**关于switch语句：**

1、 switch 后面小括号中表达式的值必须是**整型或字符型**

2、 case 后面的值**可以是常量数值**，如 1、2；也**可以是一个常量表达式**，如 2+2 ；但不能是变量或带有变量的表达式，如 a \* 2

3、 case 匹配后，执行匹配块里的程序代码，如果没有遇见 **break**会继续执行下一个的 case 块的内容，直到遇到 break 语句或者 switch 语句块结束 。

4、 可以把功能相同的 case 语句合并起来，

5、 default 块可以出现在**任意位置**，也**可以省略**

**关于数组：**

1、 声明数组

语法： 数据类型[ ] 数组名；

或者 数据类型 数组名[ ]；

2、 分配空间

简单地说，就是指定数组中最多可存储多少个元素

语法： 数组名 = new 数据类型 [ 数组长度 ];

或者直接：数据类型[ ] 数组名={1，2，3…}

Arrays.sort()可以实现数组的自动排序

使用for(变量类型 变量名称：数组名称)语句输出数组

**如何定义方法：**

访问修饰符 返回值类型 方法名（参数列表）{方法体}

1、 访问修饰符：方法允许被访问的权限范围， 可以是 public、protected、private 甚至可以省略 ，其中 public 表示该方法可以被其他任何代码调用，其他几种修饰符的使用在后面章节中会详细讲解滴

2、 返回值类型：方法返回值的类型，如果方法不返回任何值，则返回值类型指定为 void ；如果方法具有返回值，则需要指定返回值的类型，并且在方法体中使用 return 语句返回值

3、 方法名：定义的方法的名字，必须使用合法的标识符

4、 参数列表：传递给方法的参数列表，参数可以有多个，多个参数间以逗号隔开，每个参数由参数类型和参数名组成，以空格隔开

根据方法是否带参、是否带返回值，可将方法分为四类：

Ø 无参无返回值方法

Ø 无参带返回值方法

Ø 带参无返回值方法

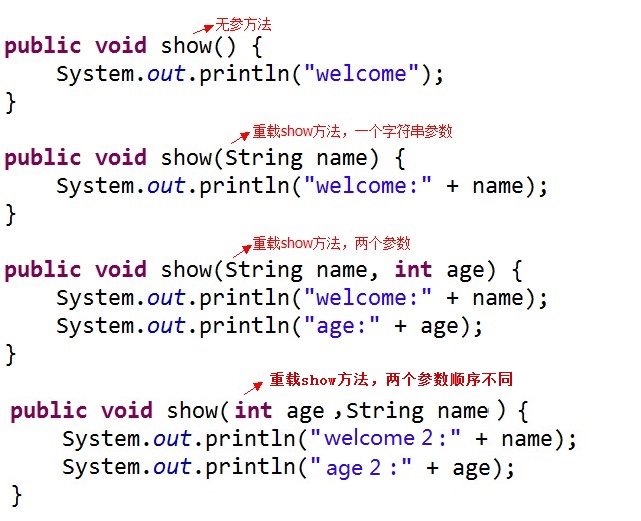
Ø 带参带返回值方法

类名 对象名=new 类名（）；

对象名.方法名（）；

方法的重载:

如果同一个类中包含了两个或两个以上方法名相同、方法参数的个数、顺序或类型不同的方法，则称为方法的重载，也可称该方法被重载了。如下所示 4 个方法名称都为 show ，但方法的参数有所不同，因此都属于方法的重载：

[](http://img.mukewang.com/5375c8f000010ecb06410521.jpg)

**类，对象，方法：**

Public class 类名{

变量1类型 变量1名；-----🡪成员变量（默认值为0）

变量2类型 变量2名；

…

}

Public 返回类型 方法1（）{

变量4类型 变量4名；-----🡪局部变量

}

Public 返回类型 方法2（）{

变量5类型 变量5名；

}

**构造方法：（创建方法调用的不是类而是构造方法）**

Public构造方法名（）{

//初始化代码

}

构造方法名与类名相同

New + 构造方法 创建一个对象

定义在类中的用来初始化对象的方法



**静态变量/方法（公用）：static 变量类型 变量名**

可以通过 类名.变量名 调用

可以通过 对象名.变量名 调用

**this关键字:**

this.属性=当前对象属性

this.方法=当前对象方法

**内部类：**

定义在普通类的内部，用以实现更好的封装

在创建内部类对象时，需要先创建外部类对象；

如外部类为outer，内部类为inner，则创建inner对象时

outer O=new outer();

inner I=O.new inner();

内部类方法调用外部类参数时：（如外部类outer中有参数name）

怎调用过程为：outer.this.name

**静态内部类**

静态内部类是 static 修饰的内部类，这种内部类的特点是：

1、 静态内部类不能直接访问外部类的非静态成员，但可以通过 new 外部类().成员 的方式访问

2、 如果外部类的静态成员与内部类的成员名称相同，可通过“类名.静态成员”访问外部类的静态成员；如果外部类的静态成员与内部类的成员名称不相同，则可通过“成员名”直接调用外部类的静态成员

3、 创建静态内部类的对象时，不需要外部类的对象，可以直接创建 内部类 对象名= new 内部类();

**继承：**

子类可以继承父类的特性，包括成员变量以及方法等

Public class father{

变量1；

方法1；

……}

Public class son extends father{

创建子类的对象

对象.变量；

对象.方法；

……}

继承父类的方法可以重写：但是注意与父类的方法名，返回值类型以及参数一致

**Super关键字**

子类方法想调用父类的属性可用 super.属性 来执行

1. 如果子类的构造方法中没有显式调用父类构造方法，则系统默认调用父类无参构造方法
2. 如果显式调用，则super()必须在子类构造方法的第一行
3. 如果子类构造方法中既没有显式调用父类构造方法，而父类又没有无参的构造方法，则编译出错

Object类中的toString()和equal()方法的重写：菜单栏中的source

**多态：**

引用多态：父类的引用可以指向本类的对象；父类的引用可以指向子类的对象

方法多态：方法可以继承父类，也可以自己写

**引用类型转换：**

Instancof运算符解决引用对象强制类型装还的安全性问题

**抽象类：定义父类用以约束子类**

规则：abstract定义抽象类；abstract定义抽象方法，只有声明，不需要实现；包含抽象方法的类是抽象类；抽象类中可以包含普通方法，也可以没有抽象方法；抽象类不能直接创建，可以定义引用变量

抽象方法没有方法体，以“；”结尾

**接口：**

类是一种具体的实现体，而接口定义了某一批类所需要遵守的规范，接口不关心这些类的内部数据，也不关心这些类里方法的实现细节，她之规定这些类里必须提供某些方法。

[修饰符](abstract)interface 接口名[extends 父接口1，父接口2…]

{

零到多个常量定义

零到多个抽象方法定义

}

接口就是用来继承的，一般用public，接口中的变量是常量，方法是抽象方法（不能有方法体）

使用接口： [修饰符]class 类名extends父类implement接口1，2

{

类体部分//如果继承了抽象类，需要实现继承的抽象方法；要实现接口中的抽象方法

}

UML（Unified Modeling Language）标准建模语言

1用例图：能够以可视化的方式，表达系统如何满足所收集的业务规则，以及特定的用户需求等信息

2序列图：用于按照交互发生的一系列顺序，显示对象之间的这些交互

3类图：UML类图、业务逻辑和所有支持结构一同被用于定义全部的代码结构