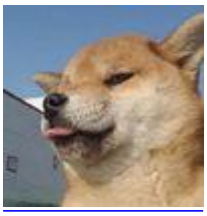
/ 算法与数据结构体系课 /

阶段三 · 冒泡排序，希尔排序和排序算法大总结

您的教学服务已到期，部分功能禁用 我要续费 什么是教学服务？

-
-
- 

您的教学服务已到期，续费即享完整服务

我们提供的教学服务

-

课程作业

-

老师 1 对 1 的作业批复

-
-

课程问答

-

不会的问题提出来，老师解惑同学互助

-
-

课程练习

-

课后即时练习与编程训练

线上考试

线上模拟测试，检验学习效果

1-6 换个方式实现冒泡排序

在这一小节，我们换一个方式，使用从后向前比较每个相邻元素的方式，来实现冒泡排序法。

首先，我们依然是需要进行 n – 1 轮的冒泡排序的过程，所以外循环依然是：
for(int i = 0; i + 1 < data.length; i ++)

不过，这次，因为我们是从后向前调整每个相邻的元素，所以，每轮循环以后，最小的元素会放到前面。

此时，我们的循环不变量变成了 arr[0, i) 已经排好序了。这个循环不变量更好理解，相当于前 i 个元素排好序了。我们每轮循环，只需要在 arr[i]的位置上放上合适的元素就好了。

至于内层循环，j 只需要从后向前遍历，每次比较 data[j – 1]和 data[j]就好。

我的参考代码如下：

```
public static <E extends Comparable<E>> void sort(E[] data){

    for(int i = 0; i + 1 < data.length; i ++){

        // arr[0, i) 已排好序
```

```
        // 通过冒泡在 arr[i] 位置放上合适的元素
        for(int j = data.length - 1; j > i; j --)
            if(data[j - 1].compareTo(data[j]) > 0)
                swap(data, j - 1, j);
    }
}
```

下面，我们开始尝试给这个代码进行优化。

第一个优化非常简单，我们使用一个叫 isSwapped 的变量跟踪内层循环的 swap 操作是否发生过，如果没有发生，说明整个数组已经有序了，就可以直接 break 了。

我的参考代码如下：

```
public static <E extends Comparable<E>> void sort2(E[] data){

    for(int i = 0; i + 1 < data.length; i ++){

        // arr[0, i) 已排好序
        // 通过冒泡在 arr[i] 位置放上合适的元素
        boolean isSwapped = false;
        for(int j = data.length - 1; j > i; j --)
            if(data[j - 1].compareTo(data[j]) > 0){
                swap(data, j - 1, j);
```

```
                isSwapped = true;

            }


        if(!isSwapped) break;

    }

}
```

第二个优化复杂一下，我们的外层循环，每一次不一定 i++，因为内层循环的结果，可能可以让我们跳过更多轮的外层循环。因此，外层循环我们写成这样 for(int i = 0; i + 1 < data.length;)，删掉了 i++，i 的变化在循环内控制。

之后，我们在每轮循环中，设立一个新的变量 lastSwappedIndex，记录最后的交换位置。因为现在，我们是从后向前查看相邻的数据对作调整，所以 lastSwappedIndex 的初值为 data.length – 1，即最后一个元素的位置。

在下面的第二轮循环过程中，一旦两个相邻的元素 data[j – 1]和 data[j]需要交换位置，我们就将 lastSwappedIndex 的值更新为 j–1，即这两个元素前面一个元素的位置。因为，如果后续不再有数据的交换的话，我们能肯定，data[0...j–1]范围的元素一定有序了。

因此，整个循环结束后，data[0...lastSwappedIndex]范围里的元素肯定有序了。那么 i 应该更新为多少？依然是，i 可以表示整个数组有多少个元素已经有序了。那么 data[0...lastSwappedIndex]范围里有多少元素？答案是 lastSwappedIndex+1。所以，最后，i 更新为 lastSwappedIndex+1。

我的参考代码如下：

```
    public static <E extends Comparable<E>> void sort3(E[] data){


        for(int i = 0; i + 1 < data.length; ){
```

```
        // arr[0, i) 已排好序
        // 通过冒泡在 arr[i] 位置放上合适的元素
        int lastSwappedIndex = data.length - 1;
        for(int j = data.length - 1; j > i; j --)
            if(data[j - 1].compareTo(data[j]) > 0){
                swap(data, j - 1, j);
                lastSwappedIndex = j - 1;
            }

        i = lastSwappedIndex + 1;
    }
}
```

怎么样？是不是通过一个小小的冒泡排序，也能学到不少东西？锻炼自己的编程水平？

大家加油！：）

此编程练习题暂不支持 WebIDE
请在本地环境完成练习
问答

续费提示

您的教学服务已到期，暂时无法提问与搜索，续费后恢复此功能

立即续费
暂无相关问题，您可点击“提问”提交
热门 待解决 我的提问 我的回答 搜索
数据加载中...
笔记

公开笔记
全部笔记 我的笔记

暂无相关笔记

帮同学解答一个问题吧

$qa1.title

巩固知识 获得学分 我来回答
换个问题

$qa2.title

巩固知识 获得学分 我来回答
换个问题

$qa3.title

巩固知识 获得学分 我来回答
换个问题

提示验证

您提交太频繁，为保证正常使用，请验证

章节 问答 课签 笔记 资料
数据加载中...
课程辅助材料
章节

这是课程的目录，可以随时唤出目录切换章节学习

不用提醒 下一步