**标识符**

**标识符是为了标识程序中诸如类、方法和变量的元素而采用的命名。**

**标识符是由字母、数字、下划线和美元符号($)构成的字符序列。**

**标识符必须以字母、下划线或美元符号开头，不能以数字开头。**

**标识符不能是保留字。**

**标识符不能是true、false或null。**

**标识符可以为任意长度。**

**由于Java是区分大小写的，所以area、Area和AREA都是不同的标识符。**

**不要使用字符$命名标识符。习惯上，字符$只用在机器自动产生的源代码中。**

**变量**

**变量用于表示在程序中可能被该边的值。**

**例如**

**int count;**

**double radius;**

**double interestRate;**

**如果几个变量为同一个类型，允许一起声明它们：**

**int i,j,k;**

**int i=2,j=2; //一步完成变量的声明和初始化**

**在赋值给变量之前，必须声明变量。方法中声明的变量在使用之前必须被赋值。**

**任何时候，都要尽可能一步完成变量的声明和初赋值。这会使得程序易读，同事避免程序设计错误。**

**在赋值语句中，左边变量的数据类型必须与右边值的数据类型兼容。例如，int x=1.0是非法的，因为x的数据类型是整型int。在不使用类型转换的情况下，是不能把double值（1.0）赋给int变量的。**

**命名常量（named constant）是一个代表不变值的标识符。**

**例如**

**final datatype CONSTANTNAME=value;**

**使用常量有三个好处：1.不必重复输入同一个值；2.如果必须修改常量值，只需在源代码的一个地方做改动；3.给常量赋值可以提供代码的可读性。**

**命名习惯**

**1.使用小写字母命名变量和方法。如果一个名字包含多个单词，就将它们连在一起，第一个单词的字母小写，而后面的每个单词的首字母大写。**

**2.类名中的每个单词的首字母大写，例如，类名ComputeArea和System。**

**3.大写常量中的所有字母，两个单词间用下划线连接，例如，MAX\_VALUE。**

**数据类型**

**每个数据类型都有它的取值范围。编译器会根据每个变量或常量的数据类型为其分配内存空间。**

**java为数值、字符值和布尔值数据提供了八种基本数据类型。**

**byte ：（-2^7）-128~（2^7-1）127 8位带符号数**

**short ：（-2^15）-32768~（2^15-1）32767 16位带符号数**

**int ：（-2^31）-2147483648~（2^31-1）2147483647 32位带符号数**

**long ： -2^63~2^63-1 64位带符号数**

**float ： 负数范围：-3.4028235E+38~~1.4E-45**

**正数范围：1.4E-45~~3.4028235E+38**

**32位，标准IEE754**

**double ： 负数范围：-1.7976931348623157E+308~~4.9E-324**

**正数范围：4.9E-324~~1.7976931348623157E+308**

**64位，标准IEE7554**

**通常情况下，应该使用double型，因为它比float型更精确。**

**默认情况下，整型直接量是一个十进制正整数。要表示一个二进制整数直接量，使用0b或者0B开头；表示一个八进制整数直接量，就用0开头，而要表示一个十六进制整数直接量，就用0x或0X开头。**

**例如**

**System.out.println(0b1111); //Displays 15**

**System.out.println(07777); //Displays 4095**

**System.out.println(0xFFFF); //Displays65535**

**浮点型直接量带小数点，默认情况下是double型的。可以通过在数字后面加字母f或F表示该数为float型直接量，也可以在数字后面加d或D表示该数为double型直接量。**

**float型和double型都是用来表示带有小数点的数。把它们称为浮点数，是因为这些数都是以科学记数法的形式进行内部存储的。当一个像50.534的数被转换成科学记数法的形式时，它就是5.0534E+1，它的小数点就移到一个新的位置。**

**为了提高可读性，java允许在数值直接量的两个数字间使用下划线。例如**

**long ssn=232\_45\_4519;**

**long creditCardNumber=2324\_4545\_4519\_3415L;**

**显示当前时间**

**可以通过调用System.currentTimeMillis()返回当前时间。**

**System类中的方法currentTimeMillis返回从GMT 1979年1月1日00:00:00开始到当前时刻的毫秒数。**

**时间戳是时间开始计时的点，因为1970年是UNIX操作系统正式发布的时间，所以这一个时间也称为UNIX时间戳（UNIXepoch）。**

**1）调用System.currentTimeMillis()方法获取1970年1月1日午夜到现在的总毫秒数totalMilliseconds。**

**2）通过将总毫秒数totalMilliseconds除以1000得到总秒数totalSeconds。**

**3）通过将totalSeconds%60得到当前的秒数。**

**4）通过将totalSeconds除以60得到总的分钟数totalMinutes。**

**5）通过totalMinutes%60得到当前分钟数**

**6）通过将总分钟数totalMinutes除以60获得总的小时数totalHours。**

**7）通过totalHours%24得到当前的小时数。**

**用long类型**

**数值类型转换**

**总是可以讲一个数值赋给支持更大数值范围类型的变量，例如，可以将long型的值赋给float型变量。但是，如果不进行类型转换，就不能将一个值赋给范围较小类型的变量。**

**将一个小范围类型的变量转换为大范围类型的变量称为拓宽类型（widening a type），把大范围类型的变量转换为小范围类型的变量称为缩窄类型（narrowing a type）。**

**java将自动拓宽类型，但是缩窄类型必须显式完成。**

**java中，x1 op=x2形式的增强赋值表达式，执行为x1=(T)(x1 op x2)，这里T是x1的类型。因此，如下代码正确。**

**int sum=0;**

**sum+=4.5; //sum becomes 4**

**sum+=4.5等价于sum=(int)(sum+4.5);**

**有两种类型的import语句：明确导入和通配符导入。明确导入是在imiport语句中指定导入单个类；通配符导入将包中所有的类导入。**