# Java基础

## 一、Java基础知识

**java一次编译，到处运行，Java语言有跨平台性**，但JVM不具有跨平台性，不同的操作系统具有不同的JVM。

ASCII码表：American Standard Code Information Interchange(美国信息交换标准代码)

Unicode：万国表。比ASCII码表有更多的字符。

关键字：全部小写

注意在TXT中写完java代码然后在命令提示框中运行时，若提示非法字符则可能是输入了中文字符：如中文下的括号，分号等。

html中不区分大小写，javascript中区分大小写，java中区分大小写。

Java中变量名、方法名第一个单词的首字母小写，后面每个单词的首字母大写。类名以大写字母开头，常量全部大写。

main为函数名，主函数，在javase中不能修改。args可以修改，是参数，其它不能动。main为系统自动调用的方法

java中函数相当于方法，即函数就是方法，方法就是函数。

**按住shift键，再按鼠标右键，则可以打开快速命令窗口。Java文件在哪里，就在哪里打开命令窗口**。

## 二、Java的基本数据类型

**浮点型默认是double类型**

字符型char默认值为’**\u0000**’,即‘空格’，字符型的空格。

布尔类型默认为false，1位。

字符“0”-“9”之间的ASCII码值为48-57。

字符“A”的ASCII码值为：65；“Z”为90.

字符“a”的ASCII码值为：97，“z”为122.



局部变量没有默认值。写在方法中的变量都是局部变量，局部变量在使用之前必须赋值，如写在public static void main(String[] args){}中的变量都是局部变量，使用时必须赋值，否则报错。

byte的取值范围为-128～127之间。即如果：byte s1=(byte)58598，则打印的时候是一个问号。Long数据的范围-263～263-1；int类型的取值范围为21个亿多，即-231～231-1。char类型的为0-6555；。

**运行java程序**：**1.javac 文件全名(包括后缀名) 2.java 文件名(不包含后缀)。javac命令是用来将xxx.java编译成xxx.class文件，java命令是用来运行xxx.class。**

长整型：long，定义长整型时要在数字后面加L。

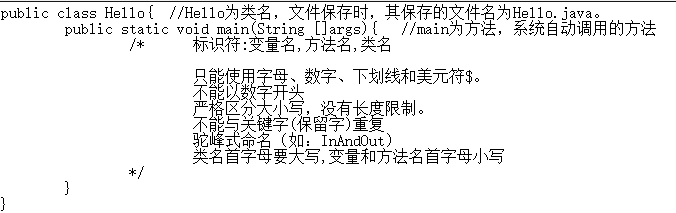
单精度浮点型：float，要在后面加f，如0.12f。

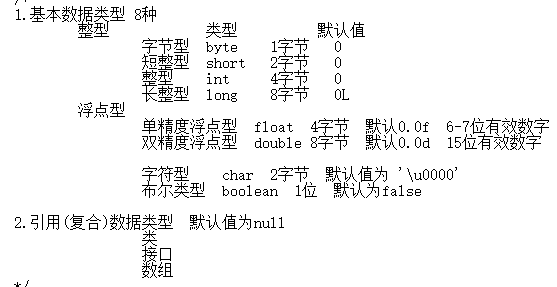
注意Double.NaN!=Double.NaN的值是true，因为NaN是not a number，不能进行比较，所以Double.NaN==Double.NaN的值为false，故上面的值为true。

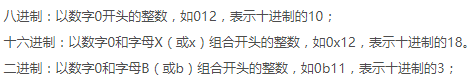
注意char类型的字符，在两个单引号之间有且仅有一个字符，**不能没有**。而字符串中，两个双引号之间可以没有字符。

空常量null不能打印输出，否则报错。

**标识符**：







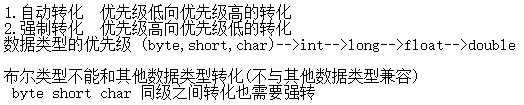
decimal：小数的、十进制的 octonary：八进制的

hexadecimal：十六进制的

字符串也属于引用数据类型.

