字符串是引用类型，比较特别

类似 包装类

new 构造，所以每次new 都不是同一个对象

但是不能输入地址，因为底层重写了toString()后面解释

== 判断的是两个对象的地址是否相同

字符串重写了equals 后面解释

字符串比较特别，可以不用new 直接赋值

可以单作基本类型使用，也可以单作引用类型来理解

直接赋值的，可以单作基本类型使用

编译阶段对字符串的拼接

# 继承特性

以上都是面向对象的基础

类和对象之间的关系 构造 构造函数 类执行过程

判断== equals 的用法

定义一个类A

10个属性

再定义一个类B

11属性 其中10个属性和前面那个类是一样的

不需要重新定义10个属性，而是继承A 类，这样B类就有A 类的10个相同的属性

避免了重复定义

is 描述

B类是一个A类

人类

学生类

老师类

学生和老师两个类不要重复定义了，继承人类

学生是人类

老师是人类

is 关系的

is 前面那个类叫子类

后面那个类叫父类 基类

特点

单一继承

子类只能继承一个父类

相对于C++语言，它是多继承的

学生 继承人类 又继承神类

人神

java中只能继承一个类型

学生 zhansan

类型不确定了。

只能单继承

但是可以延续继承

学生 》大学生类 》人类

所有的类都继承Object类型的

如果你显式继承Object类，那么就不能再继承其他类了

不需要继承Object，默认就继承了

类的执行顺序

在继承关系中

父类的构造函数先运行

现象是先调用父类构造函数，其实应该还是调用子类的

子类中有一行代码 隐式调用父类的构造函数。

为何实例化子类，还要去调用父类的构造函数

就是初始化属性的值

默认调用父类的无参的构造函数

不是说调用子类的有参就调用父类有参

显式去调用有参的构造函数

在设计类的时候最好手动写一个无参的构造函数

递归调用

排序 面试概率比较

自身调用方法形成递归调用，类似于循环

知道执行的次数，用循环

不知道次数，根据结果执行次数的，递归操作

算法

开发中

技术 逻辑 算法

冒泡排序

sort 实现了算法

普通方法是在一对大括号中实现功能，定义成抽象以后，就不需要实现方法了。抽象方法

类中只要有个方法是抽象的，那么类就定义为抽象类

抽象类是不能被实例化

实现一个类 对象就可以调用所有的方法了。

抽象类中的部分方法可以有实现。

软件开发=技术+逻辑+算法+设计+英文

总结所以为什么需要抽象类的设计

标准，没有实现。规范

final 常量

作用在标识符上的

作用在方法上面

作用在类上面

固定的意思，不能修改，不能重写 不能继承