**一、变量与标识符**

**数学名词**

变数或变量，是指没有固定的值，可以改变的数。变量以非数字的符号来表示，一般用拉丁字母。变量是常数的相反。变量的用处在于能一般化描述指令的方式。

**计算机解释**

变量就是程序分配的一块内存单元，用来存储各种类型的数据。根据所存储的数据类型的不同，有各种不同类型的变量。变量名代表这块内存中的数据。

**变量分类**

按所属的数据类型划分：

基本数据类型 引用数据类型变量

按被声明的位置划分：

局部变量：方法或语句块内部定义的变量

成员变量：方法外部、类的内部定义的变量

**标识符**

Java对包、类、方法、参数和变量等要素命名时使用的字符序列称为标识符

Java标识符命名规则：

1.由字母、数字、下划线（\_）和美元符号（$）组成。

2.不能以数字开头

3.区分大小写

4.长度没有限制

5.不能是Java中的关键字和保留关键字

标识符命名习惯：驼峰命名法、见名知意、以单词或单词组合来命名

**二、八种基本数据类型**

字节（Byte）是：计算机文件大小的基本计算单位

1个字节（Byte） = 8个位（Bit）

Bit意为“位”或“比特”，是计算机运算的基础；二进制中的一位，是二进制最小信息单位，二进位可以用来表示一个简单的正/负的判断，有两种状态的开关（如电灯开关）

**单位换算：**

1byte = 8 bit

1KB = 1024byte

1MB = 1024KB

1GB = 1024MB

1TB = 1024G

1PB = 1024TB

说明：

在内存计算过程中，字节少的数据运算速度更快；

在硬盘存储中，字节少的数据类型也充分存入更多的数据

数据类型 关键字 占用字节 取值范围

布尔型 boolean 不定值

字节型 byte 1 -128~127

短整型 short 2 -32768~32767

整型 int 4

长整型 long 8

字符型 char 2 0~65535

单精度浮点型 float 4

双精度浮点型 fouble 8

**三、基本数据类型的声明**

**声明变量的格式：**

1.声明变量

类型 变量名；

2.声明的同时给变量赋值

类型 变量名 = 值；

3.给已声明过的变量赋值

变量名 = 值；

**四、进制与转换**

**二进制**

计算机中的数据都以二进制数字保存。二进制：逢二进一。即只有0、1两个值

10----1010

**八进制**

基数为八

Java中八进制数据要以0开头。如：0123

八进制转换成二进制：只需要将每个八进制数字替换为相对应的三个二进制位即可

**十六进制**

二进制表示法太冗长，所以在程序中一般喜欢用十六进制

十六进制：基数为十六，逢十六进一。它用abcdef表示从0-9之上的值

Java中十六进制数据要以0x或0X开头。如 0x23D

十六进制转换成二进制只需将每个十六进制数字替换为相对应的四个二进制位即可

**补码：**

事实上，计算机内的二进制数值是以补码形式表示的

补码：一个正数的补码和其原码的形式是相同的

负数的补码是：将该数的绝对值的二进制形式，按位取反再加1

由此可知，二进制补码数值的最高位（最左位）是符号位：该位为0，表示数值为正数；该数为1，表示数值为负数

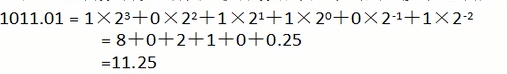
主要原因：使用补码，可以将符号位和其它位统一处理；

同时，减法可可按加法来处理

另外，两个用补码表示的数相加时，如果最高位（符号位）有进位，则进位被舍弃

**二进制数转换成十进制数：**

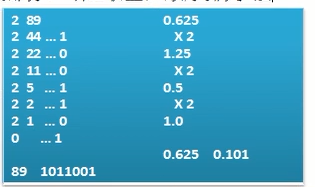
按权相加：把二进制数首先写成加权系数展开式，然后按十进制加法规则求和。



十进制转换成二进制

整数部分：除2取余，逆序排列法

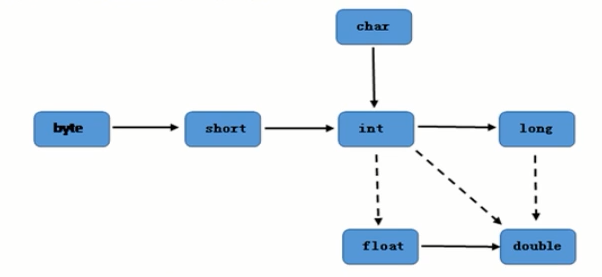
小数部分：乘2取整，顺序排列法



**五、基本数据类型转换**

**自动类型转换：**

容量小的类型自动转换成容量大的数据类型



byte，short，int ，它们三者在计算时会转换成int类型

如果把int值转换为float值，或者long转换成double值，不需要强制转换，但可能丢失精度

强制类型转换：

容量大的类型转换成容量小的数据类型时，要加上强制转换符。

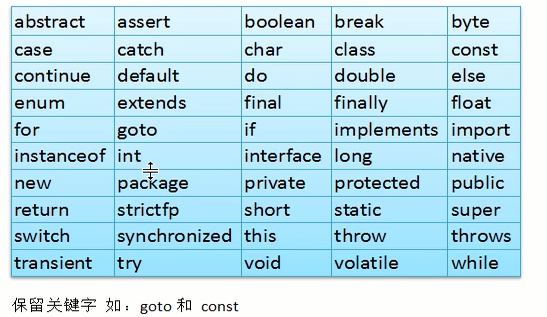
long n = 100L;

int i =(int)n;

