# 面向对象(基础)

黑马程序员
www.itheima.com

传智教育旗下
高端IT教育品牌

## 面向对象

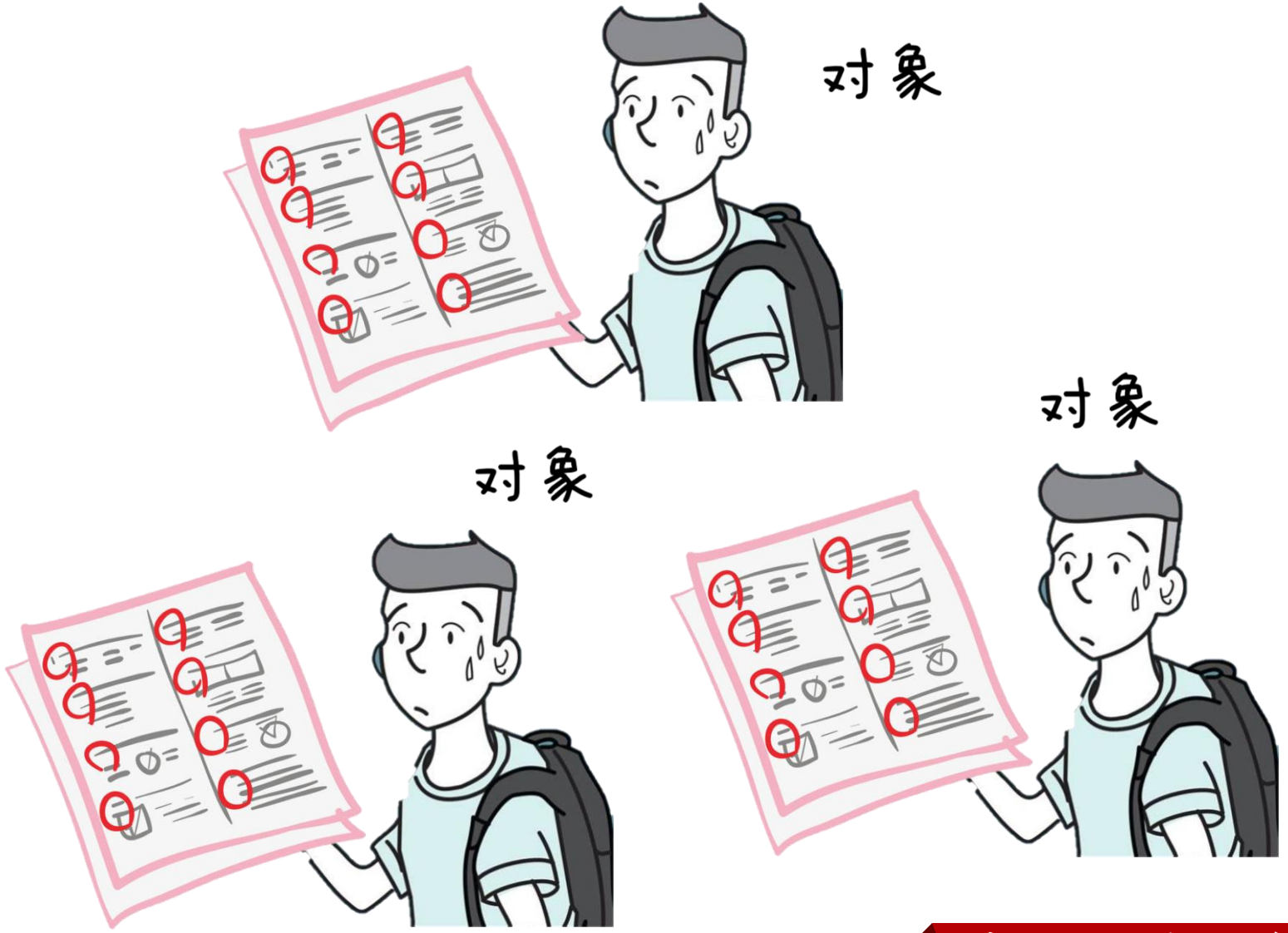
## 面向对象介绍

- 并不是一个技术，而是一种编程指导思想
- 以什么形式组织代码；以什么思路解决问题

## 为什么要用面向对象编程

- 因为生活中，我们解决问题时，就是采用这种指导思想去解决的。所以，我们写程序去解决问题时，如果也能采用这种指导思想就会使编程变得非常简单，程序也便于人理解


对象
对象


对象
对象
对象


对象
对象


对象
对象

```
public class Test {

    public static void main(String[] args) {
        老师 t = new 老师();
        试卷 s = t.出题();
        学生 stu = new 学生();
        stu.考试(s);
        t.批卷(s);
    }

}
```

```
public class Test {

    public static void main(String[] args) {
            顾客 g = new 顾客();
            售货员 s = new 售货员();
            钱 m = g.掏钱();
            手机 p = s.卖手机(m);
            g.获取手机(p);
    }
}
```

## 面向对象介绍

- 并不是一个技术，而是一种编程指导思想
- 以什么形式组织代码；以什么思路解决问题

## 为什么要用面向对象编程

- 因为生活中，我们解决问题时，就是采用这种指导思想去解决的。所以，我们写程序去解决问题时，如果也能采用这种指导思想就会使编程变得非常简单，程序也便于人理解

# 面向对象,重点学习什么?

学习如何自己设计对象并使用

学习获取已有对象并使用

# 目录

Contents

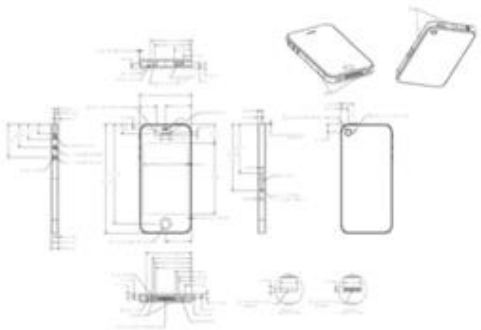
# 目录

Contents

## 什么是类

- 类是对现实生活中一类具有共同属性和行为的事物的抽象
- 我们可以将其理解为对象的设计图

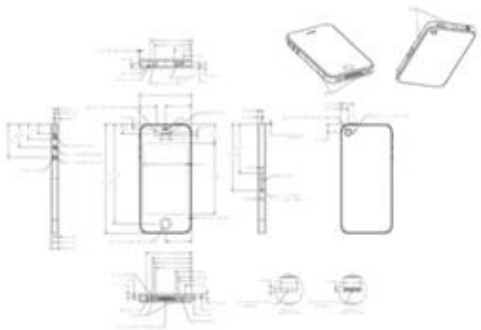
设计图



## 什么是对象

- 是能够看得到摸的着的真实存在的实体
- 小结：类是对象的抽象，对象是类的实体

设计图

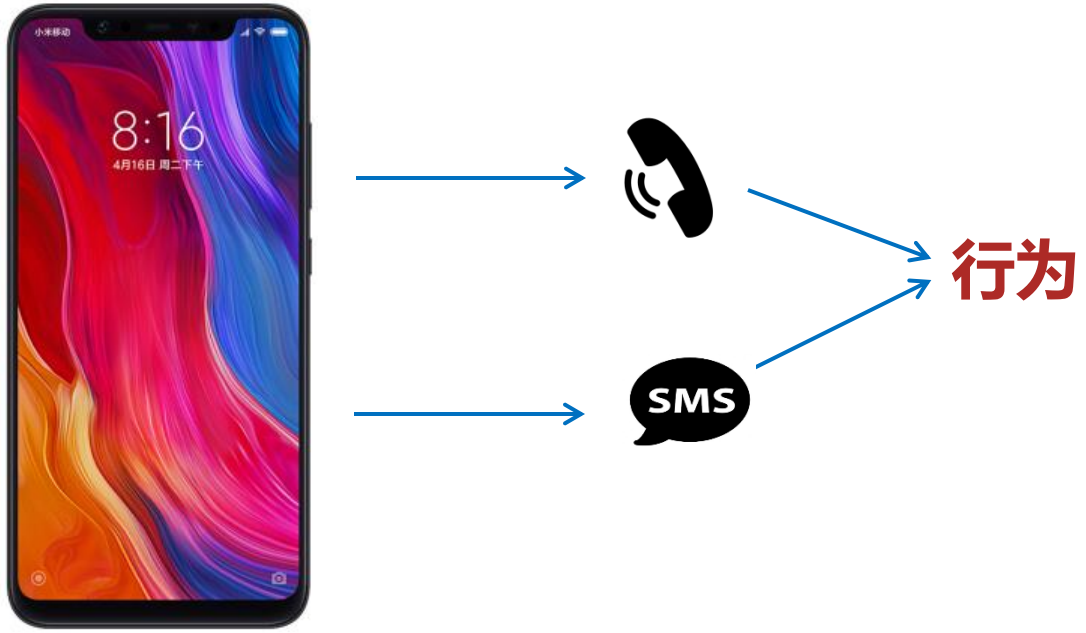






## 对象的属性和行为

- 属性：对象具有的各种特征，每个对象的每个属性都拥有特定的值
- 行为：对象能够执行的操作

| 属性 | 值 |
| --- | --- |
| 品牌： | 小米 |
| 价格： | 2999 |
| 内存： | 128G |
| … | |



## 类的定义

- 类是什么：是对现实生活中一类具有共同属性和行为的事物的抽象，确定对象将会拥有的属性和行为

类的组成：属性和行为

- 属性：在类中通过成员变量来体现（类中方法外的变量）
- 行为：在类中通过成员方法来体现（和前面的方法相比去掉static关键字即可）

# 类的定义

**类的定义步骤：**

① 定义类

```
public class 类名 {

}
```

# 类的定义

**类的定义步骤：**

① 定义类

② 编写类的成员变量

```
public class 类名 {
        // 成员变量
        变量1的数据类型 变量1;
        变量2的数据类型 变量2;
        ...



}
```

# 类的定义

**类的定义步骤：**

① 定义类

② 编写类的成员变量

③ 编写类的成员方法

```
public class 类名 {
        // 成员变量
        变量1的数据类型 变量1;
        变量2的数据类型 变量2;
        ...
        // 成员方法
        方法1;
        方法2;
        ...
}
```

## 类的定义


```
public class 类名 {
        // 成员变量
        变量1的数据类型 变量1;
        变量2的数据类型 变量2;
        ...
        // 成员方法
        方法1;
        方法2;
        ...
}
```

# 对象的使用

## 创建对象

- 格式：类名 对象名 = new 类名();
- 范例：Phone p = new Phone();

## 使用对象

**1：使用成员变量**

- 格式：对象名.变量名
- 范例：p.brand

**2：使用成员方法**

- 格式：对象名.方法名(参数)
- 范例：p.call()

练习

# 学生类的定义和使用

需求：定义一个学生类，然后定义一个学生测试类，在学生测试类中通过对象完成成员变量和成员方法的使用

分析：

① 定义一个学生类

类名：**Student**

成员变量：**name，age**

成员方法：**study()，doHomework()**

② 定义学生测试类

类名：**StudentDemo**

因为要做测试，所以有一个主方法：**main**方法

③ 在学生测试类中通过对象完成成员变量和成员方法的使用

给成员变量赋值，输出成员变量的值

调用成员方法

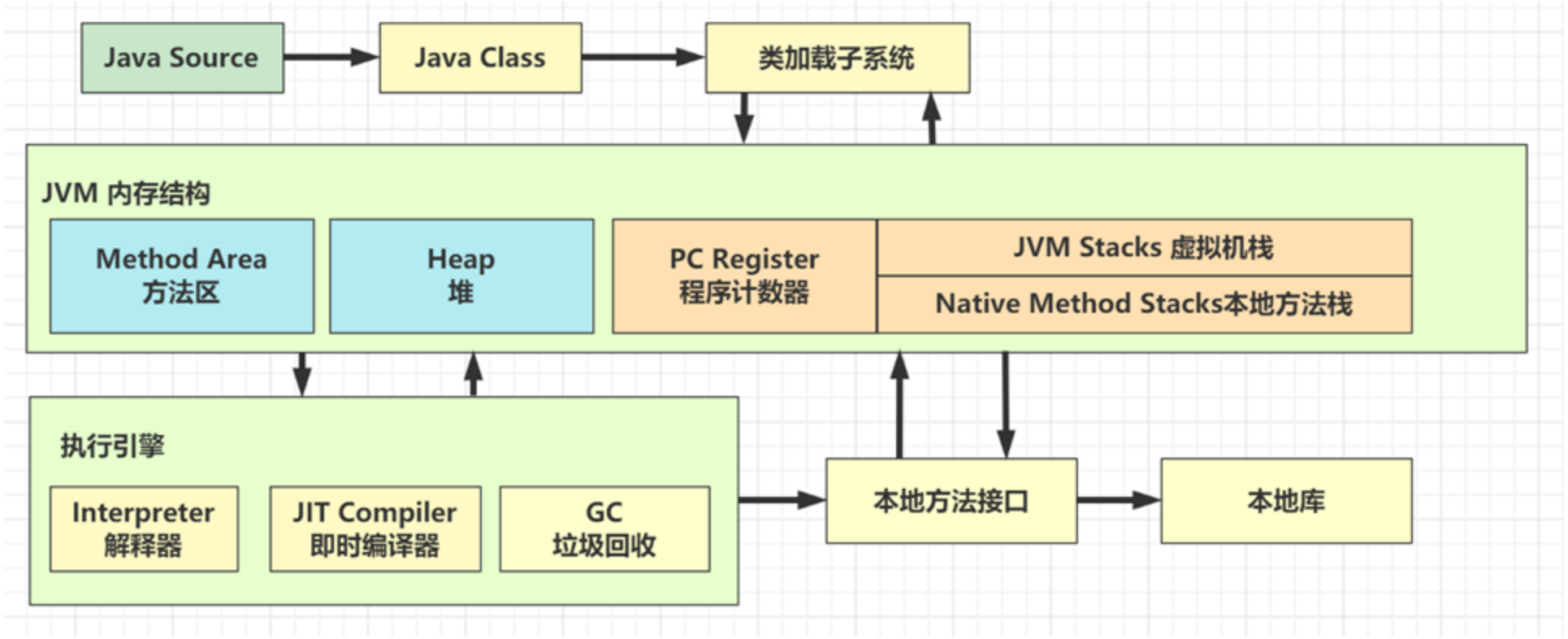
# 目录

Contents

## Java 内存分配

- Java 程序在运行时，需要在内存中分配空间。为了提高运算效率，就对空间进行了不同区域的划分，因为每一片区域都有特定的处理数据方式和内存管理方式。

## Java 内存分配




- 栈(虚拟机栈)
- 堆

## Java 内存分配

栈：所有局部变量都会在栈内存中创建

- 局部变量：定义在方法中的变量或者方法声明上的变量
- 方法执行都会加载到栈中进行
- 局部变量特点：随着方法的调用而存在，随着方法的调用完毕而消失

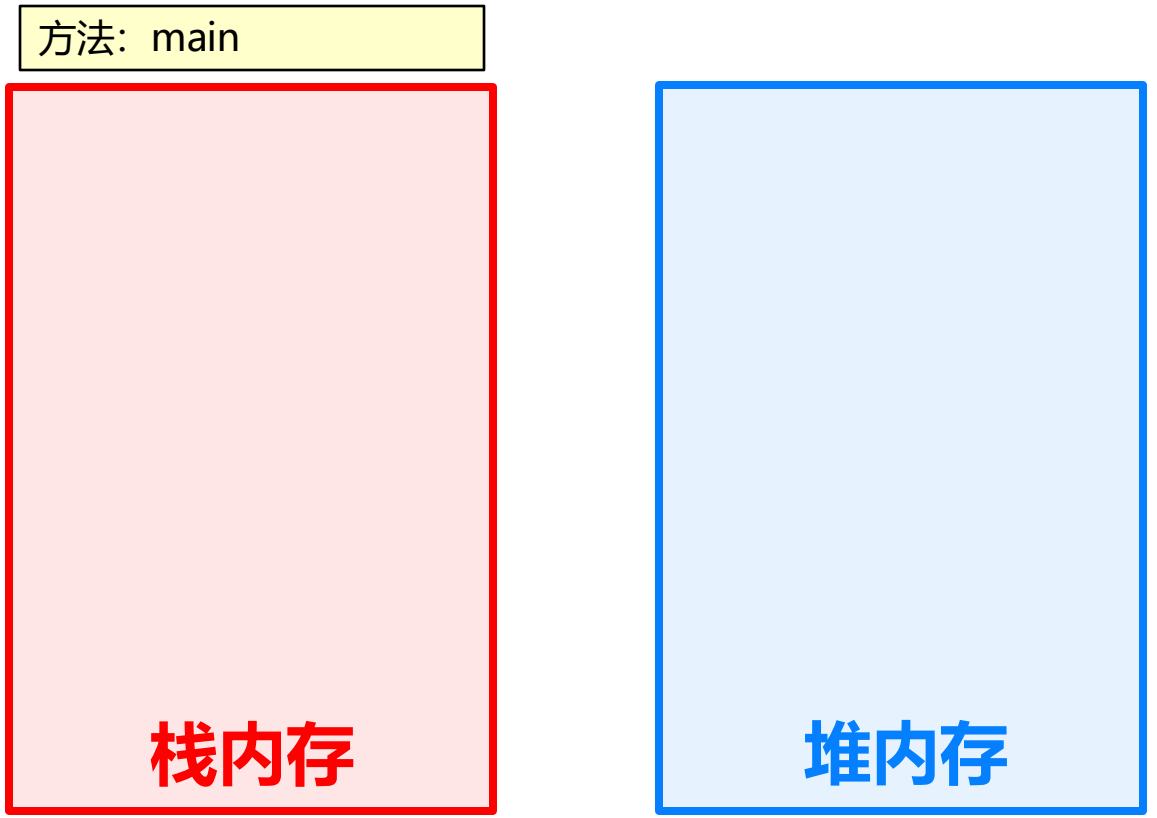
- 代码：Student s = new Student();

堆：所有对象及其对应的实例变量和数组都将存储在此处

- 简单理解为：new出来的东西，都存储在堆内存
- 每一个new出来的东西都有一个地址值，使用完毕，会在垃圾回收器空闲时被回收
- 实例变量(成员变量)有初始化值：
  - 基本数据类型：整数：0，浮点数：0.0，布尔：false，字符：空字符
  - 引用数据类型：null

