

哈尔滨工业大学(深圳)

《数据结构》实验报告

实验五

排序、查找及其应用

学 院: 计算机科学与技术

姓 名: 苏亦凡

学 号: 200111229

专 业: 计算机科学与技术

日 期: 2021-05-21

一、问题分析

题目 1：问题实质为求出总数为奇数的一组未排序数据的中位数。问题涉及数组排序以及简单的对中位数的定位。

题目 2：问题实质为求出未排序数组中最大的 k 个元素，并且按从小到大的顺序进行输出。问题涉及数组排序，由于只需求出最大的 k 个元素故合理选取排序的方法可优化算法。

题目 3：问题可抽象为一个在不同时间段不断变动的数据，学生的空闲时间段模拟成进出，在某时间段的总量对应空闲学生人数。问题即求取其中总数最大的时间段。主要涉及离散化与排序。

二、详细设计

2.1 设计思想

题目 1：要求对整个数组进行排序，考虑到时间复杂度的问题，选择快速排序。对排序后的数组，中位数的下标为 $(n-1)/2$ 。时间复杂度为 $O(n\log n)$ 。

题目 2：只需输出最大的 k 个数，可采用不完善的堆排序，输出 k 次大顶堆的堆顶即可。有新建初始堆的时间复杂度为 $O(n)$ ，输出 k 次堆顶时间复杂度为 $O(k\log n)$ 。总体时间复杂度小于 $O(n\log n)$ 。

题目 3：根据此题模型，考虑定义结构体 PERIOD，用结构体数组存储信息。

其中 time 存储时间段开始时间，结束时间用元素后继的 time 值-1 表示；nstu 储存该时间段空闲学生人数。首先对 start 和 end 数组进行排序，遍历读取两数组的元素获取每个时间段的变化量，计算当前空闲人数并将信息储存于结构体数组中。遍历结构体数组求取最大人数，最后遍历数组输出时间段。

2.2 存储结构及操作

(1) 存储结构

主要是数组。堆排序涉及到堆，但堆亦是以数组的形式储存的。

(2) 涉及的操作

快速排序：

```
void QSort(int *a, int low, int high); //部分快速排序
void QuickSort(int *a, int n); //封装的快速排序
int Partion(int *a, int low, int high); //划分
```

堆排序：

```
void HeapAdjust(int *H, int s, int m); //调整 s 到 m 使之符合堆
void ArrtoHeap(int *arr, int N); //将数组转变为堆
```

三、用户手册

如：(1)输入数据的方式；(2)实现各种功能的操作方式等。

(1) 数据输入方式：

题目 1：输入文件 5_1_input_5.in

第 1 行：奶牛数量 N（奇数）

第 2 行：N 头母牛的牛奶产量。

题目 2: 输入文件 5_2_input_5.in

第 1 行: 测试数据的组数

第 2 行: 最大 k 个元素的 k 值 (第一组测试数据)

第 3 行: 数组元素个数 n 值

第 4~n+3 行: 数组元素

第 n+4 行: 最大 k 个元素的 k 值 (第二组测试数据)

.....

