复习：

1. 数组：排序---冒泡、选择排序---时间复杂度O(n2)，空间复杂度o(1);Arrays.sort()---时间复杂度O(nlogn);查找---如果元素有序，可以使用二分查找---时间复杂度O(logn);反转---首尾交换；复制---本质上产生了一个新的数组---延伸出了数组的扩容机制

2. 二维数组：元素类型是一维数组。---二维数组的每一个位置上存放的是对应的一维数组的地址。---如果只定义了二维数组的大小，而没有定义包含的一维数组的大小，那么这时候每一个位置上存放的是一个null---如果针对null进行操作会出现NullPointerException

3. 方法/函数：为了使某段代码能够重复利用，所以将这段代码提取成方法。void--没有返回值；形参和实参；返回值---有了具体的返回执行类型就一定确保方法有返回值

public static boolean m(int number){ // 返回值类型是boolean类型，也就意味着方法必须有一个boolean类型的返回值

if(number < 4){ // 由于在返回之前需要进行判断，这个时候就意味着return语句不一定执行

return true;

}

return false;

}

public static boolean m2(int number){

for(int i = 0; i < 5; i++){ // 对于 JVM而言，i只是一个变量，具体值是未知的，所以无法判断变量i与5的大小是否一定成立，因此JVM认为return语句不一定会执行

// 凡是出现了变量参与的判断，jvm在编译的时候都不确保后续的代码是否一定执行

return number \* 2 > 3;

}

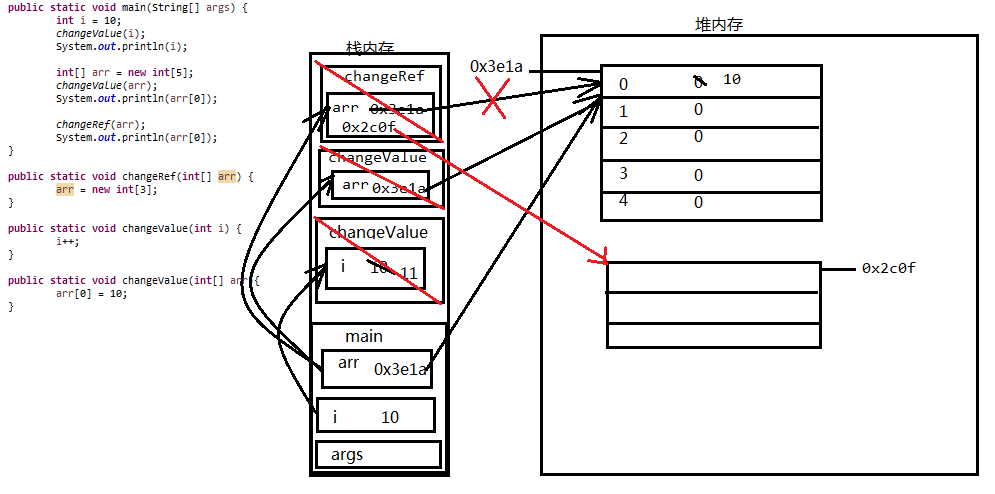
}

方法的递归---形式是方法调用自己本身。---方法是在栈中执行

### 方法的重载

在同一个类中，方法名一致而参数列表不同---方法的重载。---重点关注的就是参数列表

### 方法的传值

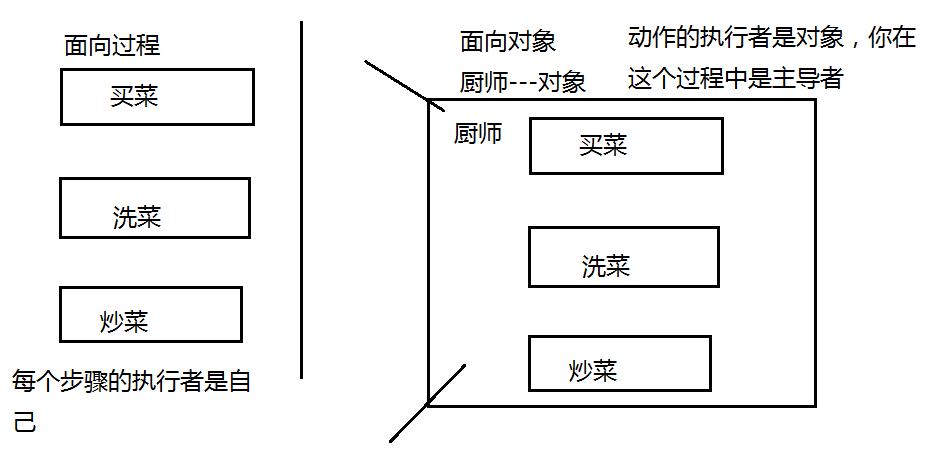


在方法传参的时候，基本类型传递的是实际的数据，所以新方法中数据发生改变不影响原方法中的数据；数组传递的是地址，所以在新方法中如果不改变地址，则会影响原来的方法中的数组；如果在新方法中改变了地址，则不会影响原来方法中的数组

## 面向对象

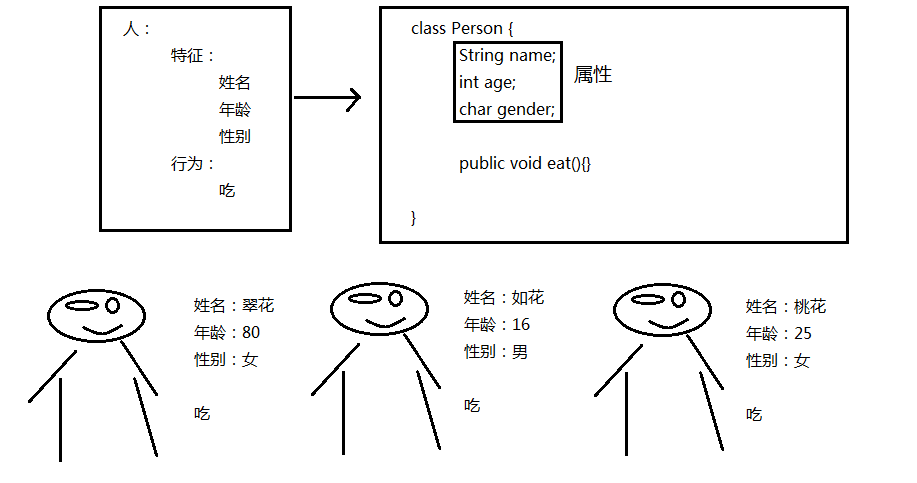
面向过程：注重的过程，强调的动作

面向对象：所关注的是对象。只要有了对应的对象，就能拥有这个对象所具有的一切功能

面向对象是基于面向过程的

面向对象一定优于面向过程吗？---不一定---当处理一些相对简单的事务的时候，建议使用面向过程；如果处理的事务比较复杂，建议使用面向对象。

### 类和对象的关系



根据一类对象的共有的特征和行为进行抽取，将这类事物用类来表示，将特征抽取成了属性，将行为抽取成方法---类是对象的抽取/概括

利用new关键字来创建对应的类的对象/实例---对象是类的具体化