Chapter – 4
Operators & Expressions

Course Code: CIS 115 & 115 L

Course Title: Structured Programming

Course Teacher: Md. Faruk Hosen

Prepared by: ABK Bhuiyan (Jehad)

Operator

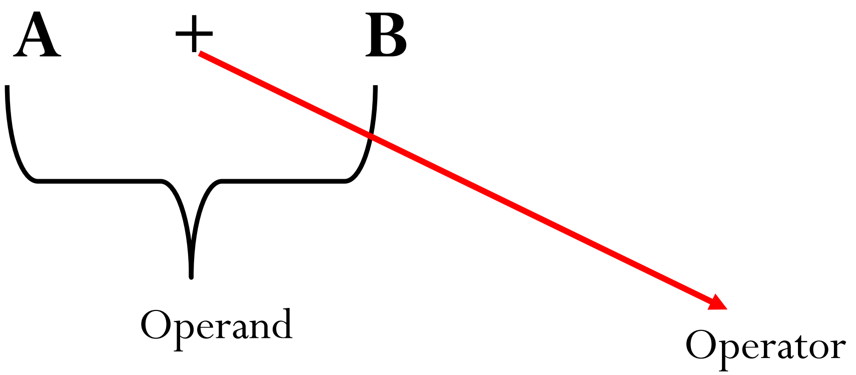
- An operator is a symbol used to indicate a specific operation on variables in a program.

- Example : symbol " + " is an **add operator** that adds two data items called operands.

Expression

- An expression is a combination of operands (constants, variables, numbers) connected by operators and parenthesis.

- Example :



Operators in C

- C language is very rich in operators.

- Main types of C operators :
    - Arithmetic
    - Relational
    - Logical
    - Bit wise
    - Assignment

Arithmetic Operators

- Arithmetic operators are used to perform mathematical calculations like **addition, subtraction, multiplication, division and modulus**.
- Following operators are used for arithmetic operations on all built in data types :

+	(unary plus)
-	(unary minus)
+	(addition)
-	(subtraction)
*	(multiplication)
/	(division or quotient)
%	(modulus or remainder)
--	(decrement)
++	(increment)

ARITHMETIC OPERATORS

Operators +, -, / and * known to us. The % operator known as modulo division produces the reminder of an integer division. Example: Suppose a and b are two variables

a+b	**This performs addition on the operands a and b**
a-b	This performs Subtraction on the operands a and b
a*b	This performs Multiplication on the operands a and b
a/b	This performs Division on the operands a and b
a%b	This produces the reminder of the integer division of a and b

Note that the modulo division operator (%) are not applicable on floating point data.

Precedence or Order of Evaluation

```
1 + 2 * 3 - 4 / 5 =
1 + (2 * 3) - (4 / 5)
```

B stands for brackets,
O for Order (exponents),
D for division,
M for multiplication,
A for addition, and
S for subtraction.

Assignment Operator

- It is used to **assign** variable a value:

  variable_name = expression;

- **lvalue** : In compiler lvalue error messages means that an object on left hand side of assignment operator is missing.

- **rvalue** : In compiler rvalue error messages means that expression on right hand side of assignment operator is erroneous.

Two cases of assignment

- **Multiple assignment**:

  int j=k=m=0;

- **Compound assignment**:

  j= j+10; this expression can be written as

  j + = 10;

  similarly

  m= m-100; is equivalent to **m - = 100;**

Relational Operator

- A relational operator is used to compare two values and the result of such operation is always either TRUE (1) or FALSE (0).

Operator	Meaning	Example
<	Less than	10 < 20, x < y
<=	Less than or equal	10 <= 10, x <= y
>	Greater than	20 > 10, x > y
>=	Greater than or equal	21 >= 20, x >= y
==	Equal to	15 == 15, x == y
!=	Not equal to	5 != 6, x != y

RELATIONAL OPERATOR

Expression containing relational operator in known as relational expression. The resulting value of a relation expression is either zero or one. **The result will be one if the condition is true and zero if false.**

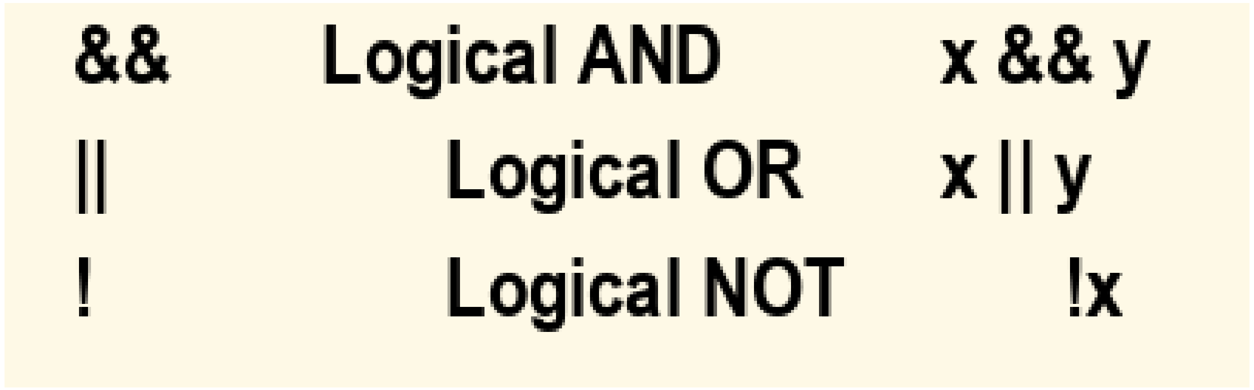
Example:

- **Suppose that i, j, and k are integer variables whose values are 1, 2 and 3, respectively.**

Expression	Value	Interpretation
i < j	1	true
(i + j) > = k	1	true
(j + k) > (i + 5)	0	false
k!=3	0	false
j = = 2	1	true

Logical Operators

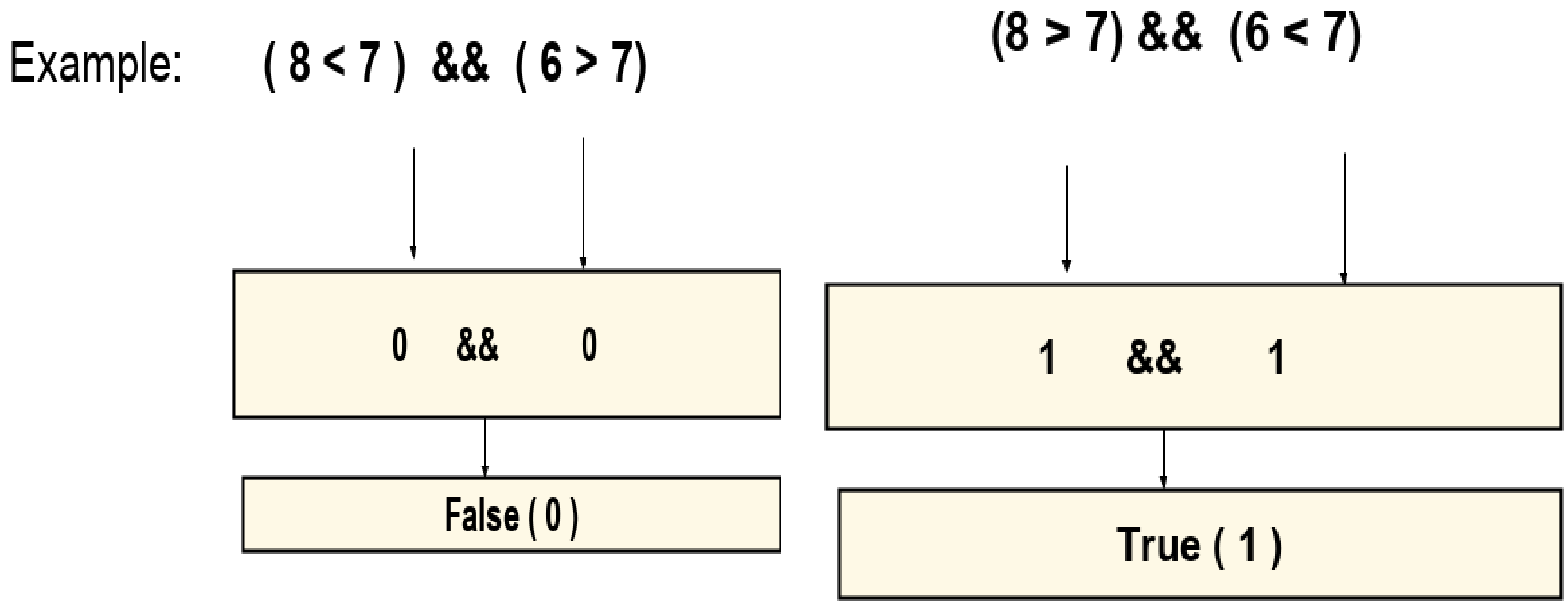
- A logical operator is used to connect two relational expressions or logical expressions.
- The result of logical expressions is always an integer value either TRUE (1) or FALSE(0).



&&	Logical AND	x && y
||	Logical OR	x || y
!	Logical NOT	!x

Logical AND Operator

- The output of the logical AND operation is TRUE if both the operands are true.



Logical OR Operator

- The result of logical OR operation will be TRUE if either operand is true or if both operands are true.
- Example:

(8 < 7) || (6 > 7) is false

(8 > 7) || (6 > 7) is true

(8 > 7) || (6 < 7) is true

Logical NOT Operator

- The Logical NOT (!) is a unary operator. It negates the value of the logical expression or operand.

- If value of X = 0   ! X = ?

  ! X = 1

- ! (5 < 6) || (7 > 7) = ???

  ! (1) || (0) = ! 1 = 0 -> false

- ! (5 > 3) = ??

  -> 0 -> false

- !(34 >= 765) = ??

  -> 1 -> True.

Exercise

x = 10 and y = 25

(x > = 10) && (x < 20)	True
(x >= 10) && (y < = 15)	False
(x = = 10) && (y > 20)	True
(x==10) || (y < 20)	True
(x ==10) &&(! (y < 20))	True

Exercise

- **Suppose that**

  j = 7, an integer variable

  f = 5.5, a float variable

  c = ‘w’

  Interpret the value of the following expressions:

(j >= 6) && (c = = ‘w’)	1
(j >= 6) | | (c = = ‘w’)	1
(f < 11) && (j > 100)	0
(c ! = ‘p’) | | ((j + f) <= 10)	1
f > 5	1
!(f > 5)	0
j < = 3	0
!(j <= 3)	1
j > (f +1)	1
!(j > (f +1))	0

Excercise

- Suppose that

  j = 7, an integer variable

  f = 5.5, a float variable

  c = ‘w’

  Interpret the value of the following expressions:

j + f <= 10	0
j >= 6 && c = = ‘w’	1
f < 11 && j > 100	0
!0 && 0 | | 0	0
!(0 && 0) | | 0	1