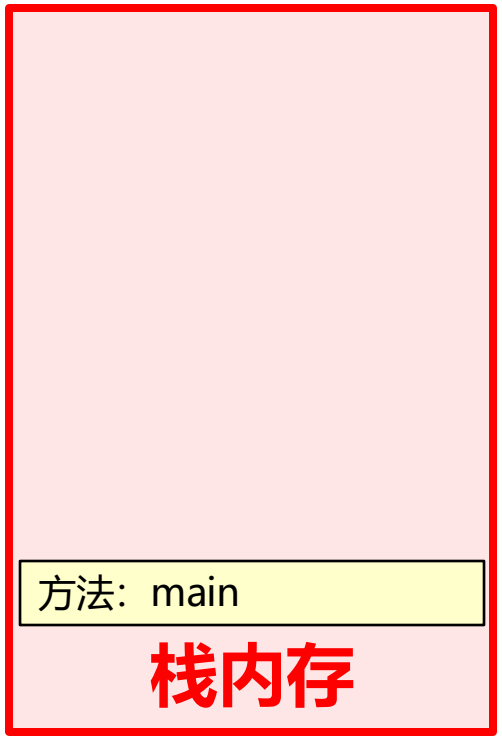
## 单个对象

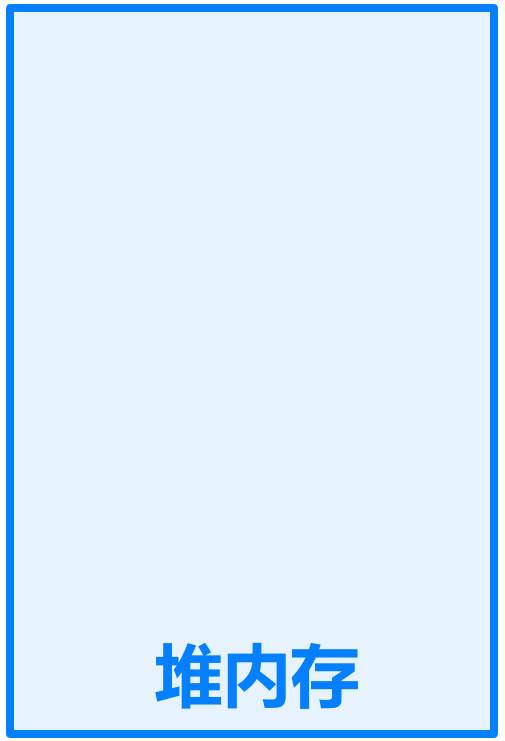
```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
        //使用成员变量
        System.out.println(s.name + "," + s.age);
        s.name = "张曼玉";
        s.age = 28;
        System.out.println(s.name + "," + s.age);
        //使用成员方法
        s.study();
        s.doHomework();
    }
}
```



## 单个对象

```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
        //使用成员变量
        System.out.println(s.name + "," + s.age);
        s.name = "张曼玉";
        s.age = 28;
        System.out.println(s.name + "," + s.age);
        //使用成员方法
        s.study();
        s.doHomework();
    }
}
```

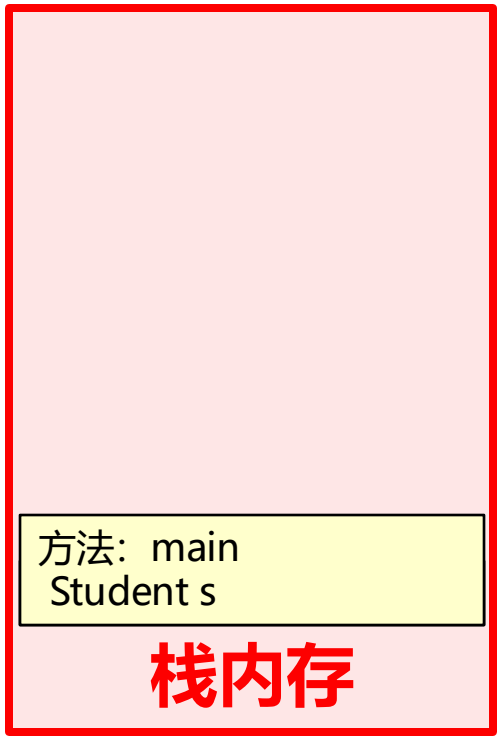


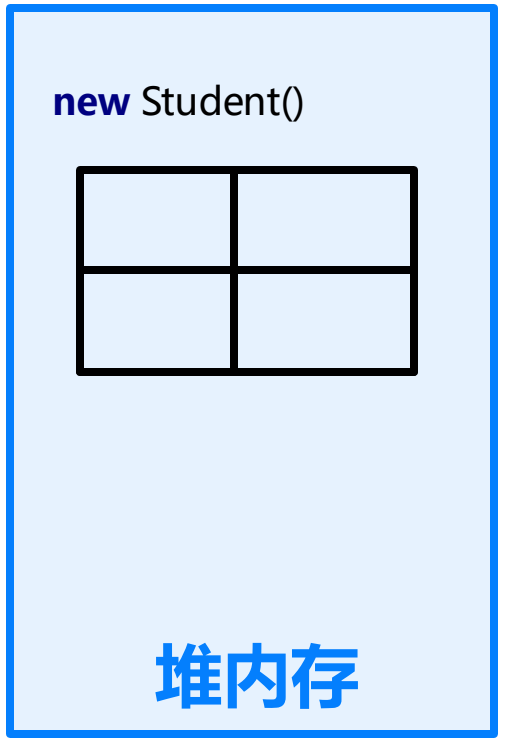



## 单个对象

```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
    }
}
```

```
public class Student {
    //成员变量
    String name;
    int age;
    //成员方法
    public void study() {
        System.out.println("好好学习");
    }
    public void doHomework() {
        System.out.println("多做练习");
    }
}
```

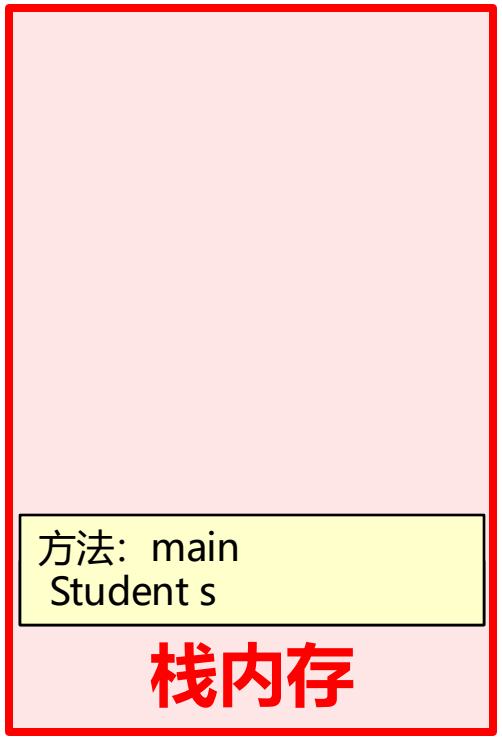


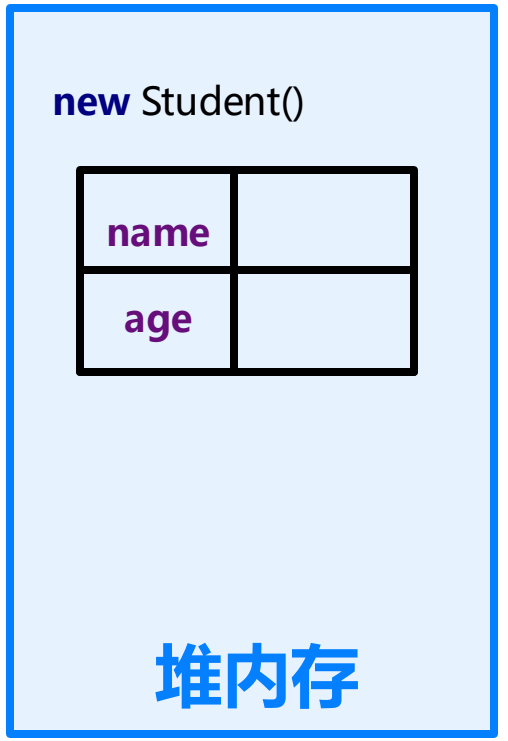



## 单个对象

```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
    }
}
```

```
public class Student {
    //成员变量
    String name;
    int age;
    //成员方法
    public void study() {
        System.out.println("好好学习");
    }
    public void doHomework() {
        System.out.println("多做练习");
    }
}
```



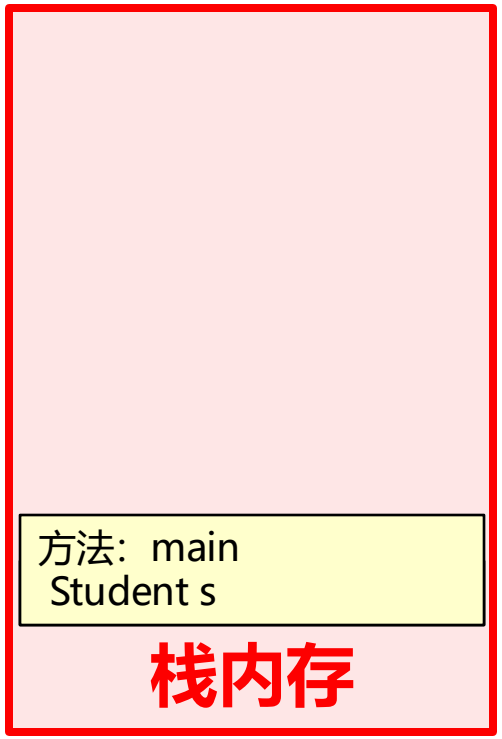



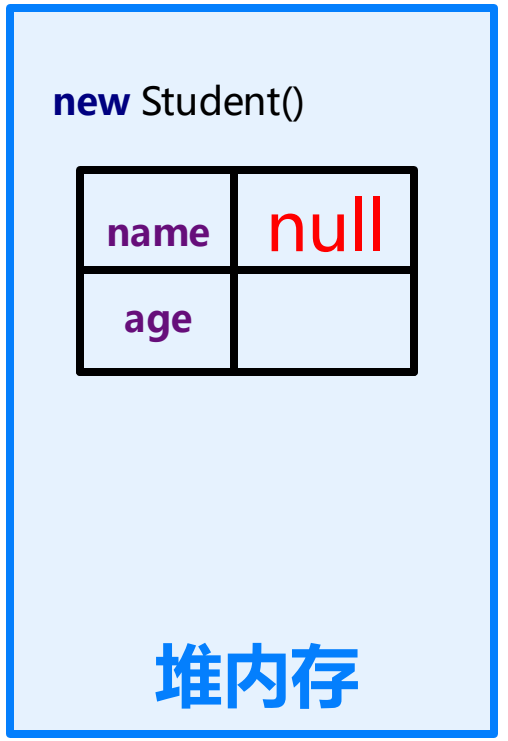

**成员变量都具有默认值**

## 单个对象

```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
    }
}
```

```
public class Student {
    //成员变量
    String name;
    int age;
    //成员方法
    public void study() {
        System.out.println("好好学习");
    }
    public void doHomework() {
        System.out.println("多做练习");
    }
}
```



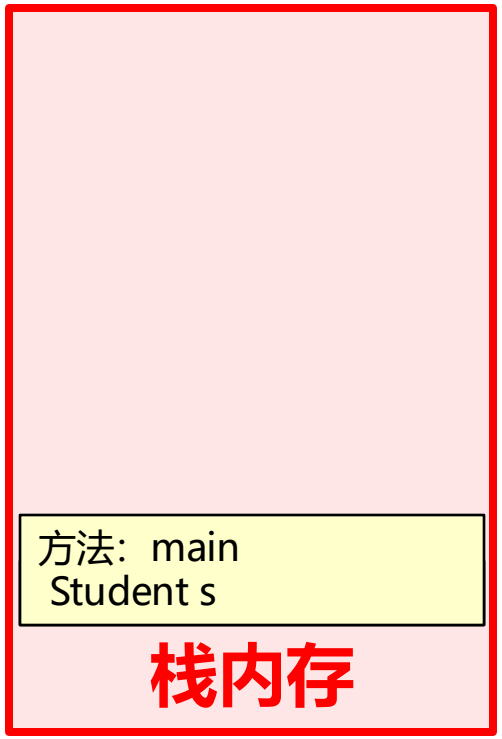



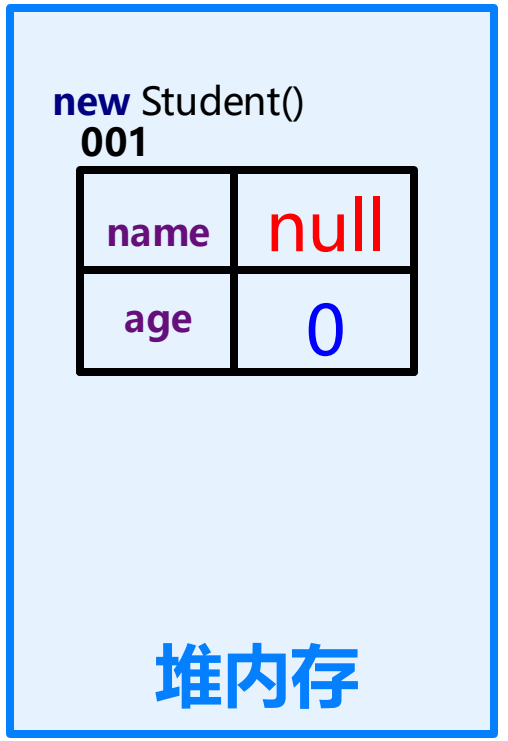

**成员变量都具有默认值**

## 单个对象

```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
    }
}
```

```
public class Student {
    //成员变量
    String name;
    int age;
    //成员方法
    public void study() {
        System.out.println("好好学习");
    }
    public void doHomework() {
        System.out.println("多做练习");
    }
}
```

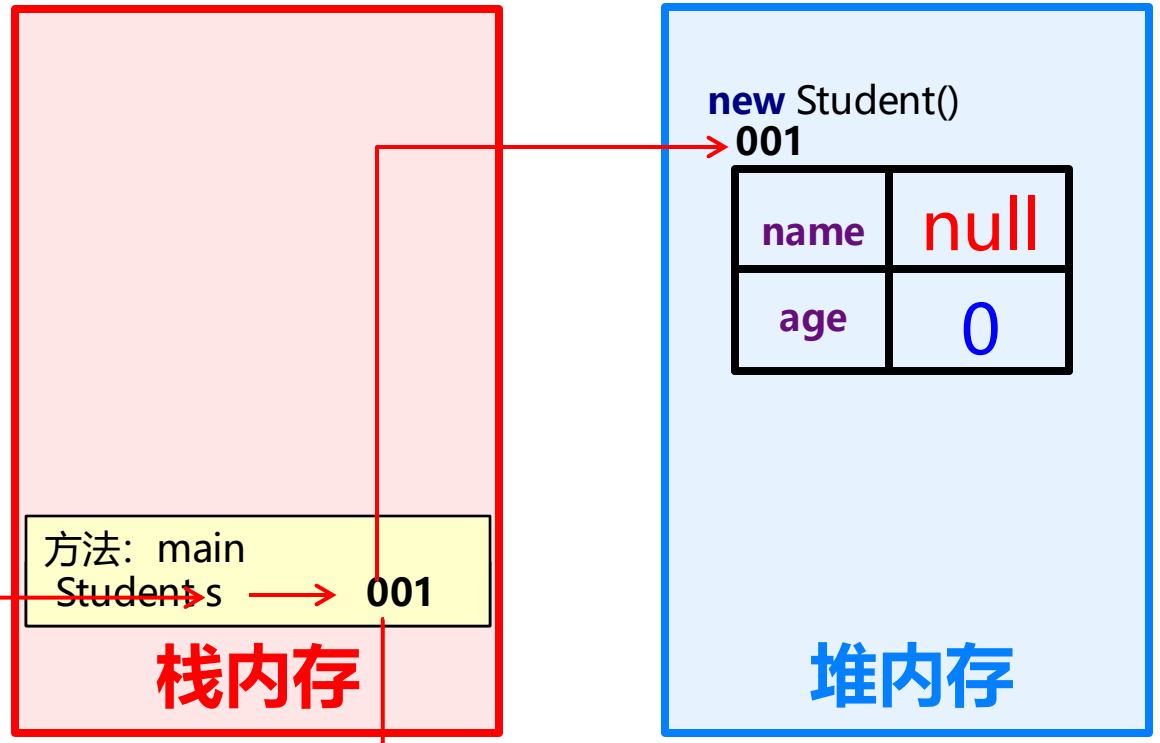



**成员变量都具有默认值**

## 单个对象

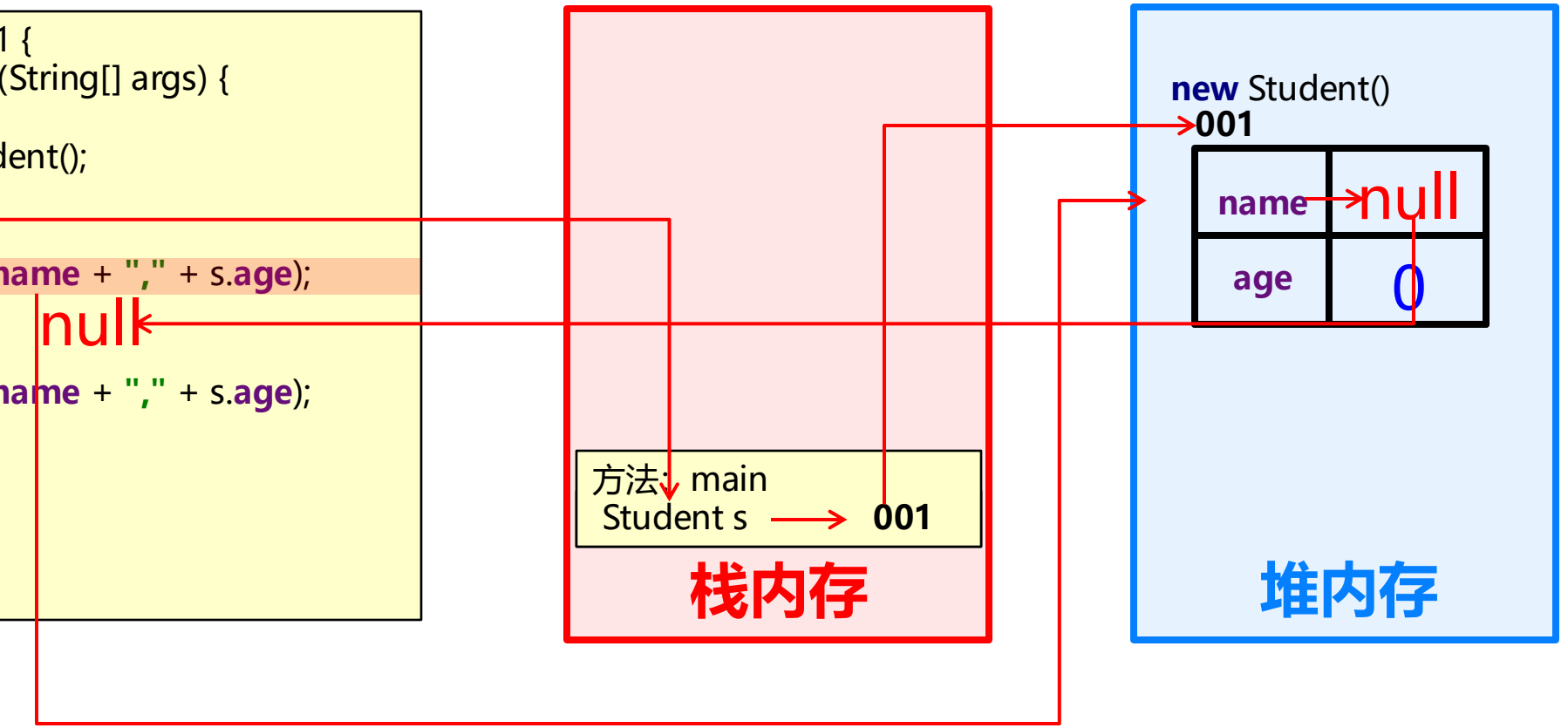
```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
        //使用成员变量
        System.out.println(s.name + "," + s.age);
        s.name = "张曼玉";
        s.age = 28;
        System.out.println(s.name + "," + s.age);
        //使用成员方法
        s.study();
        s.doHomework();
    }
}
```




