# Part 1

**比特（bit）**:一个数字0或者一个数字1，代表一位。

**字节（byte）**:每逢8位是一个字节，这是数据存储的最小单元。

命令提示符**(cmd)**:

启动： **Win+R**

切换盘符： 盘符名称：

进入文件夹： **cd** 文件夹名称

进入多级文件夹： **cd** 文件夹**\**文件夹**2\**文件夹**3**

返回上一级： **cd ..**

返回根目录： **cd \**

查看当前内容**： dir**

清屏： **cls**

退出： **exit**

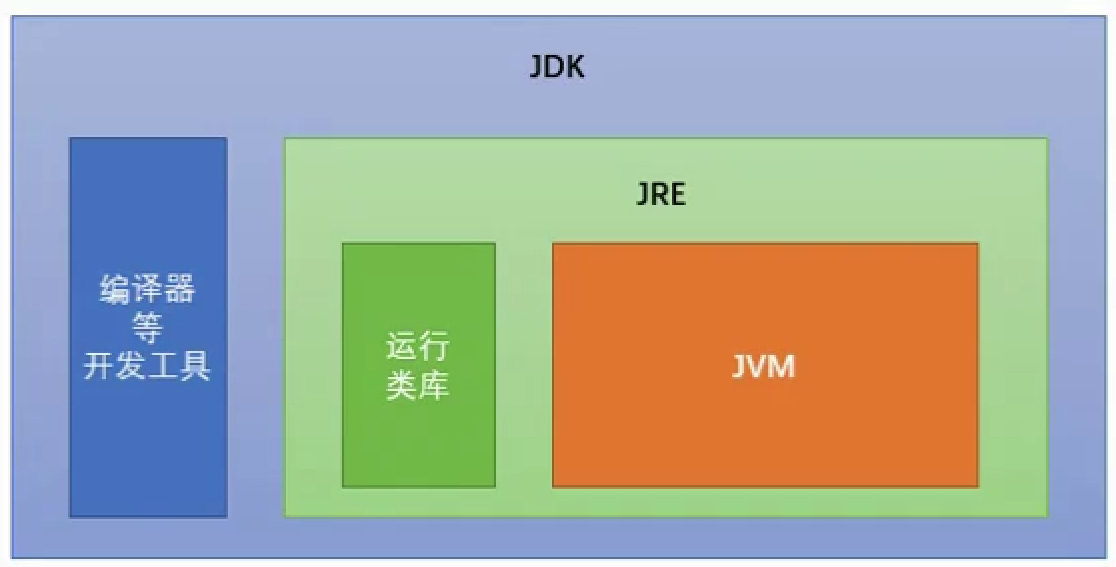
# Part 2

**Java**虚拟机**—JVM(Java Virtual Machine)**

Java语言的**跨平台特性**：Java编写的软件可以运行在任何操作系统上，该特性是由JVM实现的，我们编写的程序运行在JVM上，而JVM运行在操作系统上。

**JRE**（Java Runtime Environment）:Java程序的运行时环境，包含JVM和运行时所需要的的核心类库。

**JDK**（Java Development Kit）：Java程序开发工具包，包含JRE和开发人员使用的工具。



**ERROR**：在CMD中安装程序报C:\Program' 不是内部或外部命令,也不是可运行的程序或批处理文件。

**Answer：**文件安装在C:\Program Files\Java\jdk1.7.0\_25\bin\Javac中，但报上面错误；解决方法为C:\Progra~1\Java\jdk1.7.0\_25\bin\Javac；可以正常运行。

Java程序开大三步骤：编写、编译、运行。

# Part 3关键字&标识符

**关键字的特点：**

1. 完全小写的字母
2. 在编辑器中有特殊颜色

**标识符：**

1. 标识符是指在程序中，自己定义的内容。比如：类的名字、方法的名字和变量的名字。
2. 命名规则：

可以包含英文字母（区分大小写）、0-9数字、$、和\_

不能以数字开头

不能是关键字

1. 命名规范：

大驼峰式（类名规范，如：HelloWorld）、小驼峰式（变量名规范，如：helloWorld）

**常量：**

1. 字符串常量。例如：”abc”、“123”、“hello”
2. 整数常量。例如：100/0、-250
3. 浮点数常量。例如：2.5、0.0
4. 字符常量。例如：’A’、’b’
5. 布尔常量。例如：true、false
6. 空常量。null

