

# OUCEEHLLP 课程任务四

郑海永

2016 年

## 目录

1	数组和字符串应用	1
1.1	编程题 1：求字母的个数	1
1.2	编程题 2：忽略大小写比较字符串大小	2
1.3	编程题 3：最长单词 2	3
1.4	编程题 4：矩阵交换行	4
1.5	编程题 5：异常细胞检测	6
1.6	编程题 6：循环移动	6
1.7	编程题 7：中位数	7
1.8	编程题 8：校门外的树	8

## 1. 数组和字符串应用

### 1.1 编程题 1：求字母的个数

<http://oucee.openjudge.cn/a4/1>

**来源** POJ 1690

**注意** 总时间限制: 1000ms 内存限制: 65536kB

**描述** 在一个字符串中找出元音字母 a、e、i、o、u 出现的次数。

**输入** 输入一行字符串（字符串中可能有空格，请用 `cin.getline(s, counts)` 方法把一行字符串输入到字符数组 `s` 中，其中 `counts` 是 `s` 的最大长度，这道题里面可以直接写 80。），字符串长度小于 80 个字符。

**输出** 输出一行，依次输出 a、e、i、o、u 在输入字符串中出现的次数，整数之间用空格分隔。

**样例输入**

```
1 If so, you already have a Google Account. You can sign in on the right.
```

## 样例输出

```
1 5 4 3 7 3
```

## 提示

注意，只统计小写元音字母 a、e、i、o、u 出现的次数。

## 1.2 编程题 2：忽略大小写比较字符串大小

<http://oucee.openjudge.cn/a4/2>

## 来源

POJ 1723

## 注意

总时间限制: 1000ms 内存限制: 65536kB

## 描述

一般我们用 `strcmp` 可比较两个字符串的大小，比较方法为对两个字符串从前往后逐个字符相比较（按 ASCII 码值大小比较），直到出现不同的字符或遇到 `'\0'` 为止。如果全部字符都相同，则认为相同；如果出现不相同的字符，则以第一个不相同的字符的比较结果为准。但在有些时候，我们比较字符串的大小时，希望忽略字母的大小，例如 "Hello" 和 "hello" 在忽略字母大小写时是相等的。请写一个程序，实现对两个字符串进行忽略字母大小写的大小比较。

## 输入

输入为两行，每行一个字符串，共两个字符串。（请用 `cin.getline(s,80)` 录入每行字符串）（每个字符串长度都小于 80）

## 输出

- 如果第一个字符串比第二个字符串小，输出一个字符 "<"；
- 如果第一个字符串比第二个字符串大，输出一个字符 ">"；
- 如果两个字符串相等，输出一个字符 "="。

## 样例输入

```
1 第一组
2 Hello
3 hello
4 第二组
5 hello
6 HI
7 第三组
8 hello
9 HELL
```

**样例输出**

```
1 第一组
2 =
3 第二组
4 <
5 第三组
6 >
```

**提示**

strcmp 的实现如下，结果用result 保存。

```
1 int i = 0;
2 char result;
3 while (s1[i] != '\0' && (s1[i] == s2[i])) {
4     i++;
5 }
6 if (s1[i] > s2[i]) {
7     result = '>';
8 } else if (s1[i] < s2[i]) {
9     result = '<';
10 } else {
11     result = '=';
12 }
```

