets will have the highest precedence and have to be evaluated first, then comes of , then comes division, multiplication, addition and finally subtraction.

C language uses some rules in evaluating the expressions and they r called as precedence rules or sometimes also referred to as hierarchy of operations, with some operators with highest precedence and some with least.

The 2 distinct priority levels of arithmetic operators in c are-

Highest priority : * / %

Lowest priority : + -

Rules for evaluation of expression

1. First parenthesized sub expression from left to right are evaluated.
2. If parentheses are nested, the evaluation begins with the innermost sub expression
3. The precedence rule is applied in determining the order of application of operators in evaluating sub expressions
4. The associatively rule is applied when 2 or more operators of the same precedence level appear in a sub expression.
5. Arithmetic expressions are evaluated from left to right using the rules of precedence
6. When parentheses are used, the expressions within parentheses assume highest priority

Hierarchy of operators

Operator	Description	Associativity
( ), [ ]	Function call, array element reference	Left to Right
+, -, ++, - - ,!,~,*,&	Unary plus, minus, increment, decrement, logical negation, 1's complement, pointer reference, address	Right to Left
*, / , %	Multiplication, division, modulus	Left to Right

Example 1

Evaluate x1=(-b+ sqrt (b*b-4*a*c))/(2*a) @ a=1, b=-5, c=6

=(-(-5)+sqrt((-5)(-5)-4*1*6))/(2*1)

=(5 + sqrt((-5)(-5)-4*1*6))/(2*1)

=(5 + sqrt(25 -4*1*6))/(2*1)

=(5 + sqrt(25 -4*6))/(2*1)

=(5 + sqrt(25 -24))/(2*1)

=(5 + sqrt(1))/(2*1)

=(5 + 1.0)/(2*1)

=(6.0)/(2*1)

=6.0/2 = 3.0

Example 2

Evaluate the expression when a=4

b=a- ++a

=a – 5

=5-5

=0