# 运算符

## 运算符介绍

运算符是一种特殊的符号，用以表示数据的运算、赋值和比较等。

1) 算术运算符

2) 赋值运算符

3) 关系运算符［比较运算符］

4) 逻辑运算符

5) 位运算符［需要二进制基础］

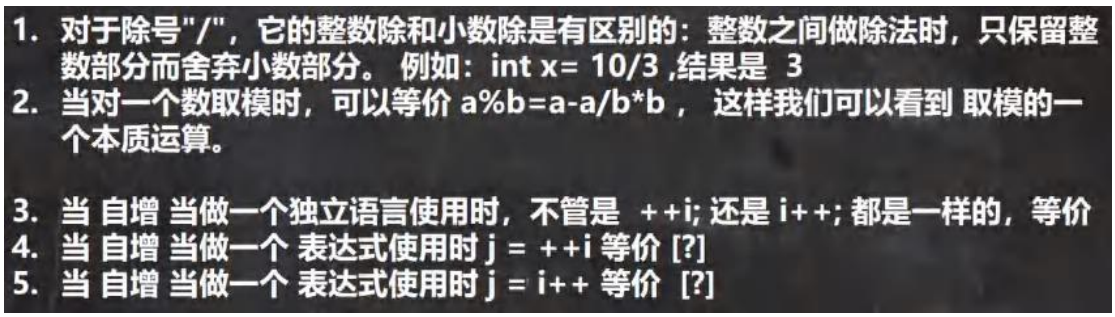
6) 三元运算符

## 算术运算符

算术运算符是对数值类型的变量进行运算的，在 Java 程序中使用的非常多。

```
public class ArithmeticOperator {
//编写一个 main 方法
public static void main(String[] args) {
// /使用
System.out.println(10 / 4); //从数学来看是 2.5, java 中 2
System.out.println(10.0 / 4); //java 是 2.5
// 注释快捷键 ctrl + /, 再次输入 ctrl + / 取消注释
double d = 10 / 4;//java 中 10 / 4 = 2, 2=>2.0
System.out.println(d);// 是 2.0
// % 取模 ,取余
// 在 % 的本质 看一个公式!!!! a % b = a - a / b * b
// -10 % 3 => -10 - (-10) / 3 * 3 = -10 + 9 = -1
// 10 % -3 = 10 - 10 / (-3) * (-3) = 10 - 9 = 1
// -10 % -3 = (-10) - (-10) / (-3) * (-3) = -10 + 9 = -1
System.out.println(10 % 3); //1
System.out.println(-10 % 3); // -1
```

```
System.out.println(10 % -3); //1
System.out.println(-10 % -3);//-1
//++的使用
//
int i = 10;
i++;//自增 等价于 i = i + 1; => i = 11
++i;//自增 等价于 i = i + 1; => i = 12
System.out.println("i=" + i);//12
/*
作为表达式使用
前++：++i 先自增后赋值
后++：i++先赋值后自增
*/
int j = 8;
//int k = ++j; //等价 j=j+1;k=j;
int k = j++; // 等价 k =j;j=j+1;
System.out.println("k=" + k + "j=" + j);//8 9
}
}
```




## 关系运算符

1）关系运算符的结果都是 boolean 型，也就是要么是 true，要么是 false

2）关系表达式 经常用在 if 结构的条件中或循环结构的条件中

关系运算符注意

1）关系运算符的结果都是 boolean 型，也就是要么是 true，要么是 false。

2) 关系运算符组成的表达式，我们称为关系表达式。 a > b

3) 比较运算符"=="不能误写成"="

## 逻辑运算符

1) 短路与 && ， 短路或 ||，取反 ！

2) 逻辑与 &，逻辑或 |，^ 逻辑异或

逻辑运算规则

1)a&b : & 叫逻辑与,当 a 和 b 同时为 true ,则结果为 true, 否则为 false

2)a&&b : && 叫短路与,当 a 和 b 同时为 true ,则结果为 true,否则为 false

3)a|b : | 叫逻辑或，当 a 和 b，有一个为 true ,则结果为 true,否则为 false

4)a||b : || 叫短路或,当 a 和 b，有一个为 true ,则结果为 true,否则为 false

5)!a : 叫取反，或者非运算,当 a 为 true, 则结果为 false, 当 a 为 false 是，结果为 true

6)a^b: 叫逻辑异或，当 a 和 b 不同时，则结果为 true, 否则为 false

&&与&区别

1) &&短路与：如果第一个条件为 false，则第二个条件不会判断，最终结果为 false，效率高

2) & 逻辑与：不管第一个条件是否为 false，第二个条件都要判断，效率低

3) 开发中， 我们使用的基本是使用短路与&&, 效率高

||与|区别

1) ||短路或：如果第一个条件为 true，则第二个条件不会判断，最终结果为 true，效率高

2) | 逻辑或：不管第一个条件是否为 true，第二个条件都要判断，效率低

3) 开发中，我们基本使用 ||

## 赋值运算符

赋值运算符就是将某个运算后的值，赋给指定的变量。

分类

基本赋值运算符 = int a = 10;

