运算符

接下来，给同学们讲解一个在开发中用得很多的一块内容，叫做运算符。

大家知道计算机是用来处理数据的，处理数据就少不了对数据的计算，想要对数据进行计算就必须用到运算符。

运算符就是参与运算的符号。Java提供的运算符有很多种，可以分为下面几种

- 基本算术运算符
- 自增自减运算符
- 赋值运算符
- 关系运算符
- 逻辑运算符
- 三元运算符

算术运算符

先从最基本的算术运算符开始学习，算术运算符有 `+ - * / %` ，其中`*`表示乘法，`/`表示除法，`%`表示取余数

需要我们注意以下几点

```
/：两个整数相除，结果也是一个整数
%：表示两个数相除，取余数
```

符号	作用	说明
+	加	参考小学一年级内容
-	减	参考小学一年级内容
*	乘	参考小学二年级内容，与“x”相同
/	除	与“÷”相同，注意：再java中两个整数相除结果还是整数
%	取余	获取的是两个数据做除法的余数

需要我们注意的是：`+`符号除了用于加法运算，还可以作为连接符。**`+`符号与字符串运算的时候是用作连接符的，其结果依然是一个字符串。**

下面通过代码演示一下各种算术运算符的运算效果

```java
public class OperatorDemo1 {
    public static void main(String[] args) {
        // 目标：掌握基本的算术运算符的使用。
        int a = 10;
        int b = 2;
        System.out.println(a + b);  // 12
        System.out.println(a - b);  // 8
        System.out.println(a * b);  // 20
        System.out.println(a / b);  // 5 整数 / 整数 = 整数
        System.out.println(5 / 2);  // 2
        System.out.println(5.0 / 2); // 2.5
        int i = 5;
        int j = 2;
        System.out.println(1.0 * i / j); // 2.5

        System.out.println(a % b);  // 0
        System.out.println(3 % 2);  // 1

        System.out.println("-----------------------------------");

        // 目标2：掌握使用+符号做连接符的情况。
        int a2 = 5;
        System.out.println("abc" + a2);  // abc5
        System.out.println(a2 + 5);  // 10
        System.out.println("kaifamiao" + a2 + 'a');  // kaifamiao5a
        System.out.println(a2 + 'a' + "kaifamiao");  // 5akaifamiao 102kaifamiao
    }
}
```

自增自减运算符

接下来，学习一种比较常用的运算符：`++`和`--`

`++`读作自增，`--`读作自减； 运算规则如下

符号	作用
自增：++	放在某个变量前面或者后面，对变量自身的值加1
自减： --	放在某个变量前面或者后面，对变量自身的值减一

需要我们注意的是，自增自减只能对变量进行操作，不能操作字面量。具体使用时也分为两种情况，如下:

```
1.单独使用: ++或者--放在变量前面没有区别
       int a = 10;
        a++;  // 1
        --a;  // 10
        System.out.println(a); // 10

2.混合使用: ++或者--放在变量或者前面运算规则稍有不通过  ++ 在后 先赋值 再自增 ++ 在前 先自增再赋值
    int a = 10;
    int b = a++;
    int c = ++a;
    System.out.println(a); // 12
    System.out.println(b); // 10
    System.out.println(c); // 12


    int x = 10;
    int y = --x;
     x = 9 y = 9
```

下面通过代码演示一下 `++` 和 `--` 的用法

```
public class OperatorDemo2 {
    public static void main(String[] args) {
        // 目标: 掌握自增自减运算符的使用。
        int a = 10;
        ++a;
        System.out.println(a); // 11

        --a;
        System.out.println(a); // 10

        System.out.println("-----------------------------------");

        int i = 10;
        int rs = ++i;
        System.out.println(rs); // 11
        System.out.println(i);  // 11

        int j = 10;
        int rs2 = j++;
```

```
        System.out.println(rs2); // 10
        System.out.println(j);   // 11


        int num1 = 2;
        int num2 = 3;
        int num3 = ++num1;  // num3 = 3 num1 = 3
        System.out.println(++num1 + num2++ + num3);  // 10 num1 = 4 num2 = 4 num3 = 3
        System.out.println(--num1 + num2-- + --num3);// 9  num1 = 3 num2 = 3 num3 = 2
        System.out.println(num1-- + num2-- + --num3);// 7  num1 = 2 num2 = 2 num3 = 1
        System.out.println(num1);
        System.out.println(num2);
        System.out.println(num3);
    }
}
```

