|  |
| --- |
| 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: | 朱海峰 | | 学 号: | 190110716 | | 专 业: | 计算机科学与技术 | | 日 期: | 2020-05-23 | |

# 一、问题分析

## 1.1 问题

**题目1**

给定一组大小为-107到107的整数，元素不重复，找到其中值差最小的元素对，如若有多对，请都输出。

首先输入数组元素个数N，其中N的范围为2-2000。接着程序输入N个不重复的整数，整数间用空格隔开。

**题目2**

对数组内元素进行排序，数组元素最高可达1000位。

首先输入数组元素个数N，其中N为1～1000。接着输入N行，每行为一个不以0开头的正整数。

**题目3**

课程即将结束迎来考试，统计同学们空闲的时间段后以少数服从多数的原则进行考试安排，故需统计人数最多的时间段，考试时间被分为了N个时间段，其中N可能会非常大，可以假设N为1亿，也就是100000000（这要求程序不能声明长度为N的数组或定义N个变量），若未考虑此情况则不能得分。

首先输入两个数字N，M，表示N个考试时间段，M个考试学生。接着程序输入M行，每行两个数字，表示第i名同学空闲时间的开始时间段与终止时间段。其中N的范围为1-100000000，M的范围为1-1000。当两个时间段的空闲人员不完全一致时，认为其为不同时间段。

要求：排序算法的平均时间复杂度不得大于O(MlogM)。

## 1.2 分析

**题目1**

可以定义一个长度大于两千的整型数组，存储所有数据，完成排序后，输出差值最小的元素对即可

**题目2**

可以定义一个长度大于1000的字符串数组，每个字符串容量为1000。跟据字符串的长度、元素的ASCII码值的比较完成排序输出即可。

**题目3**

可对所有开始时间和结束时间定义为两个数组存储，完成排序。

之后通过遍历开始时间和结束时间的数组，建立一个新的结构体数组，单个结构体记录一个**有空闲情况变化的关键时间段（后简称关键时间段）**的人数及变化的完全把握。

通过一次遍历结构体数组记录空闲人数最大值，再一次遍历结构体数组即可输出空闲人数最多的时间段。

# 二、详细设计

## 2.1 设计思想

三个题目显然都需要对相关数据进行排序，考虑到数据量不小，应使用快速排序，提高计算机工作效率。

#### 2.1.1快速排序

**快速排序（后简称快排）**用到递归的思想。对一个序列进行快排：

1. 确定中间值；
2. 分割：将其他数据按与中间值大小关系放在两侧；
3. 对中间值左边序列进行快排。
4. 对中间值右边序列进行快排。

递归执行以上操作，直到每个序列的长度均为1，则完成原序列的快排。

另外，题目二中超大整数比较大小，使用以下方法：

由于数位可能较长，整型甚至双精度浮点型都不能满足，考虑用到长度大于1000的字符串，排序中比较稍有不同——先比较字符串长度，长者为大；长度相同则直接通过ASCII码值从0号元素开始比较字符串大小。

#### 2.1.2附加操作

对于题目1，遍历数据，比较相邻两数差值，并记录最终的最小差值。再次遍历数据，将符合差值最小的输出即可。

对于题目2，只需要进行快速排序输出即可。

对于题目3，通过遍历开始时间和结束时间的数组，建立一个新的结构体数组，单个结构体表示一个关键时间段，每个关键时间段开始、结束时的人数均可存储在对应结构体中，这样完成对每个关键时间段的人数增减的完全把握。

具体建立方法是开始和结束时间两个数组中用两个指针，按时间顺序交替遍历，然后在对应关键时间结构体中，记录当前时间段开始时的人数以及结束时的人数。那么这样就可以，对于开始时和结束时人数均为最大值的表明此关键时间段就是空闲人数最多的一个时间段；而只有开始时人数是最大值的关键时间结构体表明从当前关键时间到下一个关键时间是一个空闲人数最多的时间段。

