# 注释、关键字、标识符

## 注释

* 注释概述
  + 用于解释说明程序的文字
* 注释分类
  + 单行注释

格式： //注释文字

* + 多行注释

格式： /\* 注释文字 \*/ 选中之后ctrl + shift + /

* + 文档注释

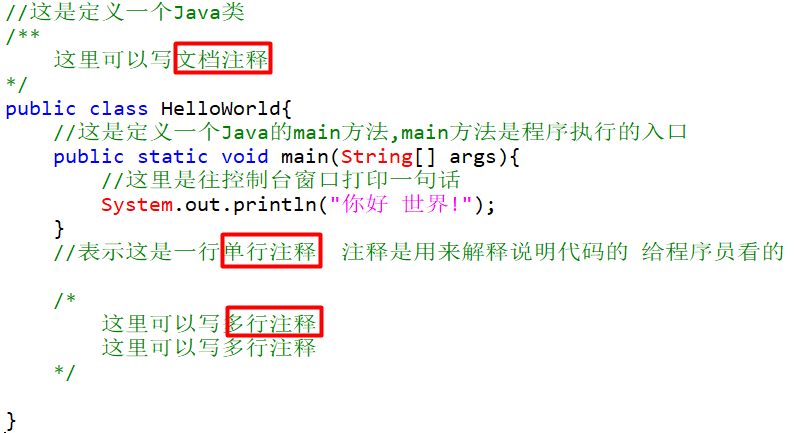
格式：/\*\* 注释文字 \*/ **用来生成帮助文档(非重点) Alt + shift + J**

1. 注释的作用：

1、对程序进行解释说明：要求代码中，有30%行数都是用来写注释的

2、对程序的思路进行描述

3、对程序进行调试：通过注释的方式，快速定位出现错误的代码位置

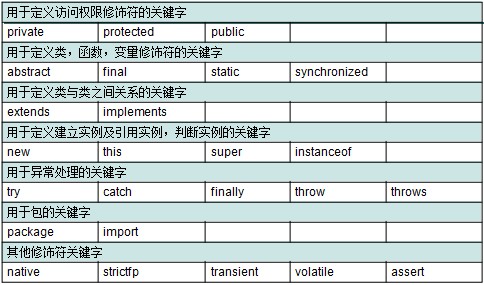


## 关键字

Eclipse中为紫色的都是关键字！

* 关键字概述
  + 关键字是被Java语言赋予特殊含义，具有专门用途的单词，比如之前接触的class，public，static，void均为Java已经预设好的。
  + 你可以先把Java关键字理解为“命令”！
* 关键字特点
  + 组成关键字的字母**全部小写**
  + 常用的代码编辑器,针对关键字有特殊的颜色标记，非常直观，所以我们不需要去死记硬背，在今后的学习中重要的关键字也会不断的出来。
* java中的所有关键字展示





特点：

1、全部小写

2、自己起名字的时候，不要和关键字重名

3、有两个**保留字**：goto、const，在当前版本不使用，但是以后是否使用未知。

**注意：边学边积累（不要背）**

## 标识符

* 什么是标识zhi符

其实在Java程序中，除了关键字以外基本上都是标识符了。

其实标识符就是名称的意思，所有的名称都统称为标识符。

Java中经常要定义类、方法、变量，在定义他们时总要给它们起名字，这些名字就是标识符。我们要学习的就是标识符的规范。

* 组成元素
* 英文字符: a-zA-Z
* 数字: 0-9
* 符号: \_与$（中文可以，但是不用）
* 标识符规则(规范)
* 数字不能开头
* 不可以使用关键字
* 严格区分大小写，不限制长度起名时，尽量达到**见名知意**
* 类名：首字母大写，其他字母小写，若有多个单词，每个单词的首字母大写（后面学习）
* 方法名：第一个单词字母小写，后面的单词首字母大写，其他字母小写
* 变量：一个单词首字母小写。多个单词，从第二个单词开始每个单词首字母大写（驼峰形式）非强制要求，约定俗称的规则

# 数据类型、常量、变量

## 程序概述

软件是什么？其实软件就是用来处理数据的程序。可以把软件分为两个部分，一个部分是“数据”；另一个部分是“数据处理的逻辑”。

**程序 = 数据 + 逻辑(算法)**

什么是数据？用电商来举例，用户就是数据，商品也是数据。那么什么是数据处理逻辑呢？购买商品的流程就是数据处理逻辑了。

程序开发离不开数据，其实说白了程序就是用来处理数据的。在Java中我们要学习数据的类型及数据量。

## 数据类型

### 数据类型分类

String以后再讲，先了解基本数据类型。

在生活中数据是有类型的，例如人的姓名是字符串类型，人的年龄是正整数类型。在Java中数据也是有类型的（任何数据必然会有类型），下面是Java中数据的类型：



* 基本数据类型是Java语言中内置的类型，分别是整数类型、小数类型、字符类型、布尔类型。这四类基本类型是最简单、最基础的类型。
* 引用数据类型是强大的数据类型，它是基于基本数据类型创建的。JavaSE中提供了一个超级类库，类库中包含了近万种引用数据类型。不过现在我们先要学习的是基本类型！

