复习：

1. this关键字：代表当前在活动的对象的引用(地址)，用于在本类中调用本类的属性和方法。--- this语句，表示在本类的构造方法中调用本类中其他形式的构造方法 -- 必须放在构造方法第一行。

2. 代码块：

构造/初始化代码块---在创建对象的时候先于构造方法执行依次。用于完成一些对象需要的初始化操作

局部代码块 --- 定义在方法中，用于限制变量的使用范围和生命周期，以提高栈内存的利用率

3. 面向对象的特征：封装、继承、多态

4. 封装：

体现形式---方法、内部类、属性的私有化

意义：提高复用性，保证数据的合法性

### 继承

对一部分的类进行分析，将这些类中共有的属性和方法提取到了一个新的类中，然后利用extends关键字让原来的类和新的类产生了关系，这种关系称之为继承。---子类在继承父类之后可以使用父类中的一部分方法和属性。

在Java中，支持单继承，支持多层继承 --- 一个子类只能继承一个父类，但是一个父类可以有多个子类。

单继承和多继承的比较：在代码的复用性上，多继承要优于单继承。单继承在调用方法的明确性上要优于多继承

class A {

public int m(){return 1;}

}

class B {

public int m(){return 3;}

}

class C extends A, B {}

C c = new C();

int i = c.m(); --- 调用的时候会产生歧义

好处：提高代码的复用性，避免方法调用的时候产生歧义

注意：子类在继承父类的时候能够继承父类所有的属性和方法，但是只有一部分方法和属性对子类可见

#### super关键字

表示父类对象的引用。通过super可以在子类中调用父类的方法和属性。

super语句---表示在子类中调用父类对应形式的构造方法，来创建一个父类对象，创建好的父类对象用super表示。---super语句必须写在子类构造方法的第一行。---每创建一个子类对象都会伴随着创建一个父类对象。

super语句和this语句能共存吗？---不能

练习：定义一个类表示矩形(Rectangle),提供获取这个矩形周长和面积的方法，提供一个子类表示正方形

考虑：在子类中使用和子类对象使用是一会事儿吗？---不是

#### 方法的重写

在父子类中存在了方法签名一致的非静态方法，就称之为是方法的重写/覆盖。

方法在重写的过程中需要遵守5个原则 --- “两等两小一大”

1. 方法签名相同

2. 子类重写的方法的权限修饰符的范围要大于等于父类对应方法的权限修饰符的范围

class A {

protected void m(){}

}

class B extends A {

public void m(){}

}

3. 如果父类方法的返回值类型是基本类型/void,那么子类在重写方法的时候要求返回值类型一致

class A {

public void m(){}

}

class B extends A {

public void m(){}

}

4. 如果父类方法的返回值类型是引用类型，那么子类重写的方法的返回值类型和父类方法返回值类型一致或者是父类方法返回值类型的子类

class A {}

class B extends A {}

class C {

public A m(){return null;}

}

class D extends C {

public B m(){return null;}

}

### 多态

编译时多态：方法的重载

add(2,3) --- add(int, int )

add(2,3,4) --- add(int, int, int)

运行时多态：向上造型和方法的重写 --- 基于继承

注意：如果是利用向上造型来创建的对象，那么这个对象能干什么看的是父类；具体的怎么干看的是子类

#### 重写的理解

1. 子类重写的方法的权限修饰符的范围要大于等于父类对应方法的权限修饰符的范围

class A {

public void m(){}

}

class B extends A {

protected void m(){} // 不行

}

A a = new B(); // a对象是用A类来声明的，意味着a对象能干什么看的A类。A类就告诉了a对象有一个m方法可以使用，并且m是使用public修饰，可以在任何一个地方使用

a.m(); // a对象的实际创建类型是B类，就意味着m方法的执行要看B类。在执行m方法的时候，m方法是用protected修饰，不能在其他的类中使用。 --- 声明的时候说的是m方法可以在任何地方使用，实际执行的时候却不能在其他类中用

2. 如果父类方法的返回值类型是引用类型，那么子类重写的方法的返回值类型和父类方法返回值类型一致或者是父类方法返回值类型的子类

class A {}

class B extends A {

public void mb(){}

}

class C {

public B m(){return new B();}

}

class D extends C {

public A m(){return new A();} // 不行

}

C c = new D(); // c对象是用C类声明的，那么c对象能干什么看的是C类。C类告诉c对象有一个m方法可以使用，并且方法的返回值类型是B。

B b = c.m(); // c对象的实际类型是D类，那么c对象如何执行m方法看的是D类。D类在定义m方法的时候告诉c对象返回值类型是A --- 用子类B接了父类A

b.mb(); // 假设子类能够接住父类，当b对象调用mb方法的时候看的是实际类，b对象的实际类是A类型，A类中没有mb方法

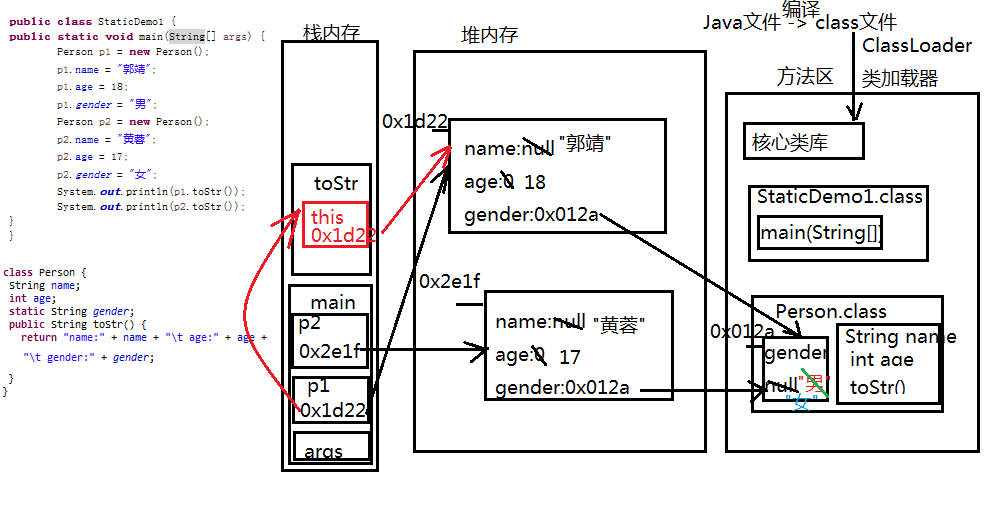
注意：在Java中，基本类型之间是没有继承关系

## static---静态

修饰变量、方法、内部类以及代码块

### 静态变量

static修饰变量称之为静态变量/类变量。静态变量是随着类的加载而加载到方法区，并且在方法区中被赋予了默认值。由于静态变量是先于对象出来的，所以可以不通过对象而是通过类名来调用静态变量。每一个对象存的是这个静态变量所对应的地址，所以静态变量是被这个类的所有对象所共享的



注意：

1. 类在内存中是存储在方法区

2. 类在第一次使用的时候才加载

静态变量可以在方法中定义吗？ --- 不可以 --- 方法在调用的时候执行，到栈内存中执行；静态变量在类加载的时候初始化，存储在方法区。

静态变量可以在构造代码块中定义吗？ --- 不可以 --- 构造代码块在创建对象的时候执行，在栈内存中执行；静态变量先于对象出现，存储在方法区

练习：定义一个类，统计这个类创建的对象的个数