Tuesday, 3 August 2021 8:24 PM

Java Operators

1. Arithmetic operators
2. Relational operators
3. Bitwise operators
4. Logical operators
5. Assignment operators
6. misc. operators

1) Arithmetic operators

+	addition
−	subtraction
*	multiplication
/	division
%	modulus (remainder)

$10 \% 5 = 0$

$6 \% 4 = 2$

2) Relational operator

$a = 10$

$b = 20$

1)	==	equal to	$(a == b) \rightarrow$ false
2)	!=	not equal to	$(a != b)$
3)	>	greater than	
4)	<	less than	
5)	>=	greater than or equal to	
6)	<=	less than or equal to	

3) Bitwise operator

$a = 11\,1\,1\,1\,1\,1\,1\,1$

$b = 11\,1\,1\,1\,1\,1\,1\,1$

1)	&	bitwise AND	$a \& b \rightarrow$
2)	|	bitwise OR	$a \mid b$
3)	^	bitwise XOR	$a \wedge b$
4)	~	complement	$a = 11\,1\,1\,1\,1\,1\,1\,1$
5)	<<	left shift	$a << 1$
6)	>>	right shift	$a >> 1$

4) Logical operator

(boolean)

$a = true$

$b = false$

1)	&&	logical AND	$(a \&\& b) \rightarrow$ false
2)	||	logical OR	$a \mid\mid b$ → true
3)	!	logical NOT	$(!a \&\& b) \rightarrow$ true

* both other is a logical operator than boolean values

Assignment operators

=

assignment (assigns value from rhs to lhs)

$a = 10$

$b = 20$

$c = a + b$

$c = 30$

$c = c + 30$

$c = 60$

$c = 30 + c$

$c = 90$

$c = c - a$

$c = 80$

$c = c - a$

$c = 70$

$c = c * a$

$c = 700$

$c = c / a$

$c = 70$

$c = c \% a$

$c = 0$

5) misc. operator

(ternary operator / conditional operator)

→ used to evaluate boolean expressions

→ goal of this operator is to decide which value to be assigned to the variable.

Syntax:

variable = (expression) ? (value if true) : (value if false);

$a = 10$

$b = 20$

$c = 30$

(boolean exp) → $(a < b)$? (true) : (false)

$a < b$

true → false

Step ONE " Step reason & concept ";

Decision control statements/ conditional statements

variable = (condition) ? (true) : (false);

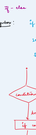
variable = (condition) ? (true) : (false);

1) if statement

Syntax: if (condition) {

// if code to be executed

}



2) if - else

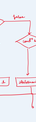
Syntax: if (condition) {

// code if cond is true.

} else {

// code if cond is false.

}



Ex: given a year, check if it is a leap year or not.

3) if - else if statement

Syntax: if (condition) {

// code if cond is true.

} else if (condition 2) {

// code if cond 2 is true.

} else if (condition 3) {

// code if cond 3 is true.

} else {

// other code.

}

Ex

A . . . . Z → Uppercase

a . . . . z → Lowercase

else → Invalid

4) nested if statement

Syntax: if (condition) {

if (condition) {

}

}

switch statement

Syntax: switch (variable) {

case value1:

break;

case value2:

break;

default:

// code to be executed if all cases not match.

}

The java switch statement is a fall-through statement.

It means that it executes all statements after the first match is found if break statement is not found.

Precedence of operators

→ operator precedence determines the order in which the operator in an expression are evaluated.

int val = 12 / 2 * 3 + (3 - (10 % 3));

(3 - (10 % 3)) → 1

12 / 2 → 6

6 * 3 → 18

18 + 1 → 19

When two operators have a common operand, the operator with the highest precedence is applied first.

Operator precedence table.

→ refers a operator applies to the nearest, higher the precedence.

	Operators	Precedence	
1)	postfix increment & decrement	++, --	left to right
2)	prefix ++, --, !, ~, (type), new	++, --, +, -, !, ~, (type), new	right to left
3)	multiplicative	*, /, %	left to right
4)	additive	+, -	L to R
5)	shift	<<, >>, >>>	L to R
6)	relational	<, >, <=, >=, instanceof	L to R
7)	equality	==, !=	
8)	bitwise AND	&	
9)	^ XOR	^	
10)	| OR	|	
11)	logical AND	&&	
12)	logical OR	||	
13)	ternary	?:	R to L
14)	assignment	=, +=, -=, *=, /=, %=	R to L

operator associativity

If an expression has two operators with the same precedence, then expression is evaluated according to associativity.

(left to right) or (right to left)