Java 采用按值调用的参数传递方式，也就是说，方法接收的是参数值的一个副本。

在 Java 中对方法参数的操作有如下性质：

- 方法不能修改基本数据类型的参数（即数值型和布尔型）。
- 方法可以改变对象参数的状态。
- 方法不能让一个对象参数引用一个新的对象。

下面对这三条性质做出说明。

1 基本数据类型

```
// 定义方法：
public static void tripleValue(double x)
{
    x = 3 * x;
}

// 调用方法：
double percent = 10;
trpleValue(percent);
```

这个方法的具体执行过程为：

1. `x` 初始化为 `percent` 值的一个副本（也就是 10）。
2. `x` 乘 3 后等于 30， `percent` 仍为 10。
3. 这个方法结束后，参数变量 `x` 不再使用。

由此可见，调用方法时传递给形参的是原来变量的值，方法中对参数的修改并不影响原来变量的值。

2 改变对象参数的状态

以前面提到的 `Employee` 类为例：

```
// 定义方法：
public static void tripleSalary(Employee x)
{
    x.raiseSalary(200);
}

// 调用方法：
Employee harry = new Employee(...);
tripleSalary(harry);
```

这个方法的具体执行过程为：

1. `x` 初始化为 `harry` 值的一个副本，`x` 和 `harry` 引用同一个对象。
2. `raiseSalary` 方法应用于这个对象引用，`x` 和 `harry` 同时引用的那个对象的工资提高了 200%。
3. 方法结束后，参数变量 `x` 不再使用。对象 `harry` 继续引用那个工资增至 3 倍的对象。

由此可见，对于对象参数，参数变量和原来的变量引用同一个对象，对参数变量的修改自然会作用到原来变量上。

3 改变对象参数引用的对象

例如：

```
// 定义方法:
public static void swap(Employee x, Employee y)
{
    Employee temp = x;
    x = y;
    y = temp;
}

// 调用方法:
Employee a = new Employee(...);
Employee b = new Employee(...);
swap(a, b);
```

这个方法的具体执行过程为：

1. `x` 初始化为 `a` 的一个副本，`x` 与 `a` 引用同一个对象；`y` 初始化为 `b` 的一个副本，`y` 与 `b` 引用同一个对象。
2. `x` 与 `y` 交换，`a` 和 `b` 不受影响。
3. 方法结束后，参数变量 `x` 和 `y` 不再使用，`a` 和 `b` 并未交换。

由此可见，方法改变的只是参数变量的引用，不能改变原来变量的引用。