

OUCEEHLLP 课程任务一

郑海永

目录

1 感性接触计算机程序	1
1.1 抄写题 1：实现冒泡程序	1
1.2 抄写题 2：奇偶排序	4
2 从现实问题到计算机程序	6
2.1 抄写题 3：晶晶赴约会	6
2.2 抄写题 4：奇数求和	8
2.3 抄写题 5：整数的个数	9
2.4 抄写题 6：1 的个数	11
2.5 抄写题 7：数组逆序重放	12

1. 感性接触计算机程序

1.1 抄写题 1：实现冒泡程序

<http://oucee.openjudge.cn/a1/1>

来源

POJ 6383

注意

总时间限制: 1000ms 内存限制: 65536kB

描述

请根据自己的理解编写冒泡排序算法，数组大小 1000 以内。

输入

- 第一行是 n ，表示数组的大小；
- 接着 n 行是数组的 n 个元素。

输出

- 排序之后的结果；

- 一个元素一行。

参考答案

请完全按照如下的程序书写代码，并在书写的过程中体会优秀的代码风格：

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     int n, a[1000]; // 一共 n 个数，n 不超过 1000。a 用来保存这些数
6     cin >> n;
7     // 输入 n 个数
8     for (int i = 0; i < n; i++) {
9         cin >> a[i];
10    }
11    // 冒泡，不断比较相邻的两个数，如果顺序错了，那么就交换
12    for (int i = 0; i < n - 1; i++) {
13        for (int j = 1; j < n - i; j++) {
14            if (a[j - 1] > a[j]) {
15                int temp = a[j];
16                a[j] = a[j - 1];
17                a[j - 1] = temp;
18            }
19        }
20    }
21    // 依次输出
22    for (int i = 0; i < n; i++) {
23        cout << a[i] << endl;
24    }
25    return 0;
26 }
```

冒泡排序的原理：[点击查看](#)

样例输入

```
1 50
2 71 899 272 694 697 296 722 12 726 899 374 541 923 904 83 462 981 929 304 550 59 860
  ↪ 963 516 647 607 590 157 351 753 455 349 79 634 368 992 401 357 478 601 239 365
  ↪ 453 283 432 223 739 487 714 391
```

样例输出

```
1 12
2 59
3 71
4 79
5 83
6 157
7 223
8 239
9 272
10 283
11 296
12 304
13 349
14 351
15 357
16 365
17 368
18 374
19 391
20 401
21 432
22 453
23 455
24 462
25 478
26 487
27 516
28 541
29 550
30 590
31 601
32 607
33 634
34 647
35 694
36 697
37 714
38 722
39 726
40 739
41 753
42 860
43 899
44 899
45 904
46 923
```

47 929
48 963
49 981
50 992

1.2 抄写题 2：奇偶排序

<http://oucee.openjudge.cn/a1/2>

来源

POJ 2165

注意

总时间限制: 1000ms 内存限制: 65536kB

描述

输入十个整数，将十个整数按升序排列输出，并且奇数在前，偶数在后。

输入

输入十个整数。

输出

按照奇偶排序好的十个整数。

提示

先排序，再分别输出奇数、偶数。

参考答案

请完全按照如下的程序书写代码，并在书写的过程中体会优秀的代码风格以及两种方法实现的不同之处：

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     int a[10];
6     for (int i = 0; i < 10; i++) {
7         cin >> a[i];
8     }
9     // 首先，我们把奇数放到数组左边，偶数放到数组右边
10    int l = 0, r = 9; //用左手和右手分别指向数组两端
11    while (l <= r) {
12        bool leftIsOdd = a[l] % 2 == 1;
13        bool rightIsEven = a[r] % 2 == 0;
14        if (leftIsOdd) {
15            l++;
16        } else if (rightIsEven) {
17            r--;
18        } else if (!leftIsOdd && !rightIsEven) {
19            int temp = a[l];
20            a[l] = a[r];
21            a[r] = temp;
22        }
23    }
24    // 对 l 左边（奇数部分）冒泡，不断比较相邻的两个数，如果顺序错了，那么就交换
25    int start = 0, end = 1;
26    for (int i = start; i < end - 1; i++) {
27        for (int j = start + 1; j < start + end - i; j++) {
28            if (a[j - 1] > a[j]) {
29                int temp = a[j];
30                a[j] = a[j - 1];
31                a[j - 1] = temp;
```

```
32     }
33     }
34 }
35 // 对 l 右边 (偶数部分) 冒泡, 不断比较相邻的两个数, 如果顺序错了, 那么就交换
36 start = 1, end = 10;
37 for (int i = start; i < end - 1; i++) {
38     for (int j = start + 1; j < start + end - i; j++) {
39         if (a[j - 1] > a[j]) {
40             int temp = a[j];
41             a[j] = a[j - 1];
42             a[j - 1] = temp;
43         }
44     }
45 }
46 for (int i = 0; i < 10; i++) {
47     cout << a[i] << ' ';
48 }
49 return 0;
50 }
```

