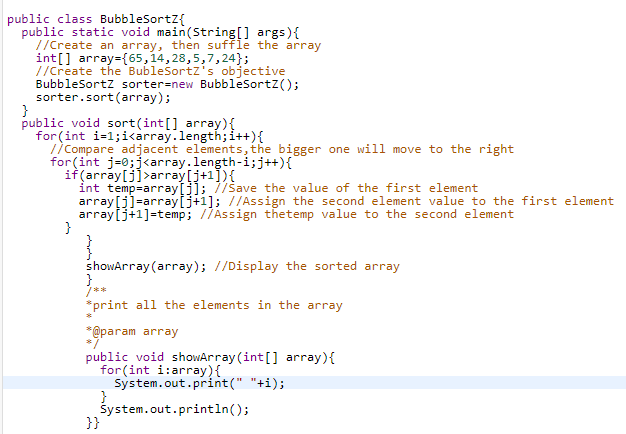
在进行Java学习中，于Java从入门到精通(第三版)第6章中6.5数组排序算法中了解到了几个常用的排序算法：冒泡排序，直接选择排序和反转排序。此文档主要用于记录以上几种排序及了解到的其他排序的具体用法以及优化方法：

冒泡排序：

冒泡排序有双层循环实现，其中外层循环用于控制排序轮数，一般为要排序的数组长度减次，而内层循环主要用于对比数组中的每个相邻元素的大小，以确定是否交换位置。(Java从入门到精通)

冒泡排序正排序演示：

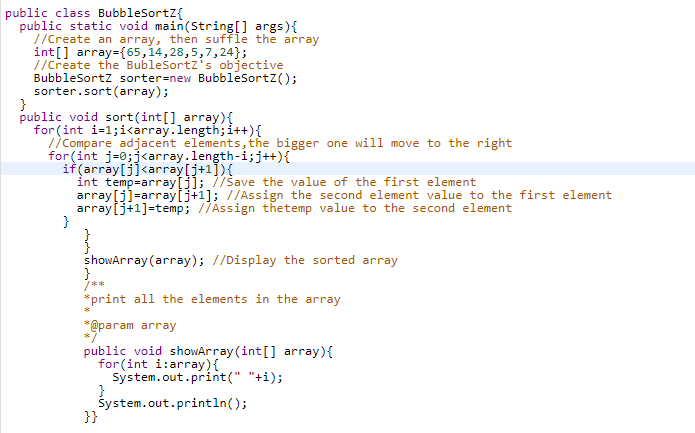


冒泡排序正排序结果：



PS：system.out.println和system.out.print的结果基本没什么差别，就是println最后会换行， print最后不会换行。搜寻了以上区别之后，发现还有system.out.printf,其中f应是指代的format。printf 可以进行格式化输出。

以上是冒泡排序正排序的代码演示以及结果，那么根据书上所示，如果我们要通过冒泡排序负排序，代码以及演示应该如何呢？其实改变if条件语句中的一句就可以了：

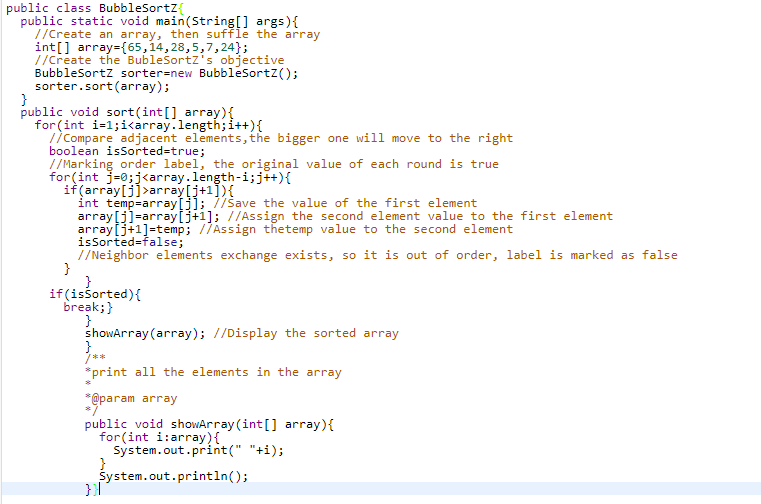
冒泡排序负排序演示：

冒泡排序负排序结果：



偶然一天看到来自于微信公众号中的@程序员小灰 介绍冒泡排序，其中了解到了冒泡排序的是一种稳定排序，由于该排序算法的每一轮要遍历所有元素，轮转的次数和元素数量相当，所以时间复杂度是O(N2)，显然这样的话执行比较缓慢。因此，@程序员小灰介绍了对于冒泡排序的两种优化点：