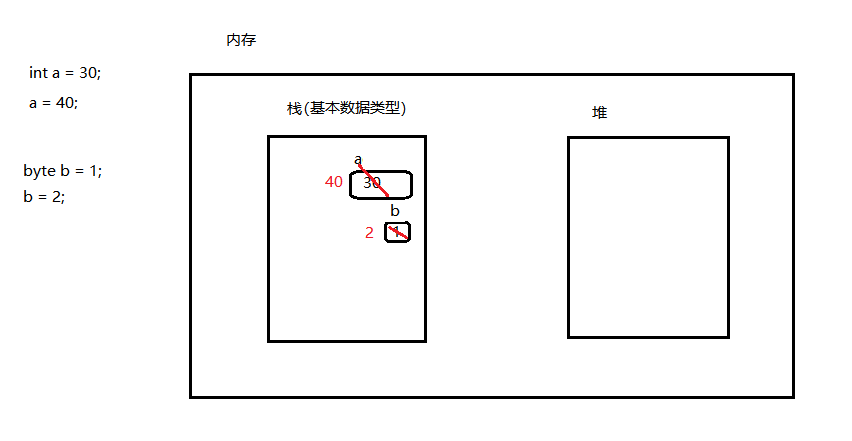
### 基本数据类型示例

* 整数类型
  + 十进制表示方式：正常数字 如 13、25等
  + 二进制表示方式：以0b(0B)开头 如0b1011 、0B1001
  + 八进制表示方式：以0开头 如01、07、0721
  + 十六进制表示方式：以0x(0X)开头 数字以0-9及A-F组成 如0x23A2、0xa、0x10
* 小数类型
  + 如1.0、-3.15、3.168等
* 布尔类型
  + true、false
* 字符类型
  + 如'a'，'A', '0', '家'
  + 字符必须使用’’ 包裹，并且其中只能且仅能包含一个字符。
* 字符串类型
  + 字符串String类型是一种引用类型，我们先了解作为常量类型的使用方式
  + 如“我爱Java”，“0123”，“”，“null”
  + 字符串必须使用“”包裹，其中可以包含0~N个字符。

### 基本数据类型占用大小和表示范围

* 基本数据类型(4类8种)

|  |  |  |  |
| --- | --- | --- | --- |
| **四类** | **八种** | **字节数** | **数据表示范围** |
| **整型**  **(精确)** | **byte** | 1 | -128～127 |
| **short** | 2 | -32768～32767 |
| **int(默认)** | 4 | -2147483648～2147483647 |
| **long** | 8 | -263～263-1 |
| **浮点型**  **(不精确)** | **float** | 4 | -3.403E38～3.403E38 |
| **double(默认)** | 8 | -1.798E308～1.798E308 |
| **字符型** | **char** | 2 | 表示一个字符，如('a'，'A'，'0'，'家') |
| **布尔型** | **boolean** | 1 | 只有两个值true与false |



## 常量

### 概念

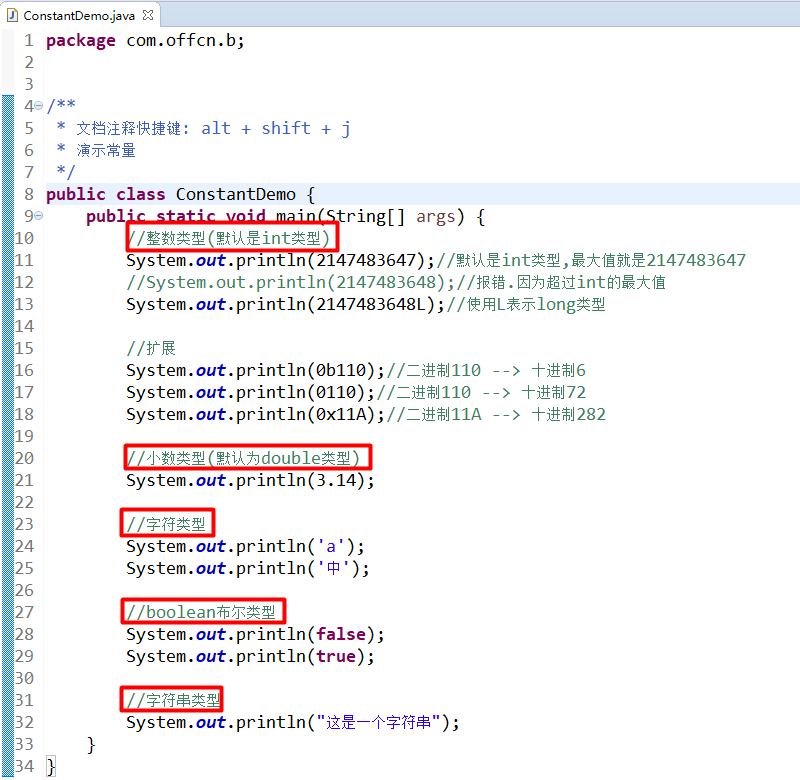
Java中的数据量分为常量和变量，我们先来学习常量！

常量就是不变的数据量，例如100就是常量，任何数据量都有其类型，那么100这个常量是什么类型呢？回忆一下基本类型中有四类，分别是整数、小数（浮点数）、字符、布尔。



### 代码演示

回顾HelloWorld案例，其中System.out.println(“Hello World!”)的圆括号中放的就是一个字符串。那么我们是否可以把圆括号中的字符串换成其他字符串呢？答案是可以的！不只是可以换成其他字符串，还可以把圆括号中的字符串换成其他类型的常量，例如：



