中国人民大学
RENMIN UNIVERSITY OF CHINA

C程序设计

第5讲 数组

余力

buaayuli@ruc.edu.cn

内容提要

5.1 数组的定义和初始化

5.2 二维数组

5.3 数组的排序问题

5.4 筛法求素数

中国人民大学
RENMIN UNIVERSITY OF CHINA

1. 数组概念定义初始化

数组

- 一组类型相同的顺序存储的数据（变量）
- 数组名、下标、元素
- 方便对一组数据进行命名和访问
  - 数组名+下标 唯一确定数组中的一个元素
  - 通过数组名+下标可以访问数组中的任意元素
- 应用：
  - 对一组数求最值、平均值
  - 对一组数据排序

一维数组的定义

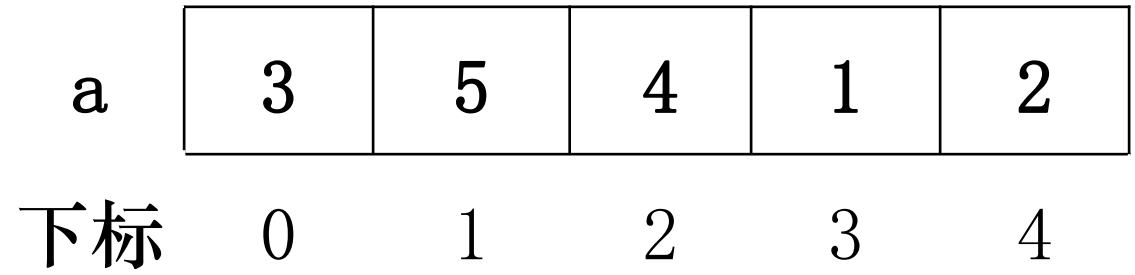
- 定义形式
  - 类型说明符 数组名[常量]
  - 例： float sheep[10]; int a2001[1000];
- 数组的命名规则
  - 数组名的第一个字符应为英文字母；
  - 用方括号将常量表达式括起；
  - 常量表达式定义了数组元素的个数；
  - 数组的下标从0开始，如果定义了5个元素，是从第0个元素到第4个元素
  - 常量表达式中不允许含有变量

一维数组的数组组织方式

int boo[4] (note: 2 bytes per int)

char foo[4] (note: 1-byte char)

数组初始化

- 直接声明时初始化
  - int a[5] = { 3, 5, 4, 1 };
  - 


```
int main() {
    int a[10];
    printf("%d", a[0]);
    printf("%d", a[1]);
    printf("%d", a[11]);
    return 0;
}
```

危险！！！

Dev C++ OK

友学网 不行

Chp05_数组初始化.cpp

数组元素的访问

- 访问一维数组中元素的形式：

  数组名[下标]

- 例如：
  - a[0] = a[1] + a[2] ;
- 其中：
  - 下标写在一个方括号中；
  - 下标是整型表达式，如果为浮点型数据，C截去小数部分，自动取整。
  - 引用时下标不能超界，否则编译程序检查不出错误，但执行时出现不可知结果。

数组与循环

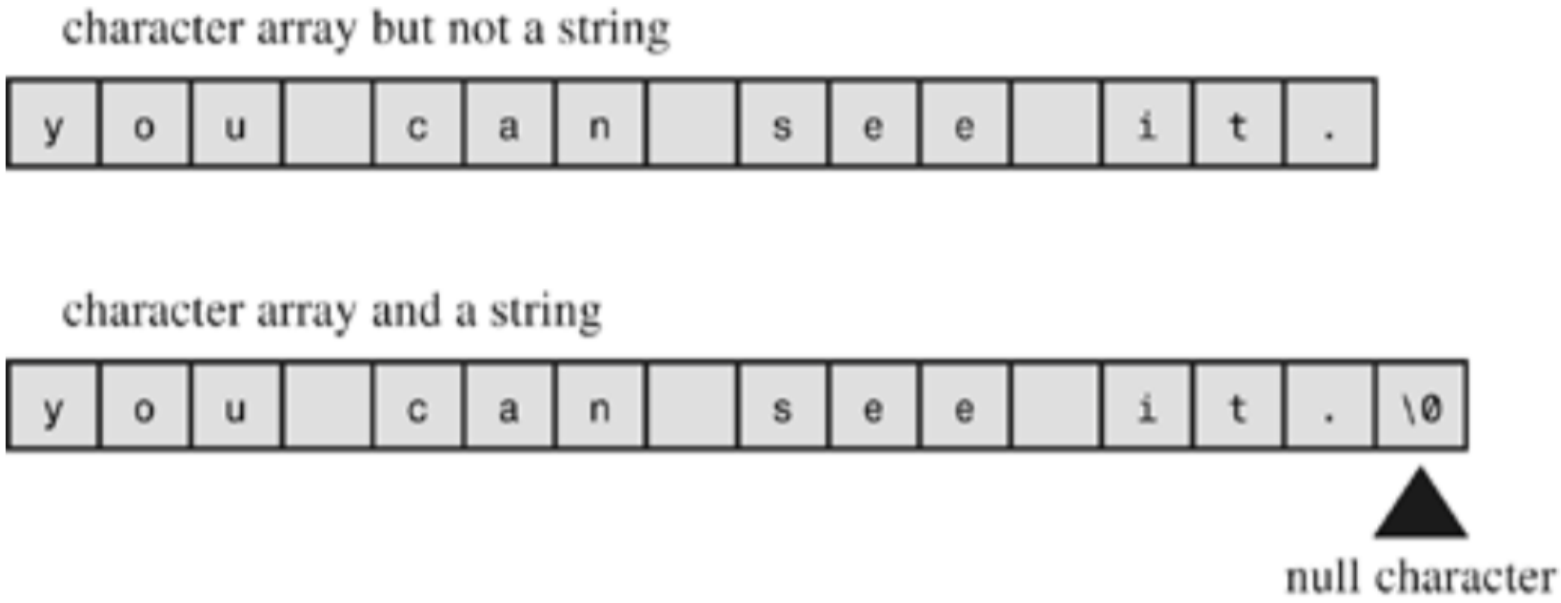
```
for (i=0;i<n;i++)
    scanf("%d", &A[i]);
printf("\n");
```