1. 判断数组已经有序，并且做出标记，剩下的几轮排序就可以不必执行，提早结束工作。

其代码以及演示如下：

其结果如下：



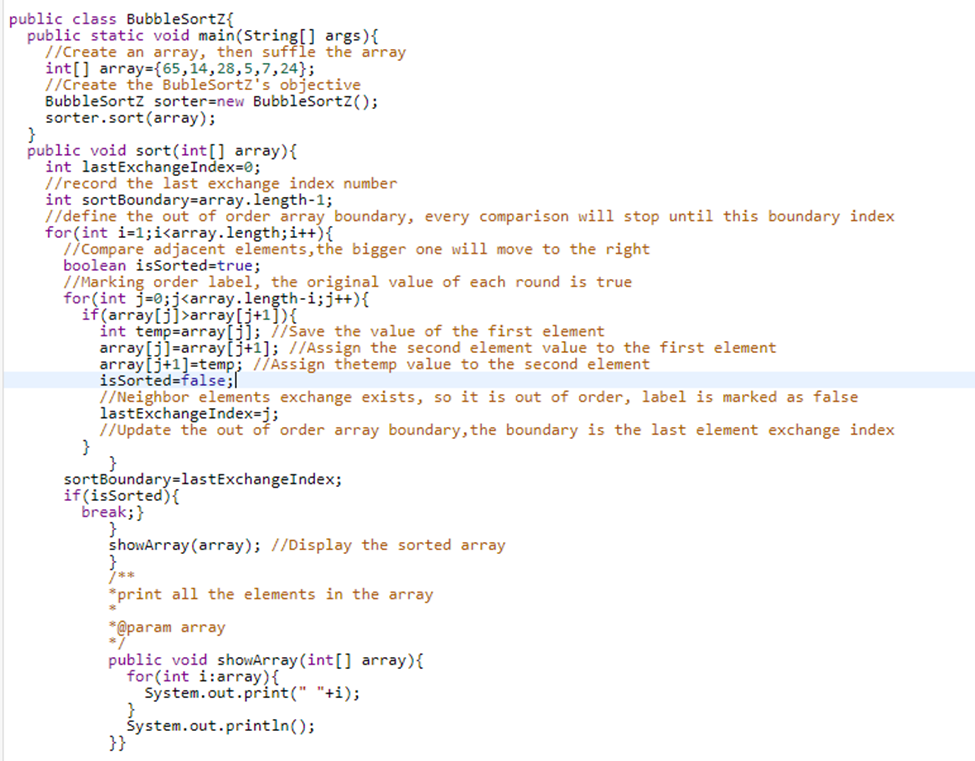
@程序员小灰 做了小小的改动，利用布尔变量isSorted作为标记。如果在本轮排序中，元素有交换，则说明数列无序；如果没有元素交换，说明数列已然有序，并没有直接跳出大循环。

之后，@程序员小灰又提出了一个问题：其实在冒泡排序中，如下数组：[3,4,2,1,5,6,7,8]

在第二轮排序中，数组的有序区包含一个元素，其实右边的许多元素已经是有序的了，可是每一轮还是白白比较了许多次。

因此@程序员小灰 提出了以下方法去避免这种情况：

可以在每一轮排序的最后，记录下最后一次元素交换的位置，那个位置也就是无序数列的边界，再往后则是有序区。

再次优化的冒泡排序如下所示：

其结果如下：



以上内容则是我在学习数组中了解到的冒泡排序的知识，作为一个基本算法，以上内容作为记录方便以后查询以及重温。