Operators

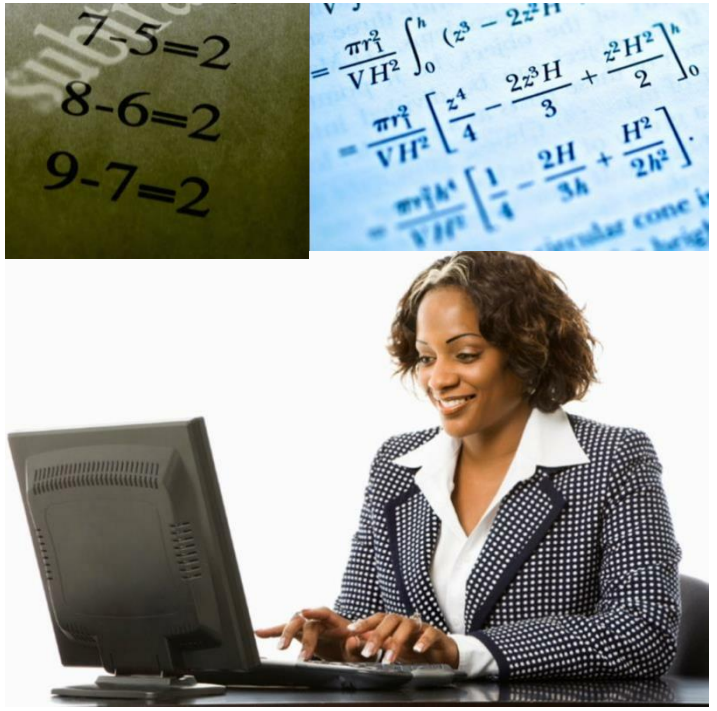
SESSION 4

Objectives

- Define operators
- List the different types of operators
- Describe the use of arithmetical operators
- Describe the use of relational operators to make comparisons
- Explain the process of associating selections with logical operators
- Identify the precedence of operators in an expression

Introduction

- Computer operations can be arithmetic such as addition, division, or even comparison where one variable is compared to another variable.
- These kinds of operations are performed using operators.



Operators

- Operators:
  - A set of symbols that help to manipulate or perform some sort of function on data

- The three types of operators are as follows:
  - Arithmetic Operators
  - Relational Operators
  - Logical Operators

Using Arithmetic Operators 1-3

- Arithmetic operators:
  - Help to manipulate numeric data
  - Help perform common arithmetic operation on the data

+ - / *

Using Arithmetic Operators 2-3

- The table shows a list of arithmetic operators common to most programming languages.

Operator	Description	Example	Result	C# Equivalent
+	Addition	9 + 2	11	+
-	Subtraction	9 – 2	7	-
/	Division	9/2	4.5	/
*	Multiplication	9 * 2	18	*
^	Exponentiation	9^2	81	^
MOD	Modulus	9 MOD 2	1	%
-	Negation	-9	-9	-

Using Arithmetic Operators 3-3

- The negation operator
  - Requires only a single operand
  - Is also known as a unary operator

- All other operators require two operands and are known as binary operators.

Precedence between Arithmetic Operators

- The table shows the order in which each arithmetic operator precedes over other arithmetic operators.

Precedence	Operator	Description
1	++	Increment
2	--	Decrement
3	*, /, MOD	Multiplication, Division, Modulus
4	+, -	Addition, Subtraction

Using Relational Operators 1-2

- Relational operators:
  - Compare two or more values or expressions and always return either ‘True’ or ‘False’
  - Are binary operators

Using Relational Operators 2-2

- The table shows a list of relational operators common to most languages.

Operator	Description	Example	Result	C# Equivalent
<	Less than	2<9	True	<
<=	Less than or Equal to	2<=9	True	<=
>	Greater than	2>9	False	>
>=	Greater than or Equal to	2>=9	False	>=
=	Equal to	2=9	False	==
<>	Not Equal to	2<>9	True	!=

Precedence between Relational Operators

- There is no precedence among relational operators.

- Therefore, they are always evaluated from left to right.