输出：**001**

## 单个对象

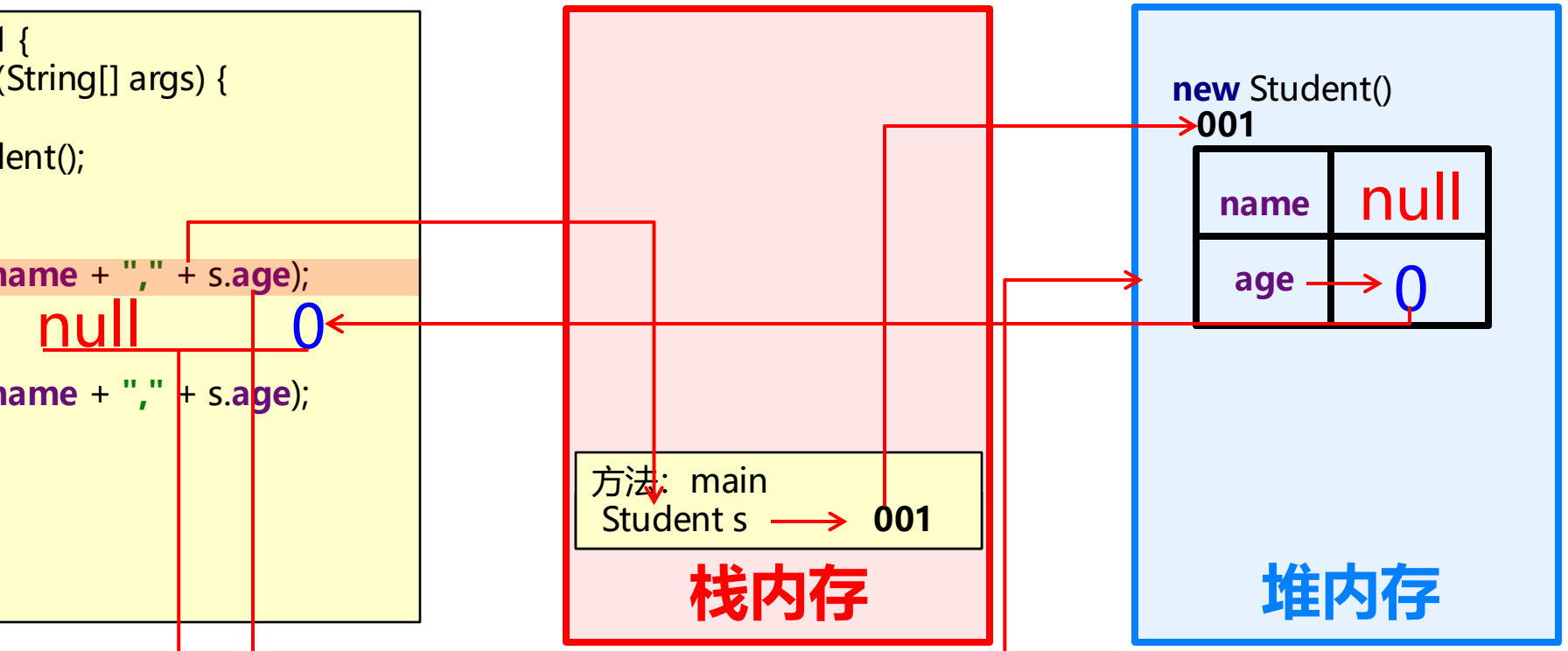
```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
        //使用成员变量
        System.out.println(s.name + "," + s.age);
        s.name = "张曼玉";
        s.age = 28;
        System.out.println(s.name + "," + s.age);
        //使用成员方法
        s.study();
        s.doHomework();
    }
}
```




输出：**001**

## 单个对象

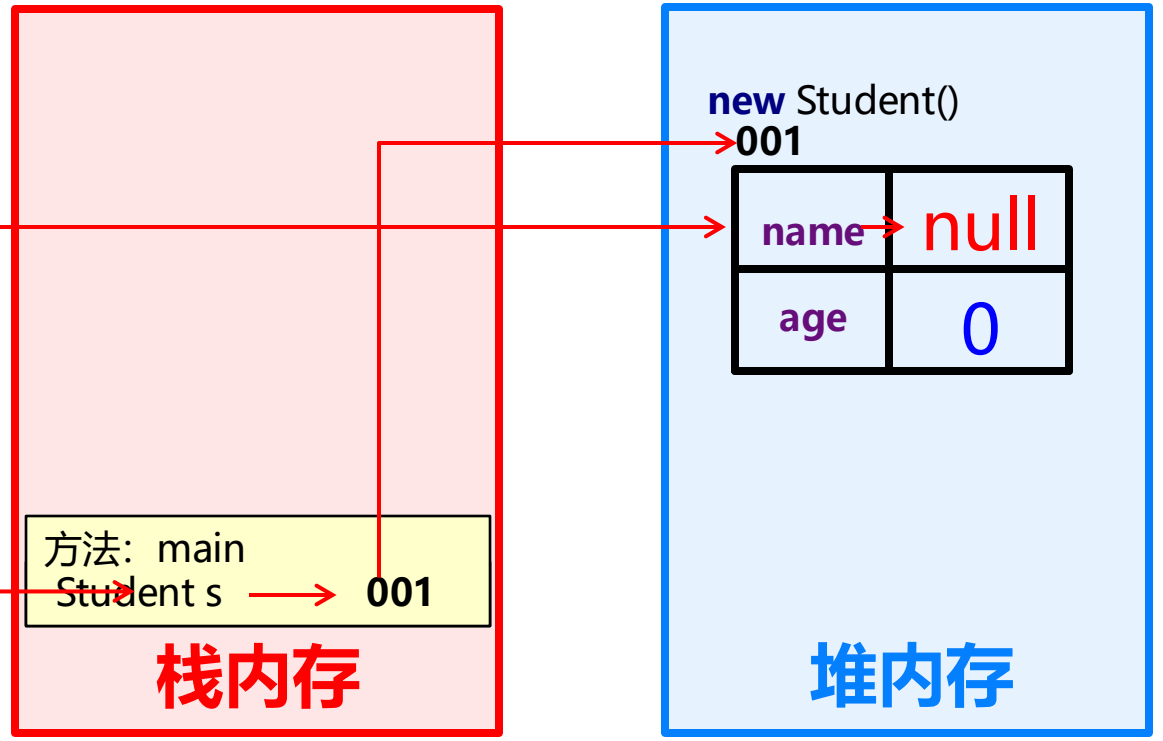
```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
        //使用成员变量
        System.out.println(s.name + "," + s.age);
        s.name = "张曼玉";
        s.age = 28;
        System.out.println(s.name + "," + s.age);
        //使用成员方法
        s.study();
        s.doHomework();
    }
}
```




输出：001
输出：null，0

## 单个对象

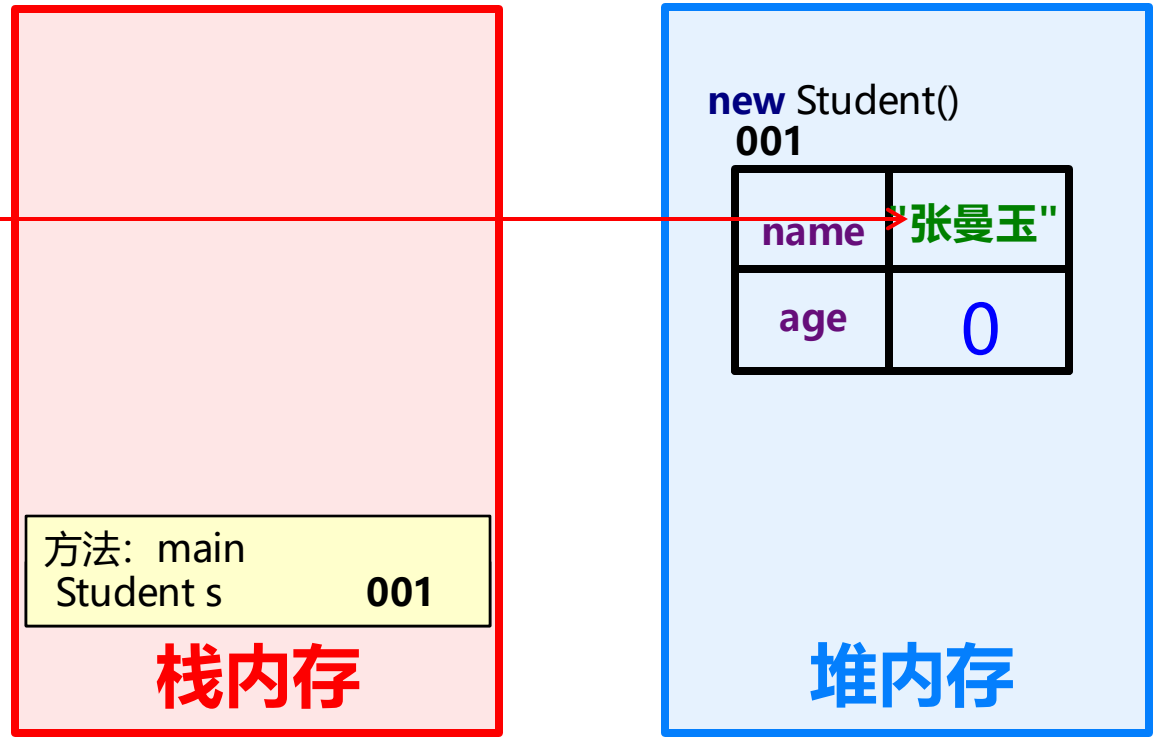
```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
        //使用成员变量
        System.out.println(s.name + "," + s.age);
        s.name = "张曼玉";
        s.age = 28;
        System.out.println(s.name + "," + s.age);
        //使用成员方法
        s.study();
        s.doHomework();
    }
}
```




输出：**001**
输出：**null**，0

## 单个对象

```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
        //使用成员变量
        System.out.println(s.name + "," + s.age);
        s.name = "张曼玉";
        s.age = 28;
        System.out.println(s.name + "," + s.age);
        //使用成员方法
        s.study();
        s.doHomework();
    }
}
```

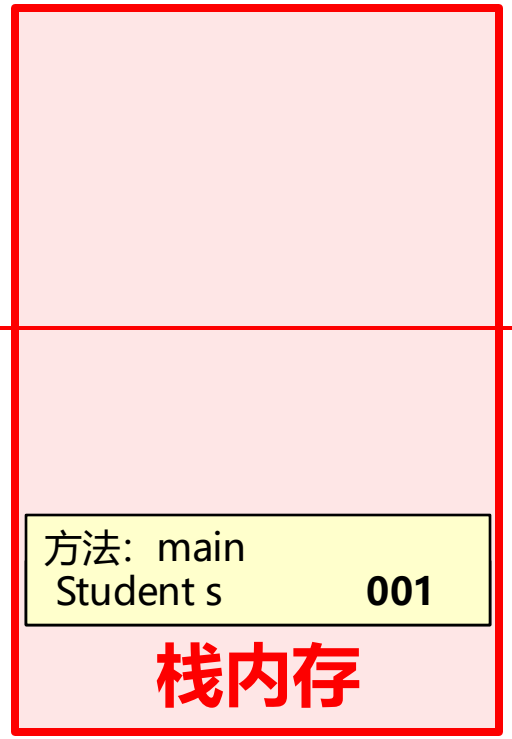


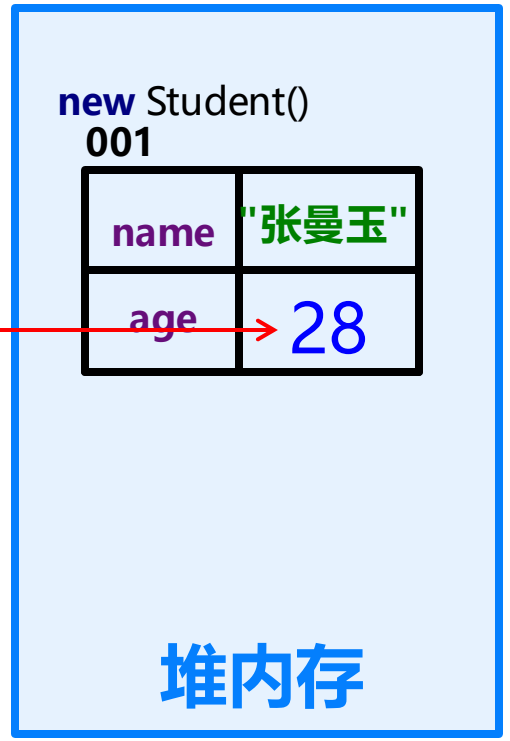

输出：**001**
输出：**null**，0

## 单个对象

```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
        //使用成员变量
        System.out.println(s.name + "," + s.age);
        s.name = "张曼玉";
        s.age = 28;
        System.out.println(s.name + "," + s.age);
        //使用成员方法
        s.study();
        s.doHomework();
    }
}
```

"张曼玉" 28




输出：**001**
输出：**null**，0
输出：**张曼玉**，28

## 单个对象

```
public class Student {
    //成员变量
    String name;
    int age;
    //成员方法
    public void study() {
        System.out.println("好好学习");
    }
    public void doHomework() {
        System.out.println("多做练习");
    }
}
```

```
pub
    pu
        System.out.println(s.name + "," + s.age);
        //使用成员方法
        s.study();
        s.doHomework();
    }
}
```

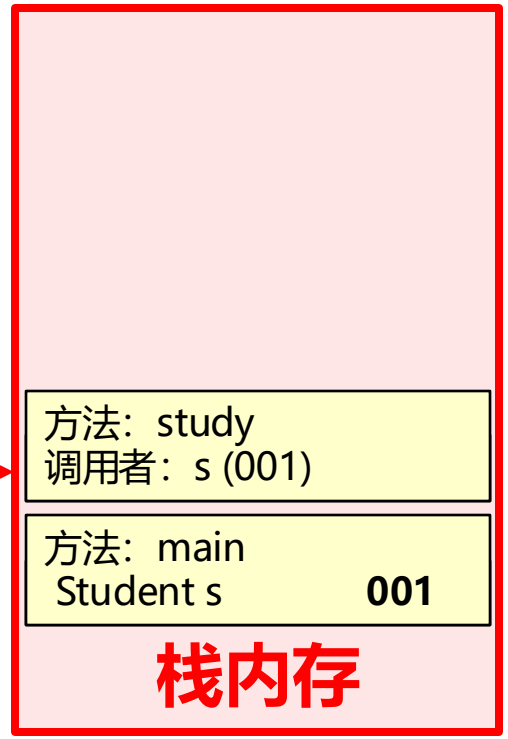


输出：**001**
输出：**null**，0
输出：**张曼玉**，28

## 单个对象

```
public class Student {
    //成员变量
    String name;
    int age;
    //成员方法
    public void study() {
        System.out.println("好好学习");
    }
    public void doHomework() {
        System.out.println("多做练习");
    }
}
```

```
pub
    pu
        System.out.println(s.name + "," + s.age);
        //使用成员方法
        s.study();
        s.doHomework();
    }
}
```





输出：001　　输出：好好学习
输出：null，0
输出：张曼玉，28

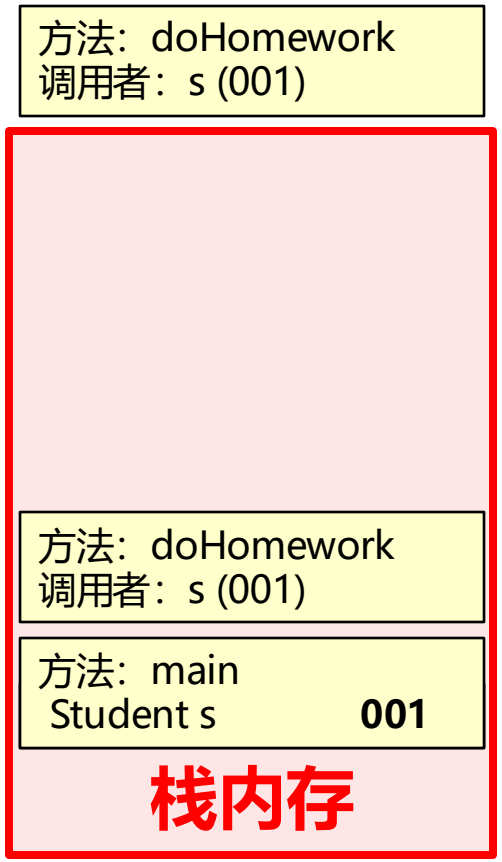
## 单个对象

```
public class Student {
    //成员变量
    String name;
    int age;
    //成员方法
    public void study() {
        System.out.println("好好学习");
    }
    public void doHomework() {
        System.out.println("多做练习");
    }
}
```

```
pub
    pu
        System.out.println(s.name + "," + s.age);
        //使用成员方法
        s.study();
        s.doHomework();
    }
}
```

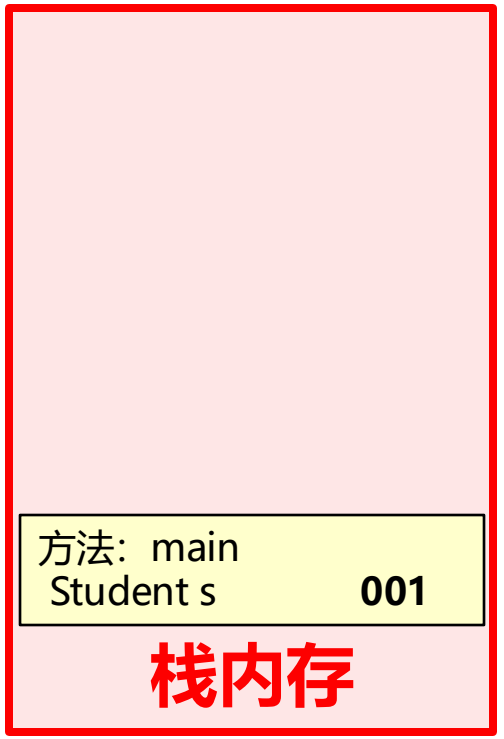
输出：001
输出：null，0
输出：张曼玉，28

输出：好好学习
输出：多做练习




## 单个对象

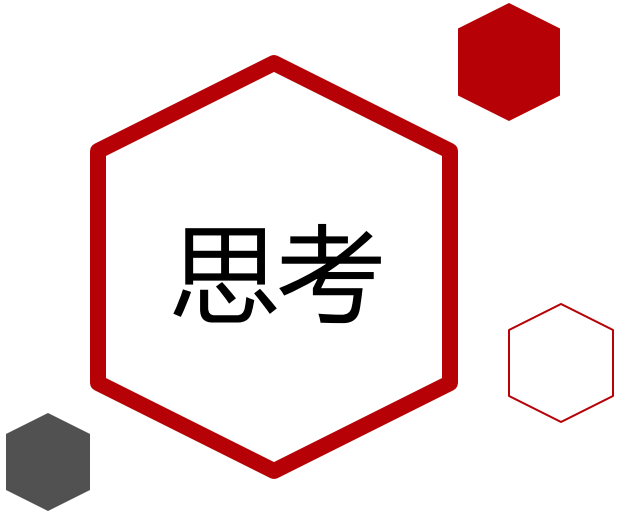
```
public class StudentTest01 {
    public static void main(String[] args) {
        //创建对象
        Student s = new Student();
        System.out.println(s);
        //使用成员变量
        System.out.println(s.name + "," + s.age);
        s.name = "张曼玉";
        s.age = 28;
        System.out.println(s.name + "," + s.age);
        //使用成员方法
        s.study();
        s.doHomework();
    }
}
```





