**PROG0101 Fundamentals of Programming **

**PROG0101**

**FUNDAMENTALS OF PROGRAMMING**

**Chapter 5**

**Operators**

**1**

**PROG0101 Fundamentals of Programming**

**Operators Topics**

• Operators

• Arithmetic Operators

• Relational Operators

• Logical Operators

• Increment and Decrement Operators

**2**

**PROG0101 Fundamentals of Programming**

**Operators Operators**

• An operator is a symbol, which helps the user to command the computer to do a certain mathematical or logical manipulations.

• Operators are used in programming language program to operate on data and variables.

**3**

**PROG0101 Fundamentals of Programming**

**Operators Operators**

• Operators can be classified as: – Arithmetic operators

– Relational Operators

– Logical Operators

– Increments and Decrement Operators

**4**

**PROG0101 Fundamentals of Programming** 

**Operators**

**Arithmetic Operators**

• You can use an arithmetic operator with one or two arguments to add, subtract, multiply, and divide numeric values.

|  |  |  |
| --- | --- | --- |
| **Operator** | **Name** | **Description** |
| + | Addition | to add two numbers together |
| - | Subtraction | to subtract one number from another |
| \* | Multiplication | to multiply one number by another. |
| / | Division | to divide one number by  another. |
| % | Modulus  (Remainder) | to find the remainder from dividing one number by another. |

**5**

**PROG0101 Fundamentals of Programming**

**Operators Arithmetic Operators** Example:

i. 5 + 3 = 8

ii. 5 – 3 = 2

iii. 5 \* 3 = 15

iv.5 / 3 = 1

v. 5 % 3 = 2

**6**

**PROG0101 Fundamentals of Programming**

**Operators Arithmetic Operators**

• \*, / and % will be performed before + or - in any expression.

• Brackets can be used to force a different order of evaluation to this.

**7**

**PROG0101 Fundamentals of Programming**

**Operators Arithmetic Operators** Example

i. 2 + 5 \* 4 – 3 = ? ii. (2 + 5) \* (4 – 3) = ?

**8**

**PROG0101 Fundamentals of Programming**

**Operators Arithmetic Operators**

• Here are some arithmetic expressions used within assignment statements:

i. z = x + y

ii. no1 = x – y

iii. age = a \* b + c

iv. velocity = distance / time

v. force = mass \* acceleration

vi. count = count + 1

**9**

**PROG0101 Fundamentals of Programming**

**Operators Integer Arithmetic**

• When an arithmetic operation is performed on two whole numbers or integers than such an operation is called as integer arithmetic.

• It always gives an integer as the result.

**10**

**PROG0101 Fundamentals of Programming**

**Operators Integer Arithmetic**

Example

Let x = 27 and y = 5 be two integer numbers. Then the integer operation leads to the following results:

i. x + y = 32

ii. x – y = 22

iii. x \* y = 115

iv. x % y = 2

v. x / y = 5

**11**

**PROG0101 Fundamentals of Programming**

**Operators Floating-point Arithmetic**

• When an arithmetic operation is performed on two real numbers or fraction numbers such an operation is called floating-point arithmetic.

**12**

**PROG0101 Fundamentals of Programming**

**Operators Floating-point Arithmetic** Example

Let x = 14.0 and y = 4.0 then

i. x + y = 18.0

ii. x – y = 10.0

iii. x \* y = 56.0

iv. x / y = 3.50

**13**

**PROG0101 Fundamentals of Programming**

**Operators Relational Operators**

• An operator that compares two values.

• For example, the expression:

x < 5

means x is less than 5

• This expression will have a value of TRUE if the variable x is less than 5; otherwise the value of the expression will be FALSE.

**14**

**PROG0101 Fundamentals of Programming**

**Operators Relational Operators**

• Relational operators are sometimes called comparison operators.

• Expressions that contain relational operators are called relational expressions.

• A simple relational expression contains only one relational operator and takes the following form:

<exp1> relational operator <exp2>

• Where exp1 and exp2 are expressions, which may be simple constants, variables or combination of them.

**15**

**PROG0101 Fundamentals of Programming** 

**Operators**

**Relational Operators**

• The following are relational operators:

|  |  |  |
| --- | --- | --- |
| **Operator** | **Name** | **Description** |
| < | Less than | Indicates whether the value of the left operand is less than the value of the right operand. |
| <= | Less than or equal to | Indicates whether the value of the left operand is less than or equal to the value of the right operand. |
| > | Greater than | Indicates whether the value of the left operand is greater than the value of the right operand. |
| >= | Greater than or equal to | Indicates whether the value of the left operand is greater than or equal to the value of the right operand. |

**16**

**PROG0101 Fundamentals of Programming** 

**Operators**

**Relational Operators**

• The following are relational operators:

|  |  |  |
| --- | --- | --- |
| **Operator** | **Name** | **Description** |
| == | Equal to | Indicates whether the value of the left operand is equal to the value of the right operand. |
| != | Not equal to | Indicates whether the value of the left operand is not equal to the value of the right operand. |

**17**

**PROG0101 Fundamentals of Programming**

**Operators Relational Operators**

Example:

Let x = 2 and y = 5 then

i. x < y = True ii. (x + 2) > (y \* 2) = False iii. (x + 3) <= y = True iv. x != y = True v. y > (3 + x) = False

**18**

**PROG0101 Fundamentals of Programming** 

**Operators**

**Logical Operators**

• An operator that compare or evaluate logical and relational expressions.