通过一次遍历结构体数组记录空闲人数最大值，再一次遍历结构体数组，通过对各个关键时间段信息的判断，分为单个关键时间段、该关键时间段到下一个关键时间段的两种情况，输出空闲人数最多的所有时间段。

## 2.2 存储结构及操作

#### 2.2.1存储结构

主要用到线性结构，包括静态数组和动态数组，不涉及树形结构和图形结构。

题目1

int arr[2002]//直接用于存储数据

题目2

typedef struct nString{//自定义字符串

char strNum[1002];//单个数据

int length;//数据位数

}nString;

nString arr[1002];//完整数据

题目3

int start[1002];//开始时间数组

int end[1002];//结束时间数组

struct time\_Point{//每个时间段

int num1,//开始时人数，无新增用不到则直接为零即可

num2;//结束时人数，什么情况都有值

int time;//当前时间

};

//动态结构体数组减小空间占用

struct time\_Point\* TIME=(struct time\_Point\*)malloc(2\*M\*sizeof(struct time\_Point));

#### 2.2.2 涉及操作

快速排序

int partion(int i,int j,int\* arr)//分割算法

void QuickSort(int i, int j,int\* arr)//快排

题目1

void findClosestPair(int \*arr, int num)//找差值最小元素组并输出

题目2

int cmpStr(nString fir,nString sec)//比较两个字符串大小

题目3

void findPeriod(int \*start,int \*end,int M,int N) //计算空闲人数最多的所有时间段

## 2.3 程序整体流程

1. 由于实验都用到快速排序，先给出快排算法流程，如下图1所示；
2. 题目1，先进行快速排序，同图1，再找差值最小元素对，如下图2所示；
3. 题目2，进行快速排序，同图1，其中比较字符串整数大小，如下图3所示；
4. 题目3，完整流程如下图4所示。