输出：**001**
输出：**null**，0
输出：**张曼玉**，28

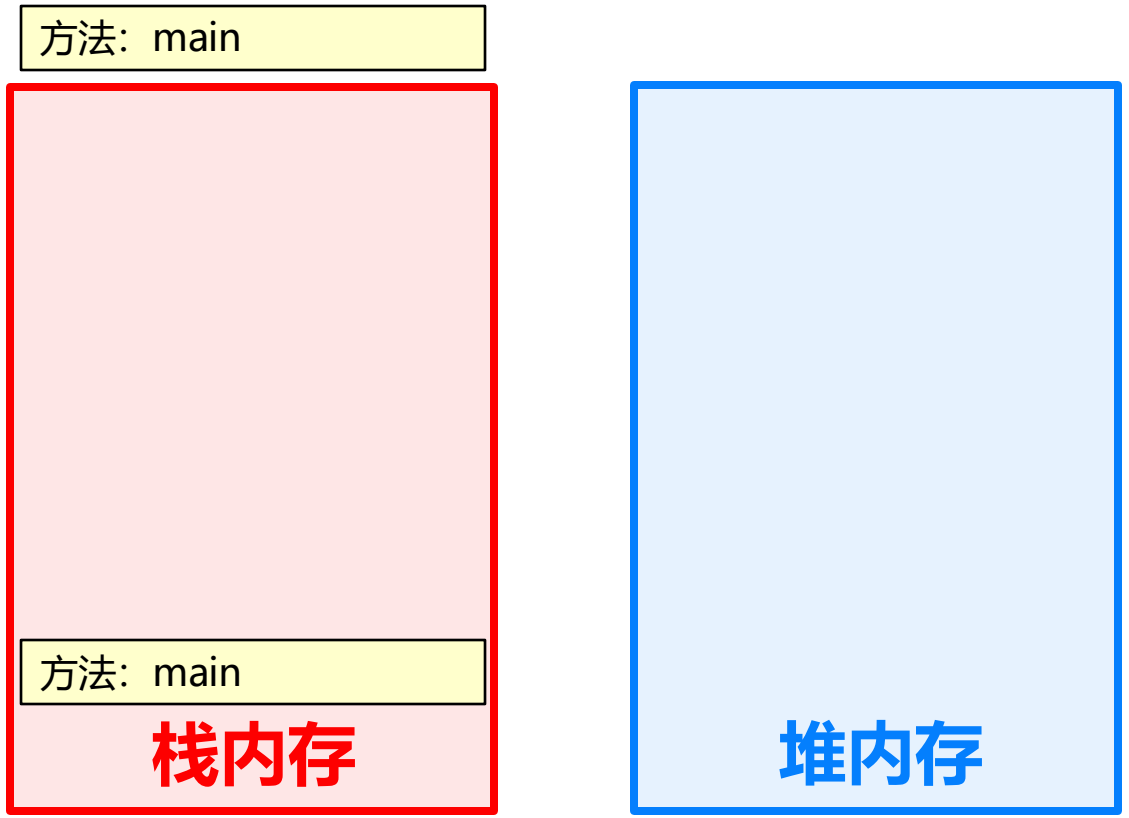
输出：**好好学习**
输出：**多做练习**




多个对象共用同一个方法内存吗

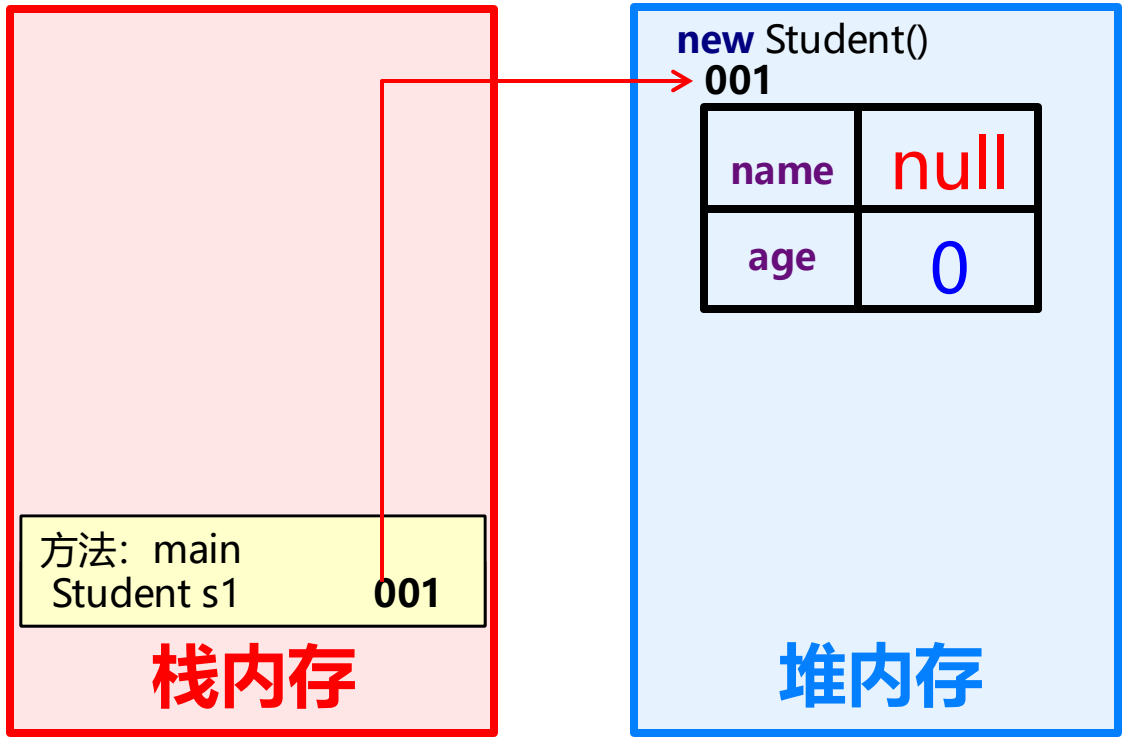
## 多个对象

```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```



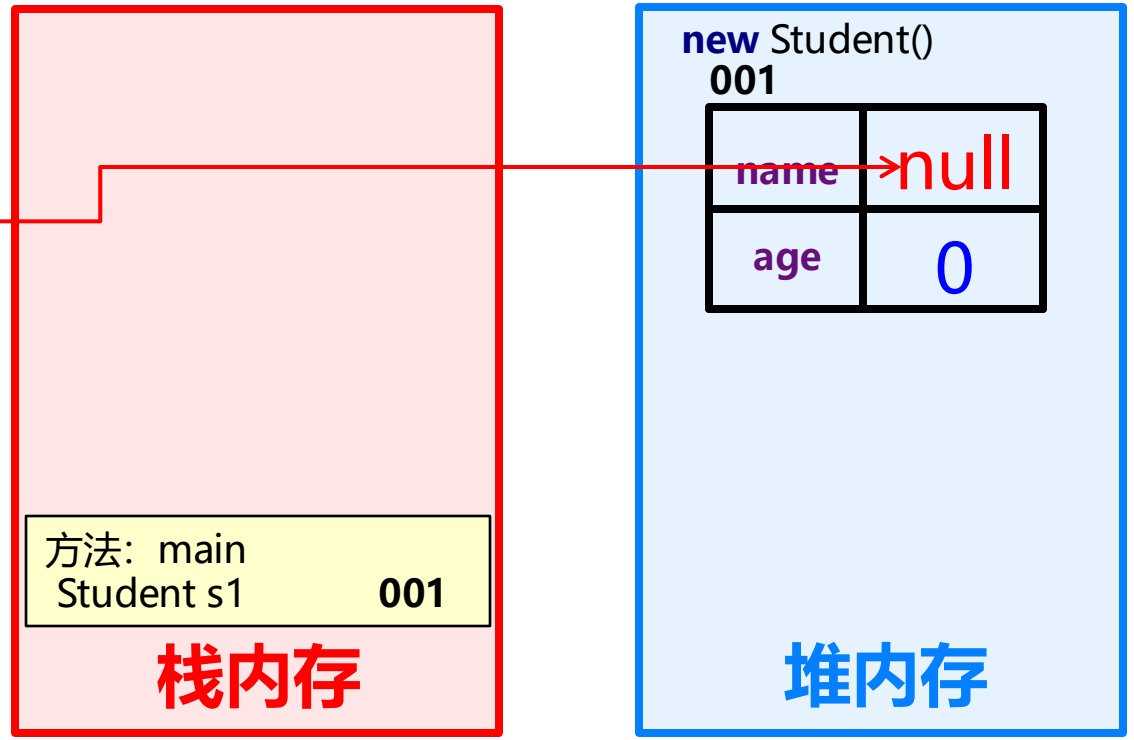
## 多个对象

```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```



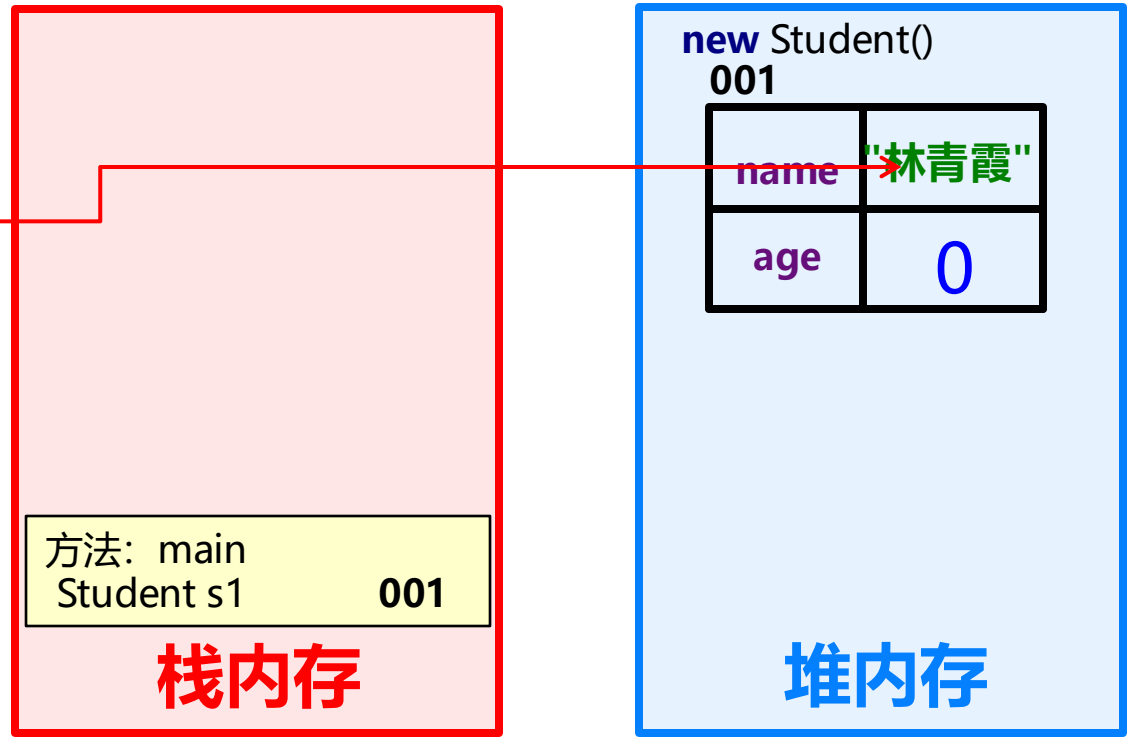
## 多个对象

```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```



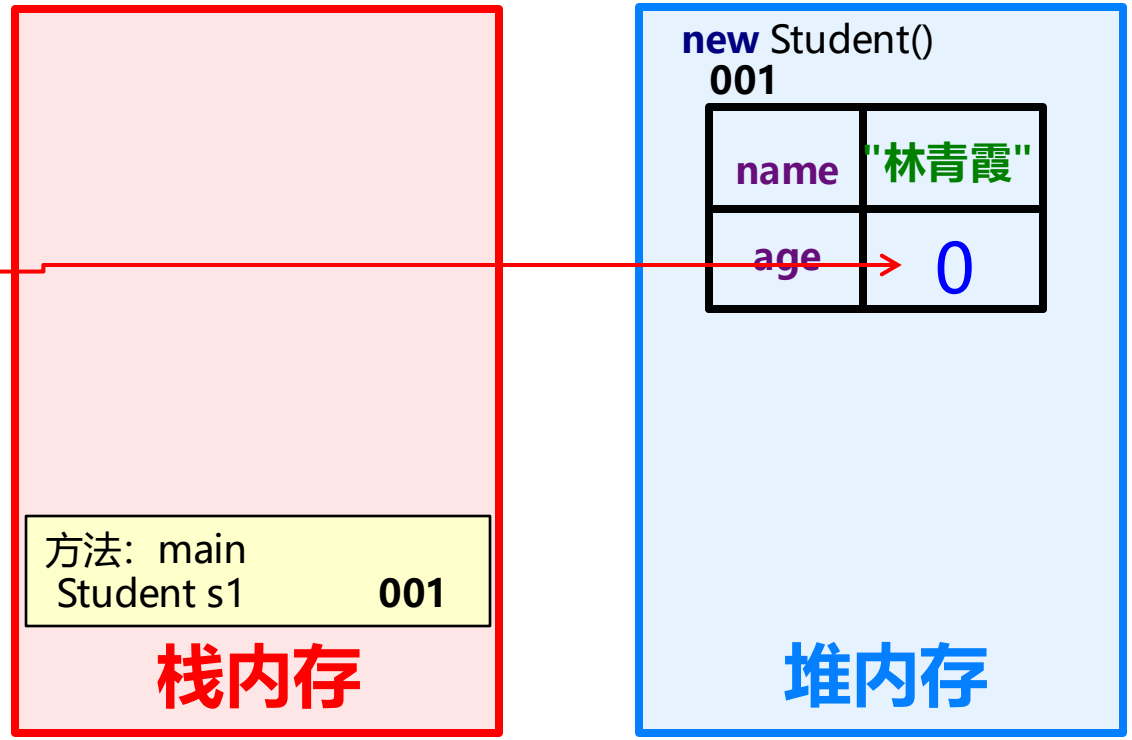
## 多个对象

```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```



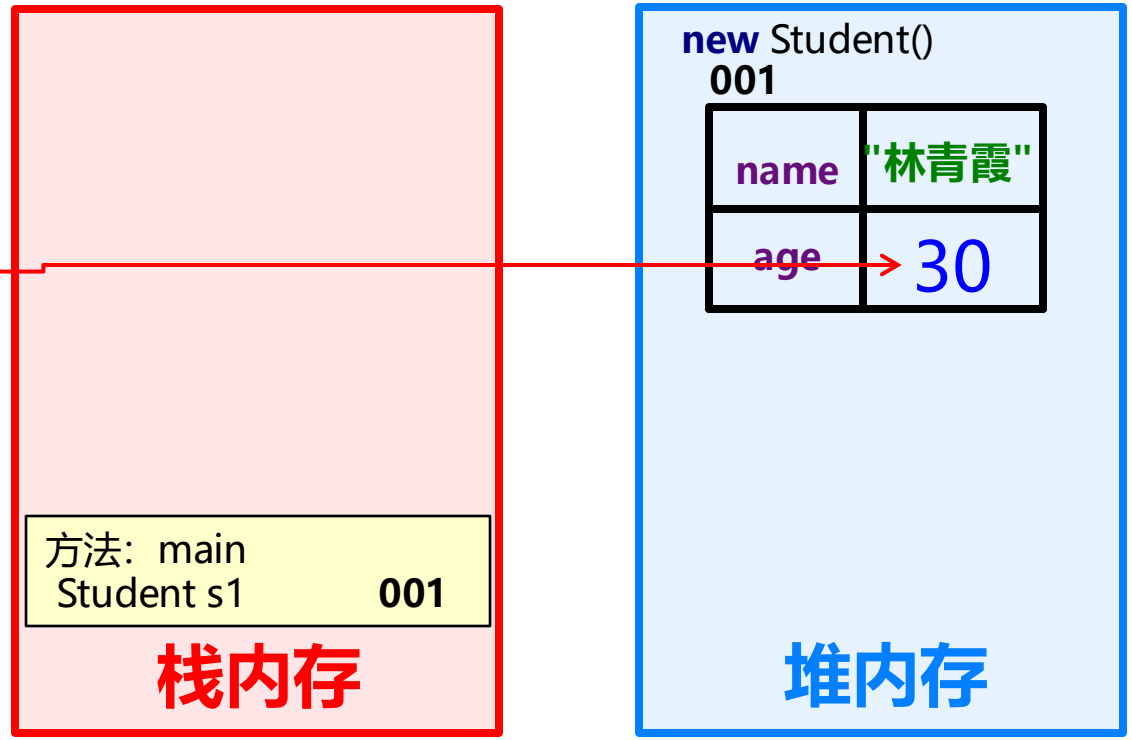
## 多个对象

```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```



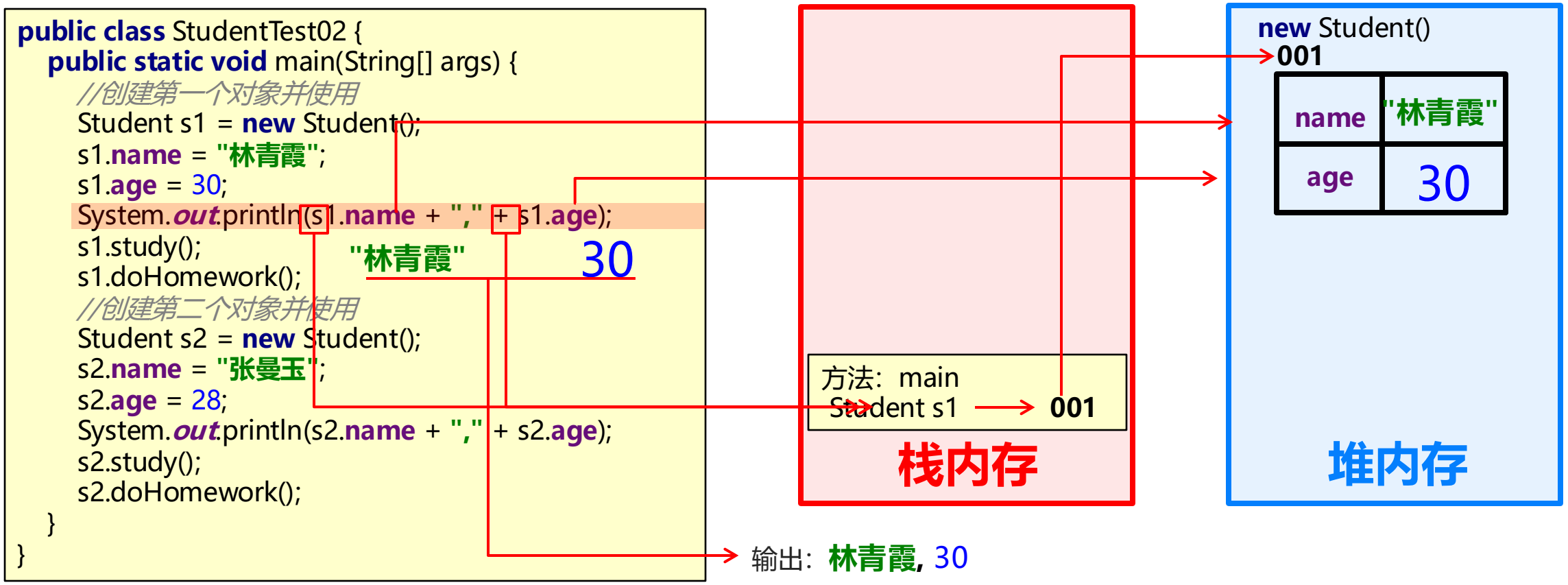
## 多个对象

