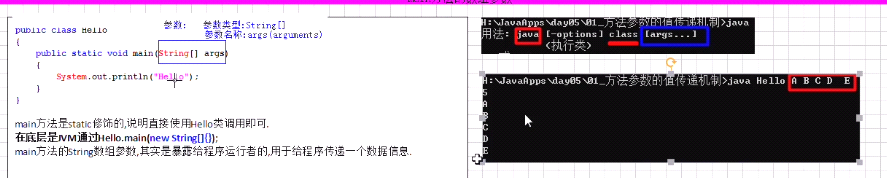
Day 05

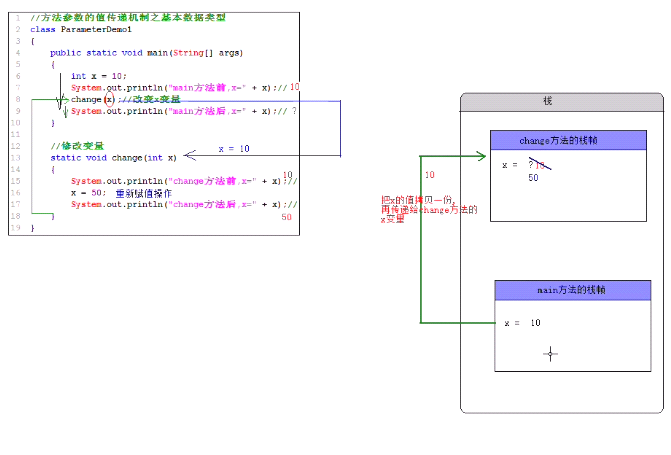
**01main方法的数组参数**



**02参数的值传递机制**

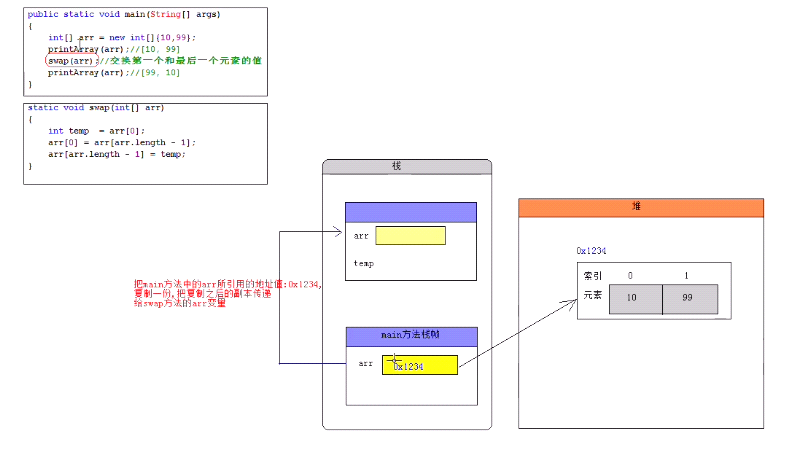
栈：每一个方法都存在一个栈帧，方法的执行和退出为压栈和出栈；

变量名和基本变量的值存储在方法的栈帧中。



方法参数的值传递时是先拷贝再传递，方法的参数在该方法的栈帧中存在。

**03 参数的值传递机制-引用数据类型**

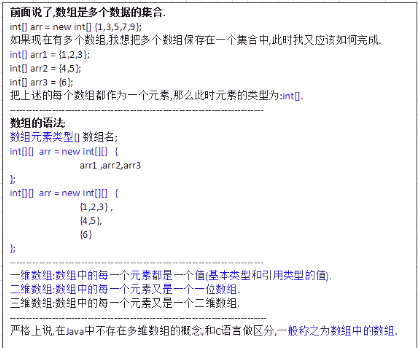


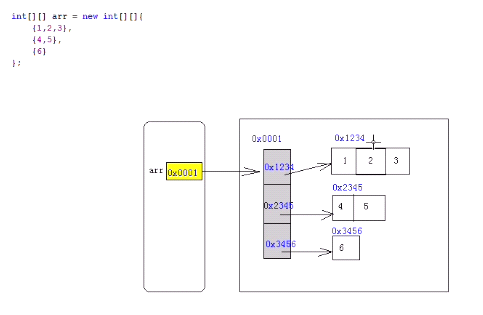
引用传递拷贝的是引用，是地址，然后将地址传递，这时两个变量实际操作的是同一个对象空间

对象空间在堆中，而堆是共享的；栈不是方法私有的。

**04多维数组**

**05多维数组的操作**





类、数组和接口是引用类型，存储在堆中。

多维数组中除了最内层存储的数实际数据外，其他存储的都是下一维数组的第一个元素的地址，所以说java中的多维数组实际不准确，严格来说应该是数组中的数组。

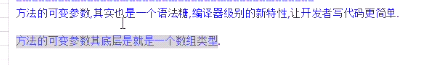
**06增强for循环**



Foreach是for的语法糖，底层仍然是由for来循环的，只是又重新将数组元素赋值给了一个新的变量。

正因为如此，不能在foreach循环中改变数组的值

**07 java5对于数组的支持-方法的可变参数**

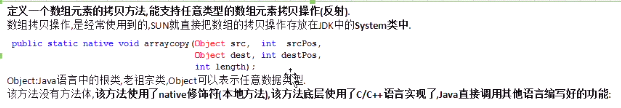


底层实现仍然是数组类型，如果定义两个同名的方法，一个参数为数组，一个为…，编译的时候会报错，定义了两个相同的方法。

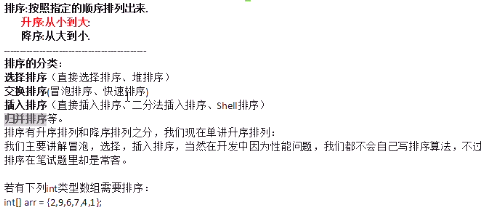
这是编译器级别的新特性；只是编译器在接受成参数时将所有的可变参数组合为一个数组；因此可变参数必须作为方法参数的最后一个。

**08实现数组元素的拷贝**

**09system中的copy方法**

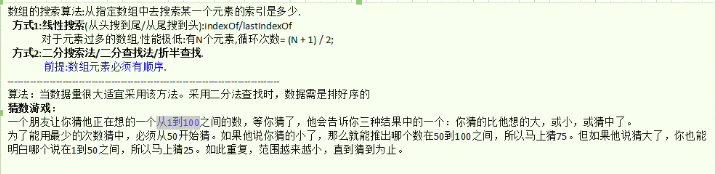


**10冒泡排序**



**11选择排序**

**12 二分查找**



**13自行封装操作数组的工具类**

定义数组工具类-ArrayUtil

**14java内置工具类**

Java默认到lang包里面寻找外部类的属性或者方法，所以lang包不用显示的引入，而其他的包需要显示的引入或者在使用时使用带包名的全名引用（完全限定名）。

15小结