• The following are logical operators:

|  |  |
| --- | --- |
| **Operator** | **Name** |
| && | Logical AND |
| || | Logical OR |
| ! | Logical NOT |

**19**

**PROG0101 Fundamentals of Programming** 

**Operators**

**Logical AND**

• This operator is used to evaluate two conditions or expressions with relational operators simultaneously. • If both the expressions to the left and to the right of the logical operator is true then the whole compound expression is true.

|  |  |  |
| --- | --- | --- |
| **Exp1** | **Exp2** | **Exp1 && Exp2** |
| False | False | False |
| True | False | False |
| False | True | False |
| True | True | True |

**20**

**PROG0101 Fundamentals of Programming**

**Operators Logical AND**

Example:

(a > b) && (x == 10)

The expression to the left is a > b and that on the right is x == 10, the whole expression is true only if both expressions are true i.e., if a is greater than b and x is equal to 10.

**21**

**PROG0101 Fundamentals of Programming**

**Operators Logical AND**

Example:

Given a = 2, b = 3 and c = 5, evaluate the following logical expressions:

i. (a > b) && (c != 5) = False

ii. (a < b) && (c < b) = False

iii. (a > b) && (c == 5) = False

iv. (a < b) && (b < c) = True

**22**

**PROG0101 Fundamentals of Programming** 

**Operators**

**Logical OR**

• The logical OR is used to combine two expressions or the condition evaluates to true if any one of the 2 expressions is true.

• The expression evaluates to true if any one of them is true or if both of them are true.

|  |  |  |
| --- | --- | --- |
| **Exp1** | **Exp2** | **Exp1 || Exp2** |
| False | False | False |
| True | False | True |
| False | True | True |
| True | True | True |

**23**

**PROG0101 Fundamentals of Programming**

**Operators Logical OR**

Example:

(a < m) || (a < n)

The expression evaluates to true if any one of them is true or if both of them are true.

**24**

**PROG0101 Fundamentals of Programming**

**Operators Logical OR**

Example:

Given a = 2, b = 3 and c = 5, evaluate the following logical expressions:

i. (a > b) || (c != 5) = False

ii. (a < b) || (c < b) = True

iii. (a > b) || (c == 5) = True

iv. (a < b) || (b < c) = True

**25**

**PROG0101 Fundamentals of Programming** 

**Operators**

**Logical NOT**

• The logical NOT operator takes single expression and evaluates to true if the expression is false and evaluates to false if the expression is true.

• In other words it just reverses the value of the expression.

|  |  |
| --- | --- |
| **Exp1** | **!Exp1** |
| True | False |
| False | True |

**26**

**PROG0101 Fundamentals of Programming**

**Operators Logical NOT**

Example:

! (x >= y)

The NOT expression evaluates to true only if the value of x is neither greater than or equal to y

**27**

**PROG0101 Fundamentals of Programming**

**Operators Logical NOT**

Example:

Given a = 2, b = 3 and c = 5, evaluate the following logical expressions:

a) !(a > b) = True

b) !(a < b) = False

c) !(a > b || c == 5) = False

**28**

**PROG0101 Fundamentals of Programming**

**Operators Increment and Decrement Operators**

• The increment and decrement operators are one of the unary operators which are very useful in programming language.

• They are extensively used in loops.

• The syntax of the operators is given below:

++ variable name

variable name++

– –variable name

variable name– –

**29**

**PROG0101 Fundamentals of Programming**

**Operators Increment and Decrement Operators**

• The increment operator ++ adds the value 1 to the current value of operand.

• The decrement operator – – subtracts the value 1 from the current value of operand.

**30**

**PROG0101 Fundamentals of Programming**

**Operators Increment and Decrement Operators** Example:

Consider the following:

m = 5;

y = ++m; (prefix)

In this case the value of y and m would be 6.

Suppose if we rewrite the above statement as

m = 5;

y = m++; (postfix)

Then the value of y will be 5 and that of m will be 6.

**31**

**PROG0101 Fundamentals of Programming**

**Operators Increment and Decrement Operators**

• A **prefix** operator first adds 1 to the operand and then the result is assigned to the variable on the left. • On the other hand, a **postfix** operator first assigns the value to the variable on the left and then increments the operand.

**32**

**PROG0101 Fundamentals of Programming**

**Operators Increment and Decrement Operators** Example 1:

x = 4

y = ++x

PRINT x

PRINT y

What is the output?

5

5

**33**

**PROG0101 Fundamentals of Programming**

**Operators Increment and Decrement Operators** Example 2:

x = 3

y = x++

PRINT x

PRINT y

What is the output?

4

3

**34**

**PROG0101 Fundamentals of Programming**

**Operators Increment and Decrement Operators** Example 3:

x = 3

y = --x

PRINT x

PRINT y

What is the output?

2

2

**35**

**PROG0101 Fundamentals of Programming**

**Operators Increment and Decrement Operators** Example 4:

x = 3

y = x--

PRINT x

PRINT y

What is the output?

2

3

**36**