```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```



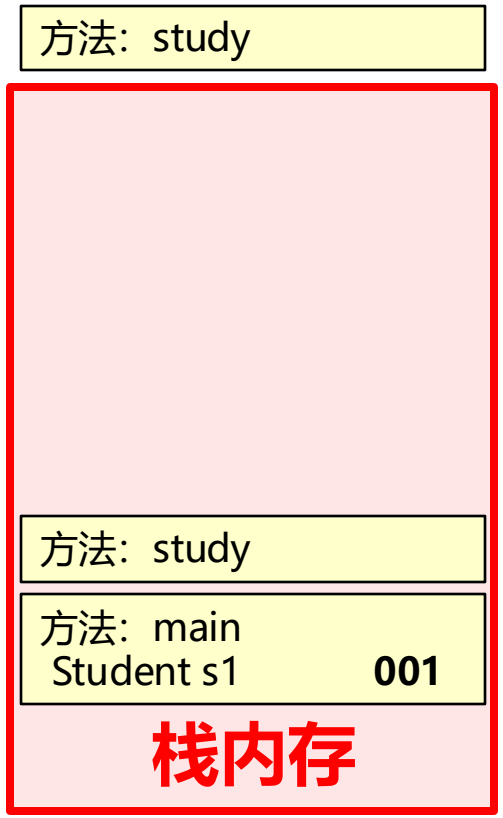
## 多个对象

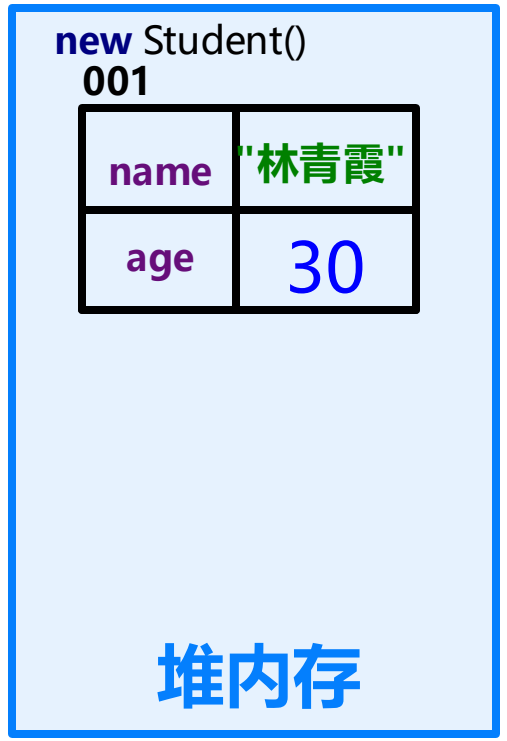
```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```



```java
public class Student {
    //成员变量
    String name;
    int age;
    //成员方法
    public void study() {
        System.out.println("好好学习");
    }
    public void doHomework() {
        System.out.println("多做练习");
    }
}
```

```java
pub
    pu
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```

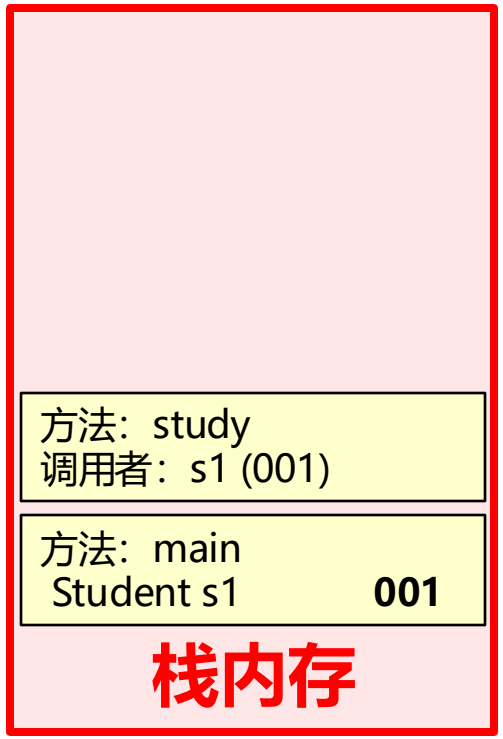


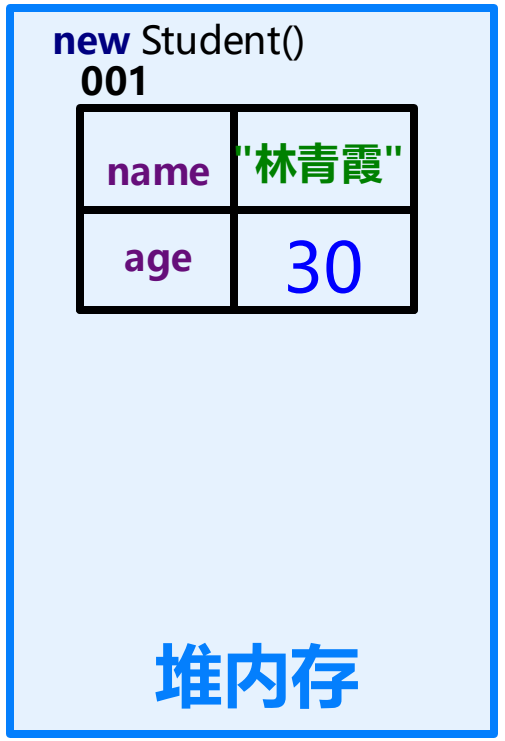




输出：**林青霞**, 30

```java
public class Student {
    //成员变量
    String name;
    int age;
    //成员方法
    public void study() {
        System.out.println("好好学习");
    }
    public void doHomework() {
        System.out.println("多做练习");
    }
}
```

```java
pub
    pu
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```

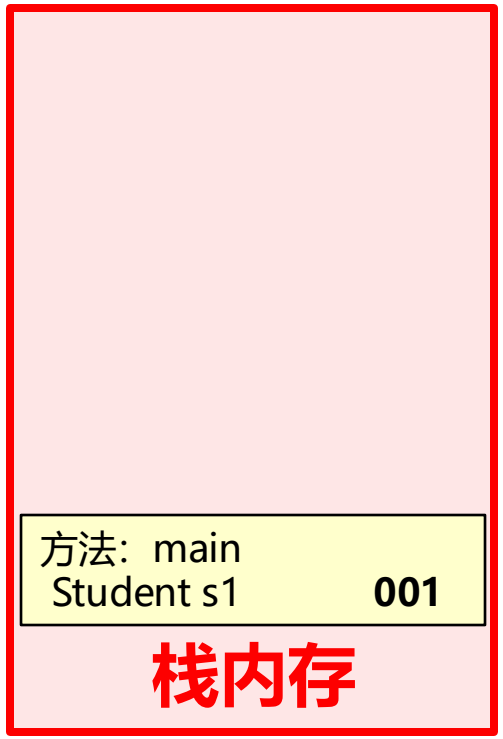


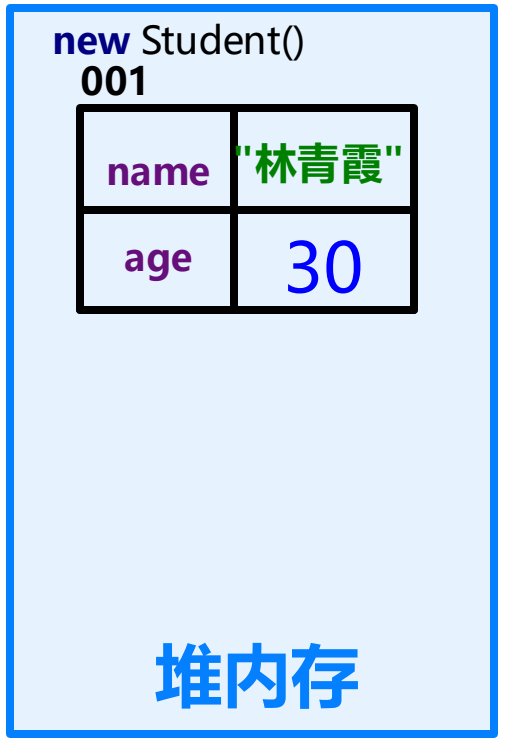




输出：林青霞, 30
输出：好好学习

## 多个对象

```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```

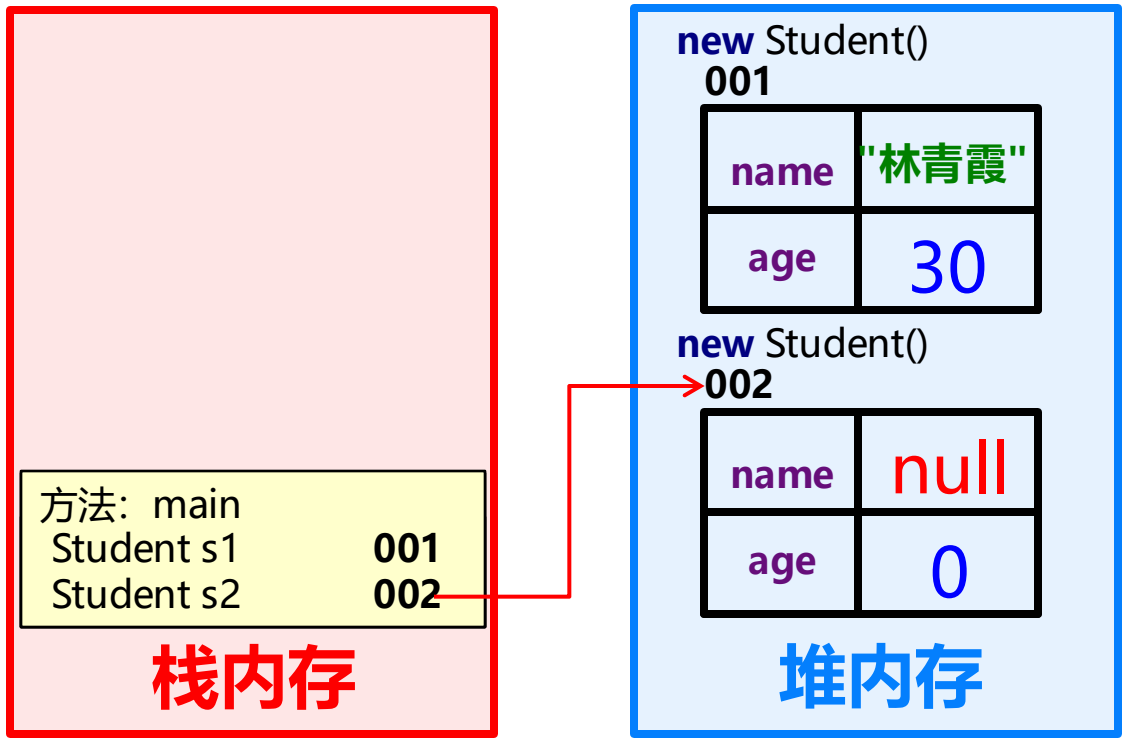



输出：林青霞，30
输出：好好学习
输出：多做练习

## 多个对象

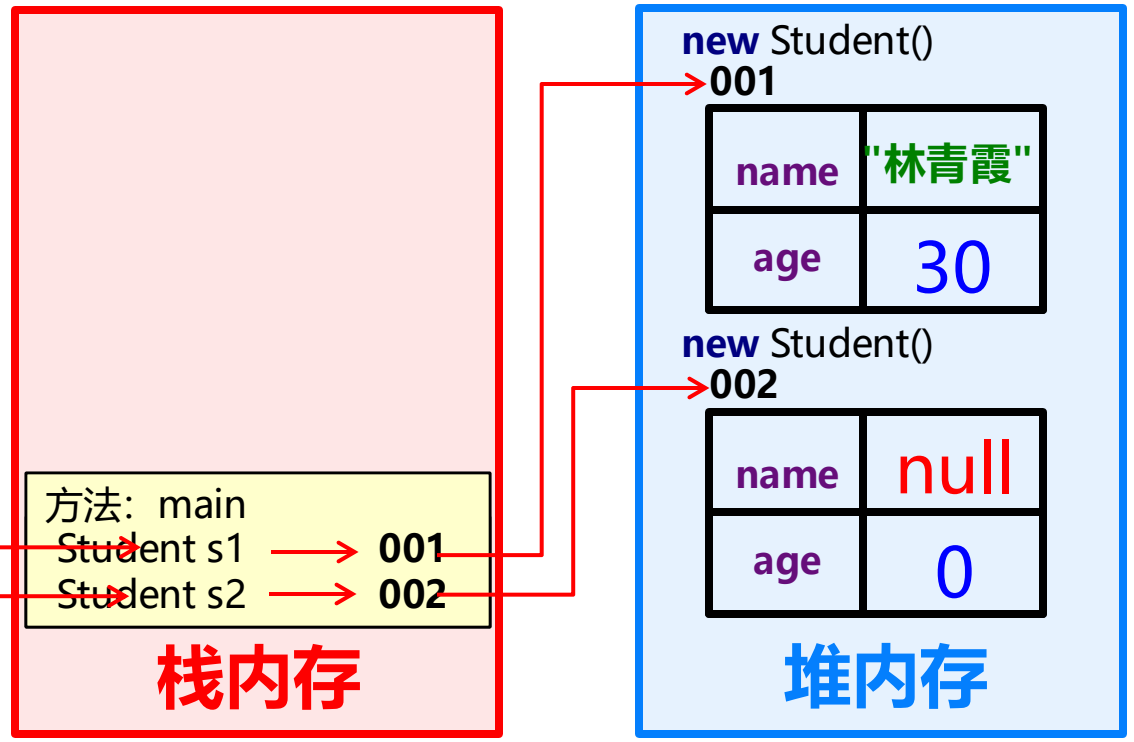
```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```




输出：林青霞, 30
输出：好好学习
输出：多做练习

## 多个对象

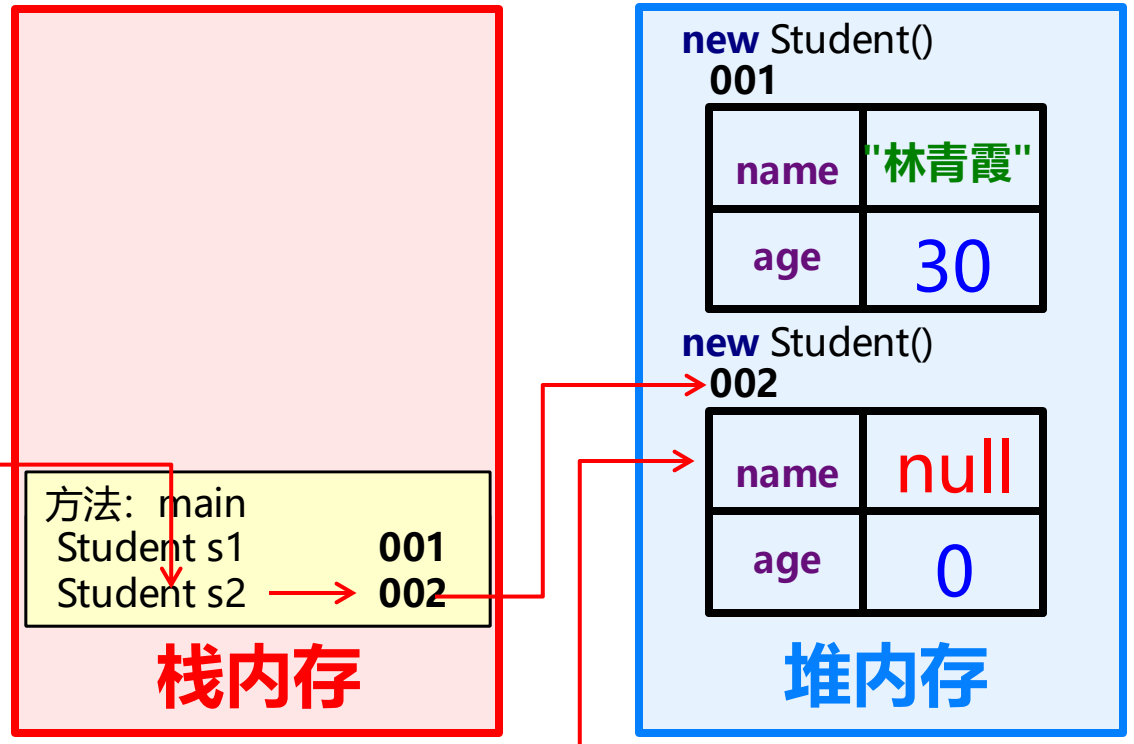
```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```




输出：林青霞，30
输出：好好学习
输出：多做练习

## 多个对象

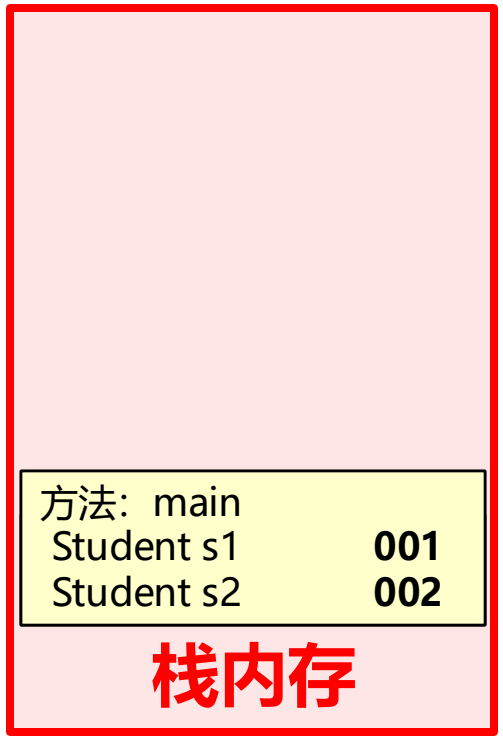
```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```

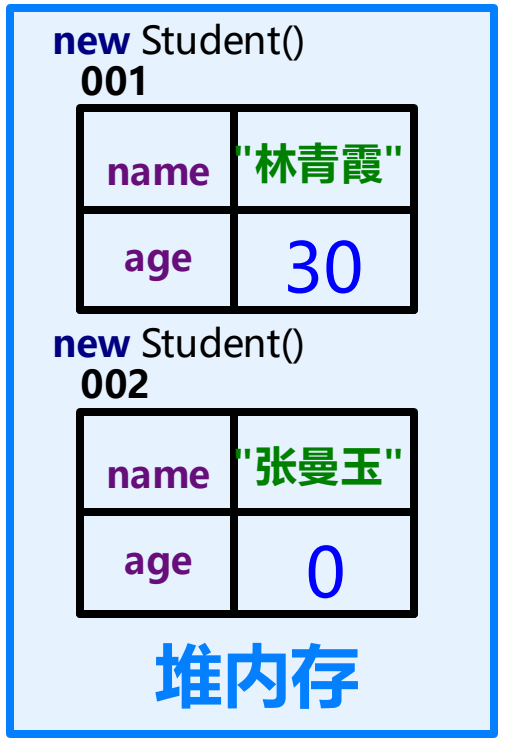



输出：林青霞，30
输出：好好学习
输出：多做练习

## 多个对象