数据范围与字节数不一定相关，如float的数据范围比long的数据范围大。浮点数有可能是一个近视值。

## 二、变量的类型转化



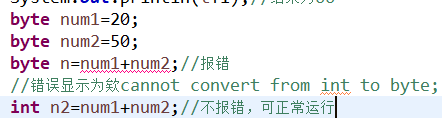
注意：强制类型转化时可能出现数据丢失。

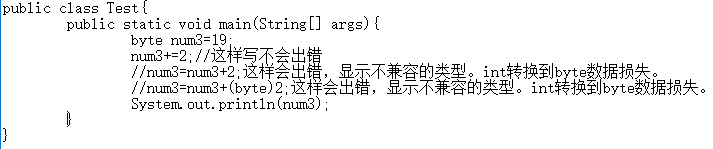
java中整型的字面量是int类型，浮点型字面量是double类型。变量参与运算时，变量类型的优先级如果小于int，则会自动提升为int类型。

下面这是因为在进行加减乘除数学运算时，byte、short、char类型的数据会自动转换成int类型的数据，然后再进行加减乘除，最后得到的结果也是int类型的。











常量有字面量和符号常量两种。字面量就是其数值意义如同字面所表示的那样，如整数18就表示18。

**将字符“a”～“z”、“A”～”Z”、“0”～“9”转换成数字，只要：**。

符号常量的定义：final 数据类型 符号常量标识符=常量值，如final double PI=3.1415.

## 三、运算符

1.算数运算符：+ - \* / % ++ --

2. 比较运算符：< > <= >= == != 结果为布尔类型。==是判断相等,不能用来判断引用数据类型，如String；只能判断基本数据类型8种。

3.逻辑运算符：& && **|** **|| !** 其中**&、|为逻辑与和逻辑或，没有短路功能**，而**&&、||则有短路功能**。&、**|**也可用在位运算符中，称为按位与、按位或，即**当&、|两边都是数字时，它两则是位运算符，将数字转换成二进制然后进行与、或运算**。**当&、|两边都是表达式时，则是逻辑运算符，在逻辑运算符中&和&&、|和||没有什么本质区别**。逻辑运算时直接用&&、**||**。

4.赋值运算符 = += -= \*= /= %=

5.**三目运算符(条件)**：当三目运算符的结果直接输出时,两个结果的类型可以不同，当三目运算符的结果用变量接受时,两个结果的类型必须相同

6.位运算符 & **| ^ << >>** **~**按位与 按位或 异或 左移 右移 按位取反（将0变为1,1变为0）。左移 n位 相当于乘以2的n次方。

## 四、输入、输出

1.输出：System.out.println()没参数时直接换行；System.out.print()不换行,**必须有参数，没参数会报错**；

2.输入：首先在顶部写入“import java.util.\*;”导入包。然后在main()方法中定义一个扫描器:**Scanner sc=new Scanner(System.in)**;

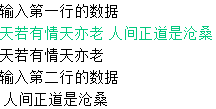
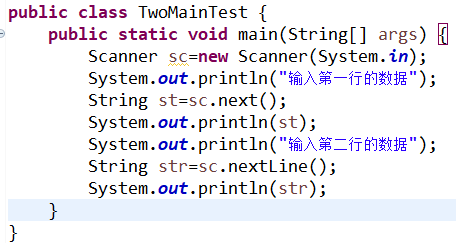
输出的形式有：String name=sc.next(); int num=sc.nextInt(); nextDouble(); nextBoolean(); nextByte();

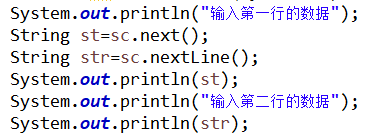
nextShort();nextLong();nextFloat();

String str=sc.nextLine();//直接回车不会继续等待,而是返回空串 "",并且不会将空格当作结束符号。注意String类型的输入是sc.nextLine()或者sc.next()，而不是sc.nextString()。