题目 3: 输入文件 5_3_input.in

第 1 行: N 值 M 值

后续行: 开始时间 结束时间

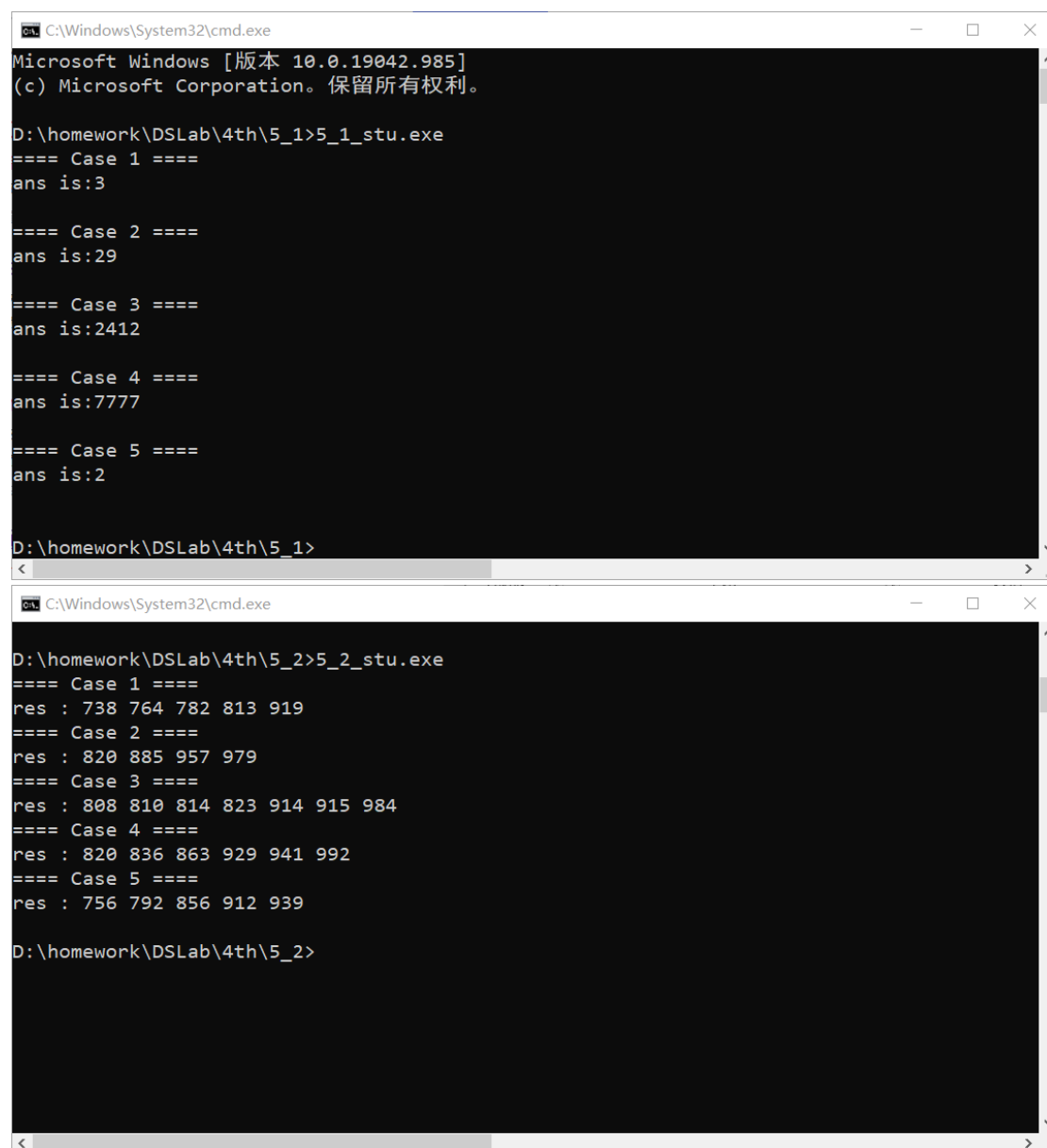
(2) 实现各种功能的操作方式:

求中位数: 调用 solve1 函数传入参数为数组名, 数组长度 (奇数)。返回值为中位数。

求最大的 k 个数: 调用 solve2 函数, 传入参数为数组名, 数组长度, k 值。返回值为存有最大 k 个数的数组头指针。

求空闲人数最多的时间段: 调用 findPeriod 函数, 传入参数为不减排序的 start 数组, 不减排序的 end 数组, M 值, N 值。

四、结果



```
C:\Windows\System32\cmd.exe
Microsoft Windows [版本 10.0.19042.985]
(c) Microsoft Corporation。保留所有权利。

D:\homework\DSLab\4th\5_1>5_1_stu.exe
==== Case 1 ====
ans is:3

==== Case 2 ====
ans is:29

==== Case 3 ====
ans is:2412

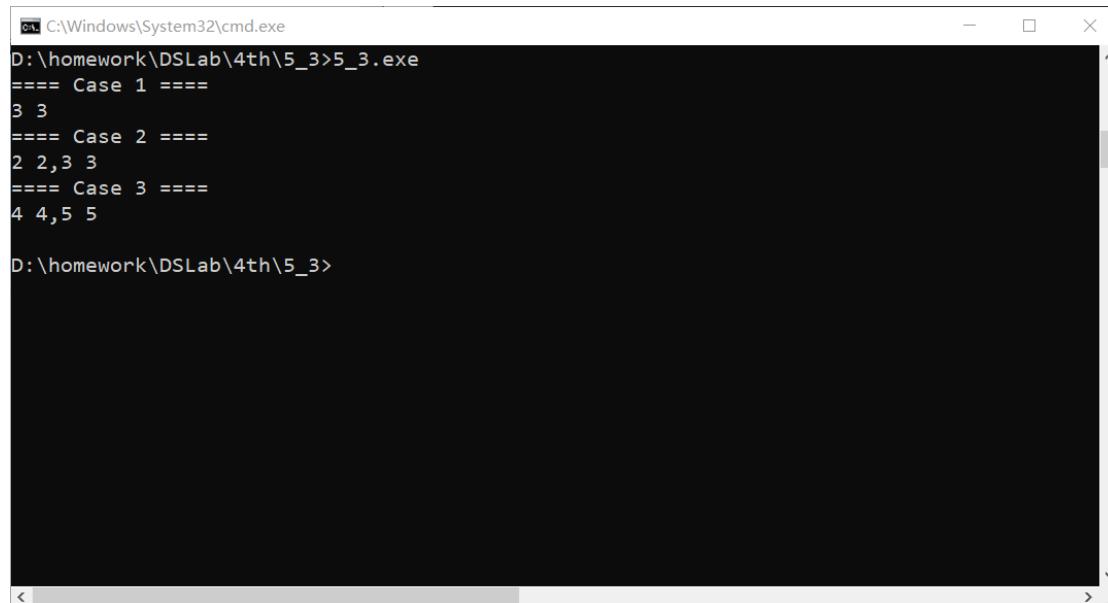
==== Case 4 ====
ans is:7777

==== Case 5 ====
ans is:2

D:\homework\DSLab\4th\5_1>

C:\Windows\System32\cmd.exe
D:\homework\DSLab\4th\5_2>5_2_stu.exe
==== Case 1 ====
res : 738 764 782 813 919
==== Case 2 ====
res : 820 885 957 979
==== Case 3 ====
res : 808 810 814 823 914 915 984
==== Case 4 ====
res : 820 836 863 929 941 992
==== Case 5 ====
res : 756 792 856 912 939

D:\homework\DSLab\4th\5_2>
```



```
C:\Windows\System32\cmd.exe
D:\homework\DSLab\4th\5_3>5_3.exe
==== Case 1 ====
3 3
==== Case 2 ====
2 2,3 3
==== Case 3 ====
4 4,5 5

D:\homework\DSLab\4th\5_3>
```

五、总结

该实验涉及到的数据结构和算法，以及遇到的问题和收获。

实验涉及的数据结构主要是线性结构中的数组，以及用数组储存的堆。涉及的算法为各种排序算法，包括快速排序、堆排序。

此次实验只要对各种排序算法熟悉便没有太多难点，困难主要是在第三题中。在第三题离散化是重要的思想。题目虽然说 N 为一个很大的数，但在实际解决问题时并不怎么涉及对 N 这一数据的处理。问题的关键是抽象出此题的模型。在实际操作中，由于此题涉及的数据较多，要时刻考虑数组越界等问题。虽然题目只对 start 与 end 排序的时间复杂度做了要求，但在其他部分的时间复杂度只为 $O(M)$ ，完全可以将整个算法的时间复杂度限制在 $O(M\log M)$ 以下。

在第二题算法的优化中涉及堆排序的特点。堆排序是先对整个数组进行一次处理，后面逐步输出最大（或最小）元素并进一步调整，因此不需要对整个数组进行排序，最后综合时间复杂度为 $O(n+k\log n)$ ，相比之下用快速排序的时间复

杂度为 $O(n \log n)$ 。当然，当 k 较小时此题可能可以采取选择排序或冒泡排序，亦如堆排序一样只需要 k 趟排序，对应时间复杂度为 $O(kn)$ 。