```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```

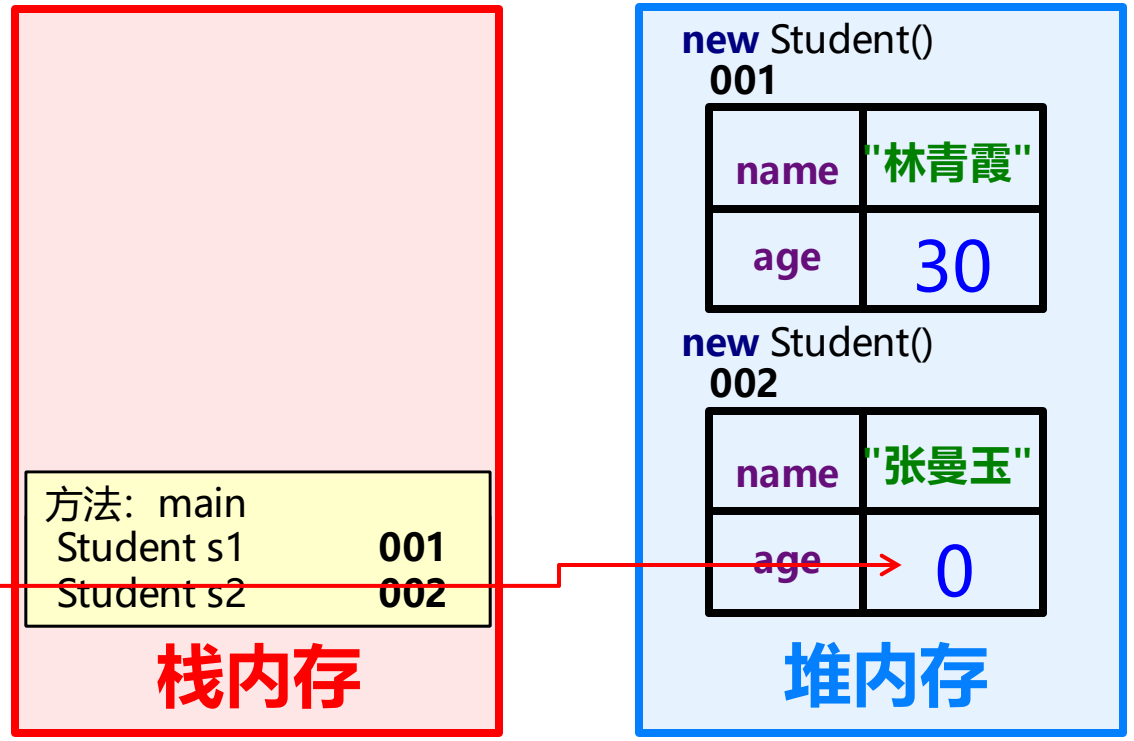



输出：林青霞, 30
输出：好好学习
输出：多做练习

## 多个对象

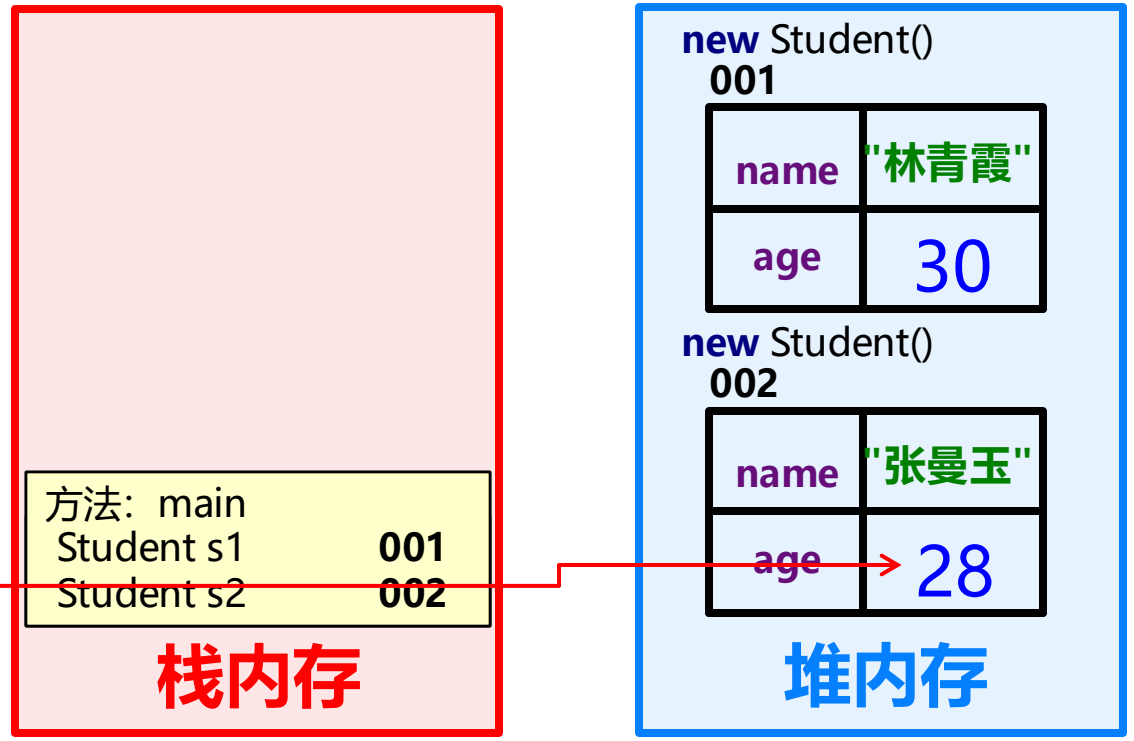
```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```




输出：林青霞，30
输出：好好学习
输出：多做练习

## 多个对象

```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```

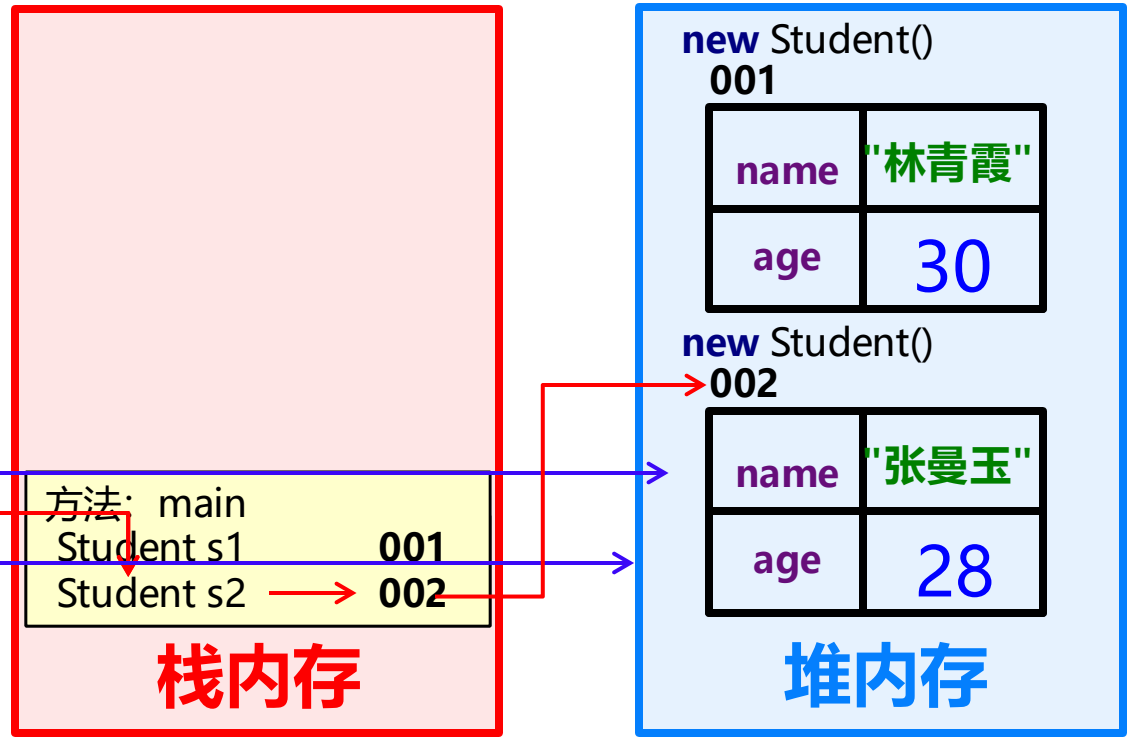



输出：林青霞, 30
输出：好好学习
输出：多做练习

## 多个对象

```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```

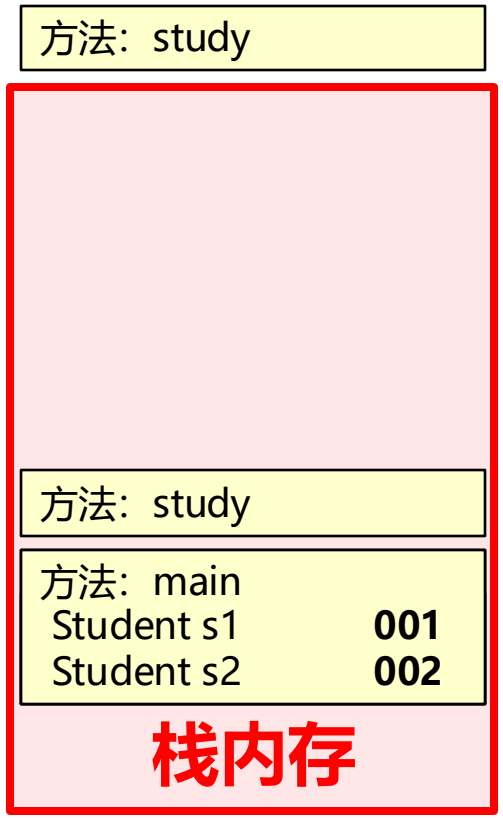
"张曼玉" 28

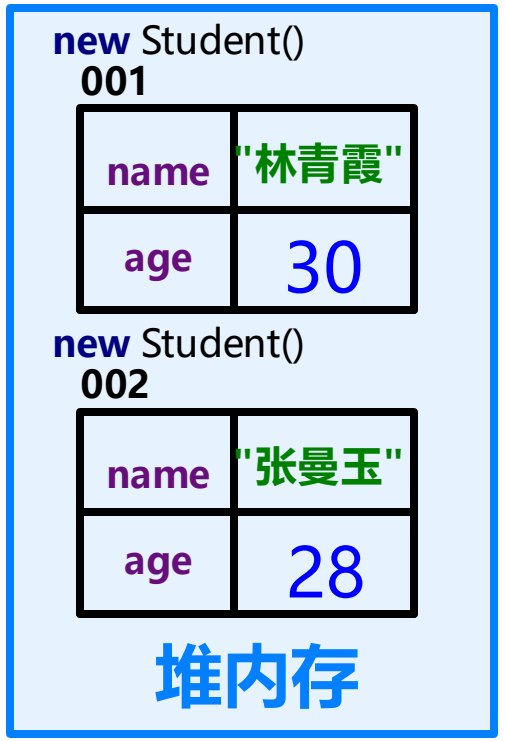



输出：林青霞, 30 → 输出：张曼玉, 28
输出：好好学习
输出：多做练习

```java
public class Student {
    //成员变量
    String name;
    int age;
    //成员方法
    public void study() {
        System.out.println("好好学习");
    }
    public void doHomework() {
        System.out.println("多做练习");
    }
}
```

