# 环境变量

右键 计算机属性 》高级系统设置》高级》环境变量 》

系统变量》

创建一个 变量名JAVA\_HOME 变量值java安装路径（一直到根目录）

再在path 中添加 %JAVA\_HOME%\bin

Eclipse 中 警告和错误的区别

警告的问题 不影响代码执行， 但是影响效率，浪费资源；

错误 ：编译就出错；

* 1. 第一个程序；

public class Hello{

public static void main(String[] args){

System.out.println(“Hello World!”);

}

}

# 语法基础：

* 1. 关键字： 其实就是某种语言赋予了特殊的单词。
  2. 标示符：其实就是在程序中自定义的名词。比如类名，变量名，函数名。包含0-9，a-z、$、\_;

注意：

数字不可以开头；

不可以使用关键字。

* 1. 常量： 程序中不会变化的数据。
  2. 变量：就是内存中的一个储存空间，用于存储常量数据。

作用： 方便于运算。能在存储空间找到该变量所蕴含的数据

特点： 变量空间可以重复使用。

变量的作用域：定义该变量使用的范围 花括号{开始到花括号}结束 要有开有闭；

# 数据类型

* 1. **数据类型：byte、short、int（整形）、long、float、**

**double（双精度浮点型）、char（字符型）、boolean、String(字符串型)**

**String 变量 =”字符串”；当给字符串赋值给变量时 string 首字母要大些**

**char 变量=’一个字符型’**

* + 1. **自动类型转换：从低级到高级别，系统自动转换的。**
    2. **强制转换类型： 高的优先级到低的优先级就需要强制转换；例如：int i= 3；**

**Double j=4.50；**

**i = (int) j;**

1. 运算符号:

同种类型参与运算(可能需要自动类型转换)

整数的除法是整除；

当运算类型不同时 需要类型转换

* 1. 算术运算符

**+ - \* / %**

**++与--的操作是**

**b=a++;//表示的是b=a,而后a++**

**b=++a;//表示的是b等于a+1后的值**

**% ：表是取余运算**

**0对其他数的余数是0**

例如 int i=0;

i++ 就是 i=i+1

余数的应用场景

Int a=10；

Int b=3；

Int c=a/b；//商；--两整数相除，结果 也是整数；

Int d=b\*c；//可以除净的数值；

Int yushu（余数）=a - d；

Yush= a%d；

**余数**的特点： 整数；小于除数，并且大于等于0；

Random ran=new Random();

Ran.nextlnt();//-xxx~~~xxx-1;

ASCII 的取值范围：0~255；

**取余就是 把比较大的范围，按照指定的方式，缩小到一定范围内容**

* 1. **逻辑运算符。**

**&&（且） ||（或） ！非**

True&&true= true 、true&&false=false 、 false&&false=false

True||true=true、true||false=true 、false||true=true、false||false=false

# If语句

条件控制语句

* 1. **If语句**

当判断数据范围，获取判断运算结果boolean类型时，需要使用if。

如果 （条件满足） {

**if (Boolean类型的值（）条件)**

做什么事件//执行语句；代码块

System.out.println();

}

否则{

**else**

做其它事件

System.out.println();

}

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){

}

}

* 1. **switch语句**

工作原理：用小括号中的变量的值依次和case后面的值进行对比，和那个case后面的值相同了 就执行那个case后面的语句，如果没有相同的则执行default后面的语句

**Break（中断、跳出）** 是可以省略的 如果省略了就一直执行到遇到break为止；

Switch后面的括号中的变量应该是byte、char、short、int四种类型中的

Switch(变量){

case 值： 要执行的语句；

break；（中断跳出）

default（默认）：要执行的语句；

# 循环

* 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

用于存储同一类型的一个容器。

好处可以对该容器中的数据进行编号，数组的下标从0开始。

* 1. **元素类型[]变量名=new 元素类型[元素的个数]；**
  2. **元素类型[]变量名={元素1，元素...};**

**元素类型[]变量名=new 元素类型[]{元素1，元素2，...};**

**二维数组**

**元素类型[][]变量名=new 元素类型[长度][长度]；**

**元素类型[][]变量名={元素1，元素2，。。。}**

**元素类型[][]变量名=new 元素类型[]{元素1，元素2.。。。}，**

# 其他

**Random r= new Random();//产生随机数**

**Int can=r.nextInt();//产生随机整数**

**Int rab=Math.abs(can%100);给产生的随机数限定范围**

**///r.nextInt（10）；//0到9之间的随机数**

**Scanner sc=new Scanner（System.in）;//键盘输入**

**Int a=sc.nextInt（）；**