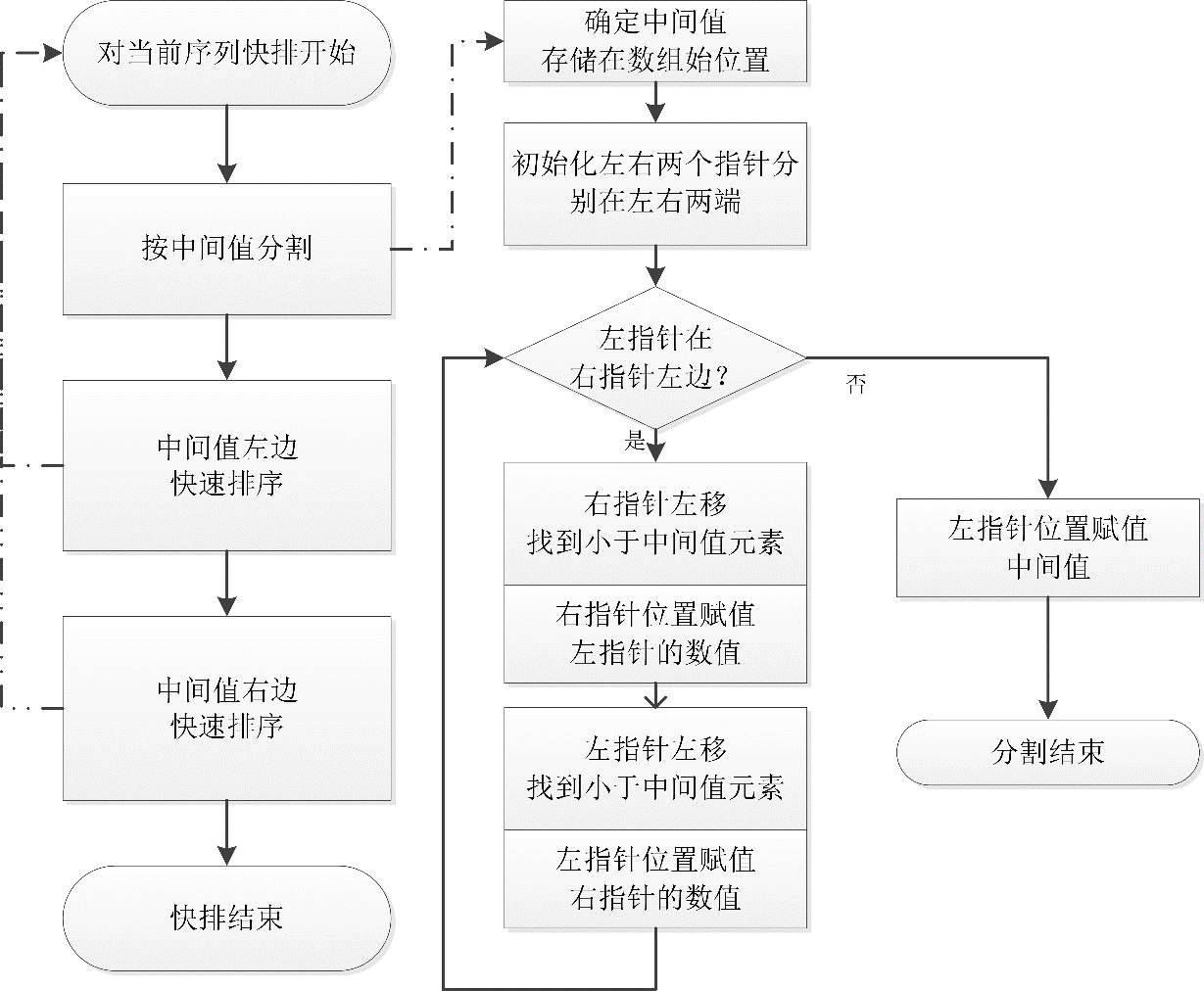


图 1

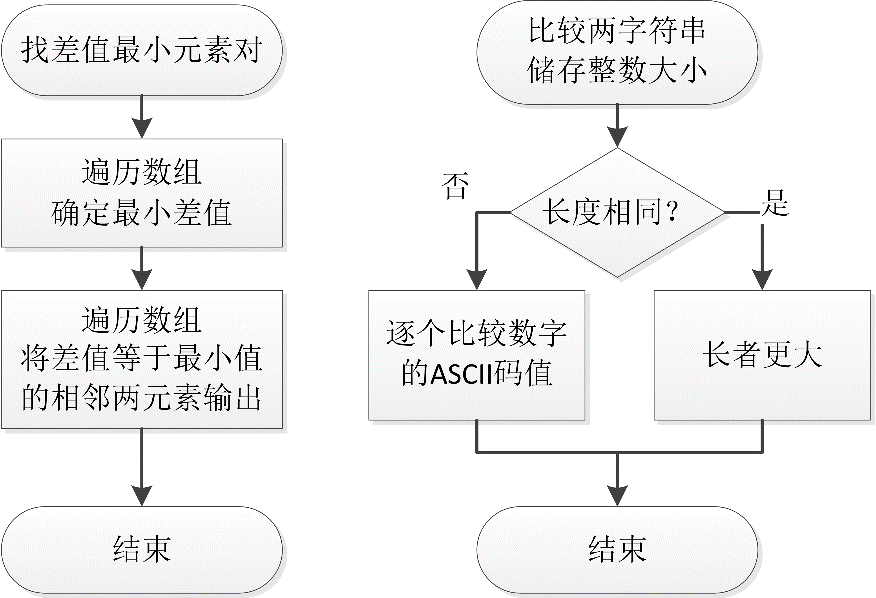


图 2 图 3

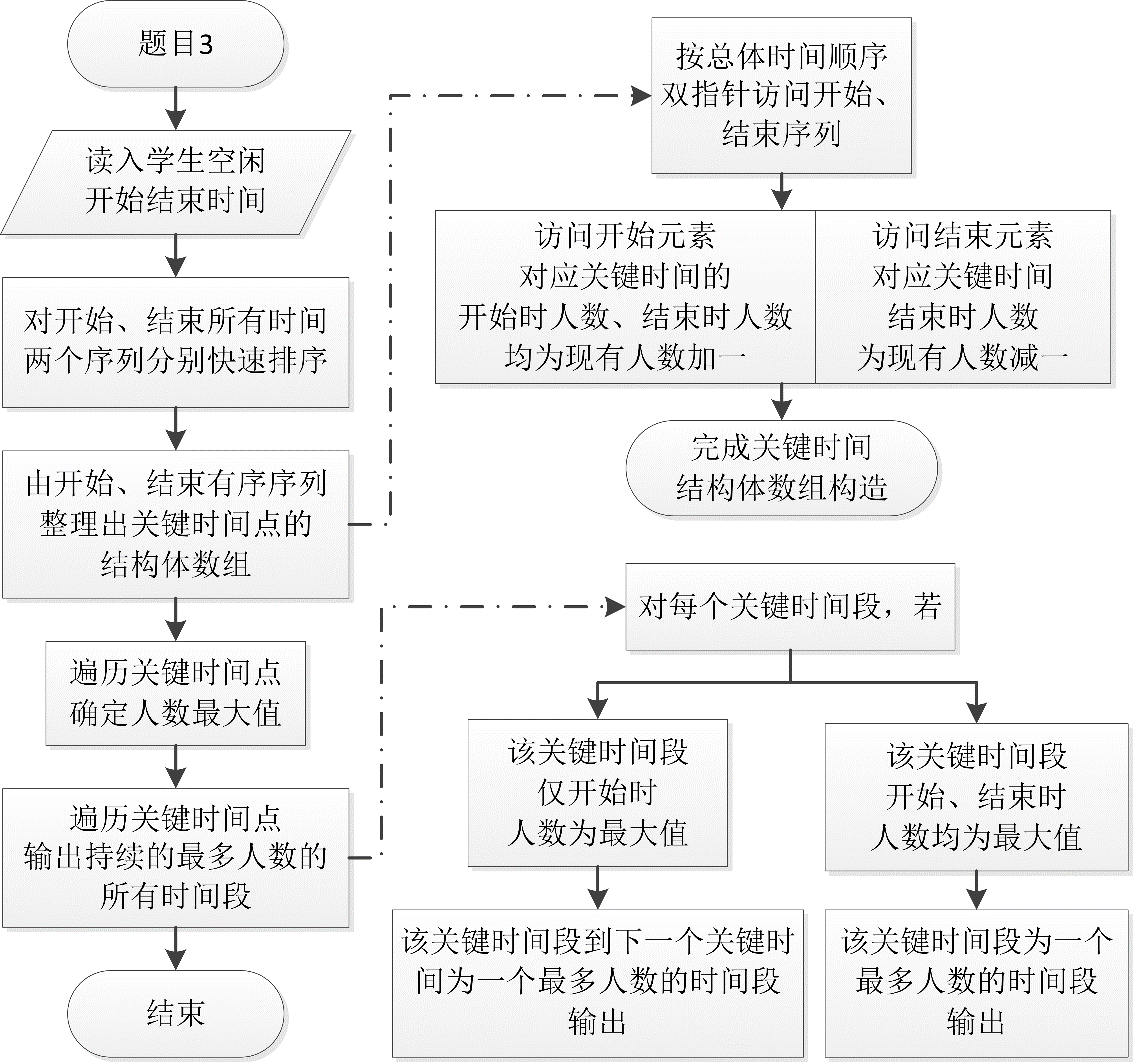


图 4

# 三、用户手册

欢迎使用本程序！

本程序由三个子程序构成：

* 求大小为-107到107，且元素不重复的一组整数中值差最小的元素对。
* 对小于1000个的1000位以内的正数进行排序。
* 由数名同学不同空闲时间段，统计空闲人数最多且人员一致的连续时间段。