### 1.3 编程题 3：最长单词 2

<http://oucee.openjudge.cn/a4/3>

**来源**

POJ 6248

**注意**

总时间限制: 1000ms 内存限制: 65536kB

**描述**

一个以'.'结尾的简单英文句子，单词之间用空格分隔，没有缩写形式和其它特殊形式。

**输入**

一个以'.'结尾的简单英文句子（长度不超过 500），单词之间用空格分隔，没有缩写形式和其它特殊形式。

**输出**

该句子中最长的单词。如果多于一个，则输出第一个。

**样例输入**

```
1 第一组
2 I am a student of Peking University.
3 第二组
4 Hello world.
```

#### 样例输出

```
1 第一组
2 University
3 第二组
4 Hello
```

## 1.4 编程题 4：矩阵交换行

<http://oucee.openjudge.cn/a4/4>

**来源** POJ 1901

**注意** 总时间限制: 1000ms 内存限制: 65536kB

**描述** 在main函数中，生成一个 $5 \times 5$ 的矩阵，输入矩阵数据，并输入 $n, m$ 的值。判断 $n, m$ 是否在数组范围内，如果不在，则输出error；如果在范围内，则将 $n$ 行和 $m$ 行交换，输出交换 $n, m$ 后的新矩阵。

**输入**  $5 \times 5$  矩阵的数据，以及 $n$ 和 $m$ 的值。

**输出**

- 如果不可交换，则输出error。
- 如果可交换，则输出新矩阵。

**样例输入**

```
1 第一组
2 1 2 2 1 2
3 5 6 7 8 3
4 9 3 0 5 3
5 7 2 1 4 6
6 3 0 8 2 4
7 0 4
8 第二组
9 1 2 2 1 2
10 5 6 7 8 3
11 9 3 0 5 3
12 7 2 1 4 6
13 3 0 8 2 4
14 5 1
```

### 样例输出

```
1 第一组
2   3   0   8   2   4
3   5   6   7   8   3
4   9   3   0   5   3
5   7   2   1   4   6
6   1   2   2   1   2
7 第二组
8 error
```

### 提示

输出error 格式如下：

```
1 cout << "error" << endl;
```

输出矩阵格式如下：

```
1 cout<< setw(4) << num;
```

输出矩阵一行后要输出cout<<endl;。

setw 是iomanip 库里定义的格式控制操作符，需要#include <iomanip> 包含这个头文件。

## 1.5 编程题 5：异常细胞检测

<http://oucee.openjudge.cn/a4/5>

**来源** POJ 1939

**注意** 总时间限制: 1000ms 内存限制: 65536kB

**描述** 我们拍摄的一张 CT 照片用一个二维数组来存储, 假设数组中的每个点代表一个细胞。每个细胞的颜色用 0 到 255 之间 (包括 0 和 255) 的一个整数表示。我们定义一个细胞是异常细胞, 如果这个细胞的颜色值比它上下左右 4 个细胞的颜色值都小 50 以上 (包括 50)。数组边缘上的细胞我们不检测。现在我们的任务是, 给定一个存储 CT 照片的二维数组, 写程序统计照片中异常细胞的数目。

**输入**

- 第一行包含一个整数  $N$  ( $100 \geq N > 2$ )。
- 下面有  $N$  行, 每行有  $N$  个  $0 \sim 255$  之间的整数, 整数之间用空格隔开。

**输出**

输出只有一行, 包含一个整数, 为异常细胞的数目。

**样例输入**

```
1 4
2 70 70 70 70
3 70 10 70 70
4 70 70 20 70
5 70 70 70 70
```

**样例输出**

```
1 2
```

## 1.6 编程题 6：循环移动

<http://oucee.openjudge.cn/a4/6>

**来源** POJ 1684

**注意** 总时间限制: 1000ms 内存限制: 65536kB

**描述** 给定一组整数, 要求利用数组把这组数保存起来, 再利用实现对数组中的数循环移动。假定共有  $n$  个整数, 则要使前面各数顺序向后移  $m$  个位置, 并使最后  $m$  各数变为最前面的  $m$  各数。

注意, 不要用先输出后  $m$  个数, 再输出前  $n - m$  个数的方法实现, 也不要两个数组的方式实现。

要求只用一个数组的方式实现，一定要保证在输出结果时，输出的顺序和数组中数的顺序是一致的。

**输入** 输入有两行：第一行包含一个正整数  $n$  和一个正整数  $m$ ，第二行包含  $n$  个正整数。每两个正整数中间用一个空格分开。

**输出** 输出有一行：经过循环移动后数组中整数的顺序依次输出，每两个整数之间用空格分隔。

**样例输入**

```
1 11 4
2 15 3 76 67 84 87 13 67 45 34 45
```

**样例输出**

```
1 67 45 34 45 15 3 76 67 84 87 13
```

**提示** 这是一道经典的算法问题，在企业面试里出现概率很高。除了循环  $m$  次每次移动一个数以外（这样需要对数组操作  $m * n$  次），你还能想到更高效的算法吗（只用操作  $3 * n$  次）？依然要求不使用额外数组，在原数组上移位之后顺序输出。