总结：

1. **System.out.println()输出的整数默认为int类型十进制**
2. **System.out.println()输出的小数类型默认为double**

## 变量

### 概念

变量：在程序中可以变化的量称为变量(常量是固定不变的量)

数学中，可以使用字母代替数字运算,例如 x=1+5 或者 6=x+5。

程序中，可以使用字母保存数字的方式进行运算，提高计算能力，可以解决更多的问题。比如x保存5，x也可以保存6，这样x保存的数据是可以改变的，也就是我们所讲解的变量。

Java中要求一个变量每次只能保存一个数据，必须要明确保存的数据类型。

### 定义格式

格式：数据类型 变量名 = 数据值；

如: int a = 100;

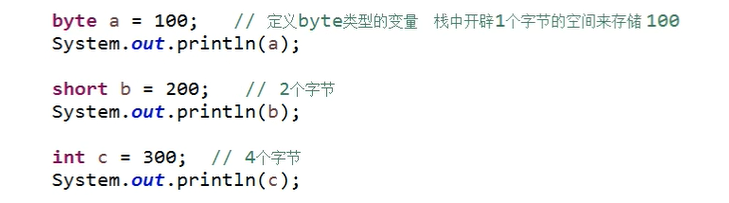
其中int是数据类型，指定了变量只能存储整数，而且指定了存储范围为-2147483648～2147483648。

其中a表示变量名，变量名是标识符，这说明只要是合法的标识符都可以用来做变量名。在程序中可以通过变量名来操作变量

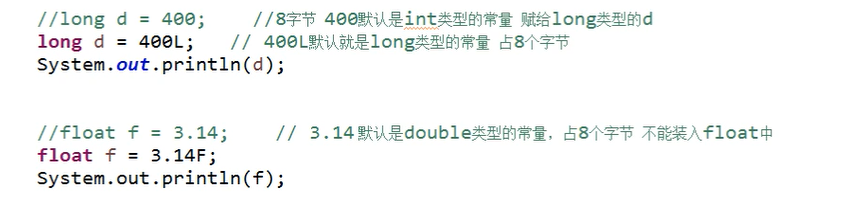
其中“=100”是给变量赋值，即向a变量中写入100。

注意，给变量赋的值一定要与类型符合，也就是说int类型只能存储整数，而且必须是在-2147483648～2147483648范围内的整数。

### 代码演示



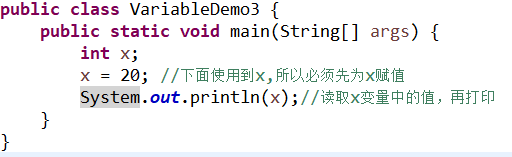
例外：



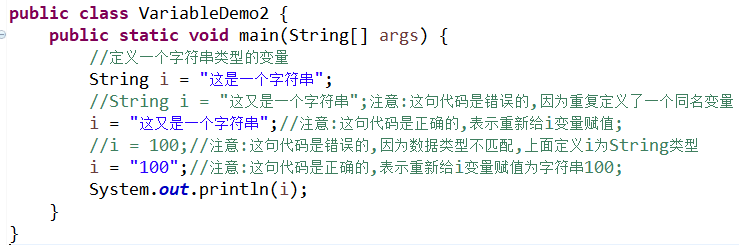
### 注意事项

我们使用变量的时候需要注意，要满足变量的使用规则才可以使用的，我们来看看都有哪些注意事项。

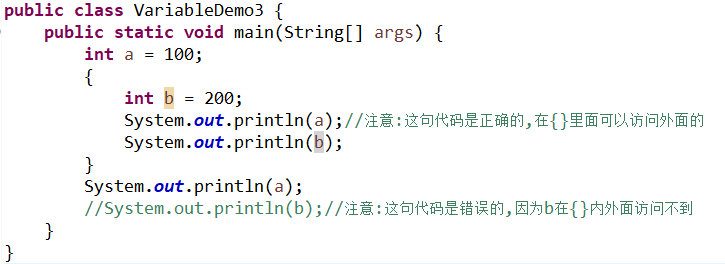
* 变量使用的注意事项
  + 变量定义后可以不赋值，使用时再赋值。不赋值不能使用。



* + 变量不能重复定义,并且赋值时类型得匹配。



* + 变量使用时有作用域的限制。



## 数据类型转换

### 概念

　　不同类型的变量是否可以在一起运算呢？答案是可以的，但要先进行类型转换再运算。下面我们来学习一下类型转换。

其实，我们所学习的数据，它的表示方式是可以灵活多变的，比如把小数转换成整数的操作

* 关于数据类型转换有两种方式：
* 自动类型转换

表示范围小的数据类型（如byte）可以直接转换成范围大的数据类型（如int），这种方式称为自动类型转换

格式：范围大的数据类型 变量 = 范围小的数据类型值；

* + 强制类型转换

表示范围大的数据类型（如int）不可以转换成范围小的数据类型（如byte），但可以强制转，这种方式称为强制类型转换

格式：范围小的数据类型 变量 = (范围小的数据类型) 范围大的数据类型值;