子程序一，首先输入数组元素个数N（2≤N≤2000）。接着程序输入N个不重复的整数，用空格隔开。

系统自动输出绝对值差若干对最小的元素对，每对两个数字。

示例如下：

输入：

4

2 3 4 5

输出：

2 3,3 4,4 5

子程序二，首先输入数组元素个数N（1≤N≤2000）。接着输入N行，每行为一个不以0开头的正整数。

系统自动按升序逐行输出N行元素。

示例如下：

输入：

3

2718281828459

31415926535897932385626897067

3

输出：

3

2718281828459

31415926535897932385626897067

子程序三，首先输入两个数字N（1≤N≤100000000），M（1≤M≤1000），表示N个考试时间段，M个考试学生。接着程序输入M行，每行两个数字，表示第i名同学空闲时间的开始时间段与终止时间段。

系统自动输出若干对起止时间，每对两个数字，表示空闲人数最多且空闲人员完全一致时间段的起始和终止时间。

示例如下：

输入1：

5 5

1 2

2 3

1 5

3 5

3 4

输出1：

3 3

输入2：

5 5

1 2

2 3

1 5

2 5

3 4

输出2：

2 2,3 3

感谢您的使用！欢迎您宝贵的提出意见与建议！

# 四、结果

1. 题目1运行结果如下图5所示；
2. 题目2运行结果如下图6所示；
3. 题目3运行结果如下图7所示。

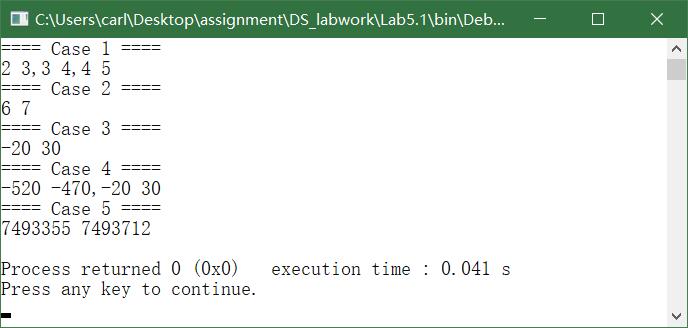