## 1.7 编程题 7：中位数

<http://oucee.openjudge.cn/a4/7>

**来源** POJ 3085

**注意** 总时间限制: 2000ms 内存限制: 65536kB

**描述** 中位数定义：一组数据按从小到大的顺序依次排列，处在中间位置的一个数或最中间两个数据的平均值（如果这组数的个数为奇数，则中位数为位于中间位置的那个数；如果这组数的个数为偶数，则中位数是位于中间位置的两个数的平均值）。

给出一组无序整数，求出中位数，如果求最中间两个数的平均数，向下取整即可（不需要使用浮点数）。

**输入** 该程序包含多组测试数据，每一组测试数据的第一行为  $N$ ，代表该组测试数据包含的数据个数， $1 \leq N \leq 15000$ 。

接着  $N$  行为  $N$  个数据的输入， $N = 0$  时结束输入。

**输出** 输出中位数，每一组测试数据输出一行。

**样例输入**

```
1 4
2 10
3 30
4 20
5 40
6 3
7 40
8 30
9 50
10 4
11 1
12 2
13 3
14 4
15 0
```

#### 样例输出

```
1 25
2 40
3 2
```

**提示** 这是一道经典的算法问题，在企业面试里出现概率很高，是“找到第  $K$  大的数”的变种。先排序再找中位数自然是很直接的做法，但排序本身很慢。我们只想找到第  $n/2$  大的数，对于其他数的顺序我们并不关心。那么怎么在不排序的前提下找到第  $n/2$  大的数呢？

## 1.8 编程题 8：校门外的树

<http://oucee.openjudge.cn/a4/8>

**来源** POJ 1810

**注意** 总时间限制: 1000ms 内存限制: 65536kB

**描述** 某校大门外长度为  $L$  的马路上有一排树，每两棵相邻的树之间的间隔都是 1 米。我们可以把马路看成一个数轴，马路的一端在数轴 0 的位置，另一端在  $L$  的位置；数轴上的每个整数点，即  $0, 1, 2, \dots, L$  都种有一棵树。



马路上有一些区域要用来建地铁，这些区域用它们在数轴上的起始点和终止点表示。已知任一区域的起始点和终止点的坐标都是整数，区域之间可能有重合的部分。现在要把这些区域中的树（包括区域端点处的两棵树）移走。你的任务是计算将这些树都移走后，马路上还有多少棵树。

**输入** 输入的第一行有两个整数  $L$  ( $1 \leq L \leq 10000$ ) 和  $M$  ( $1 \leq M \leq 100$ )， $L$  代表马路的长度， $M$  代表区域的数目， $L$  和  $M$  之间用一个空格隔开。接下来的  $M$  行每行包含两个不同的整数，用一个空格隔开，表示一个区域的起始点和终止点的坐标。

**输出** 输出包括一行，这一行只包含一个整数，表示马路上剩余的树的数目。

**样例输入**

```
1 第一组
2 500 3
3 150 300
4 100 200
5 470 471
6 第二组
7 500 3
8 100 200
9 150 160
10 180 190
```

**样例输出**

```
1 第一组
2 298
3 第二组
4 400
```

**提示**

由于数据范围不大 ( $L \leq 10000$ )，我们可以使用一个 10001 长度的数组来记录每一个坐标上有没有树。但想象一下如果数据范围很大，比如下面这个情况，你怎么办呢？

```
1 输入
2 5000000 3
3 1500000 3000000
4 1000000 2000000
5 4700000 4700001
6 输出
7 2999998
```