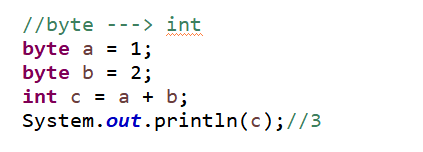
* 转换规律

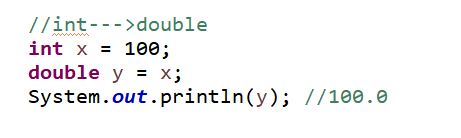
各种数据类型按照数据范围从小到大依次列出：范围小的类型向范围大的类型提升：byte < short = char‐‐>int‐‐>long‐‐>float‐‐>double

byte、short、char 运算时直接提升为int，同样的道理，当一个int 类型变量和一个double 变量运算时， int 类型将会自动提升为double 类型进行运算

### 代码演示

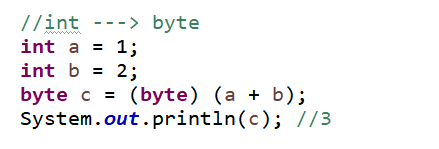
1. **小转大，直接转**



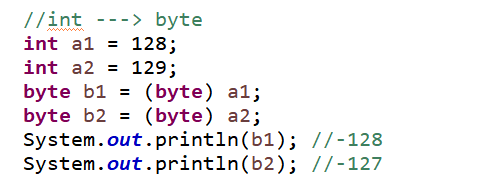


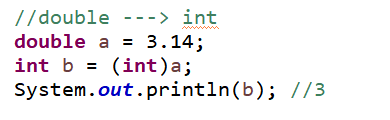
1. **大转小，强制转换（强转），丢失精度，不建议**

**在前面加想转成的类型，如（byte）**



看似没问题，其实可能造成丢失精度。第一位符号位。例如：





## ASCII编码表

### 概念

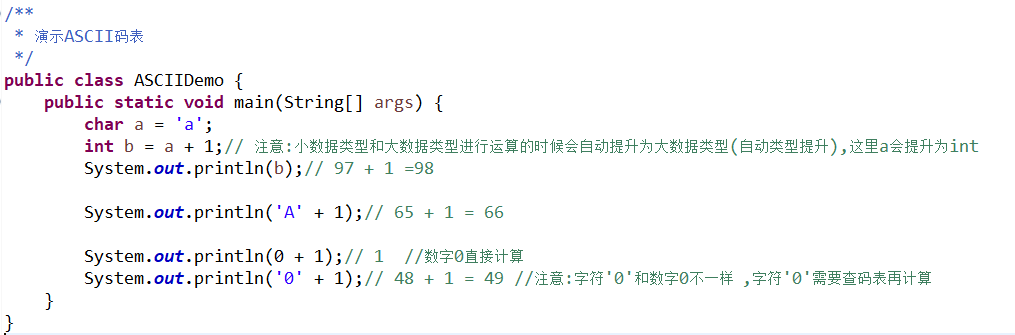
在计算机的内部都是二进制的0、1数据，如何让计算机可以直接识别人类文字的问题呢？就产生出了编码表的概念。

编码表 ：就是将人类的文字和一个十进制数进行对应起来组成一张表格

将所有的英文字母，数字，符号都和十进制进行了对应，因此产生了世界上第一张编码表ASCII（American Standard Code for Information Interchange 美国标准信息交换码）。



### 代码演示



在char类型和int类型计算的过程中，char类型的字符先查询编码表，得到97，再和1求和，结果为98。char类型提升为了int类型。char类型内存2个字节，int类型内存4个字节。

# 运算符

## 概述

* 运算符
  + 对常量和变量进行操作的符号称为运算符
* 表达式
  + 用运算符把常量或者变量连接起来符号java语法的式子就可以称为表达式。不同运算符连接的式子体现的是不同类型的表达式。
  + 定义两个int类型的变量a,b，做加法(a + b)
* 常用运算符
  + 算术运算符
  + 赋值运算符
  + 关系运算符
  + 逻辑运算符
  + 三元运算符

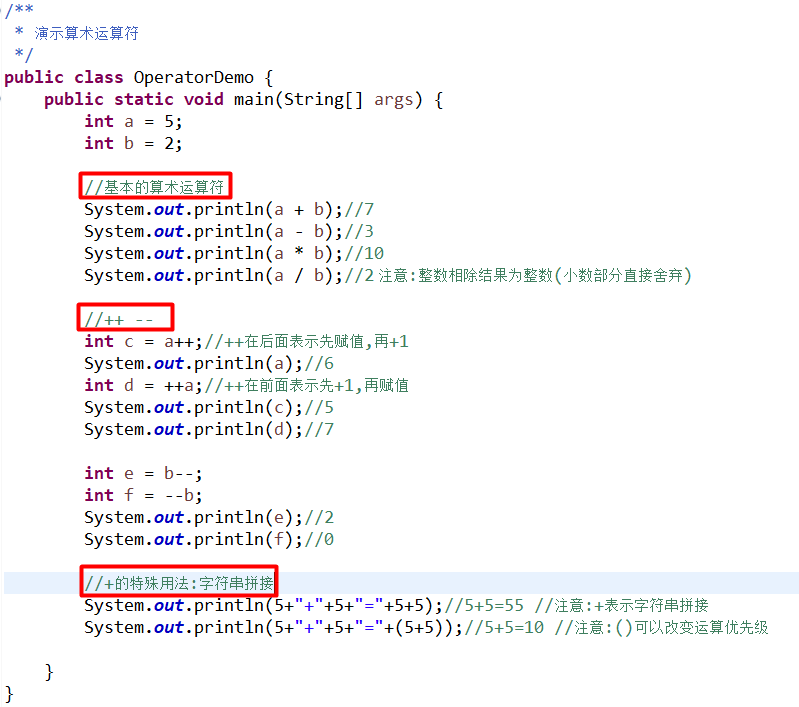
## 算术运算符

### 概念

和数学中的算术运算符基本一致,用法如下:

|  |  |  |  |
| --- | --- | --- | --- |
| **运算符** | **运算规则** | **范例** | **结果** |
| + | 正号 | +3 | 3 |
| + | 加 | 2+3 | 5 |
| + | 连接字符串 | “中”+“国” | “中国” |
| - | 负号 | int a=3; -a | -3 |
| - | 减 | 3-1 | 2 |
| \* | 乘 | 2\*3 | 6 |
| / | 除 | 5/2 | 2 |
| % | 取模(取余) | 5%2 | 1 |
| ++ | 自增 | int a=1; a++ / ++a | 2 |
| -- | 自减 | int b=3; b-- / --b | 2 |

### 基本演示



### ++、--

++ 运算，变量自己增长1

-- 运算，变量自己减少1

* 独立运算：

变量前++ ，例如 ++i

变量后++ ，例如 i++

变量在独立运算时， 前++ 和后++ 没有区别 。

|  |
| --- |
| **public** **static** **void** main(String[] args) {  **int** a = 1;  a++;  System.***out***.println(a);//2  ++a;  System.***out***.println(a);//3  } |

* 和其他变量放在一起运算，就产生了不同：

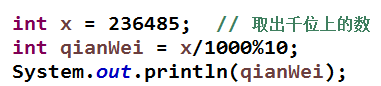
前++计算方式：变量a自己加1，将加1后的结果赋值给b，也就是说a先计算。a和b的结果都是2

后++计算方式：变量a先把自己的值1，赋值给变量b，此时变量b的值就是1，变量a自己再加1。a的结果是2，b的结果是1。

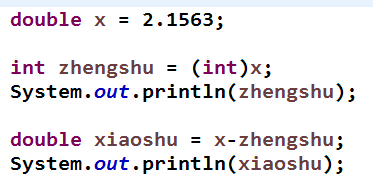
|  |
| --- |
| **public** **static** **void** main(String[] args) {  **int** a = 1;  //int b = ++a;//a:2 b:2 注意:++在前表示先计算++ 再赋值  **int** b = a++;//a:2 b:1 注意:++在后表示先赋值 再计算++  System.***out***.println(a);  System.***out***.println(b);  } |

### 练习

取出一个整数的千位上的数



浮点数 取出整数部分，取出小数部分



## 赋值运算符

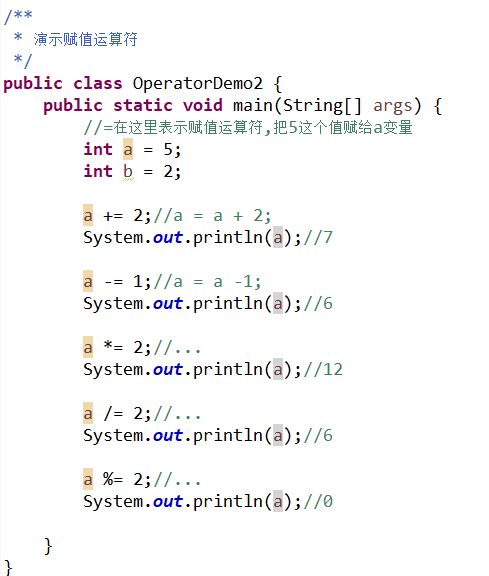
### 概念

所谓赋值运算符，就是将符号右边的值，赋给左边的变量。

如下所示:

|  |  |  |  |
| --- | --- | --- | --- |
| **运算符** | **运算规则** | **范例** | **结果** |
| = | 赋值 | int a=2 | 2 |
| += | 加后赋值 | int a=2，a+=2 | 4 |
| -= | 减后赋值 | int a=2，a-=2 | 0 |
| \*= | 乘后赋值 | int a=2，a\*=2 | 4 |
| /= | 整除后赋值 | int a=2，a/=2 | 1 |
| %= | 取模后赋值 | int a=2，a%=2 | 0 |

### 代码演示



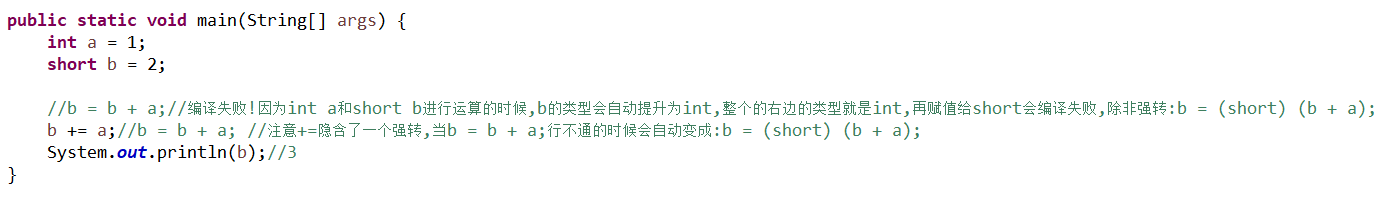
### 面试题

* 面试题1

下面代码的运行结果为?

