**常用数学函数**

Math类中的方法分为三类：三角函数方法（trigonometric method）、指数函数方法（exponent method）和服务方法（service method）。服务方法包括取整、求最小值、求最大值、求绝对值和随机方法。Math类还提供了两个double型常量，PI（π）和E（自然对数的底）。可以在任意程序中用Math.PI和Math.E的形式来使用。

指数函数方法

exp(x)返回e的x次方

log(x)返回x的自然底数

log10(x)返回x的以10为底的对数

pow(a,b)返回a的b次方

sqrt(x)对于x>=0的数字，返回x的平方根

取整方法

ceil(x) x向上取整为它最接近的整数。该整数作为一个双精度值返回

floor(x) x向下取整为它最为接近的整数。该整数作为一个双精度值返回

rint(x) x取整为它最接近的整数。如果x与两个整数的距离相等，偶数的整数作为一个双精度值返回

roud(x) 如果x是单精度数，返回(int)Math.floor(x+0.5);如果x是双精度数，返回(long)Math.floor(x+0.5)，即四舍五入

min和max方法用于返回两个数（int、long、float或double型）的最小值和最大值。例如，max(4.4,5.0)返回5.0，而min(3,2)返回2。

abs方法以返回一个数（int、long、float或double型）的绝对值。

(int)(Math.random()\*10)=======返回0-9之间的一个随机整数

50+(int)(Math.random()\*50)======返回50-99之间的一个随机整数

通常a+Math.random()\*b=====返回a-a+b之间的一个随机数，不包括a+b

**字符数据类型和操作**

字符串直接量必须括在双引号中。而字符直接量是括在单引号中的单个字符。

一个Unicode码占两个字节，用以\u开头的4位十六进制数表示，范围从'\u0000'到'\uFFFF'。

大多数计算机采用ASCII码（美国标准信息交换码），它是表示所有大小写字母、数字、标点符号和控制字符的8位编码表。Unicode码包括ASCIII，从'\u0000'到'\uoo7F' 对应128个ASCII字符。

特殊字符的转义序列

\b 退格键 \u0008

\t Tab键 \u0009

\n 换行符 \u000A

\f 换页符 \u000C

\r 回车符 \u000D

\\ 反斜杠 \u005C

\" 双引号 \u0022

字符型数据与数值型数据之间的转换

char型数据可以转换成任意一种数据类型，反之亦然。将整数转换成char型数据时，只用到该数据的低十六位，其余部分都被忽略。例如

char ch=(char)0xAB0041 //变为0x0041

要将一个浮点值转换成char型时，首先将浮点值转换成int型，然后将这个整型值转换成char型。

//65赋给ch

char ch=(char)65.25

当一个char型数据转换成数值型时，这个字符的Unicode码就被转换成某个特定的数值类型。

//i是A的Unicode码65

int i=(int)'A';

如果转换结果适用于目标变量，就可以使用隐式转换方式；否则，必须使用显式转换方式。例如，因为'a'的Unicode码是97，它在一个字节的范围内，所以就可以使用隐式转换方式：

byte b='a';

int i='a';

但是因为Unicode码\uFFF4超过了一个字节的范围，所以下面的转换就是不正确的；

byte b='\uFFF4';

为了强制赋值，就必须使用显式转换方式，如

byte b=(byte)'\uFFF4';

0-FFFF的任何一个十六进制正整数都可以隐式地转换成字符型数据。而不再此范围内的任何其他数值都必须显式地转换成char型。

所有数值操作符都可以用在char型操作数上。如果另一个操作数是一个数字或字符，那么char型操作数就会自动转换成一个数字。如果另一个操作数是一个字符串，字符就会与该字符串相连。例如

//(int)'2'是50 (int)'3'是51 结果i是101

int i='2'+'3';

//(int)'a'是97 结果j是99

int j=2+'a';