图 5

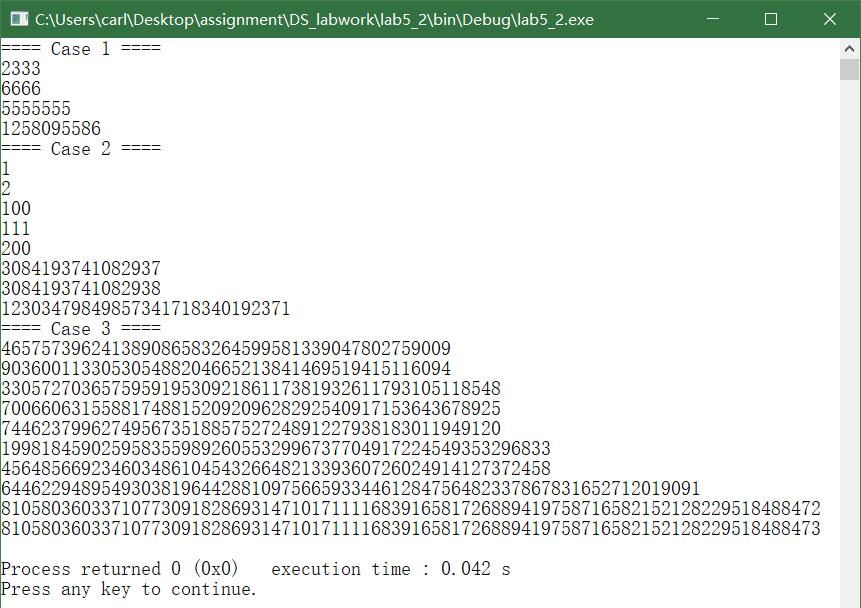


图 6

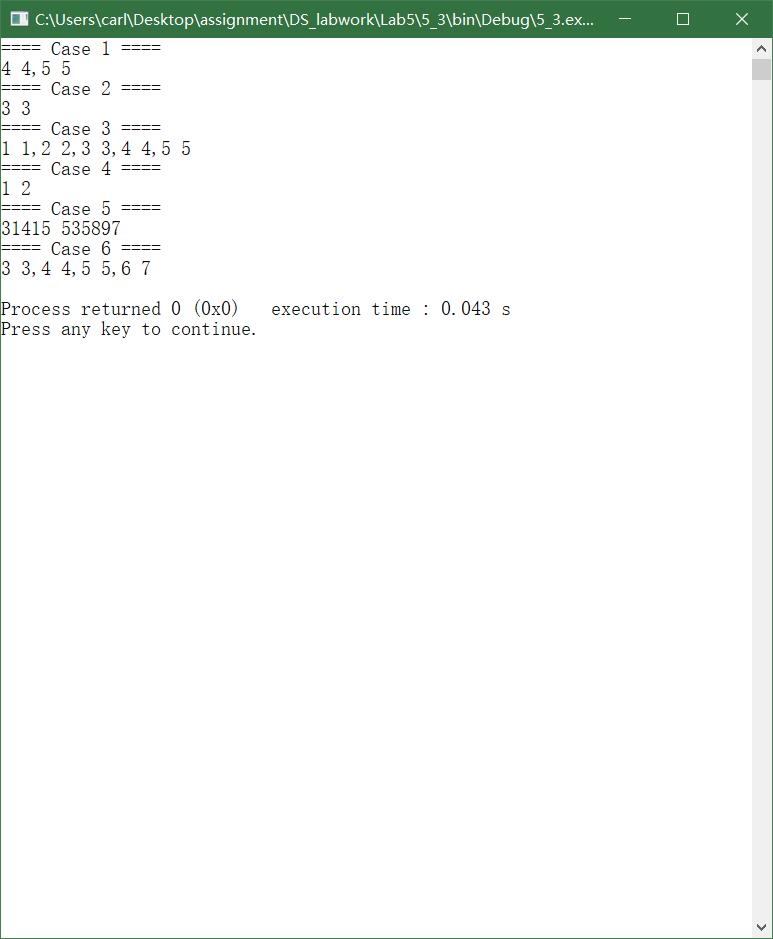


图 7

# 五、总结

该实验主要涉及到线性结构，包括静态数组和动态数组，不涉及树形结构和图形结构。涉及顺序结构，不涉及链式结构。

实验考察排序算法的应用，这里我选择使用快速排序。快速排序的平均时间复杂度为O(nlogn)，实际操作中，相较于堆排序、归并排序运行更快，读写内存次数少，在大规模排序中效率优势极其明显。而本实验要求的数据量最大可以到千级，相较于简单选择排序，≈10<<1000，节省时间更明显。

本次实验题目三的实际应用性很强，也没有现成的算法。相较于去年的题目，考虑的东西更多，实现起来就更复杂。我真就完全一点一点地试，一点一点地调，特殊情况特殊解决，包括考虑现入现出的，是不是最后一个，打不打印逗号，一个个地添加了好多个辅助的参数。一不小心调试到了凌晨两点，还好这是最后一次了，我也不用自责这次实验打乱了我整个周末的生物钟，实验做出了很有成就感。

在C语言的应用上，

1. 不断尝试学习，包括新的算法、数据结构和一些以前没有遇到的函数；
2. 在考虑边界情况、特殊数据、参数设置、循环选择控制上，我都提升了思考的缜密程度；
3. 还有就是调试方法，遇到ide崩盘时，强行用printf比对参数变化情况，正常情况下，单行调试在watches里面查看异常情况。虽然有的问题暂时没解决，程序运行也不影响，在编程上我也已经熟练更多。

完结！撒花！