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     int a[10];
6     for (int i = 0; i < 10; i++) {
7         cin >> a[i];
8     }
9     // 冒泡, 不断比较相邻的两个数, 如果顺序错了, 那么就交换
10    for (int i = 0; i < 9; i++) {
11        for (int j = 1; j < 10 - i; j++) {
12            // 与刚才的冒泡排序不同, 我们不只是通过较数字的大小决定顺序
13            // 如果左边的为偶数, 右边的为奇数, 那么顺序也需要颠倒
14            bool leftIsEven = a[j - 1] % 2 == 0;
15            bool rightIsEven = a[j] % 2 == 0;
16            if ((leftIsEven && !rightIsEven) ||
17                (leftIsEven == rightIsEven && a[j - 1] > a[j])) {
18                int temp = a[j];
19                a[j] = a[j - 1];
20                a[j - 1] = temp;

```

```
21     }
22 }
23 }
24 for (int i = 0; i < 10; i++) {
25     cout << a[i] << ' ';
26 }
27 return 0;
28 }
```

样例输入

```
1 10 9 8 7 6 5 4 3 2 1
```

样例输出

```
1 1 3 5 7 9 2 4 6 8 10
```

2. 从现实问题到计算机程序

2.1 抄写题 3：晶晶赴约会

<http://oucee.openjudge.cn/a1/3>

来源 POJ 1720

注意 总时间限制: 1000ms 内存限制: 65536kB

描述 晶晶的朋友贝贝约晶晶下周一起去看展览，但晶晶每周的 1、3、5 有课必须上课，请帮晶晶判断她能否接受贝贝的邀请，如果能输出 YES；如果不能则输出 NO。

输入 输入有一行，贝贝邀请晶晶去看展览的日期，用数字 1 到 7 表示从星期一到星期日。

输出 输出有一行，如果晶晶可以接受贝贝的邀请，输出 YES，否则，输出 NO。注意 YES 和 NO 都是大写字母！

样例输入

```
1 第一组
2 1
3 第二组
4 2
5 第三组
6 3
```

样例输出

```
1 第一组
2 NO
3 第二组
4 YES
5 第三组
6 NO
```

参考答案

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int a;
5     cin >> a;
6     if (a == 1 || a == 3 || a == 5)
7         cout << "NO" << endl;
8     else
9         cout << "YES" << endl;
10    return 0;
11 }
12 //这道题比较简单啦，和 A+B 是一个级别的，看懂应该没什么问题。
```

2.2 抄写题 4：奇数求和

<http://oucee.openjudge.cn/a1/4>

来源 POJ 6171

注意 总时间限制: 1000ms 内存限制: 65536kB

描述 计算正整数 m 到 n (包括 m 和 n) 之间的所有奇数的和, 其中, m 不大于 n , 且 n 不大于 300。例如 $m = 3$, $n = 12$, 其和则为: $3 + 5 + 7 + 9 + 11 = 35$ 。

输入 两个数 m 和 n , 两个数以空格间隔, 其中 $0 \leq m \leq n \leq 300$ 。

输出 奇数之和。

样例输入

```
1 第一组
2 7 15
3 第二组
4 0 1
5 第三组
6 3 3
7 第四组
8 100 100
```

样例输出

```
1 第一组
2 55
3 第二组
4 1
5 第三组
6 3
7 第四组
8 0
```

参考答案


```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int m, n, result = 0;
5     cin >> m >> n;
6     while (m <= n) {
7         //对于 m 和 n 之间的每一个数, 如果它是奇数, 那么就加入到我们的结果里。如果不是就跳
           ↳ 过。
8         if (m % 2 == 1)
9             result += m;
10        m++;
11    }
12    //最后输出
13    cout << result << endl;
14    return 0;
15 }
16 //其实还有更快的算法你能想到吗?
```