数组不是一个变量

不能直接操作！！！

```
for (i=0;i<n;i++)
    printf("%d", A[i]);
printf("\n");
```

字符数组与字符串


character array but not a string
y o u c a n s e e i t .
character array and a string
y o u c a n s e e i t . \0
null character

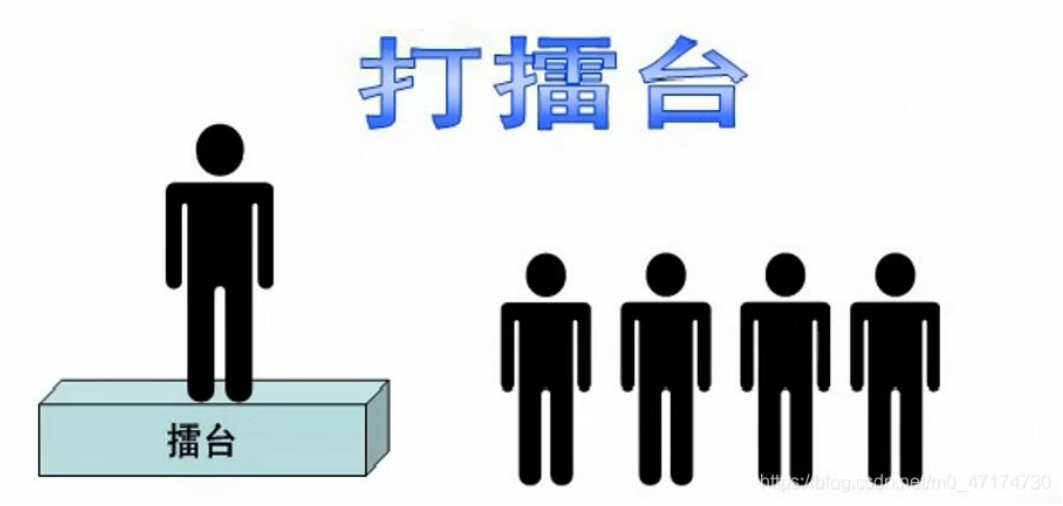
例：输出每月天数

```
#include <stdio.h>
#define MONTHS= 12
int main() {
  int days[MONTHS] = {31,28,31,30,31,30,31,31,30,31,30,31};
  int index;
  for (index = 0; index < MONTHS; index++)
      printf("Month %d has %2d days.\n", index +1, days[index]);
  return 0;
}
```

