# 方法的重写（override）

在子类继承父类时，子类的方法和父类的方法相同（访问修饰限制符， 返回值类型， 方法名， 参数列表），方法体不同。这种子类的方法将父类的方法覆盖叫做重写。

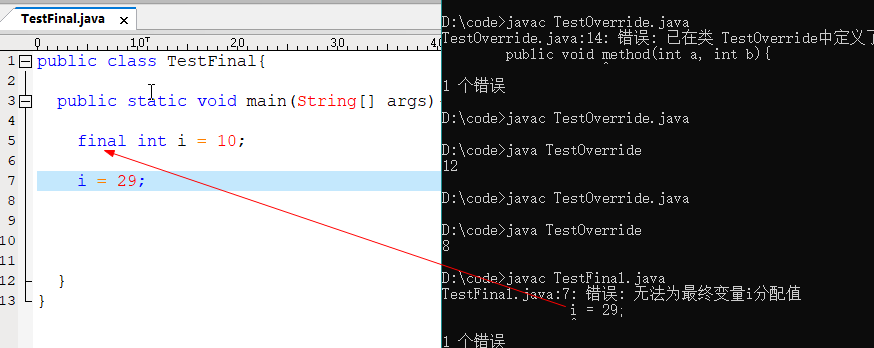
**重写和重载的区别：**

重写（override）：在父子类的继承中有相同的方法，唯一不同就是方法体，一般是父类的该方法满足不了子类的需求所以才发生重写。

重载（overload）：重载是在同一个类中，有着相同的方法名但是参数的数据类型或者参数的个数不同这两个方法就是重载。重载的目的：节省类中的命名资源和提高代码的可读性。

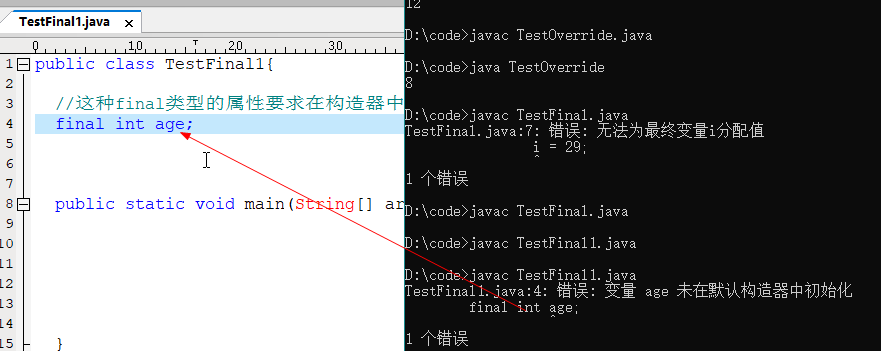
# final关键字

## final修饰在局部变量上



Final修饰局部变量的话，这个变量不能修改，就变成了常量。

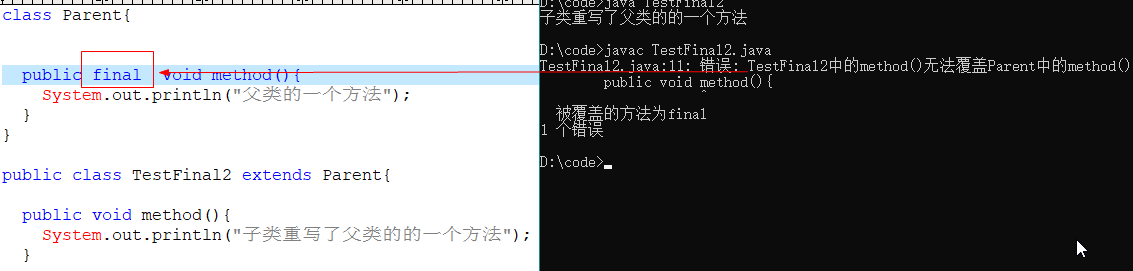
## final修饰在属性上



属性都有默认值，但是final修饰的话就没有默认值，要求要在构造器中给值， 一旦这个值赋值完毕就不能修改了，如果不在构造器中赋值，就直接给一个默认值

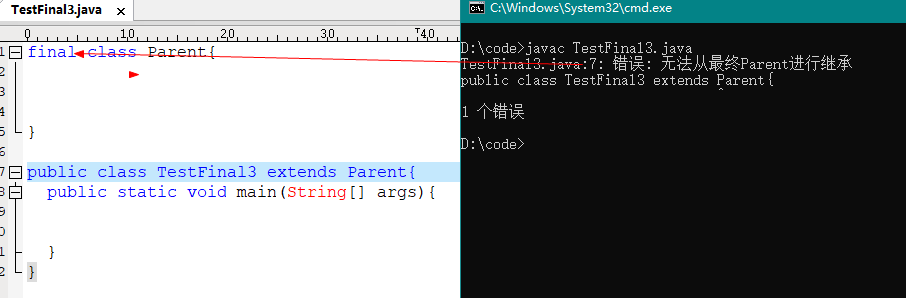
|  |
| --- |
| public class TestFinal1{    //这种final类型的属性要求在构造器中初始化  final int age = 10;      public TestFinal1(){  //age = 10;  }    public static void main(String[] args){    TestFinal1 tf = new TestFinal1();  //tf.age = 11;  }  } |

## final修饰在方法上



如果final修饰在方法上的话，不可以被子类重写。

## final修饰在类上



Final修饰在类上，不可以被继承。

# 抽象类

当多个具体的实体类存在着共同的行为，但是有不同的表现，我们在父类继承过程中父类的方法具体实现不同方式，但是能确定的是他们都有这种行为。

我们要把这种行为方法作为**抽象**方法。

Teacher

String name

Int age

Public void sleep(){

}

JavaTeacher PhpTeacher

Public void teachPhp(){

}

Public void teachJava(){

}

Teacher

Teacher

String name

Int age

Public void sleep(){

}

Public void teach();

JavaTeacher PhpTeacher

Public void teach (){

//教php

}

Public void teach (){

//教java

}

public abtract [返回值][void] 方法名 (参数列表); 没有方法体。如果在一个类中存在一个方法，那么这个类一定是抽象类。这个类上也要有abstract标识，

抽象类的语法：

abstract class 类名{

属性

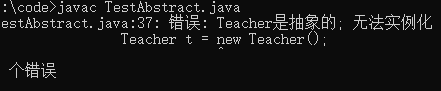
方法

}

注意：抽象类中可以没有抽象方法，但是有抽象方法的类一定是抽象类

## 抽象类的特点

1. 抽象类不能被实例化。



1. 抽象类必须被子类继承才能使用子类实例化。
2. 继承了抽象类的非抽象类必须实现抽象类的所有抽象方法
3. 抽象类也可以继承抽象类，这样不需要实现父抽象类的方法。
4. 抽象类的抽象方法不能和private，final， static共存。