NAME:

Sadam Barkat

Roll:

221698

ASSIGNMENT no:

2

Subject:

Programming Fundamental

Submitted to:

Dr Khurram

C++ operator precedence

In C++, operator precedence determines the order in which operators are evaluated in an expression. operators with higher precedence are evaluated before operators with lower precedence. If operators have the same precedence their evaluation order is determined by their associativity, which are can be either left-to-right or right-to-left.

Here is a list of C++ operators categorized by their precedence, starting from highest.

1 Postfix operators:

(•) Function call: '()'
(•) Array subscripting: '[]'
(•) Member access through pointer: '->'
(•) Member access through object: '.'

2 unary operators:

(.) Increment : '++'
(.) Decrement : '--'
(.) unary plus : '+'
(.) unary minus : '-'
(.) Logical negation : '!'
(.) Bitwise negation : '~'
(.) Indirection (dereference) : '*'
(.) Adress of : '&'
(.) size of : 'sizeof'
(.) Type cast : '(type)'

3 Mutiplicative operators:

(.) Multiplication : '*'
(.) Division : '/'
(.) Modulus : '%'

4 Additive operators:

(.) Addition : '+'
(.) subtraction : '-'

5 shift operators:

(•) Left shift : '<<'
Right shift : '>>'

6 Relational and equality operators:

(•) Less than : '<'
(•) Greater than : '>'
(•) Less than or equal to : '<='
(•) Greater than or equal to : '>='
(•) Equality : '=='
(•) Inequality : '!='

7 Bitwise operators:

(•) Bitwise AND : '&'
(•) Bitwise XOR : '^'
(•) Bitwise OR : '|'

8 Logical operators:

(•) Logical AND : '&&'
(•) Logical OR : '||'

9 conditional (tenary) operators:

conditional expression : 'condition? expression 1 : expression 2'

10 Assignment operators:

(•) simple assignment : '='
(•) compound assignment : '+=', '-=', '*=', '/=', '%=', '<<=', '>>=', '&=', '^=', '|='

The comma operator ',' is the least (lowest) precendence operator, and it is used to seperate expressions.

It is important to note that prenthesis '()' can be used to override the default precendence and enforce a specific evaluation order within an expression.

C++ Operator associativity

In c++, operator associativity determins the order in which operators with the same precedence are evaluated. There are two types of associavity.

1 Left - to - right associavity:

(•) Most operators in c++ have left-to-right associavity. This means that when multiple operators with the same precedence appear in an expression, they are evaluated from left to right. For example in the expression 'a+b-c', the addition ('+') is evaluated first, followed by the subtraction ('-')

2 Right to left associativity:

(•) The assignment operator ('=') and some unary operators, such as the unary minus ('-') and unary plus ('+')

have right to left associaulty. This means that when multiple operators with the same precedence appear in an expression, they are evaluated from right to left. For example in the expression 'a = b = c' the assignment ('=') opperator are evaluated from right to left.

Here are some examples to illustrate the associavity of operators:

(1) Example 1: (Left to right associavity):

```
int result = 5-3-1;
// Evaluation as :((5-3)-1))
// result = 1
```

(2) Example 2: (Right to left)

```
int a=5, b=3, c=1;
int resut = a=b=c;
```

// Eluates as : a = (b = c)
// result = 1 , a = 1 , b = 1 , c = 1

It's important to note that the associavity of an operator is independent of its precedence. precedence determines the evaluation order b/w operators of different precedence levels, while associativity determines the evaluation order among operators of the same precedence level. Perenthesis can be used to explicitly specify the evaluation order when needed.