**数据类型：（基本数据类型、引用数据类型）**

1. 基本数据类型：

整数型 byte、short、int、long

浮点型 float、double

字符型 char

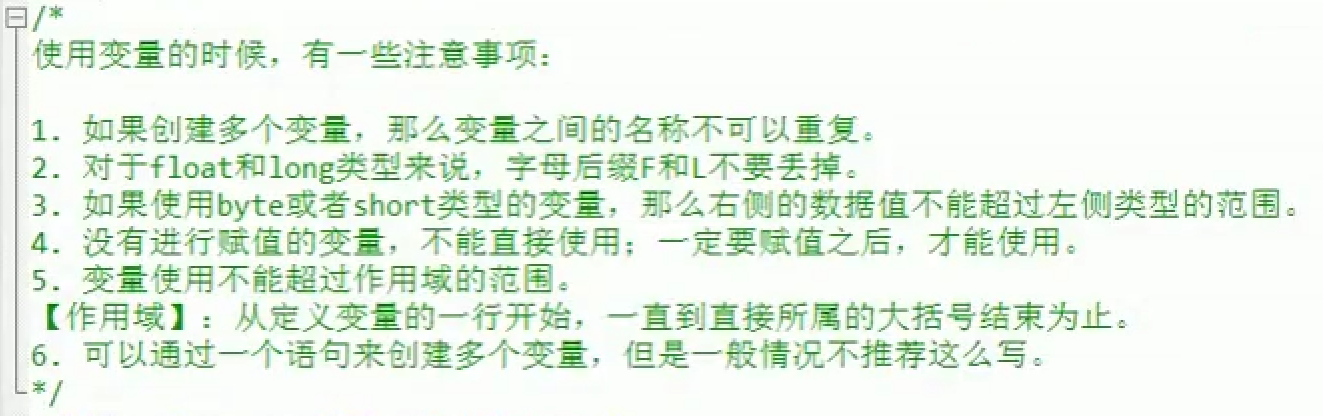
布尔型 Boolean



1. 引用数据类型（字符串、数字、类、接口、Lambda）