|  |
| --- |
| **public** **static** **void** main(String[] args) {  **int** a = 1;  **short** b = 2;    b = b + a;//3  //b += a;//3  System.***out***.println(b);  } |

* + 分析:

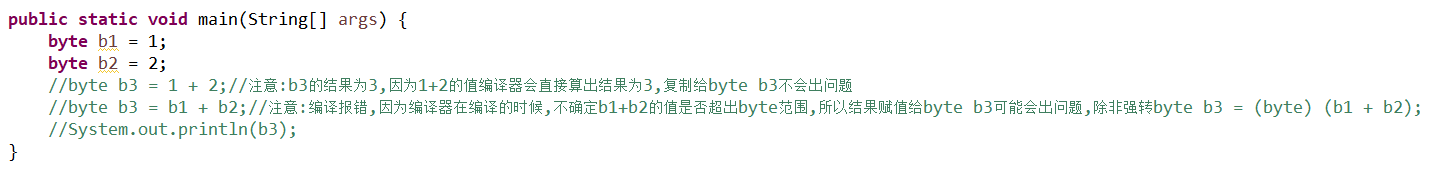


* 面试题2

下面代码的运行结果为?

|  |
| --- |
| **public** **static** **void** main(String[] args) {  **byte** b1 = 1;  **byte** b2 = 2;  //**byte** b3 = 1 + 2;  //byte b3 = b1 + b2;  //System.***out***.println(b3);  } |

* 分析



分析：

b3=1+2 ，1和2是常量，为固定不变的数据，在编译的时候（编译器javac），已经确定了1+2的结果并没有超过byte类型的取值范围，可以赋值给变量b3，因此b3=1+2 是正确的。

**常量优化机制：**

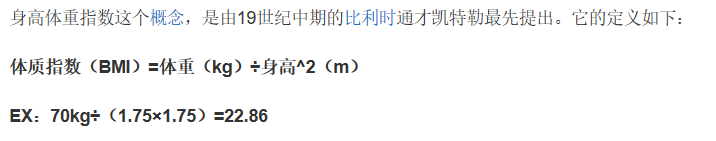
**有一些计算，非常简单，例如常量和常量的计算就非常简单，在编译阶段就可以把这些简单的运算计算完。**

反之， b4=b2+b3 ，b2和b3是变量，变量的值是可能变化的，在编译的时候，编译器javac不确定b2+b3的结果是什么，因此会将结果以int类型进行处理，所以int类型不能赋值给byte类型，因此编译失败。

### 小练习-计算BMI（作业）

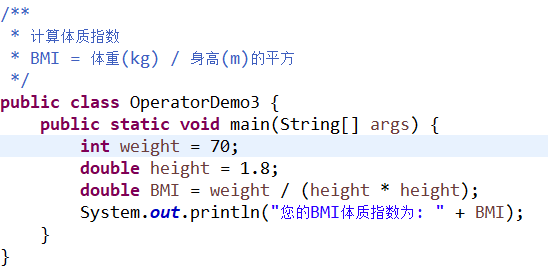
* BMI概念

BMI = 体重(kg) / 身高(m)的平方





* 代码演示



**浮点数运算问题：浮点数做数学运算，计算结果不精确！**

## 关系运算符(比较运算符)

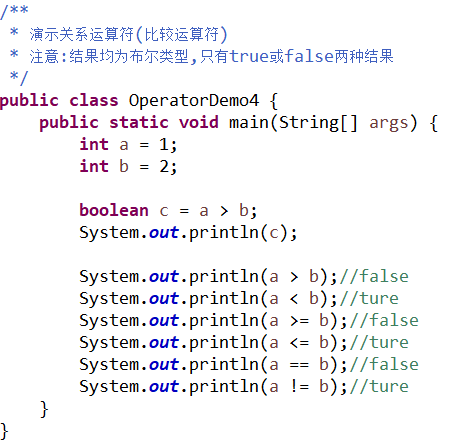
### 概念

关系运算符，是两个数据之间进行比较的运算，运算结果都是布尔值true(正确的,真的,满足条件的) 或者false(错误的,假的,不满足条件的) 。

如下所示:

|  |  |  |  |
| --- | --- | --- | --- |
| **运算符** | **运算规则** | **范例** | **结果** |
| == | 相等于 | 4==3 | false |
| != | 不等于 | 4!=3 | true |
| < | 小于 | 4<3 | false |
| > | 大于 | 4>3 | true |
| <= | 小于等于 | 4<=3 | false |
| >= | 大于等于 | 4>=3 | true |

### 代码演示



## 逻辑运算符

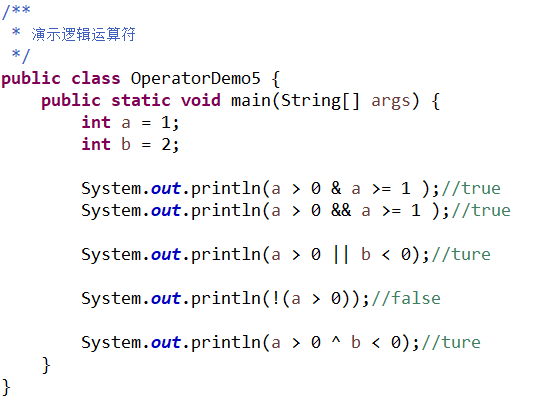
### 概念

逻辑运算符，是用来连接两个布尔类型结果的运算符，运算结果都是布尔值true 或者false

如下所示:

|  |  |  |  |
| --- | --- | --- | --- |
| **运算符** | **运算规则** | **范例** | **结果** |
| & | 与 | false&true | False |
| | | 或 | false|true | True |
| ^(了解) | 异或  (相同:false,不同:true) | true^flase | True |
| ! | 非 | !true | Flase |
| && | 短路与 | false&&true | False |
| || | 短路或 | false||true | True |