```java
pub
    pu
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```



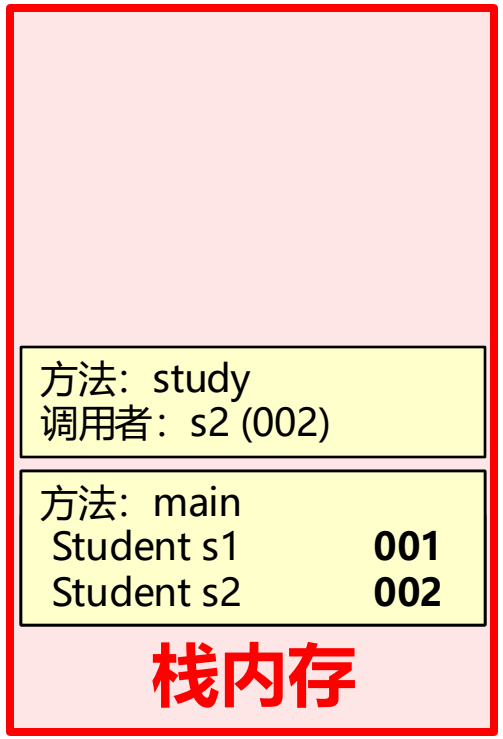



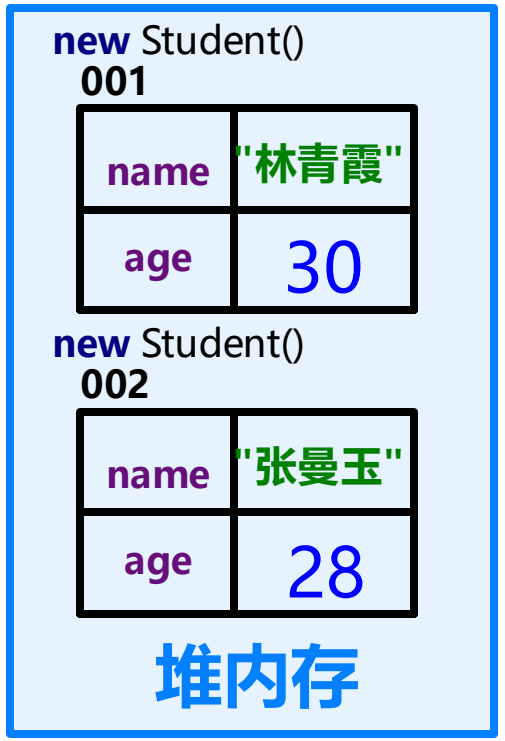

输出：林青霞, 30
输出：好好学习
输出：多做练习

输出：张曼玉, 28

```java
public class Student {
    //成员变量
    String name;
    int age;
    //成员方法
    public void study() {
        System.out.println("好好学习");
    }
    public void doHomework() {
        System.out.println("多做练习");
    }
}
```

```java
pub
    pu
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```



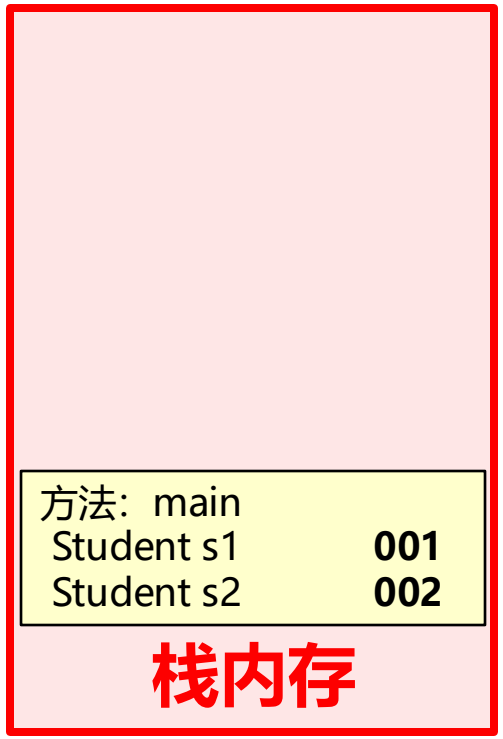



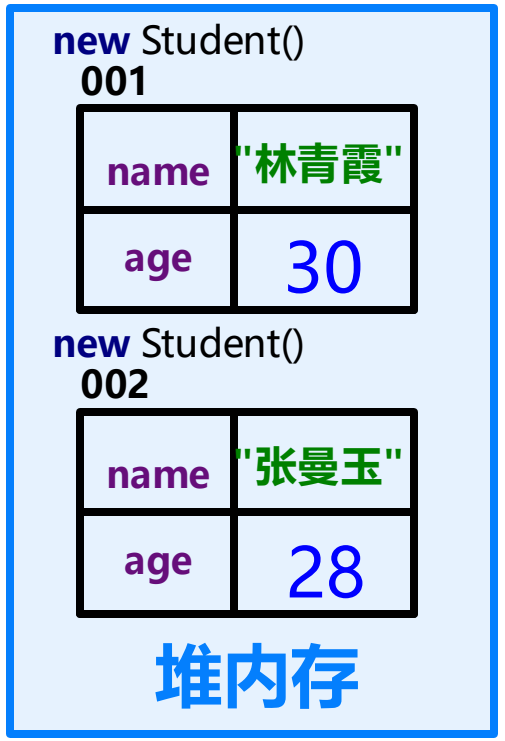

输出：林青霞, 30
输出：好好学习
输出：多做练习

输出：张曼玉, 28
输出：好好学习

## 多个对象

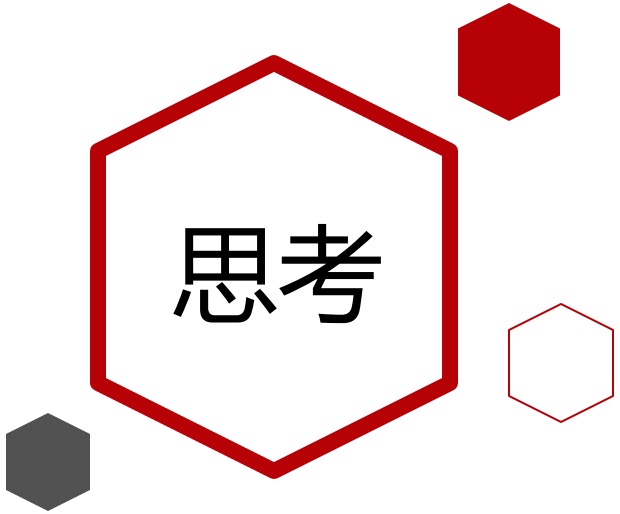
```
public class StudentTest02 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        s1.study();
        s1.doHomework();
        //创建第二个对象并使用
        Student s2 = new Student();
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s2.name + "," + s2.age);
        s2.study();
        s2.doHomework();
    }
}
```




输出：林青霞, 30
输出：好好学习
输出：多做练习

输出：张曼玉, 28
输出：好好学习
输出：多做练习

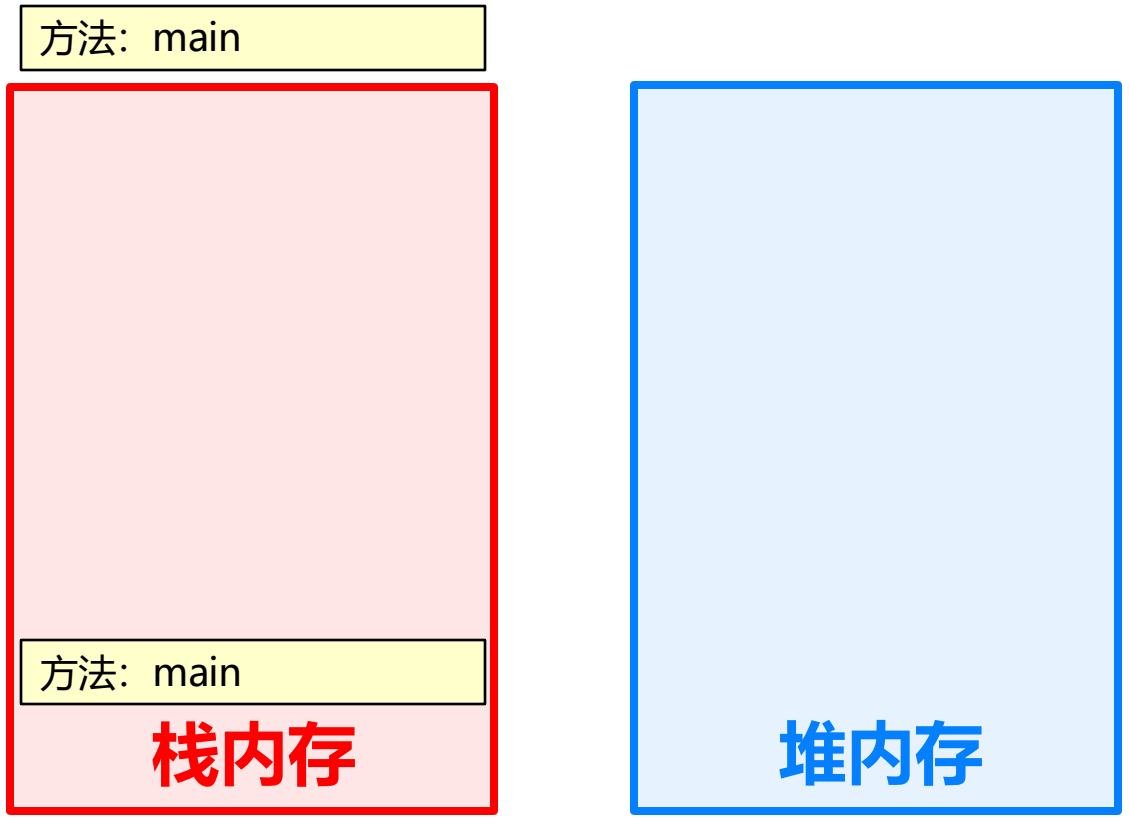



多个引用可以指向同一个对象吗？

如果可以，通过其中一个引用改变数据了，其他引用访问的数据改变了吗？

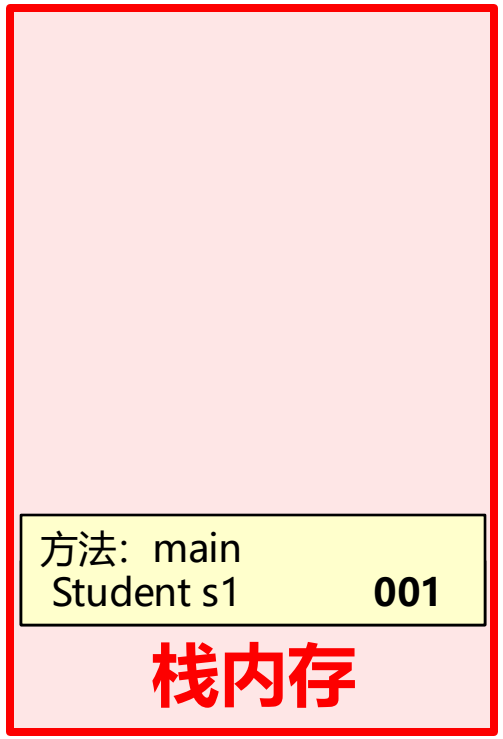
## 多个引用指向相同

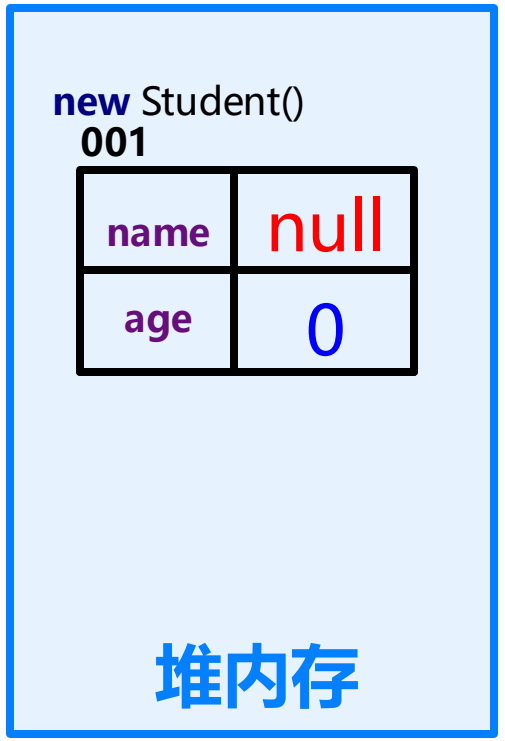
```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```



## 多个引用指向相同

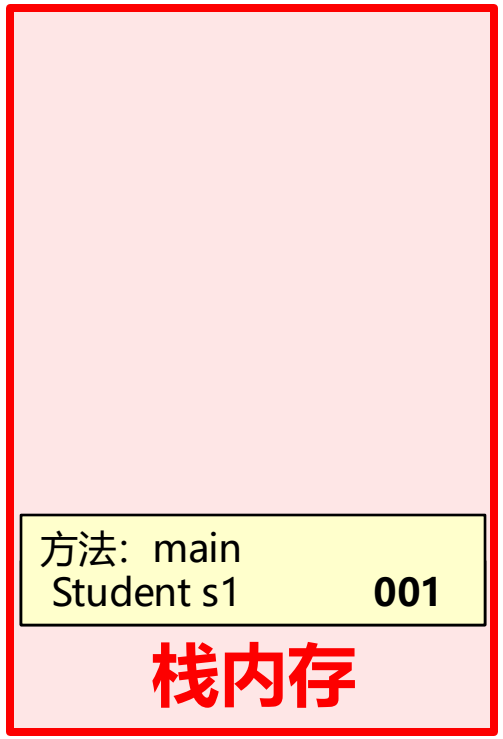
```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```

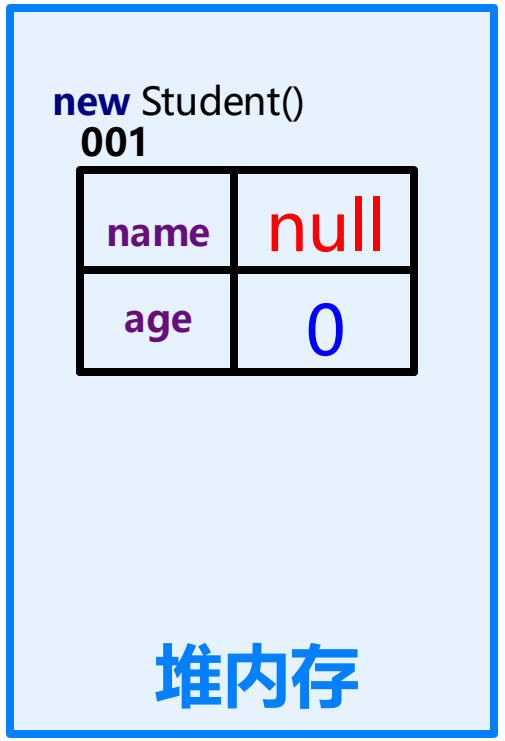


## 多个引用指向相同

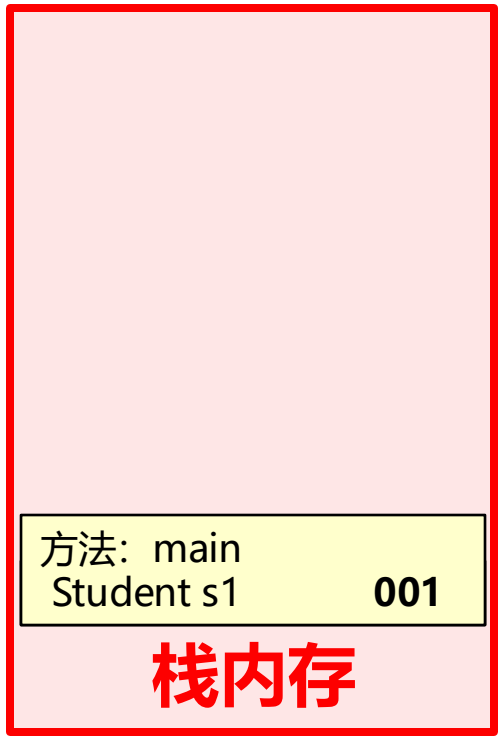
```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```



## 多个引用指向相同

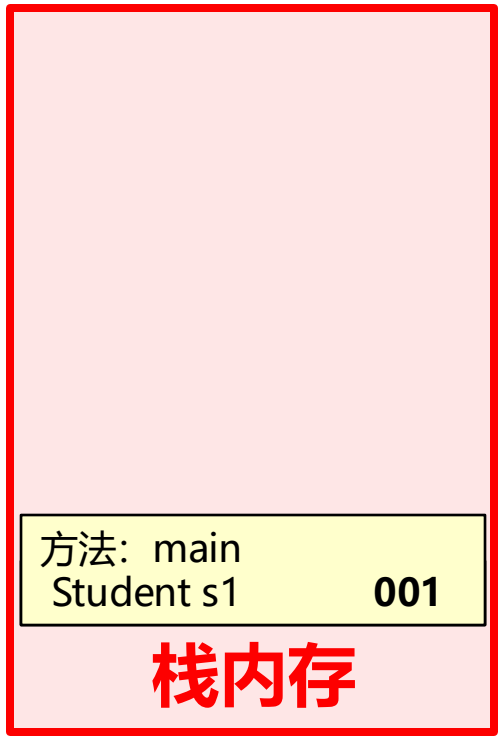
```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```




## 多个引用指向相同

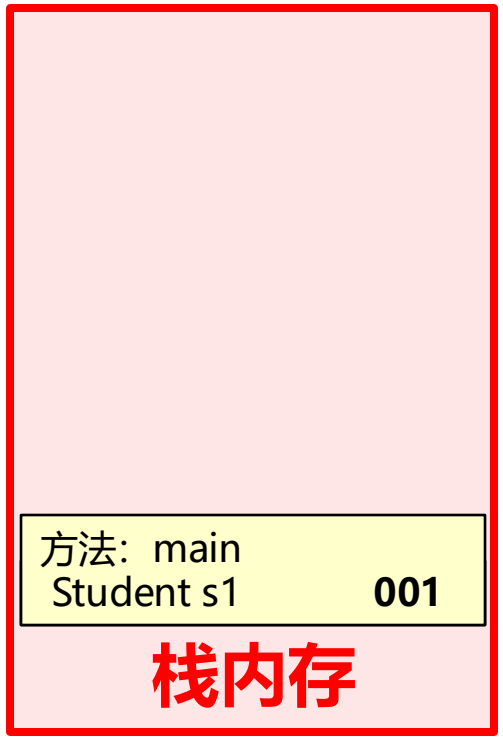
```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```

输出：林青霞，30




## 多个引用指向相同

```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```

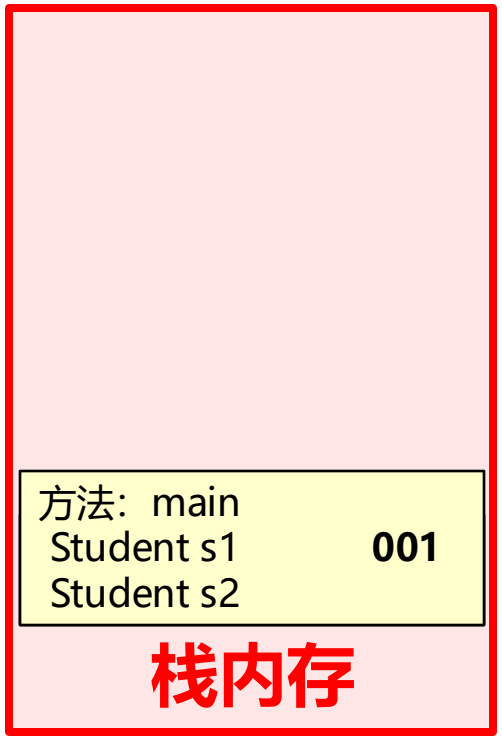




输出：林青霞，30

## 多个引用指向相同

```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```

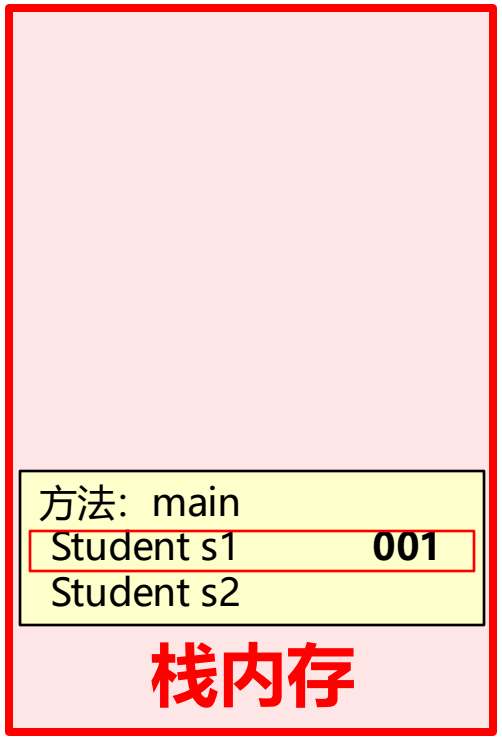




输出：林青霞，30

## 多个引用指向相同

```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```

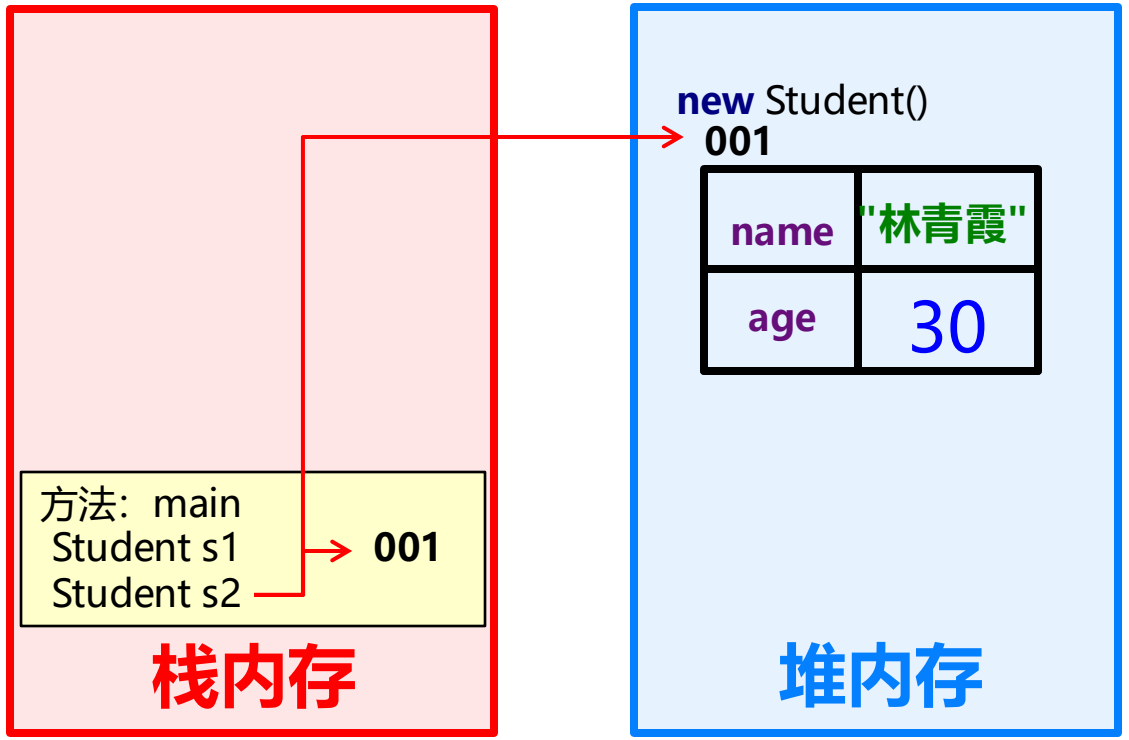




输出：林青霞，30

## 多个引用指向相同

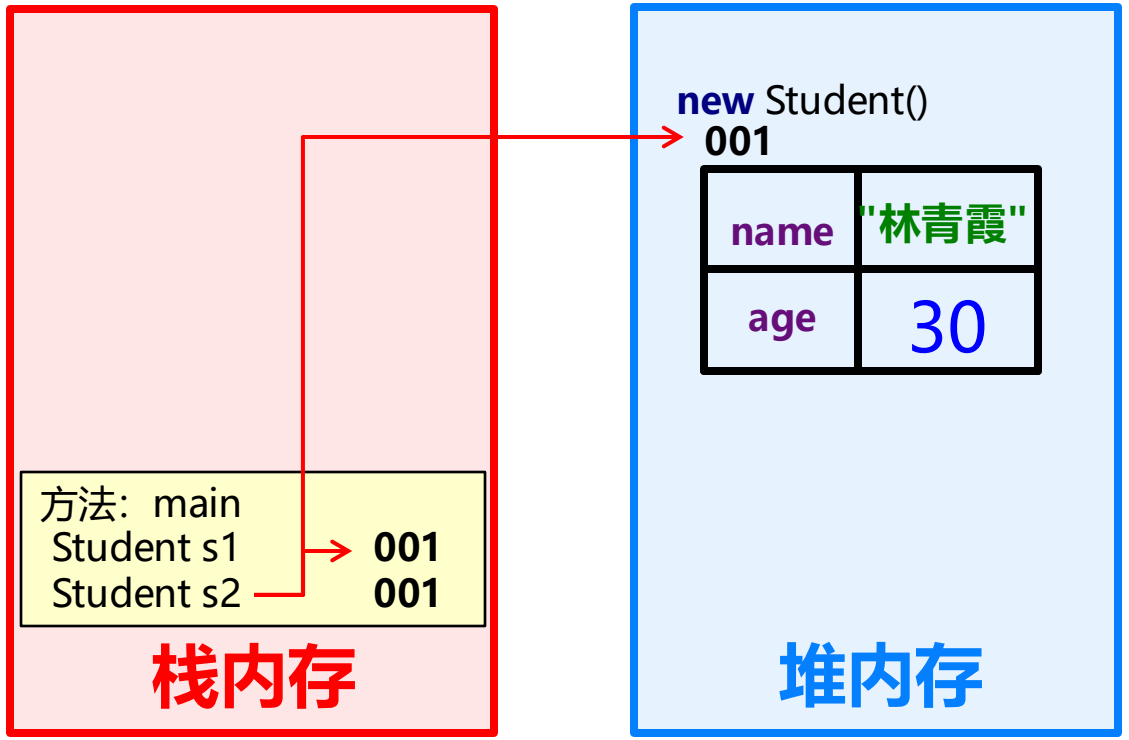
```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```