Using Logical Operators 1-2

- Logical operators:
  - Are used in situations where multiple conditions need to be satisfied
  - Combine the results of several comparisons, as required, to present a single answer
  - Return the results in either ‘True’ or ‘False’

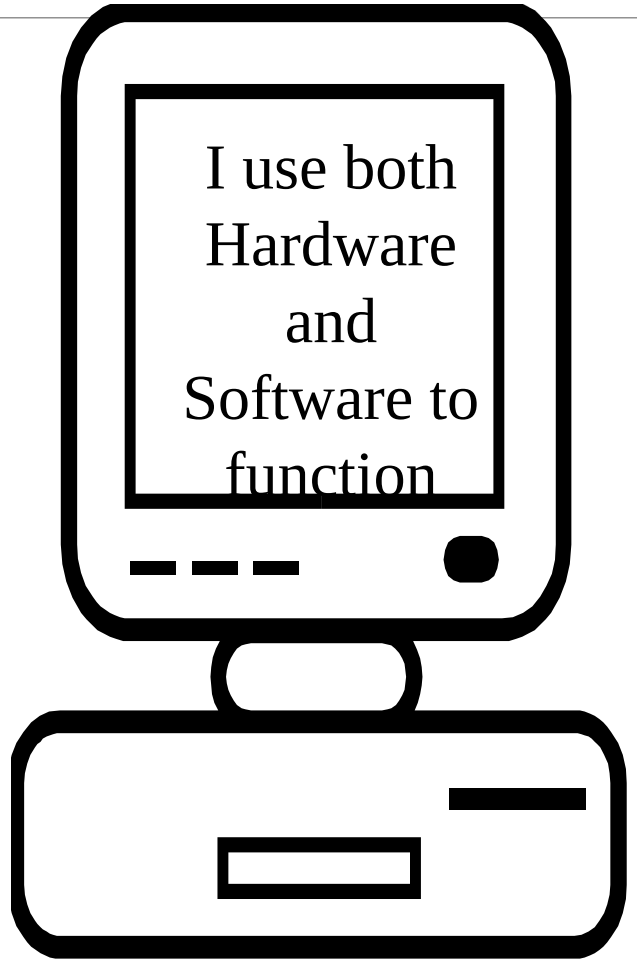
Age > 18 AND City = ‘New York’

Using Logical Operators 2-2

- The table shows a list of logical operators.

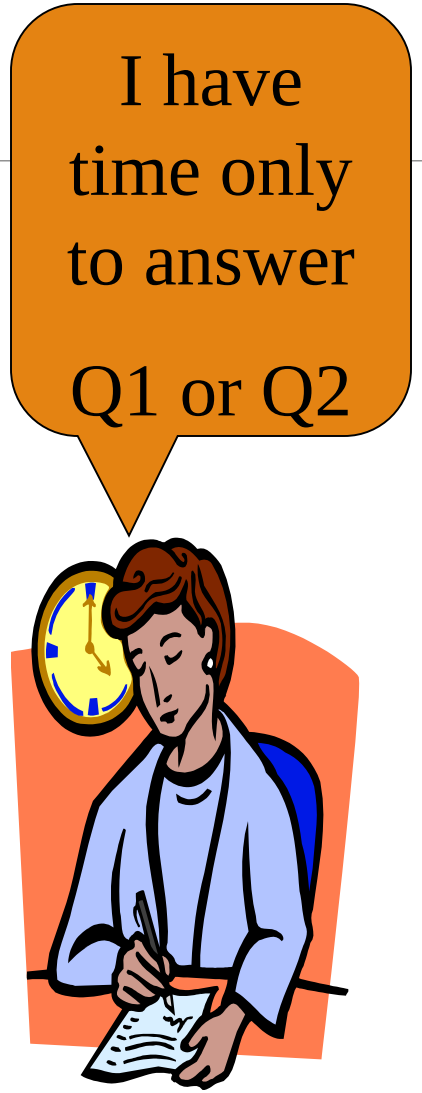
Operator	Description	C Equivalent
AND	Result is 'True' only when both conditions are 'True'	&&
OR	Result is 'True' when either of the two conditions is 'True'	||
NOT	Operates on a single value and converts 'True' to 'False' and vice-versa	!

AND Operator – Truth Table



Condition 1	Condition 2	Result
True	True	True
True	False	False
False	True	False
False	False	False

OR Operator – Truth Table



Condition 1	Condition 2	Result
True	True	True
True	False	True
False	True	True
False	False	False

NOT Operator

- Unary Operator used to Negate a condition

Condition 1	Result
True	False
False	True

Precedence between Logical Operators

- The table shows the precedence order for logical operators.

Precedence	Operator
1	NOT
2	AND
3	OR

Precedence of Operators in an Expression

- The table shows the precedence among the different types of operators.

Precedence	Type of Operator
1	Arithmetic
2	Relational
3	Logical

The Parenthesis

- Sometimes, for certain formulas, the programmer may need to override the precedence rules.

- These rules can be overridden with the help of parenthesis.

```
10/100 * ((basicSal+hra)-tax)
```