赋值运算符

接下来，我们学习赋值运算符。基本的赋值运算符其实就是`=`号，意思就是把右边的数据赋值给左边的变量。

```
int a = 10; // 将数据10赋值给左边的变量a
```

除了基本的赋值运算符，我们这里主要学习一下扩展的赋值运算符。有`+=` `-=` `*=` `/=` `%=`

符号	用法	作用	底层代码形式
+=	a+=b	加后赋值	a = (a的类型)(a + b);
-=	a-=b	减后赋值	a = (a的类型)(a - b);
*=	a*=b	乘后赋值	a = (a的类型)(a * b);
/=	a/=b	除后赋值	a = (a的类型)(a / b);
%=	a%=b	取余后赋值	a = (a的类型)(a % b);

我们以`+=`为例来看一下它的运算规则，其他的运算符运算同理分析即可

```
int a = 10;
a += 5;  // a = a + 5;
System.out.println(a);  // 15
```

下面通过一个首发红包的例子给大家演示一下

```
public class OperatorDemo3 {
    public static void main(String[] args) {
        // 目标：掌握扩展赋值运算符的使用。
        // +=
        double a = 9.5;
        int b = 520;
        a += b;  // a = (double)(a + b);
        System.out.println(a); // 529.5

        // -= 需求：发红包
        double i = 600;
        double j = 520;
        i -= j;  // i = (double)(i - j);
        System.out.println(i); // 80

        int m = 10;
        int n = 5;
        // m *= n; // 等价形式：  m = (int)(m * n)
        // m /= n; // 等价形式：  m = (int)(m / n)
        m %= n;    // 等价形式：  m = (int)(m % n)
        System.out.println(m); // 0
    }
}
```

学完扩展赋值运算符的基本使用之后，接下来我们看一个面试题

```
问题1：下面的代码否有问题？
    byte x = 10;
    byte y = 30;
    x = x + y;  // byte char short 转换成int类型进行计算
    System.out.println(x); // 报错

问题2：下面的代码是否有问题？
    byte x = 10;
    byte y = 30;
    // x+=3;
    x += y; // x = (byte) (x + y);
    System.out.println(x); // 40
```

到这里赋值运算符就学习完了，稍微总结一下

```
1.基本赋值运算符：
    =符号含义：  把右边的值赋值给左边的变量

2.扩展赋值运算符：
    += -= *= /= %=符号含义：将右边的数据和左边的变量相加、相减、相乘、相除、取余数后，将结果重新赋值给左边的变量。

    int j = 2;
    int i = 1;
    // i = j ++; // i = 2 j = 3
    i = ++ j;  // j = 3 i = 3


    int b = 2;
    b = b++;
    System.out.println(b); // 2

    int b = 2;
    b = ++b;
    System.out.println(b); // 3
```

关系运算符

接下来我们学习一个，在实际代码中用得很多，但是又非常简单的运算符，叫关系运算符。关系运算符（也叫比较运算符）。

下图是每一种关系运算符的符号及作用，每一个关系运算符的结果都是boolean值（true、false）

符号	例子	作用	结果
>	a > b	判断a是否大于b	成立返回true，不成立返回false
>=	a >= b	判断a是否大于等于b	成立返回true，不成立返回false
<	a < b	判断a是否小于b	成立返回true，不成立返回false
<=	a <= b	判断a是否小于等于b	成立返回true，不成立返回false
==	a == b	判断a是否等于b	成立返回true，不成立返回false
!=	a != b	判断a是否不等于b	成立返回true，不成立返回false