有可能造成精度降低或数据溢出，使用时要小心。

boolean类型不能转换成任何其他数据类型

**六、关键字**



**七、转义字符**



**八、运算符与优先级**

**算数运算符：**

表倒是：由变量、常量运算符组成的式子

a = b++; 先赋值后自增

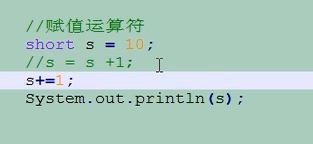
a = ++b; 先自增后赋值



**赋值运算符：**

作用是将一个值赋给一个变量，运算顺序从右到左





+= 不会有数据转换的问题

**关系运算符**

作用是比较两边的操作数，结果总是boolean型的



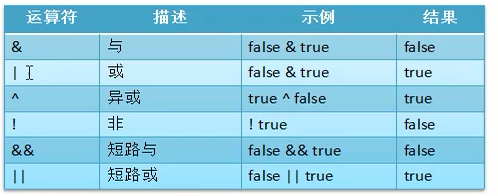
//创建一个键盘输入工具

Scanner input = new Scanner(System.in);
int x = input.nextInt();//从键盘接收一个整数
System.out.println(x)

**九、运算符与优先级**

**逻辑运算符：**

用于对boolean型结果的表达式进行运算，运算结果总是boolean型。



在计算机中，非0即真，0为假

逻辑或：或1则1

逻辑与：全1才1

与运算，分别计算表达式两边的结果再做&运算，只有两个都为true，结果才为true，否则为false

或运算，分别计算表达式两边的结果再做|运算，只要有一个为true，结果就为true

异或运算，两边相同为false，不同为true

非运算，两边相同为true,不同为false

短路与，&&

短路或，||

&和&&的区别？

& ： 在逻辑运算式分别计算表达式两边的结果再做&运算，只有两个都为true，结果才为true，否则为false，在做位运算时&表示按位与。

&& ：短路与运算，先计算左边的表达式，如果结果是false，那么不用计算右边表达式，直接返回false，如果左边的表达式结果是true，再计算右边的表达式，如果右边的表达式为true，结果为true，否则为false

|和||原理同上

所以短路与 或 短路或的计算效率更高，建议使用

**位运算符**

对两个操作数中的每一个二进制位都进行运算

位运算符功能：

1.按位取反~

2.按位取与& 有0则0

3.按位取或| 有1则1

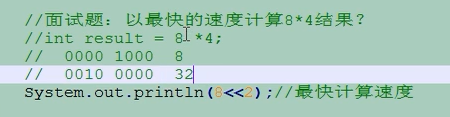
4.按位异或^ 相同为0 不同为1

**位移运算符**

左移：“a<<b” 将二进制形式的a逐位左移b位，最低位空出的b位补0

带符号右移：“a>>b”将二进制形式的a逐位右移b位，最高位空出的b位补原来的符号位

无符号右移：“a>>>b”将二进制形式的a逐位右移b位，最高位空出的b位补0



左移算法：位移数作为2的次幂与操作数相乘

右移算法：操作数除以位移数的2次幂

初始化两个数将这两个数据交换位置后，输出显示交换后的数据

1.通过第三个变量来交换

int a = 1;

int b = 2;

int c = a;

b = c;

2.不通过第三个变量交换两个数

(1)通过表达式

a = a+b-(b=a)

(2)通过+-计算

a = a + b;

b = a - b;

a = a - b;

(3)两个数交换位置最快的方法 两个数异或三次（不能用两个相同的值）

a = a ^ b;

b = a ^ b;

a = a ^ b;

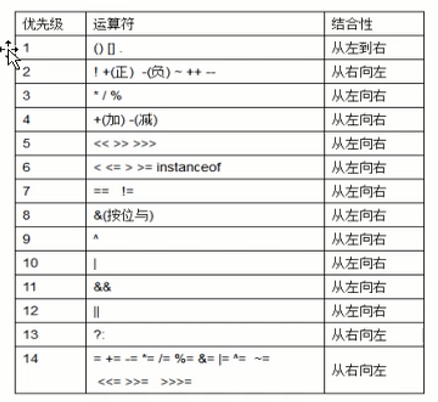
(4)乘除

**三目运算符**

X ? Y : Z

先计算x的值，若为true，返回Y，若为false,返回Z

**运算符优先级表**



**十、分支语句**