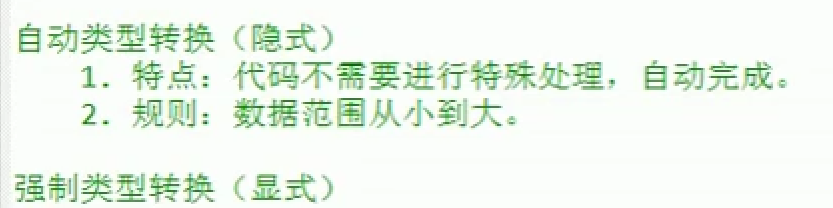
# Part 4常量变量

常量、变量

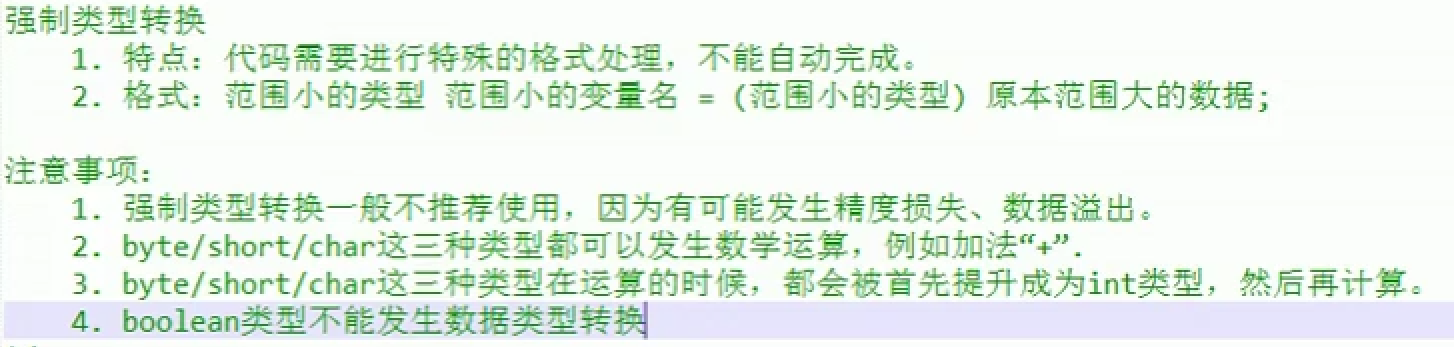


# Part 5数据类型转换

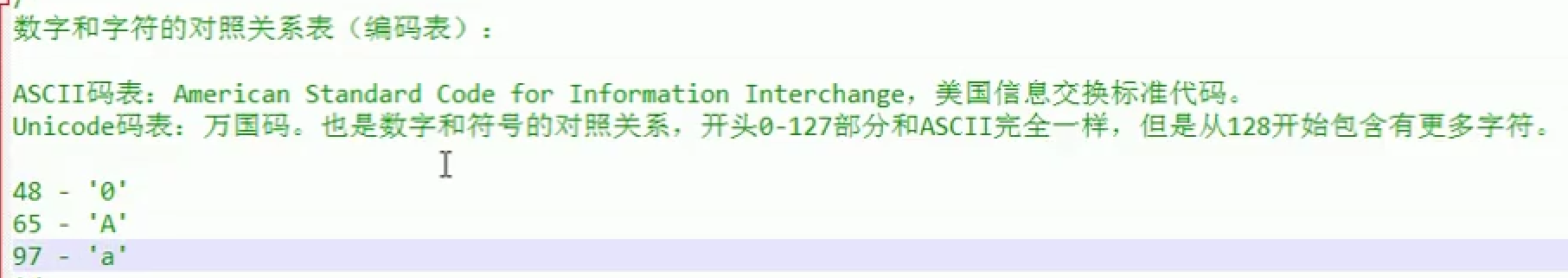