下面通过代码来家演示一下，各种关系运算符的效果

```
public class OperatorDemo4 {
```

```
    public static void main(String[] args) {
        // 目标：掌握关系运算符的基本使用。
        int a = 10;
        int b = 5;
        boolean rs = a > b;
        System.out.println(rs);  // true

        System.out.println(a >= b);  // true
        System.out.println(2 >= 2);  // true
        System.out.println(a < b);   // false
        System.out.println(a <= b);  // false
        System.out.println(2 <= 2);  // true
        System.out.println(a == b);  // false
        System.out.println(5 == 5);  // true

        // 注意了：判断是否相等一定是用 == ，=是用来赋值的。
        System.out.println(a = b);  // 5 报错
        System.out.println(a != b); // true
        System.out.println(10 != 10); // false

    }
}

```

现在我们只需要知道每一种关系运算符的运算效果就行了，**关于关系运算符的实际运用需要在后面学习了流程控制语句才能实际用到。**

关系运算符在程序中常用于条件判断，根据条件判断的结果是true还是false，来决定后续该执行哪些操作。

逻辑运算符

学习完关系运算符之后，接下来我们学习一下逻辑运算符。我们来看一下逻辑运算符有哪些。

逻辑运算符是用来将多个条件放在一起运算，最终结果是true或者false

符号	称呼	例子	运算逻辑
&	逻辑与	2 > 1 & 3 < 2	多个条件必须是true，结果才是true；有一个是false，结果就是false
|	逻辑或	2 > 1 | 3 > 5	多个条件中主要有一个是true。结果就是true
!	逻辑非	!(2 > 1)	就是取反：你真我假，你假我真。
^	逻辑异或	2 > 1 ^ 3 > 1	前后条件的结果相同，就直接返回false，前后条件的结果不同，才返回true
&&	短路与	2 > 10 && 3 > 2	判断结果与"&"一样，过程不同：左边为false，右边就不执行
||	短路或	2 > 1 || 3 < 5	判断结果与"|"一样，过程不同：左边为true，右边就不执行

下面我们通过几个案例来演示一下逻辑运算符的使用

```
// 需求1：要求手机必须满足尺寸大于等于6.95，且内存必须大于等于8.
// 需求2：要求手机要么满足尺寸大于等于6.95，要么内存必须大于等于8.
```

案例：键盘输入一个年份，判断是否是闰年

```
import java.util.Scanner;
public class IsHeap {

    public static void main(String[] args) {

        // 能被4整除但不能被100整除，或者被400整除
        Scanner sc = new Scanner(System.in);
        System.out.println("请输入一个年份");
        int year = sc.nextInt();
        // 能被4整除  year % 4 == 0
        // 不能被100 整除  year % 100 != 0
        // 能被400整除  year % 400 == 0
        boolean a = year % 4 == 0;
        boolean b = year % 100 != 0;
        boolean c = year % 400 == 0;
        boolean res = a && b || c;
        System.out.println("你输入的" + year + "年是否是闰年呢？" +
res);
```

```
18      }
19  }
```

到这里关于逻辑运算符的规则记学习完了，给你一个运算表达式你能分析出结果就行。**至于逻辑运算符的实际运用，需要在学习了流程控制语句之后，才能实际用到。**

逻辑运算符在程序中常用于组合几个条件判断，根据条件判断的结果是true还是false，来决定后续该执行哪些操作。

练习：键盘输入一个数字，判断是否是奇数。

位运算符

在计算机内部，数据以二进制位（0和1）的形式存储和处理。计算机通过将数据转换为二进制形式，将其存储在内存中，并对其进行操作和计算。不同的数据类型在内存中占据不同长度的二进制位，以便正确地表示和处理数据。

1. 原码

原码是一种最简单的机器数表示法，其中最高位表示符号位，其余位表示该数的二进制的绝对值。

2. 反码

正数的反码与原码相同，而负数的反码是其原码除符号位外，按位取反的结果。

3. 补码

正数的补码等于其原码，而负数的补码等于其反码加1

运算符	含义	运算规则
&	按位与	两个操作数都是1，结果才是1。
|	按位或	两个操作数任意一个是1，结果就是1。
^	按位异或	两个操作数相同，结果为0；两个操作数不同，结果为1.
~	按位非	操作数为1，结果为0；操作数为0，结果为1。
<<	左移	右侧空位补0。
>>	右移	左侧空位补最高位，即符号位。
>>>	无符号右移	左侧空位补0。

```
计算5&6的结果
计算5|6的结果
计算3<<2的结果
计算16>>2的结果
计算-12 | 2的结果
```