输出：林青霞，30

## 多个引用指向相同

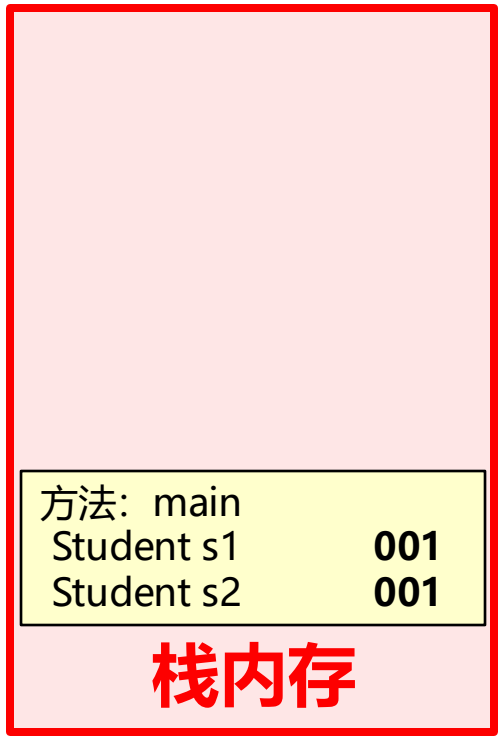
```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```




输出：林青霞，30

## 多个引用指向相同

```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```

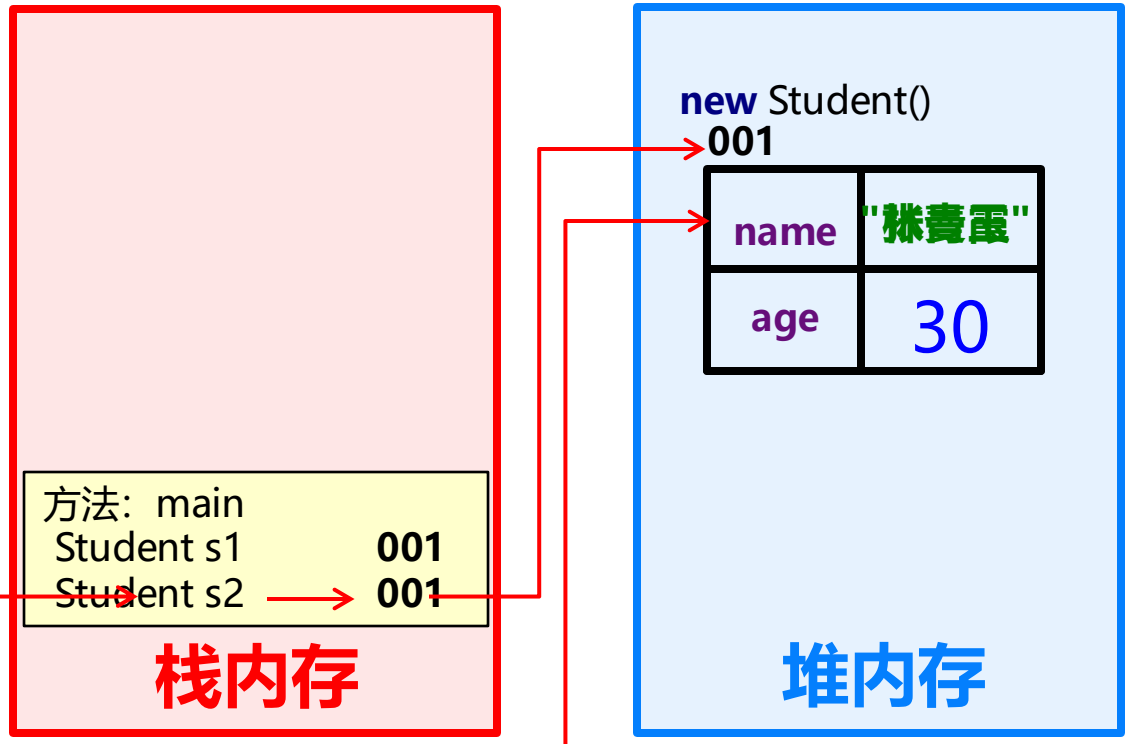




输出：林青霞，30

## 多个引用指向相同

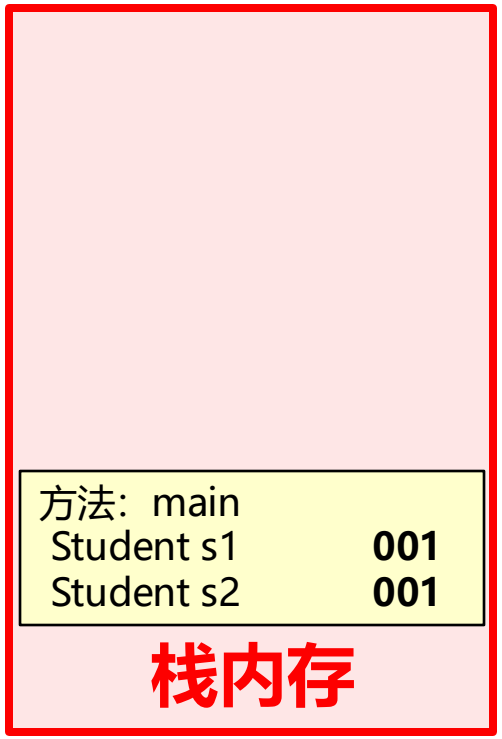
```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```

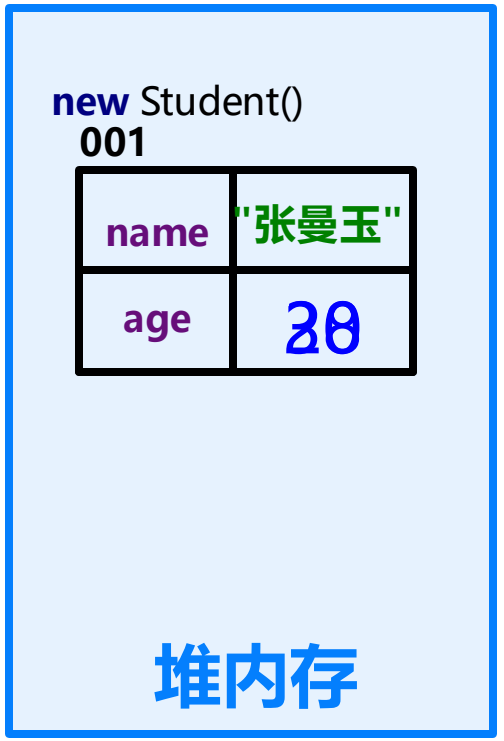



输出：林青霞，30

## 多个引用指向相同

```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```

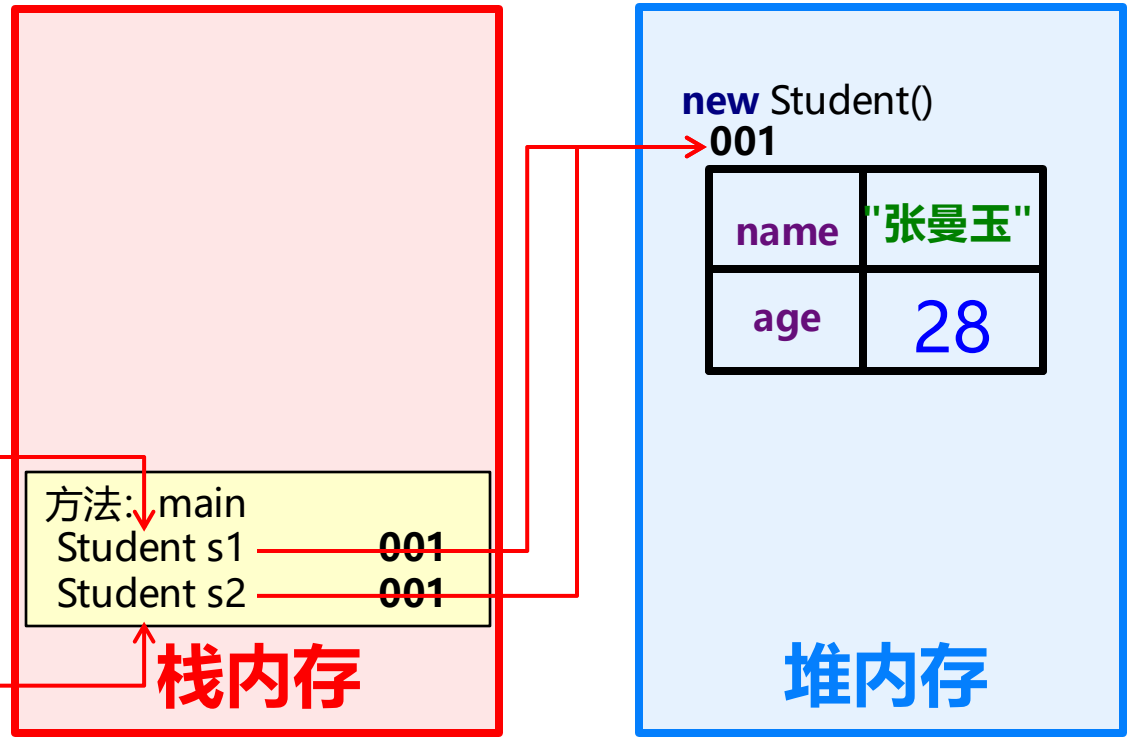



输出：林青霞，30

## 多个引用指向相同

```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```

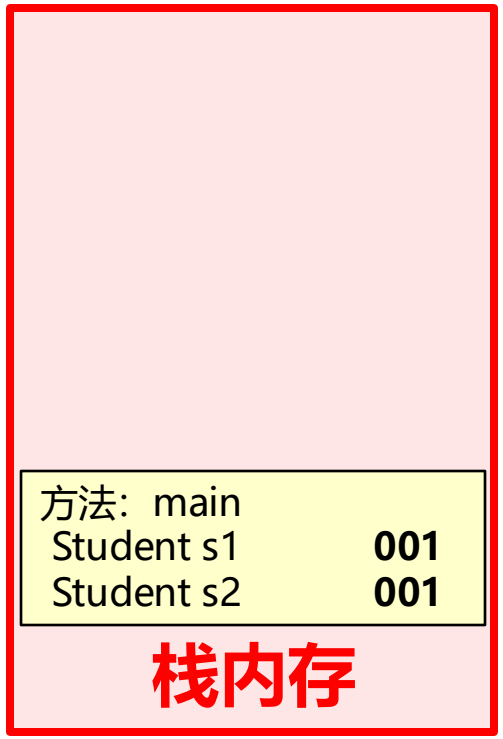



输出：林青霞，30

## 多个引用指向相同

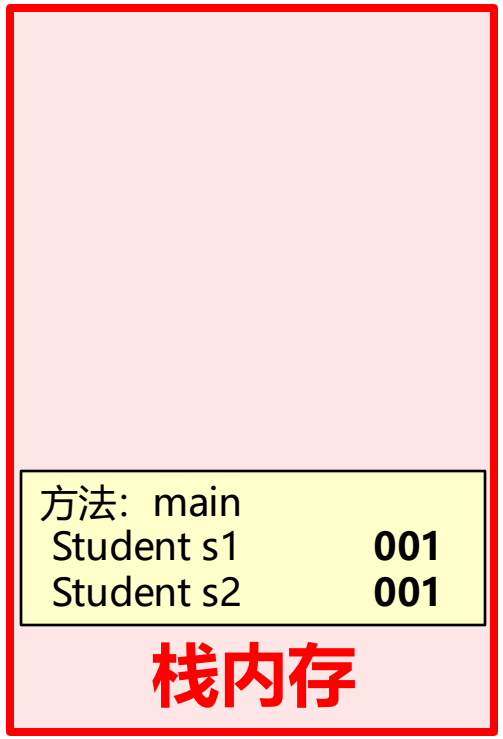
```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```

输出：林青霞，30
输出：张曼玉，28
输出：张曼玉，28



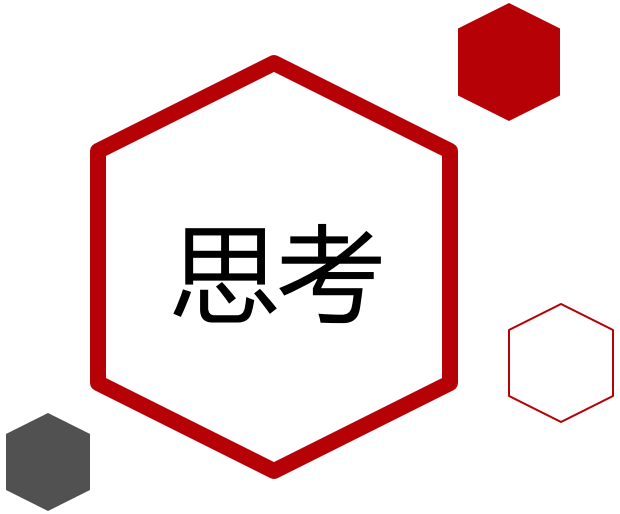

## 多个引用指向相同

```
public class StudentTest03 {
    public static void main(String[] args) {
        //创建第一个对象并使用
        Student s1 = new Student();
        s1.name = "林青霞";
        s1.age = 30;
        System.out.println(s1.name + "," + s1.age);
        //把第一个对象的地址赋值给第二个对象
        Student s2 = s1;
        s2.name = "张曼玉";
        s2.age = 28;
        System.out.println(s1.name + "," + s1.age);
        System.out.println(s2.name + "," + s2.age);
    }
}
```





输出：林青霞，30
输出：张曼玉，28
输出：张曼玉，28




成员变量和局部变量的区别?

## 成员变量和局部变量的区别

```
public class Student {

    String name;

    public void study() {
        int i = 0;
        System.out.println("好好学习");
    }

    public void doHomework() {
        System.out.println("多做练习");
        int j = 0;
    }

    int age;

}
```

## 成员变量和局部变量的区别

| 区别 | 成员变量 | 局部变量 |
| --- | --- | --- |
| 类中位置不同 | 类中方法外 | 方法内或者方法声明上 |
| | | |
| | | |
| | | |

## 成员变量和局部变量的区别

| 区别 | 成员变量 | 局部变量 |
| --- | --- | --- |
| 类中位置不同 | 类中方法外 | 方法内或者方法声明上 |
| 内存中位置不同 | 堆内存 | 栈内存 |
| | | |
| | | |

## 成员变量和局部变量的区别

| 区别 | 成员变量 | 局部变量 |
| --- | --- | --- |
| 类中位置不同 | 类中方法外 | 方法内或者方法声明上 |
| 内存中位置不同 | 堆内存 | 栈内存 |
| 生命周期不同 | 随着对象的存在而存在，随着对象的消失而消失 | 随着方法的调用而存在，随着方法的调用完毕而消失 |
| | | |

## 成员变量和局部变量的区别

| 区别 | 成员变量 | 局部变量 |
| --- | --- | --- |
| 类中位置不同 | 类中方法外 | 方法内或者方法声明上 |
| 内存中位置不同 | 堆内存 | 栈内存 |
| 生命周期不同 | 随着对象的存在而存在，随着对象的消失而消失 | 随着方法的调用而存在，随着方法的调用完毕而消失 |
| 初始化值不同 | 有默认的初始化值 | 没有默认的初始化值，必须先定义，赋值，才能使用 |

# 目录

Contents

## private 关键字

- 是一个权限修饰符
- 可以修饰成员（成员变量和成员方法）
- 作用是保护成员不被别的类使用，被 **private** 修饰的成员只在本类中才能访问

针对 **private** 修饰的成员变量，如果需要被其他类使用，提供两个相应的操作：

- 提供“**get变量名()**”方法，用于获取成员变量的值，方法用 **public** 修饰
- 提供“**set变量名(参数)**”方法，用于设置成员变量的值，方法用 **public** 修饰

# 目录

Contents

## this关键字

① this限定的变量用于指代成员变量

- 方法的形参如果与成员变量同名，

  不带this修饰的变量指的是形参，而不是成员变量

- 方法的形参没有与成员变量同名，

  不带this修饰的变量指的是成员变量

② 什么时候使用this呢？**解决局部变量隐藏成员变量**

```
public class Student {
    private String name;

    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

## this关键字

① this限定的变量用于指代成员变量

- 方法的形参如果与成员变量同名，

  不带this修饰的变量指的是形参，而不是成员变量

- 方法的形参没有与成员变量同名，

  不带this修饰的变量指的是成员变量

② 什么时候使用this呢？**解决局部变量隐藏成员变量**

③ this：方法被哪个对象调用，this就代表哪个对象

```
public class Student {
    private String name;

    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

```
public class StudentDemo {
    public static void main(String[] args) {
        Student s1 = new Student();
        s1.setName("林青霞");

        Student s2 = new Student();
        s2.setName("张曼玉");
    }
}
```

# 目录

Contents

# 封装

## 1. 封装概述

- 是面向对象三大特征之一（**封装，继承，多态**）
- 是面向对象编程语言对客观世界的模拟，客观世界里成

## 2. 封装原则

- 将类的某些信息隐藏在类内部，不允许外部程序直接访问
- 成员变量**private**，提供对应的**getXxx()/setXxx()**方

## 3. 封装好处

- 通过方法来控制成员变量的操作，提高了代码的安全性
- 把代码用方法进行封装，提高了代码的复用性

```
public class Student {
    private String name;

    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }

    private int age;

    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
}
```

# 目录

Contents

## 构造方法概述

- 构造方法是一种特殊的方法
- 作用：创建对象

格式：

```
public class 类名{
        修饰符 类名( 参数 ) {
        }
}
```

- 功能：主要是完成对象数据的初始化

```
public class StudentDemo {
    public static void main(String[] args) {
        Student s1 = new Student();
    }
}
```

```
public class Student {
    public Student() {
        //构造方法内书写的内容
    }
}
```

## 构造方法的注意事项

① 构造方法的创建

- 如果没有定义构造方法，系统将给出一个**默认**的**无参数构造方法**
- 如果定义了构造方法，系统将不再提供默认的构造方法

② 构造方法的重载

- 如果自定义了带参构造方法，还要使用无参数构造方法，就必须再写一个无参数构造方法

③ 推荐的使用方式

- **永远提供无参数构造方法**

## JavaBean

- 就是一个Java中的类，其对象可以用于在程序中封装数据
- 举例：学生类，手机类

标准 JavaBean 须满足如下要求：

- 成员变量使用 private 修饰
- 提供每一个成员变量对应的 setXxx() / getXxx()
- 提供一个无参构造方法

练习

# 手机 JavaBean

需求：编写手机 JavaBean，并在测试类中使用

提示：手机有成员变量brand(品牌)和price(价格)


传智教育旗下高端IT教育品牌