1. 自动类型转换



1. 强制类型转换

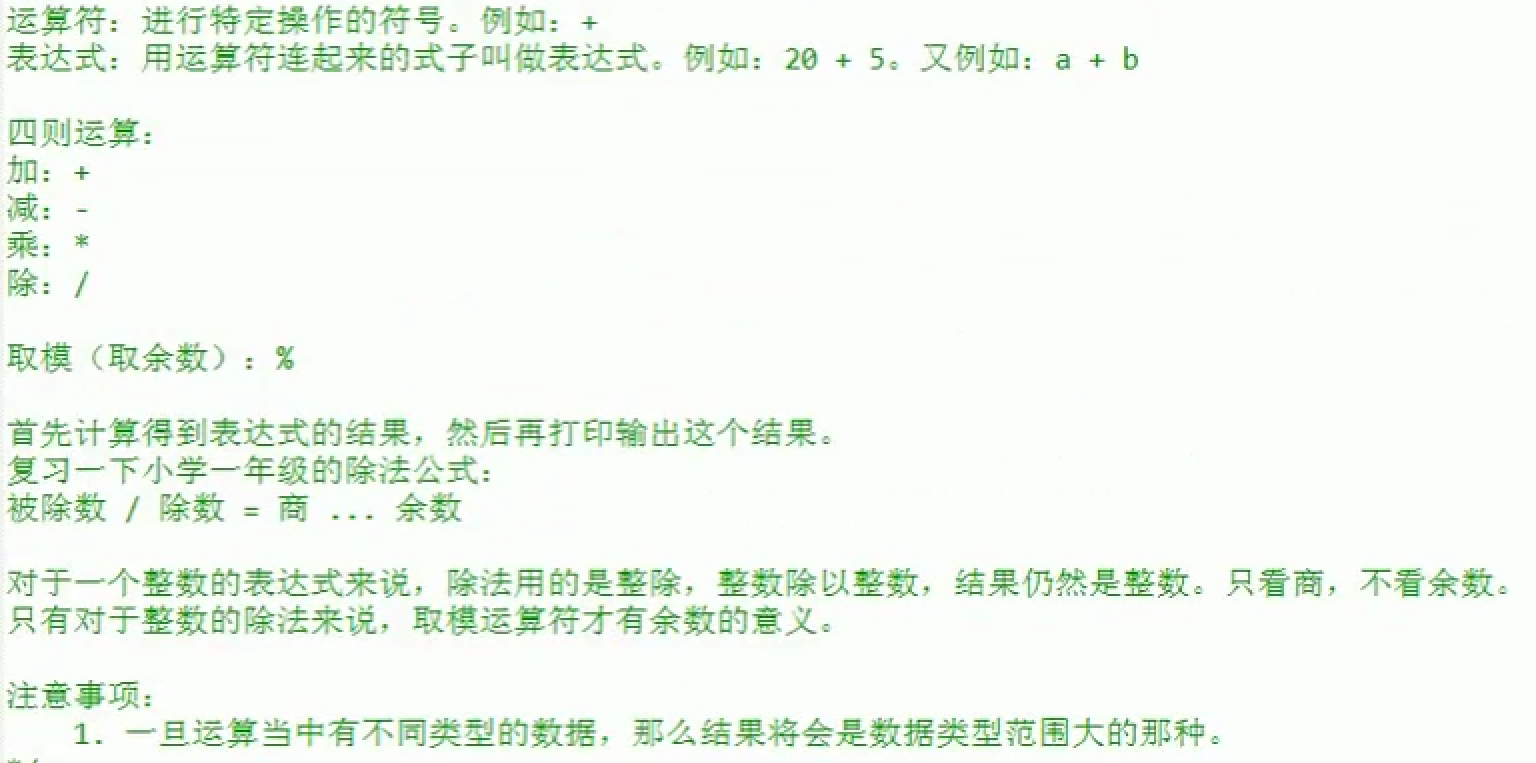


1. 编码表

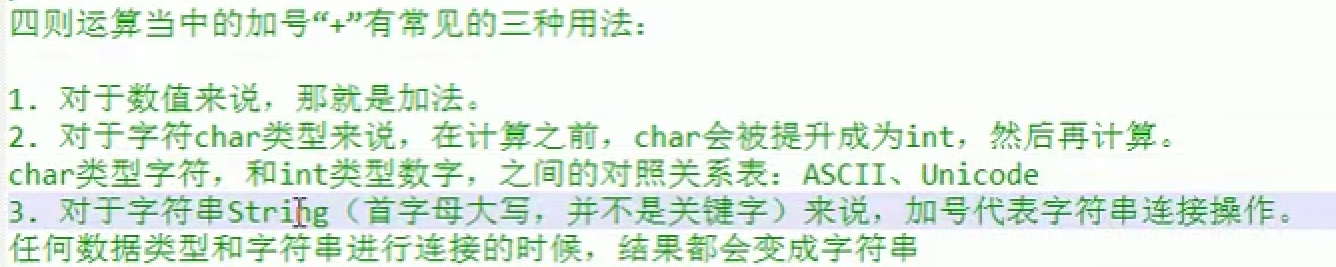


# Part 6运算符

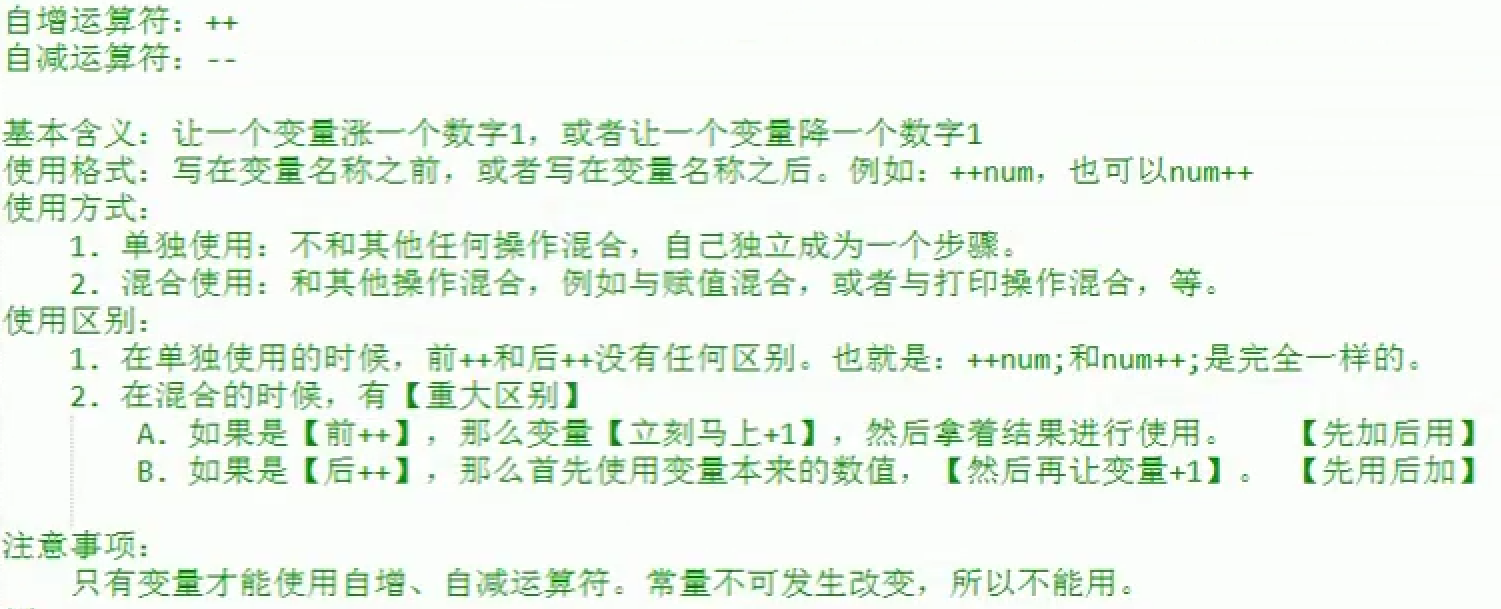