三元运算符

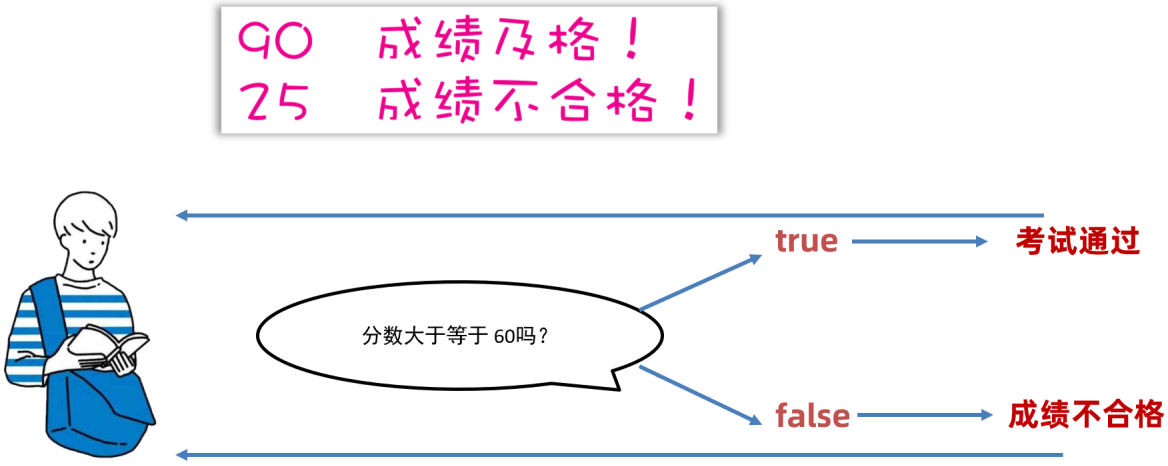
接下来，我们学习今天最后一种运算符，叫做三元运算符。

先认识一下三元运算符的格式：

```
数据类型 变量名 = 关系表达式 ? 值1 : 值2;
```

三元运算的执行流程：首先计算关系表达式的值，如果关系表达式的值为true，则返回值1；如果关系表达式的值为false, 则返回值2；

如下图所示：判断学生的成绩是否>=60，如果为true，就考试通过；如果为false，就成绩不合格。



接下来通过代码来演示一下，目的是让大家掌握三元运算符的格式和执行流程。

```java
public class OperatorDemo6 {
    public static void main(String[] args) {
        // 目标：掌握三元运算符的基本使用。
        double score = 58.5;
        String rs = score >= 60 ? "成绩及格" : "成绩不及格";
        System.out.println(rs);

        // 需求2：找出2个整数中的较大值，并输出。
        int a = 99;
        int b = 69;
        int max = a > b ? a : b;
        System.out.println(max);

        // 需求3：找3个整数中的较大值。
        int i = 10;
        int j = 45;
        int k = 34;

        // 找出2个整数中的较大值。
        int temp = i > j ? i : j;
        // 找出temp与k中的较大值。
        int max2 = temp > k ? temp : k;
        System.out.println(max2);
    }
}

```

运算优先级

最后我们在给大家介绍一下运算符的优先级，如果你想要知道各个运算符的优先级，哪些先算哪些后算，可以参考下面这张图

优先级	运算符
1	()
2	!、-、++、--
3	*、/、%
4	+、-
5	<<、>>、>>>
6	<、<=、>、>=、instanceof
7	==、!=
8	&
9	^
10	|
11	&&
12	||
13	?:
14	=、+=、-=、*=、/=、%=、&=、

从图中我们发现，&&运算比||运算的优先级高，所以&&和||同时存在时，是先算&&再算||；

比如下面的代码

```
// 这里&&先算 相当于 true || false 结果为true
System.out.println(10 > 3 || 10 > 3 && 10 < 3); // true
```

最后给大家说一下，在实际开发中，其实我们很少考虑运算优先级， 因为如果你想让某些数据先运算，其实加`()`就可以了，这样阅读性更高。

```
//有括号先算 相当于 true && false 结果为false
System.out.println((10 > 3 || 10 > 3) && 10 < 3); //false
```