2.3 抄写题 5：整数的个数

<http://oucee.openjudge.cn/a1/5>

来源 POJ 1678

注意 总时间限制: 1000ms 内存限制: 65536kB

描述 给定 k ($1 < k < 100$) 个正整数, 其中每个数都是大于等于 1, 小于等于 10 的数。写程序计算给定的 k 个正整数中, 1, 5 和 10 出现的次数。

输入 输入有两行: 第一行包含一个正整数 k , 第二行包含 k 个正整数, 每两个正整数用一个空格分开。

输出 输出有三行, 第一行为 1 出现的次数, 第二行为 5 出现的次数, 第三行为 10 出现的次数。

样例输入

```

1 第一组
2 5
3 1 5 8 10 5
4 第二组
5 5
6 2 2 2 2 2

```

样例输出

```

1 第一组
2 1
3 2
4 1
5 第二组
6 0
7 0
8 0

```

参考答案

```

1 #include <iostream>
2 using namespace std;
3 int main(){
4     int k;
5     cin>>k;
6     int n1=0, n5=0, n10=0;
7     for (int i=0;i<k;i++){
8         int n;
9         cin>>n;
10        if (n == 1) n1++;
11        else if (n == 5) n5++;
12        else if (n == 10) n10++;

```

```
13 }
14 cout<<n1<<endl;
15 cout<<n5<<endl;
16 cout<<n10<<endl;
17 return 0;
18 }
```

2.4 抄写题 6:1 的个数

<http://oucee.openjudge.cn/a1/6>

来源 POJ 2709

注意 总时间限制: 1000ms 内存限制: 65536kB

描述 给定一个十进制整数 N , 求其对应 2 进制数中 1 的个数。

输入 第一个整数表示有 N 组测试数据, 其后 N 行是对应的测试数据, 每行为一个整数。

输出 N 行, 每行输出对应一个输入。

样例输入

```
1 5
2 2
3 100
4 1000
5 66
6 0
```

样例输出

```
1 1
2 3
3 6
4 2
5 0
```

提示

- 这道题有一个特点，那就是输入的数据的个数不止一组了。但是我们知道数组的总数量，所以可以用计数循环来读入数据。
- 那么输出应该怎么办呢？在我们的评分系统上，输入和输出是分别放在两个地方处理的，所以可以提早把输出结果打印出来，不会干扰后面的输入。

参考答案

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     int n;
6     cin >> n;
7     for (int i = 0; i < n; i++) {
8         int x, ans = 0;
9         cin >> x;
10        while (x > 0) {
11            ans += x % 2;
12            x /= 2;
13        }
14        cout << ans << endl;
15    }
16    return 0;
17 }
18 //这道题的解法就是反复地除以 2，看最低位是 1 还是 0。有些知道位运算的同学喜欢用位运算，
    ↪ 但其实没必要，因为编译器优化之后的程序其实效率是一样的。
```

2.5 抄写题 7：数组逆序重放

<http://oucee.openjudge.cn/a1/7>

来源 POJ 1689

注意 总时间限制: 1000ms 内存限制: 65536kB

描述 将一个数组中的值按逆序重新存放。例如，原来的顺序为 8,6,5,4,1。要求改为 1,4,5,6,8。

输入 输入为两行：第一行数组中元素的个数 n ($1 < n < 100$)，第二行是 n 个整数，每两个整数之间用空格分隔。

输出 输出为一行：输出逆序后数组的整数，每两个整数之间用空格分隔。

样例输入

```
1 5
2 8 6 5 4 1
```

样例输出

```
1 1 4 5 6 8
```

参考答案

```
1 #include <iostream>
2 using namespace std;
3 int a[100];
4 int main() {
5     int n;
6     cin >> n;
7     for (int i = 0; i < n; i++)
8         cin >> a[i];
9     while (n--) { //常用的倒序计数循环，等价于 while(n-->0)
10        cout << a[n];
11        if (n > 0) cout << " "; //如果不是最后一个数那么就要用空格分隔开
12    }
13    return 0;
14 }
15 //顺序输入倒序输出，但是如果我们要求通过修改数组本身实现逆序重放应该怎么做呢？
```