1. 算数运算符



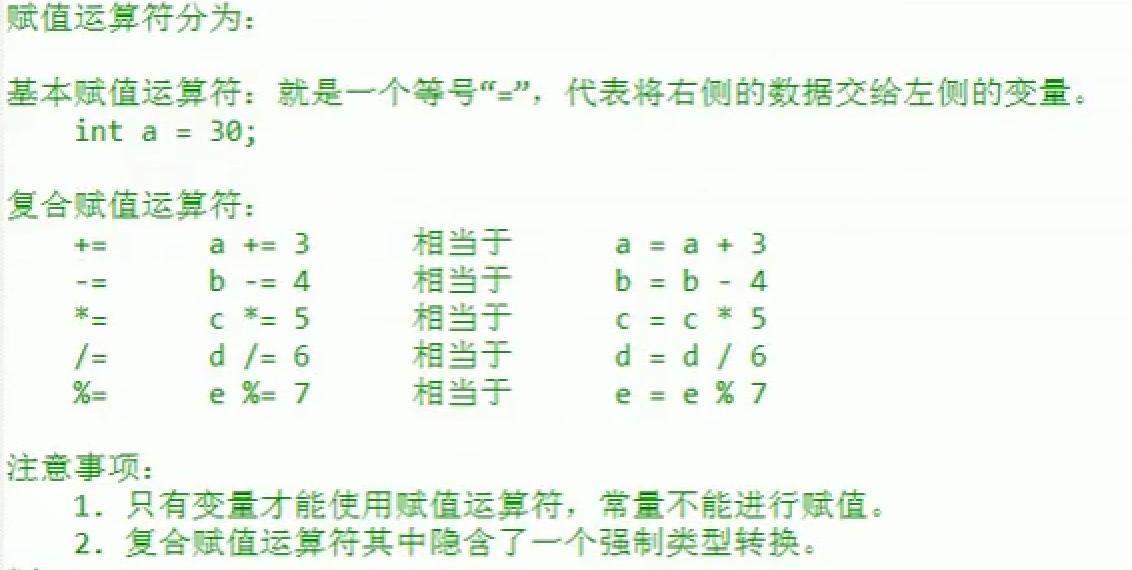
1. 加号的多种用法



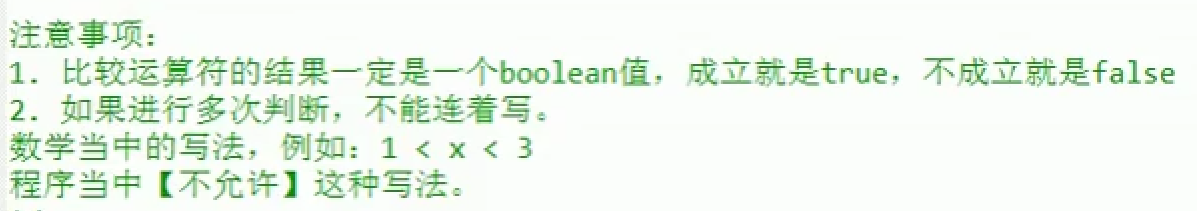
1. 自增自减运算



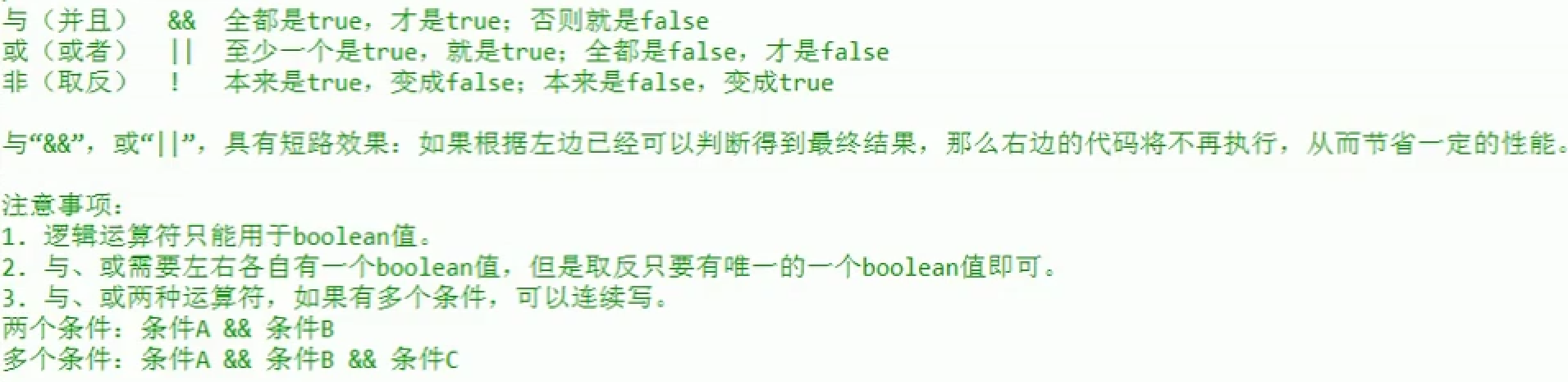