**sc.nextLine()和sc.next()的区别**：

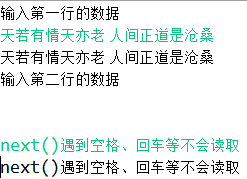
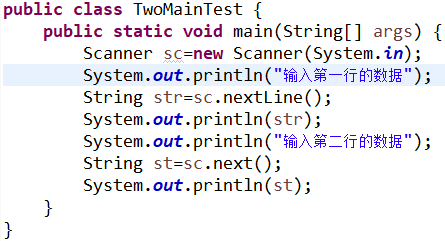
**next()方法读取到空白符(空格、回车、tab键)就结束**(注意不是结束输入，结束输入都是按Enter键)，**nextLine()方法是读取到换行符(即回车)就结束**。**next()不会读取空格、tab键的空内容，而nextLine()会读取空格、tab键的空内容。**



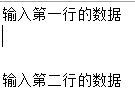
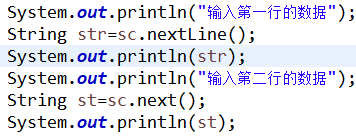


按照上面的顺序，在键盘中输入“天若有.....”，输入结束时直接按回车，则黑色部分的内容都是打印，没有第二次输入。我们在键盘中输入的内容先存到缓冲区中等待读取，边输入边读取，判断读取结束的标志是空白符(如：空格、回车、tab键)，读取结束并不是说输入结束，只有按了回车键，输入才结束。而next()读取时，遇到空格或者回车则会结束读取(这不会影响输入)，当next()下面还有nextLine()时，nextLine()就会从上次结束读取的地方开始读取，直到遇到回车就结束读取。所以上面的输入的“天若有情天亦老(空格)人间正道是沧桑(回车)”，next()读取到空格前面的内容，遇到空格就结束读取了，而后面的nextLine()接着空格处开始读取(空格也被读取到了)，一直到回车，则结束。

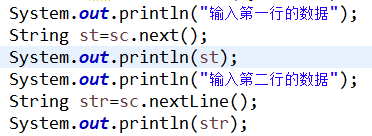
注意上面的代码中，将sc.next ()和sc.nextLine()放在一起的结果也是一样的，如上：也就是说用next()、nextLine()等读取键盘的内容时，只要保证它们前后的顺序是一致的就不会影响输出的结果，可以将它们直接上下连着写，也可以中间隔这许多代码。



同时要注意：nextLine()遇到空格是会读取的，而next()遇到空格是不会读取的。下面这个，nextLine()遇到空格读取了或遇到回车则读取空内容，从而往下走，而next()则没有读取。



下面这种，next()下，按空格、回车则一直光标则一直往下走，没输入也没输出：

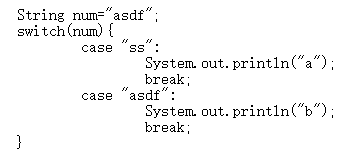
 

**注意没有nextChar()!!**!

Java中，注意布尔类型与任何数据都不能转换，if语句括号内只能放布尔类型，因此在if()语句中，括号内的内容不能直接写数字或其他字符，若if(1){}，则会出错。在其他语言中if(1){}可能不会报错，因为其他语言中if括号内只要不是0，其他数字或字符都默认为boolean的true，即1。

**判断奇偶法二：int num=11;if((num&1)==1)则为奇数，注意要将num和1括起来，不然会报错**。

switch语句中,switch括号内可以是int类型以及自动转换成int类型的数据类型如byte、short、char，也可以是String类型，因为String类型系统会自动转换成一个int值，如果要填入如double类型时，要进行强制转换成int类型，如(int)num。case 标签值不能重复，同时类型也不能大于int，如case 1.0则会报错。