复合赋值运算符

+= ，-= ，*= ， /= ，%= 等

a += b;[等价 a = a + b; ]

a -= b;[等价 a = a - b; ]

特点

1）运算顺序从右往左 int num = a + b + c;

2）赋值运算符的左边 只能是变量,右边 可以是变量、表达式、常量值

int num = 20; int num2= 78 * 34 - 10; int num3 = a;

3）复合赋值运算符等价于下面的效果

比如：a+=3;等价于 a=a+3; 其他类推

4）复合赋值运算符会进行类型转换。

## 三元运算符

条件表达式 ? 表达式 1: 表达式 2;

运算规则：

1. 如果条件表达式为 true，运算后的结果是表达式 1；

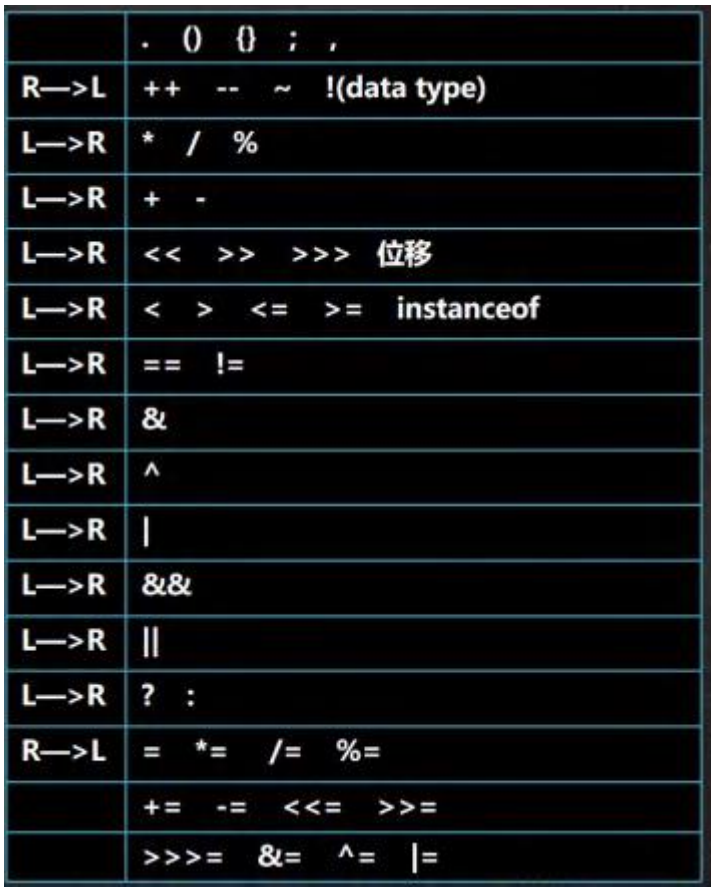
2. 如果条件表达式为 false，运算后的结果是表达式 2；

使用细节

1）表达式 1 和表达式 2 要为可以赋给接收变量的类型(或可以自动转换)

2）三元运算符可以转成 if--else 语句

## 运算符优先级




## 运算符命名规则

1）由 26 个英文字母大小写，0-9，_或$组成

2）不可以以数字开头

3）不可使用保留字与关键字

关键字

被 Java 语言赋予了特殊含义，用做 专门用途的字符串（单词），均为小写。

保留字

Java 保留字:现有 Java 版本 尚未使用,但 以后版本可能会作为关键字使用。自己命名标识符时要避免使用这些保留字 byValue、cast、future、 generic、inner、 operator、 outer、 rest、 var 、 goto 、const