### 代码演示



### && 和 ||

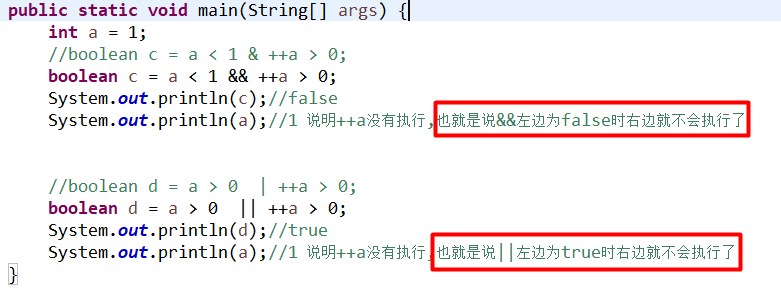
&& 短路与

短路特点：符号左边是false，右边不运算

|| 短路或

短路特点： 符号左边是true，右边不运算

* 结论:实际开发中都是使用短路与&&和短路或||,因为效率更高



## 三元运算符

### 概念

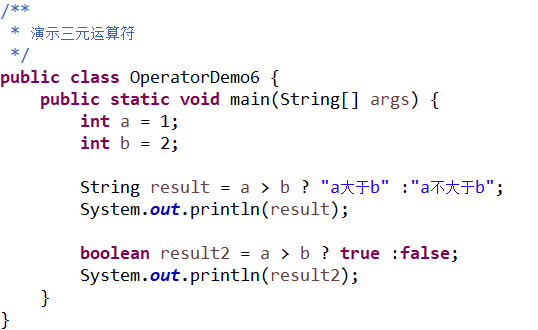
* 三元运算符格式：

数据类型 变量名 = 布尔类型表达式？值1：值2

* 三元运算符计算方式：

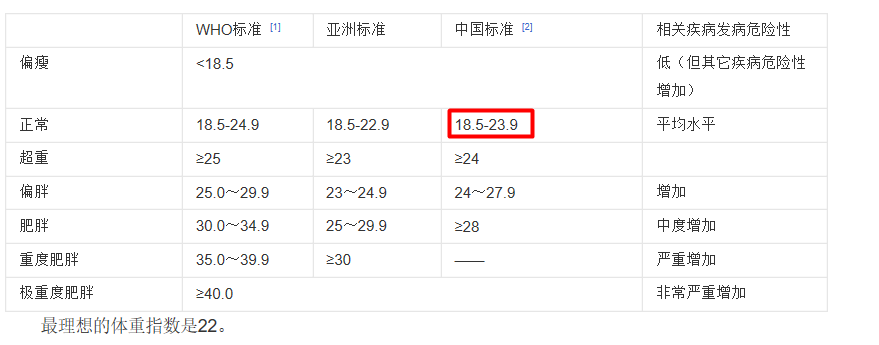
布尔类型表达式值是true时，三元运算符整体值为值1，赋值给变量。

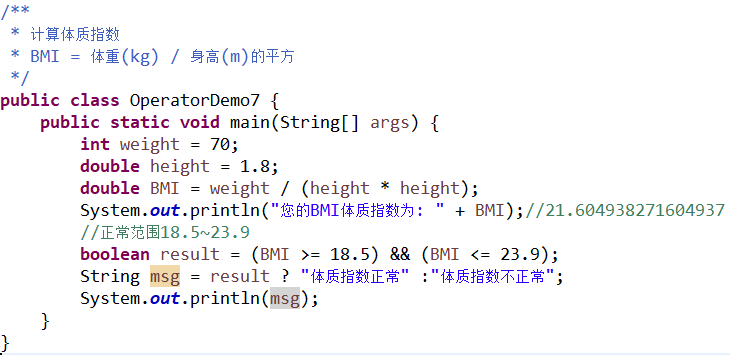
布尔类型表达式值是false时，三元运算符整体值为值2，赋值给变量。



### 小练习-计算体质指数并给提示（作业）

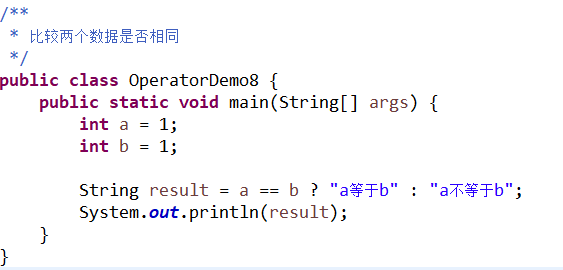
BIM = 体重(kg) / 身高(m)的平方



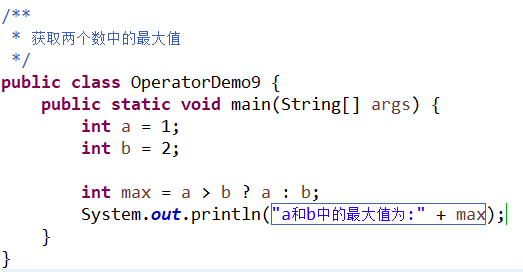


## 综合练习

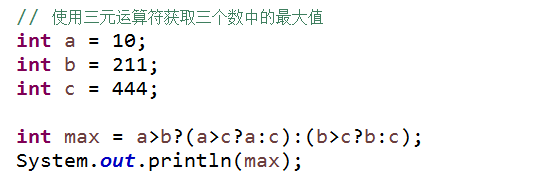
### 比较两个数据是否相同



### 获取两个数中的最大值

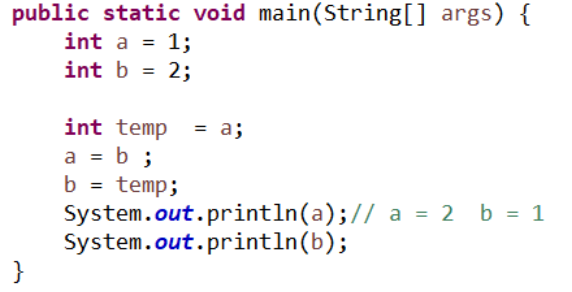