1. 赋值运算符



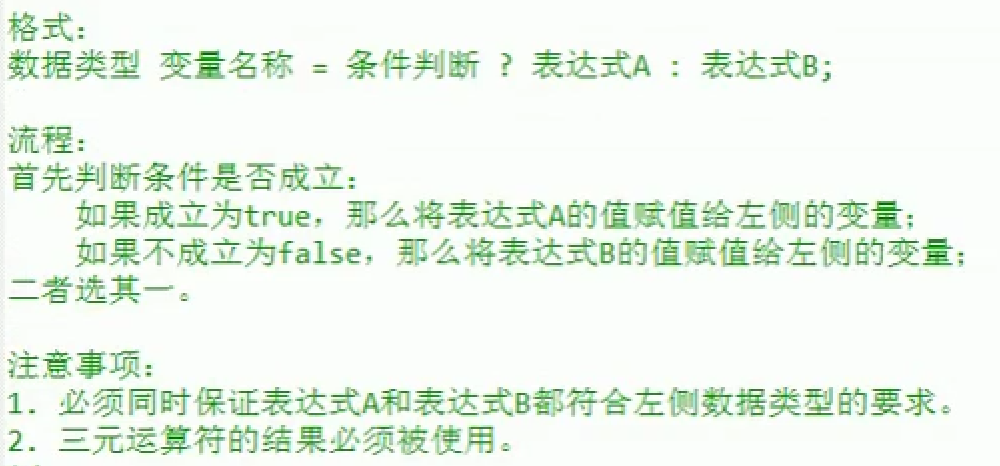
1. 比较运算符



1. 逻辑运算符

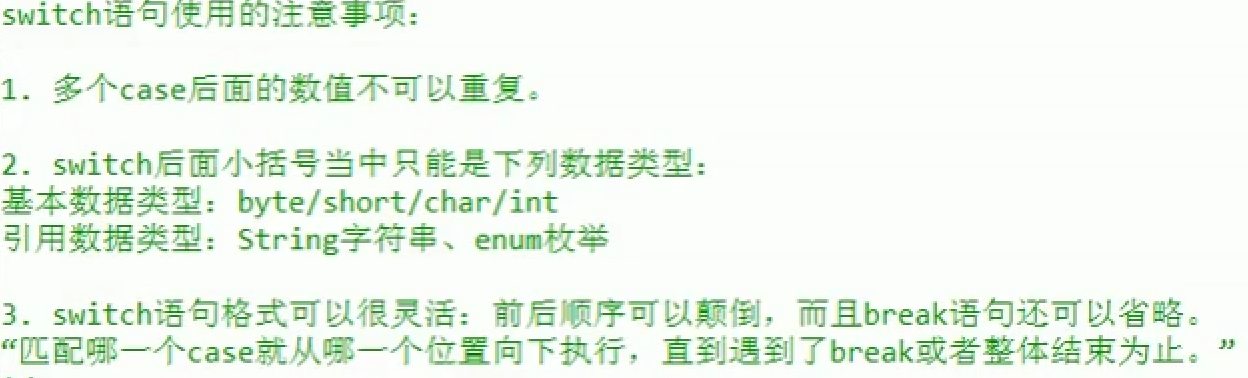
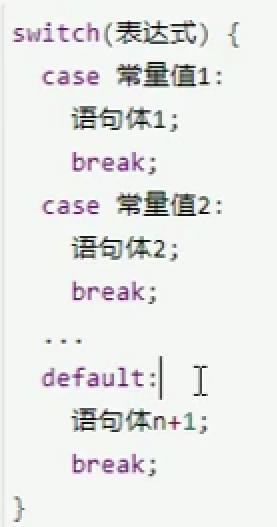


