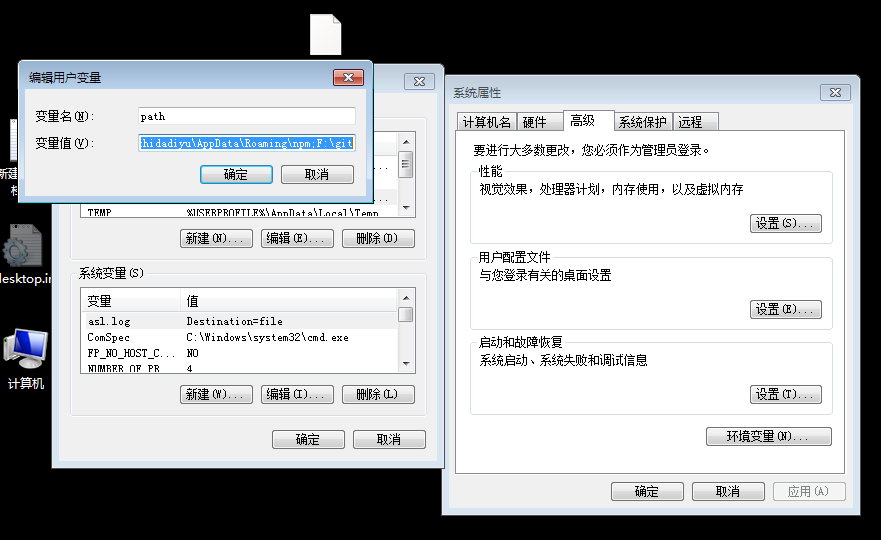
1. 环境变量



2、第一个程序；

case 1:

public class test1{

public static void main(

system.out.println(“Holle .word”);

)  
}

1. 系统输出语句；

case 1:

system.out.println(“ ”)

1. 基本数据类型：

分为8种：

整数类型

|  |  |
| --- | --- |
| byte | 8 位，用于表示最小数据单位，如文件中数据，-128~127 |
| short | 16 位，很少用，-32768 ~ 32767 |
| int | 32 位、最常用，-2^31-1~2^31 （21 亿） |
| long | 64 位、次常用 |

注意事项：

int i=5; // 5 叫直接量（或字面量），即 直接写出的常数。

整数字面量默认都为 int 类型，所以在定义的 long 型数据后面加 L或 l。

小于 32 位数的变量，都按 int 结果计算。

强转符比数学运算符优先级高。见常量与变量中的例子。

浮点数类型：

|  |  |
| --- | --- |
| double | 32 位，后缀 F 或 f，1 位符号位，8 位指数，23 位有效尾数。 |
| float | 64 位，最常用，后缀 D 或 d，1 位符号位，11 位指数，52 位有效尾 |

注意事项：

二 进 制 浮 点 数 ： 1010100010=101010001.0\*2=10101000.10\*2^10（2次方)=1010100.010\*2^11(3次方)= . 1010100010\*2^1010(10次方)

尾数： . 1010100010 指数：1010 基数：2

浮点数字面量默认都为 double 类型，所以在定义的 float 型数据后面加F 或 f；double 类型可不写后缀，但在小数计算中一定要写 D 或 X.X

float 的精度没有 long 高，有效位数（尾数）短。

float 的范围大于 long 指数可以很大。

浮点数是不精确的，不能对浮点数进行精确比较。

3）一种字符类型(char)：

字符类型：

|  |  |
| --- | --- |
| char | 16 位，是整数类型，用单引号括起来的 1 个字符（可以是一个中文字符），使用 Unicode 码代表字符，0~2^16-1（65535） 。 |

注意事项：

不能为 0个字符。

转义字符：\n 换行 \r 回车 \t Tab 字符 \" 双引号 \\ 表示一个\

两字符 char 中间用“+”连接，内部先把字符转成 int 类型，再进行加法运算，char 本质就是个数！二进制的，显示的时候，经过“处理”显示为字符。

布尔类型：

|  |  |
| --- | --- |
| boolean | true 真 和 false 假。 |

记忆：8位：Byte（字节型）

16位：short（短整型）、char（字符型）

32位：int（整型）、float（单精度型/浮点型）

64位：long（长整型）、double（双精度型）

最后一个：boolean(布尔类型)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

5.0类型转换：

自动转换：byte-->short-->int-->long-->float-->double

强制转换：①会损失精度，产生误差，小数点以后的数字全部舍弃。②容易超过取值范围。

6、**变量**

为什么要有变量；-- 计算 数据（0~9）、字符|字符串、布尔值、

数据类型：《重点》

人脑：计算区域、存储区域

电脑：CPU、内存

存储计算后的结果数据（中间数据）

有效范围、有效期：作用域；

输入的数据，也是映射到大脑皮层；

变量：在内存中开僻出的一个空间（区域），用来 存储、暂时的；该区域可以重复存放不同数据；

数据类型 变量名 ；//申明变量；

变量名 = 数据 ；//赋值语句；赋值符号；

**变量命名规范**

1. 运算符

算术运算符： + - \* / %（取余|取模） ；

比较运算符|关系运算符： > 、<、 ==、>=、 <=、 !=；

逻辑运算符： &&(and) 、||(or)、 !(not)；

**优先级：**

1. 条件控制语句

如果 （条件满足） {

做什么事件//执行语句；代码块

}

否则{

做其它事件

}

Int i =scanner.nextInt();

If(i < 5){

}else{

// i>=5

}

* 多重条件控制语句：

If (条件表达式1){

}else if(条件表达式2){

}else{

}

~~If (条件表达式1){~~

~~}~~

~~if(！条件表达式1 && 条件表达式2){~~

~~}~~

这种方式：代码复杂；计算运行效率低；

* 嵌套语句

条件表达式的代码块中（大括号中），再放一个条件表达式；

If(条件表达式1){

If(条件表达式2){

}

}

* Switch语句

Switch(待比较的表达式)

只能做等值比较；

Case 关键字；

Case 比较值 **：{**

**}**

Break 关键字；

中断、跳出当前switch语句；

Default 关键字；

1. 循环语句；

* While循环语句
* 初始化语句、比较语句、变量叠代；

While(比较语句){

//循环体；需要重复执行的代码；

//添加退出机制（步长）

//或者用break退出;

}

* Do While

//先执行一次，再判断是否继续循环

// while: 先判断是否继续循环，满足条件才执行

Dowhile最后的分号“；”不要漏掉；

**循环语句，也可以嵌套；**

* For循环

把数据初始化、比较语句、叠代语句，放在一起；

For(数据初始化 **;** 比较语句 **;** 叠代语句){

//循环的内容

}

Break:中断或者退出循环体;

Continue ： 本次循环中，关键字后面的不执行，直接转到下次循环;

**Break 和continue，只在当前循环中有效；**

* **数组**

方便管理大量数据（变量）

数据类型 [] 数组名；//申明；

两种初始化

**基本初始化**

* New 数据类型 [数组长度]

数组中，没有任何数据值，只默认值 ；

带数据的初始化

* New 数据类型[]{数据1，数据2}；

此种情况，不用指定长度，使用后面数据个数作为长度；

给数组赋值

**数组名[索引|下标] = 数据**

动车001 ，第8号车厢；

使用数组：

**数组名[索引|下标]**

***二维数组***

条件

人与人交流：自然的语言

人与机交流：程序语言；

计算机内部：计算机语言；01