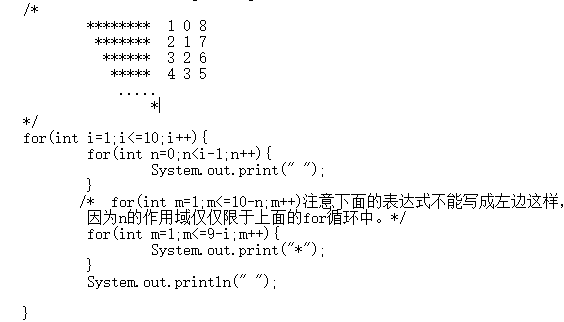


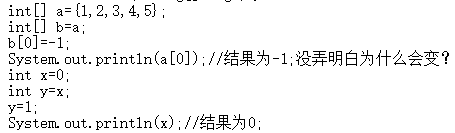
**基本数据类型的值存放在栈里面，引用数据类型存放在堆里面**。

数组引用的值等于首元素的地址，数组中的元素在内存中的地址是连续的。

break：跳出该循环

continue：结束本次循环，继续下一次循环





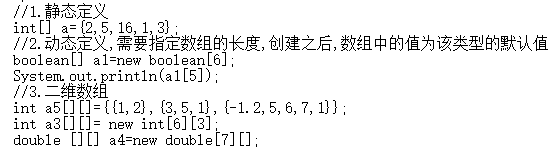
## 五、定义数组

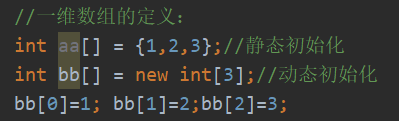
**数组只能存储同一种数据类型的数据**，指向数组的变量的值为数组首元素的地址。数组类型可以说是java中任何数据类型。

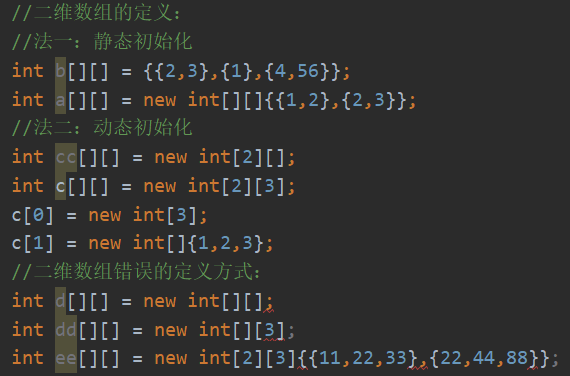
若方法接收一个数组，则此方法对数组的修改将会全部保留下来。方法可接收一个数组，也可返回一个数组（在方法的声明中明确声明返回值为数组即可）。将数组传递到方法中，传递时不需要写“[]”,只需写数组名即可。定义初始化数组时必须指定数组的长度。

数组的排序：Arrays.sort();

下面的数组静态定义也称为静态初始化，动态定义也称为动态初始化，数组静态初始化采用{}，动态初始化采用new。







注意在java的数组中不能用负号的索引来遍历数组，如a[-1]会出错。在其他编程语言中可以用负号的索引来遍历数组。

void表示方法没有返回值，当方法有返回值时，需要在方法声明中将void改成指定返回值的类型。

遍历数组可用for循环，也可用**for each**语句，详见自己的笔记。

**Arrays.sort(Object[] a)**：数组的升序排序，没有返回值，括号内参数为要排序的数组名。**该方法会改变数组的内容**。

**Arrays.toString(Object [] a)**：将数组转换成字符串，**返回值为一个字符串**，而不是一个字符串数组，参数为数组名，对原来的数组并不影响。

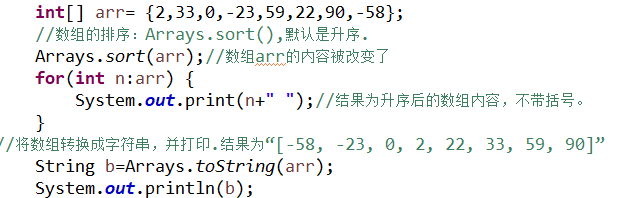
**Arrays.fill(Object[] a,Object value)**：没有返回值，给数组填充内容。Object value为要给数组中填充的元素。