**字符的比较和测试**

两个字符可以使用关系操作符进行比较，如同比较两个数字一样。这是通过比较两个字符的Unicode值实现的。

Java的Character类提供的方法进行字符测试

isDigit(ch) 如果指定的字符是一个数字，返回true

isLetter(ch) 如果指定的字符是一个字母，返回true

isLetterOrDigit(ch) 如果指定的字符是一个字母或者数字，返回true

isLowerCase(ch) 如果指定的字符是一个小写字母，返回true

isUpperCase(ch) 如果指定的字符是是一个大写字母，返回true

toLowerCase(ch) 返回指定的字符的小写形式

toUpperCase(ch) 返回指定的字符的大写形式

**String类型**

String类型不是基本类型，而是引用类型（reference type）。

任何java类都可以将变量表示为引用类型。使用引用类型声明的变量称为引用变量，它引用一个对象。

length() 返回字符串中的字符数

charAt(index) 返回字符串s中指定位置的字符

concat(s1) 将本字符串s1连接，返回一个新字符串

toUpperCase() 返回一个新字符串，其中所有的字母大写

toLowerCase() 返回一个新字符串，其中所有的字母小写

trim() 返回一个新字符串，去掉了两个边的空白字符

String是java中的对象。以上方法只能从一个特定的字符串实例来调用。由于这个原因，这些方法称为实例方法。

非实例方法称为静态方法。静态方法可以不使用对象来调用。定义在Math类中的所有方法都是静态方法。它们没有绑定到一个特定的对象实例上。

java允许在不创建新变量的情况下，使用字符串直接量直接引用字符串。例如

"Welcome to java".length()是正确的

"".length()为0

方法s.charAt(index)可用于提取字符串s中的某个特定字符，其中下标index的取值范围在0-s.length()-1之间。

在字符串s中越界访问字符是一种常见的程序设计错误。为了避免此类错误，要确保使用的下标不会超过s.length()-1。例如

s.charAt(s.length())会造成一个StringIndexOutOfBoundsException异常。

可以使用concat方法连接两个字符串。例如将s1和s2连接构成s3

Stirng s3=s1.concat(s2);

如果i=1并且j=2，下面语句的输出是

System.out.println("i+j is"+i+j);

输出是"i+j is 12"，因为"i+j is"首先和i的值连接。要强制i+j先执行，将i+j放在括号里。

从控制台读取字符串，调用Scanner对象上的next()方法。next()方法读取以空白字符结束的字符串（即''、'\t' '\f' '\r' '\n'）

可以使用nextLIne()方法读取一整行文本。nextLine()方法读取以按下回车键为结束标志的字符串。

为了避免输入错误，不要在nextByte()、nextShort()、nextInt()、NextLong()、和next()之后使用nextLine()。

**字符串比较**

equals(s1) 如果该字符串等于字符串s1，返回true

equalsIgnoreCase(s1) 如果该字符串等于字符串s1，返回true，不区分大小写

compareTo(s1) 返回一个大于0、等于0、小于0的整数，表明一个字符串是否大于、等于、小于s1

compareToIgnoreCase(s1) 和compareTo一样，除了是比较是不区分大小写之外

startsWith(prefix) 如果字符串以特定的前缀开始，返回true

endsWith(suffix) 如果字符串以特定的后缀结束，返回true

contains(s1) 如果s1是该字符串的子字符串，返回true

操作符==只能检测string1和string2是否指向同一个对象，但它不会告诉你它们的内容是否相同。

获得子字符串

substring(beginIndex) 返回该字符串的子串，从特定位置beginIndex的字符开始到字符串的结尾

substring(beginIndex,endIndex)返回该字符串的子串，从特定位置beginIndex的字符开始到下标为endIndex-1的字符。

indexOf(ch) 返回字符串中出现的第一个ch的下标。如果没有匹配的，返回-1

indexOf(ch,fromIndex) 类似上面

indexOf(s)

indexOf(s,formIndex)

lastIndexOf(ch)

lastIndexOf(ch,fromIndex)

lastIndexOf(ch)

字符串和数字间的转换

可以将数值型字符串转换为数值。要将字符串转换为int值，使用Integer.parseInt方法，例如

//intString是一个数值型字符串'123'

int intValue=Integer.parseInt(intString);

可以将数值转换为字符串，只需简单使用字符串的连接操作符，例如

String s=number+"" ;

格式化控制台输出

可以使用System.out.printf方法在控制台上显示格式化输出

%b 布尔值

%c 字符

%d 十进制整数

%f 浮点数

%e 标准科学记数法形式的数

%s 字符串

默认情况下， 输出是右对齐的。可以在格式标识符中放一个负号（-），表明该条目在特定区域中的输出是左对齐的。