Increment And Decrement And Assignments

|  |  |  |  |
| --- | --- | --- | --- |
| Expression | Initial Vaule of x | Value of y | Final value of x |
| Y = ++X | 10 | 11 | 11 |
| Y = X++ | 10 | 10 | 11 |
| Y = --X | 10 | 9 | 9 |
| Y = X-- | 10 | 10 | 9 |

Perfectly valid:

**int** x = 10;

x++;

System.***out***.println(x); //11

**Case 1:** We can apply increment and decrement operators only for variables but not for constant values. If we are trying to apply for constant values then we will get compile time error

Example: **int** y = ++10;

**Case 2: Nest**ing of increment and decrement operators are not allowed.

**Example:**

**int** x = 0;

**int** y = ++(++x);

**Case 3:** For final variables we can’t apply increment and decrement and operators.

**final int** x = 0;

x++;

**Case 4:** We can apply increment and decrement operator for every primitive type except **boolean**

**char** a = **'a'**;

a++;

System.***out***.println(a); *//b*

**double** d = 10.5;

d++;

System.***out***.println(d); *//11.5*

**boolean** b = **true**;

b++; //Operator '++' cannot be applied to 'boolean

**Difference between and b++ and b = b+1**:

Example 1**:**

**byte** a = 10;

**byte** b = 20;

**byte** c = a + b;*//incompatible type because this rule* *max(int, type of a, type of b)*

*// and we cant assign int to byte*

*// then (a+b) return int type*

*If we apply any arithmetic operator between variables a & b the result type is always max(int, type of a, type of b)*

We can solve the above problem by using type casting

**byte** a = 10;

**byte** b = 20;

**byte** c = (**byte**)(a + b);

System.***out***.println(c);

Example 2:

**byte** a = 10;

a = a + 1; //Incompatible types. Required: byte Found: int

System.***out***.println(a);

We can solve this issue using type casting

a = (byte) a + 1;

But in the case of increment and decrement operator internal type casting will be performed automatically

**byte** a = 10;

a++;

System.***out***.println(a);

a++ ==> a = (type of a) (a+1)