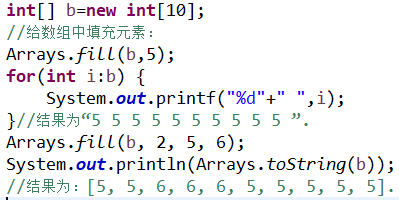
**Arrays.fill(Object[] a,int fromIndex,int toIndex,Object value)**：从索引**fromIndex(包含)**开始，一直到**toIndex(不包括)**给数组填充元素，无返回值。超过数组的长度会抛出异常。

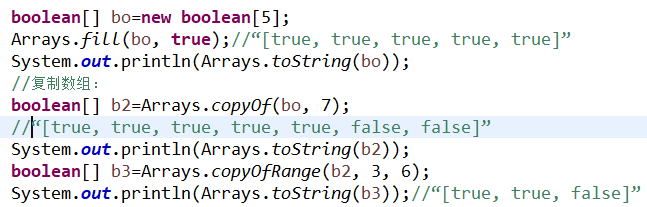
**Arrays.copyOf(Object[] original,int newLength)**：返回值为一个数组，将一个数组中的内容以某个长度复制到一个新数组中，原来的数组不会发生改变。若该长度超过原来数组的长度，则超出的部分是该数组类型的默认值。

**Arrays.copyOfRange(Object[] original,int from,int to)**：从索引fromIndex(包含)开始一直到toIndex(不包含)，将原来数组中的内容复制到新数组中，原来的数组不会改变。若toIndex超出了原来数组的范围，则新数组中超出的部分为默认值。

**Arrays.boinarySearch(Object[] a,object key)**：返回值为int类型的索引。采用二分搜索法搜索数组a中是否包含某个元素，若不包含这个元素，则返回一个带负号的数，若有这个元素则返回这个元素的索引。注意在使用这个方法之前必须先使用Arrays.sort()先将数组进行排序，再搜索，否则得到的结果将会是错误的。同时使用这个方法时，数组中的数不能有重复的，若有重复的，则查找这个重复的内容时返回的结果不一定是正确的。

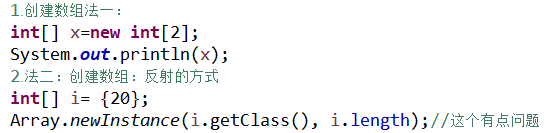






**注意Arrays是在java.util包中的，而Array是在java.lang.reflect包中的。Array类是采用反射的方法对数组进行操作。**

**Array.newInstance(Class<?> component，int length)**：采用反射的方式创建数组。(有空深入学习，下面反射方式创建数组有点问题)



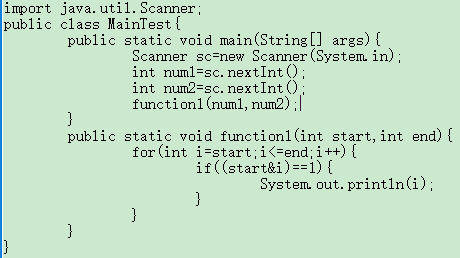
## 六、方法：即函数

方法：1.要有方法的声明 2.方法的调用。在方法调用中一定要保证实际参数和形式参数的一一对应关系。

自己定义一个方法要放在main方法之外。

1.无返回值、无参数的方法。

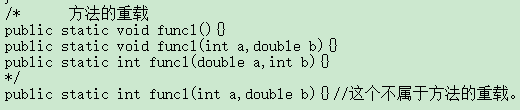
2.有参数、无返回值的方法。Java中方法声明中的形参也要具体规定属于什么数据类型。



3．有参数、有返回值的方法。当方法有返回值时，必须在方法声明中声明返回值的类型。如public static int function2(int num)，表示返回值的类型为int类型，用return语句返回，为void的时，表示没有返回值。