1. 三元运算符



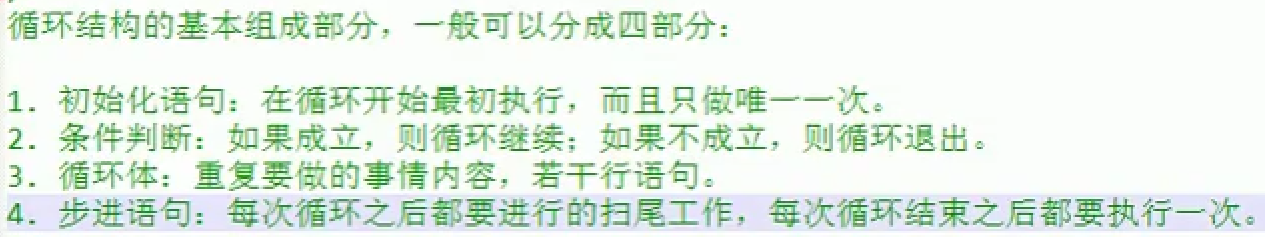
# Part 9选择结构

1. 顺序结构
2. 判断语句
3. 选择语句



# Part 10循环结构

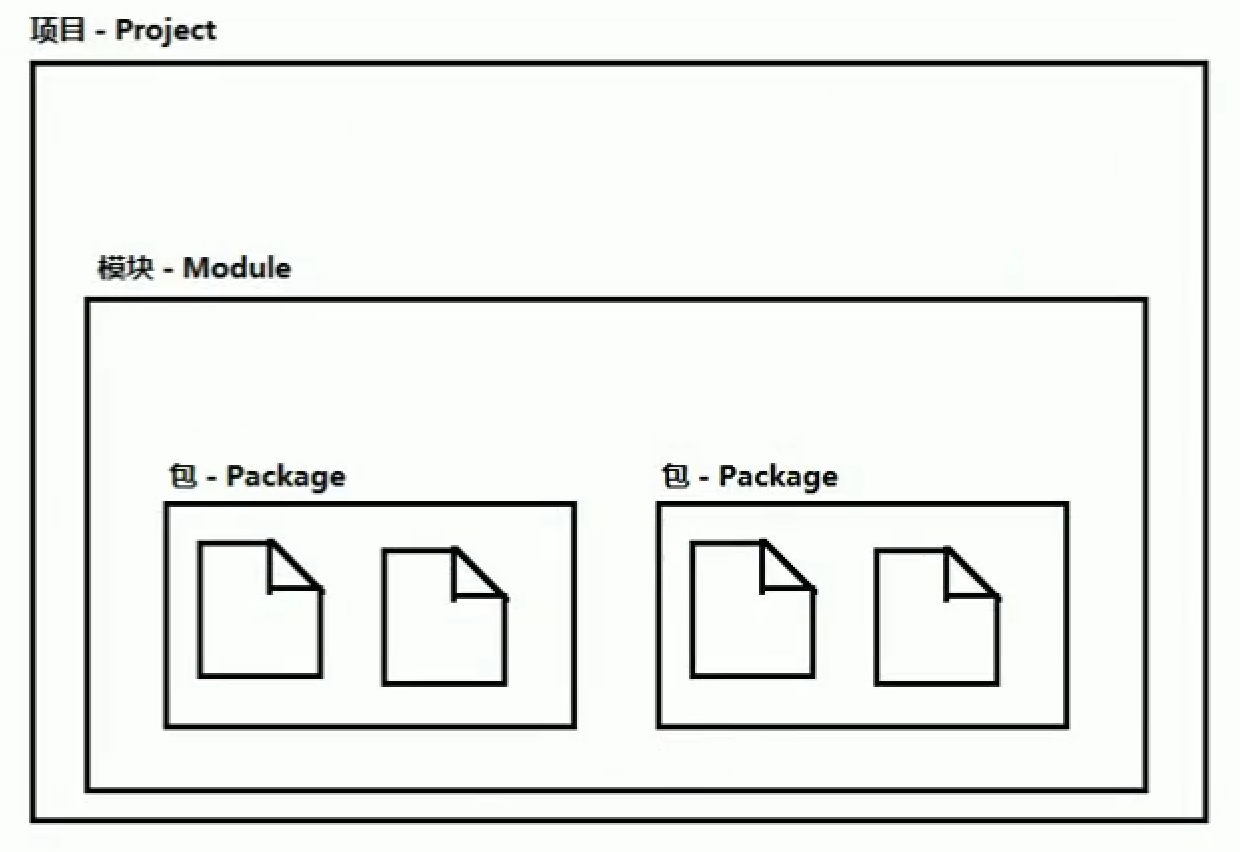
1. 循环结构的组成：初始化语句、条件判断、循环体、步进语句。



1. 循环语句包括3种：for、while、do…while
2. 三种循环的区别
3. 条件判断不满足，for、while一次都不执行；do…while执行一次
4. for循环的初始化变量在小括号里，只在循环内部使用；while、do…while的初始化变量在外部，循环执行完毕仍然可以使用变量。
5. break和continue

break跳出循环；continue跳出当前次循环，开始执行下一次循环。

# Part 11开发工具



常用快捷方式：

psvm+回车：public static void main(String [] args){}；

sout+回车：System.out.println();



# Part 12方法

1. 方法三要素：返回值类型、方法名称、参数列表。
2. 使用方法的注意事项：
3. 方法必须定义在类当中；方法不能嵌套。
4. 方法定义的前后顺序无所谓。
5. 要执行方法，必须调用方法（单独调用、打印调用、赋值调用）。
6. 如果方法有返回值，必须写“return 返回值；”。
7. Return后面的返回值数据，必须和方法的返回值类型对应。
8. 对于一个void没有返回值的方法，不能写return后面的返回值，只能是return自己。(return;)
9. 对于void方法当中最后一行的“return ;”可以省略。
