# 一、this关键字

1、this关键字

* this关键字代表了所属函数的调用者对象（调用该方法的是哪个对象，this就代表了这个对象）

2、this关键字的作用

* 如果存在同名的成员变量与局部变量时，方法内部默认调用的是局部变量数据（java就近原则）。可以通过this关键字指定访问成员变量数据。
* 在一个构造函数中，可以调用本类另一个构造函数初始化对象
  + - this(name) 表示调用本类中一个参数的构造函数

3、this关键字注意事项

* 如果在一个方法中访问一个变量，该变量只存在成员变量的情况下（不存在同名的局部变量），那么java编译器会在该变量的 前面添加this关键字

4、this关键字调用其他构造函数时要注意的事项

* this关键字调用其他构造函数时，必须位于构造函数的第一个语句
* this关键字在构造函数中不能出现互相调用，因为这是一个死循环。

# 二、static关键字

## 1、static关键字（公共的、共享的）

* static（静态\修饰符）
* 如果有数据需要共享给所有对象使用时，那么就可以使用static修饰

## 2、static修饰成员变量

（1）静态成员变量的访问方式

* 方式一：可以使用对象进行访问

格式：对象.变量名

* 方式二：可以使用类名进行访问

格式：类名.变量名

（2）注意：

* 非静态成员变量只能使用对象进行访问，静态成员变量可以使用 对象 和 类名 两种方式进行访问。
* 只有成员变量的数据需要共享的时候，才使用static关键字
* 共享的（static）成员变量，被任意修改一次的话，以后的数据都是被修改后的数据。

（3）static修饰成员变量的应用场景

* 如果一个变量需要被所有对象共享使用时，这个时候即可使用static关键字

## 3、static修饰函数（静态的成员方法）

### （1）访问方式：

* 方式一：可以使用对象进行访问。

格式：对象.静态的函数名();

* 方式二：可以使用类名进行访问。

类名.静态函数名字。

* **推荐使用是类名直接访问静态的成员。**

### （2）静态的成员变量与非静态的成员变量的区别：

1. 作用上的区别：

* + 静态的成员变量的作用共享一个 数据给所有的对象使用。
  + 非 静态的成员变量的作用是描述一类事物的公共属性。

2. 数量与存储位置上的区别：

* 静态成员变量是存储方法 区内存中，而且只会存在一份数据。
* 非静态的成员变量是存储在堆内存中，有n个对象就有n份数据。

3. 生命周期的区别：

* 静态的成员变量数据是随着类的加载而存在，随着类文件的消失而消失。
* 非静态的成员数据是随着对象的创建而存在，随着 对象被垃圾回收器回收而消失。

### （3）静态函数要注意的事项：

1. 静态函数是可以使用类名或者对象进行调用的，而非静态函数只能使用对象进行调用。

2. 静态的函数可以直接访问静态的成员，但是不能直接访问非静态的成员。

* 原因：静态函数是可以使用类名直接调用的，这时候可能还没有存在对象，而非静态的 成员数据是随着对象 的存在而存在的。

3. 非静态的函数是可以直接访问静态与非静态的成员。

* 原因：非静态函数只能由对象调用，当对象存在的时候，静态数据老早就已经存在了，而非静态数据也随着对象的创建而存在了。

4. 静态函数不能出现this或者super关键字。

* 原因：因为静态的函数是可以使用类名调用的，一旦使用类名调用这时候不存在对象，而this关键字是代表了一个函数 的调用者对象，这时候产生了冲突。

### （4）静态的数据的生命周期：

* 静态的成员变量数据是优先于对象存在的。

### （5）static什么时候修饰一个函数？

* 如果一个函数没有直接访问到非静态的成员时，那么就可以使用static修饰了。 **一般用于工具类型的方法**

### （6）静态函数不能访问非静态的成员？

* 静态函数只要存在有对象，那么也可以访问非 静态的数据。只是不能直接访问而已。

# 三、super关键字

## 1、super关键字代表了父类空间的引用

## 2、super关键字的作用

* 父子类存在着同名的成员（函数和成员变量）时，在子类中默认是访问 子类的成员，可通过super关键字指定访问父类的成员。
* 创建子类对象时，默认会先调用父类的 无参构造函数，可通过super调用父类指定的构造函数。

## 3、super关键字调用父类构造方法的注意事项

1. 如果在子类的构造方法上没有指定调用父类的构造方法，那么java编译器会在子类的构造方法上面加上super()语句。

2. super关键字调用父类的构造函数时，该语句必须要是子类构造函数中的第一个语句。

3. super与this关键字不能同时出现在同一个构造函数中调用其他的构造函数。因为两个语句都需要第一个语句。

## 4、super关键字与this关键字的区别：

1. 代表的事物不一致。

1. super关键字代表的是父类空间的引用。

2. this关键字代表的是所属函数的调用者对象。

2. 使用前提不一致。

1. super关键字必须要有继承关系才能使用。

2. this关键字不需要存在继承关系也可使用。

3. 调用构造函数的区别：

1. super关键字是调用父类的构造函数。

2. this关键字是调用本类的构造函数。

# 四、instanceof关键字

1、instanceof关键字的作用

* 判断一个对象是否属于指定的类别
* **多态时有用，一般 我们做强制类型转换之前，都会使用instanceof关键字判断一下，然后进行转换。**

2、instanceof关键字的使用前提

* 判断的对象与指定的类别必须要存在继承或实现的关系

3、instanceof关键字的使用格式

* 对象 instanceof 类别

# 五、final关键字

## 1、final（最终的、修饰符）

* 被final修饰的变量，变量名需要全部大写，单词之间用 下划线（\_）连接

## 2、final关键字的用法

（1）final 修饰 基本类型（8个）变量：

* final修饰基本类型变量时，该变量不能重新赋值，第一次值为最终值

（2）final修饰 引用类型变量

* fianl修饰引用类型变量时，该变量不能重新指向新的对象

（3）final修饰 函数

* 该函数不能被重写

（4）final修饰 类

* final修饰的类不能被继承

**常量：一般常量都是用 public static final 修饰的**

* public 公共的，放开权限，可以被任何地方访问到
* static 静态的，在类被编译器加载时就被加载，不会出现找不到该常量的情况
* final 最终的，该常量不能被修改