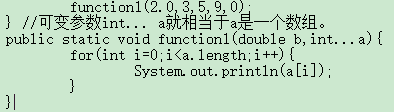
## 七、方法重载

**方法的重载：在同一个类中，方法名相同，参数列表不同(参数的个数、顺序、类型)，与返回值和权限无关。**

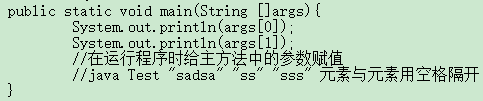


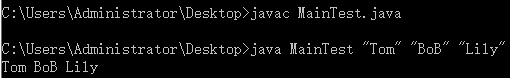
## 八、可变参数

可变参数：参数的个数可以修改。实际上就是一个数组。**注意可变长参数只能当最后一个参数**。表示方法：**数据类型...变量名**



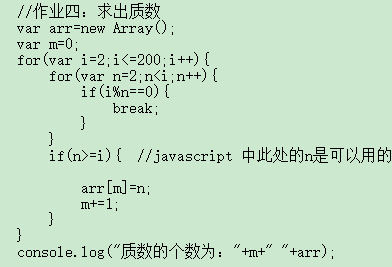
**main()方法的参数是在程序运行时通过命令行输入。元素与元素之间不能用逗号，应该用空格**。







## 八、作用域

在javascript中，变量的作用域是以函数为作用域的，若变量在某个函数中定义，则在函数之外是不可用的。但是，若变量是定义在if或者for这样的代码块中，它在代码块中是可以用的。如下：

注意在java中上面的n就不能用，否则会报错。即在java中for循环、while循环里声明的变量，只要跳出循环该变量就不能用了。

文件.class中的内容是对文件.java编译后的信息。

## 九、长度

**数组的长度用length**

**xml中结点集合的长度用getLength()**

**文件字节的大小：File.length()**

**列表List、集合set、Map的大小：size()**

**String的长度：length()**

**(在js中字符串的定义var s=’’sfafsf’’，字符串的长度用length，没有括号。)**

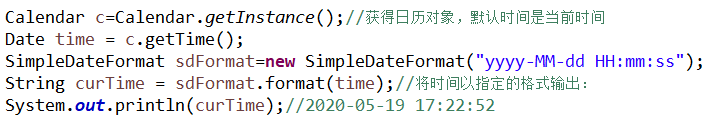
## 十、日历Calendar

在java中获得当前的年月日可用Calendar的方法，也可用new Date()的方法。**注意获得一个Calendar对象采用的是getInstance()方法，而不是new 出来的**。



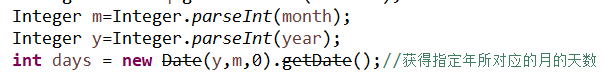
***Calendar.HOUR\_OF\_DAY、Calendar.MINUTE、Calendar.SECOND***分别获取当前时间的时分秒。

将时间以指定的格式输出：采用SimpleDateFormat类。



有空学习一下SimpleDateFormat类、Date类。

根据指定的年、月获得该月的总天数：



## 十一、Math.random

注意java中Math.random得到的是0～1之间的小数(包含0，不包含1)。若要强转成整数应该将整个Math.random部分用大括号括起来，然后在强转：这样写是正确的。

不能写成下面这样，否则得不到效果：



生成min~max之间的随机整数(包含min但不包含max，parseInt(Math.random()\*(max-min))+min。

生成min~max之间的随机整数（同时包含min和max）：parseInt(Math.random()\*(max-min+1))+min。

## 十二、反编译java的class文件

找到存放java代码的class文件的目录，一般Eclipse中存放java项目的目录下，src存放的是.java类型的文件，而bin目录下存放的是对应的.class文件。进入.class文件的目录，按住shift键，然后右键选择快速打开命令提示窗口，在命令模式下，输入**javap –c 类名.class**即可反编译对应的类的class文件。

## 十三、return、break、continue的区别