### 获取三个数中的最大值

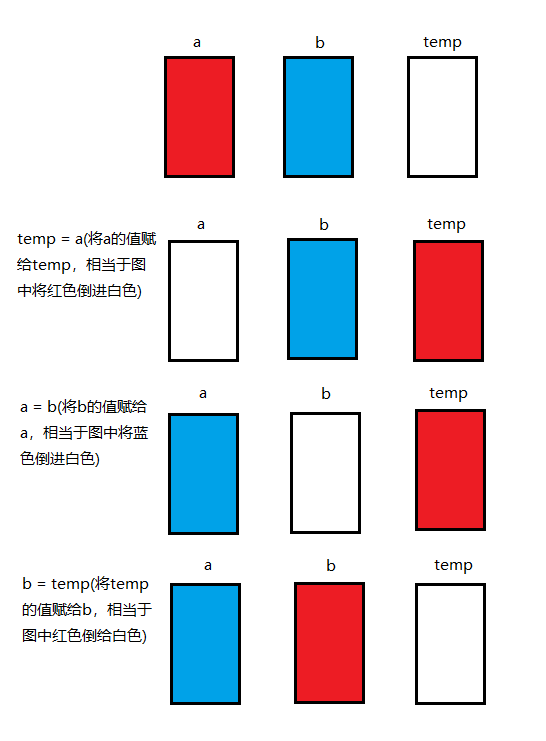


### 面试题-交换两个变量的值

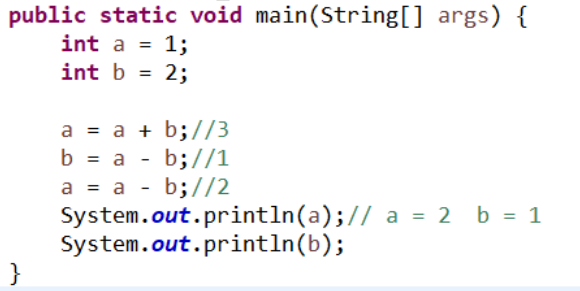
* 方式一:使用temp临时变量

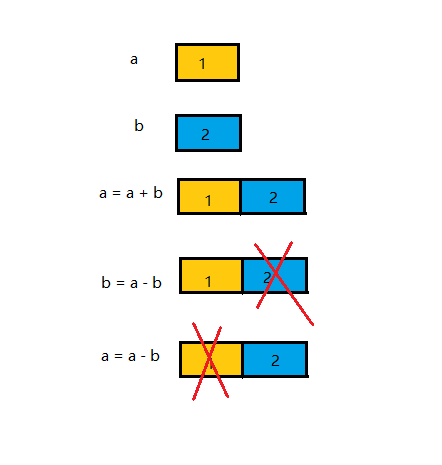


图示：

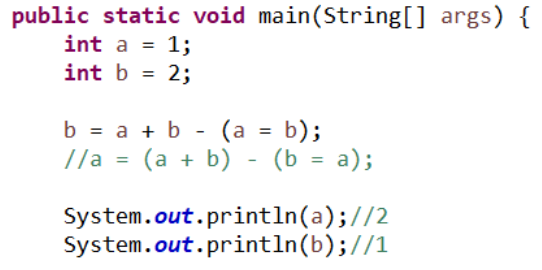


* 方式二:不使用临时变量





* 方式三:了解即可



# 键盘录入Scanner类

## 概述

我们目前在写程序的时候，数据都是代码中定义好的，我们想让数据由键盘中输入，所以可以考虑把数据改进为键盘录入，提高程序的灵活性。

那么该如何实现呢?这就需要使用到引用数据类型Scanner类,它是JDK给我们提供的,使用方式也得按照一定的步骤来

## 使用步骤

1.定义变量，并相应类型的对象赋值给变量：

数据类型 变量名 = new 数据类型();

如:Scanner scanner = new Scanner(System.in);

2.导包：

使用import导包,告诉JVM从哪里去找Scanner类

3.调用Scanner对象方法：

变量名.方法名();

## 代码演示

综合运用之前的知识,完成控制台版本的求体质指数BIM并给出相应友好提示

BIM = 体重(kg) / 身高(m)的平方



## 注意