10. 一个方法中可以有多个return语句，但是必须保证同时只有一个会被执行，两个return不能连写。

# Part 13方法重载

1. 方法的重载（Overload）：多个方法的名字一样，但是参数列表不一样。
2. 方法重载的注意事项：

方法重载与下列因素相关：参数个数不同、参数的类型不同、参数的多类型顺序不同。

方法重载与下列因素无关：参数的名称、与方法的返回值类型无关。

# Part 14数组

1. 数组：一种容器，可以同时存放多个数据。
2. 数组的**特点**：
3. 数组是一种引用数据类型；
4. 数组当中的多个数据，类型必须统一；
5. 数组的长度在程序运行期间不可改变。
6. 数组的初始化：在内存中创建一个数组，并且赋予默认值。
7. 动态初始化（指定长度）

格式：数据类型[] 数组名称 = new 数据类型[数组长度]；

元素将自动拥有默认值。规则如下：

如果是整数类型：默认为0；

如果是浮点类型：默认为0.0；

如果是字符类型：默认为‘\u0000’；

如果是布尔类型：默认为false；

如果是引用类型：默认是null。

1. 静态初始化（指定内容）

基本格式：数据类型[] 数组名称 = new 数据类型[] {元素1，元素2，…}；

省略格式：数据类型[] 数组名称 = {元素1，元素2，…}；

1. 注意事项

静态初始化没有指定长度，但是仍然会自动推算得到长度；

静态初始化标准格式可以拆分为两个步骤；

动态初始化也可以拆分为两个步骤；

静态初始化使用了省略格式，就不能拆分为两个步骤了。

1. 使用建议

如果不确定数组当中的具体内容，用动态初始化；否则，已经确定了具体的内容，用静态初始化。

1. 直接打印数组名称，得到的是数组对应的：内存地址哈希值。
2. 数组索引从0开始，一直到“数组长度-1”为止。
3. **Java的内存划分**：