4）严格区分大小写，无长度限制

5）不能包含空格

运算符命名规则

1）包名：多单词组成时所有字母都小写：aaa.bbb.ccc //比如 com.hsp.crm

2) 类名、接口名：多单词组成时，所有单词的首字母大写：XxxYyyZzz［大驼峰］

比如： TankShotGame

3) 变量名、方法名：多单词组成时，第一个单词首字母小写，第二个单词开始每个单词首字母大写：xxxYyyZzz［小驼峰， 简称 驼峰法］

比如： tankShotGame

4) 常量名：所有字母都大写。多单词时每个单词用下划线连接：XXX_YYY_ZZZ

比如 ：定义一个所得税率 TAX_RATE

## 键盘输入语句

在编程中，需要接收用户输入的数据，就可以使用键盘输入语句来获取。

Input.java ， 需要一个 扫描器(对象)，就是 Scanner

使用步骤：

1) 导入该类的所在包， java.util.*

2) 创建该类对象（声明变量）

3) 调用里面的功能

## 进制

对于整数，有四种表示方式：

二进制：0,1 ，满 2 进 1.以 0b 或 0B 开头。

十进制：0-9 ，满 10 进 1。

八进制：0-7 ，满 8 进 1. 以数字 0 开头表示。

十六进制：0-9 及 A(10)-F(15)，满 16 进 1. 以 0x 或 或 0X 开头表示。此处的 A-F 不区分大小写。

二进制-十进制

从最低位开始，将每个位上的数提取出来，乘 2 的（位数-1）次方，然后求和。

八进制-十进制

从最低位开始，将每个位上的数提取出来，乘 16 的（位数-1）次方，然后求和。

十六进制-十进制

从最低位(右边)开始，将每个位上的数提取出来，乘以 16 的(位数-1)次方，然后求和。

十进制-二进制

将该数不断除以 2，直到商为 0 为止，然后将每步得到的余数倒过来，就是对应的二进制。

十进制-八进制

将该数不断除以 8，直到商为 0 为止，然后将每步得到的余数倒过来，就是对应的八进制。

十进制-十六进制

将该数不断除以 16，直到商为 0 为止，然后将每步得到的余数倒过来，就是对应的十六进制。

二进制-八进制

从低位开始,将二进制数每三位一组，转成对应的八进制数即可。

二进制-十六进制

从低位开始，将二进制数每四位一组，转成对应的十六进制数即可。

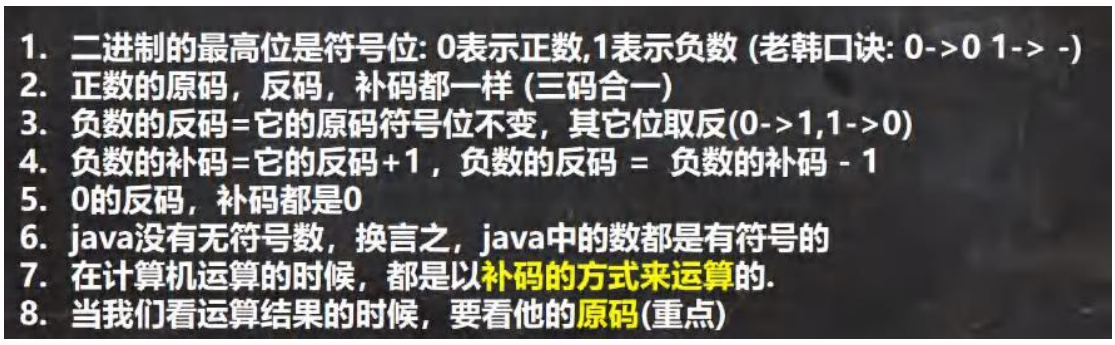
八进制-二进制

将八进制数每 1 位，转成对应的一个 3 位的二进制数即可。

十六进制-二进制

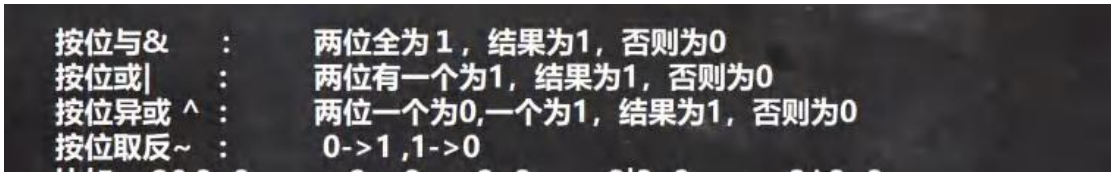
将十六进制数每 1 位，转成对应的 4 位的一个二进制数即可。

## 源码、补码、反码




## 位运算

&、|、^、~、>>、<<和 >>>




1）算术右移 >>：低位溢出,符号位不变,并用符号位补溢出的高位

2) 算术左移 <<: 符号位不变,低位补 0

3) >>> 逻辑右移也叫无符号右移,运算规则是: 低位溢出，高位补 0

4) 特别说明：没有 <<< 符号