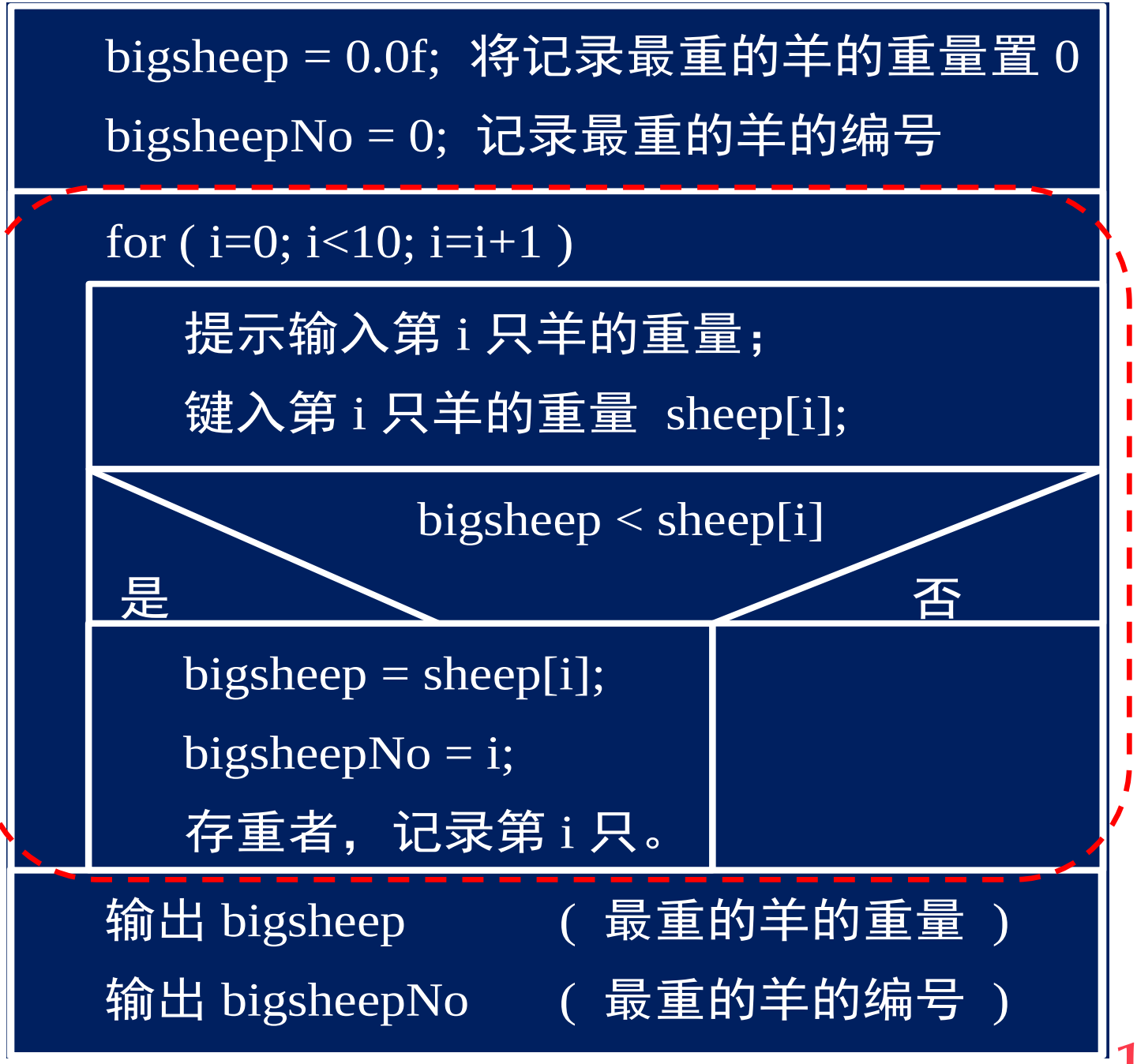
Chp05_输出月天数.cpp

例：哪只羊最重？

- 中秋佳节，有贵客来到草原，主人要从羊群中选一只肥羊宴请宾客，当然要选最重者。
  - 要记录每只羊的重量，如果有成千上万只羊，不可能用一般变量来记录。可以用带有下标的变量，即数组
  - 将羊的重量读入存放到数组中
  - 声明一个变量保存最大的重量，不断更新之



程序框图


bigsheep = 0.0f; 将记录最重的羊的重量置 0
bigsheepNo = 0; 记录最重的羊的编号
for (i=0; i<10; i=i+1)
提示输入第 i 只羊的重量;
键入第 i 只羊的重量 sheep[i];
bigsheep < sheep[i]
是
否
bigsheep = sheep[i];
bigsheepNo = i;
存重者，记录第 i 只。
输出 bigsheep (最重的羊的重量)
输出 bigsheepNo (最重的羊的编号)

```
#include <stdio.h>   // 预编译命令
int main()                              // 主函数
{   float sheep[10] ={0};               // 用于存10只羊每一只的重量
    float  bigsheep=0;                  // 浮点类型变量，存放最肥羊的重量
    int  i=0, bigsheepNo=0;             // 整型变量，i 用于计数循环，
                                        // bigsheepNo用于记录最肥羊的号
    for ( i=0; i<10; i=i+1 )
    {       printf("请输入羊的重量sheep[%d]=", i);
            scanf("%f", &sheep[i]);         // 输入第i只羊的重量
            if ( bigsheep < sheep[i] )      // 如果第i只羊比当前最肥羊大
            {       bigsheep = sheep[i];// 让第i只羊为当前最肥羊
                    bigsheepNo = i;         // 纪录第i只羊的编号
            }
    }                                       // 循环结束
    printf("最肥羊的重量为%f\n", bigsheep);
    printf("最肥羊的编号为%d\n", bigsheepNo);
    return 0;
}
```

Chp05_ 最重肥羊.cpp

中国人民大学
RENMIN UNIVERSITY OF CHINA

2. 二维数组

二维应用场景

- 有n个学生，每个学生学m门课，已知所有学生的各门课的成绩，分别求每门课的平均成绩和每个学生的平均成绩。设各学生成绩如下：

课程 姓名	课程 1	课程 2	课程 3
学生 1	89	78	56
学生 2	88	99	100
学生 3	72	80	61
学生 4	60	70	75

#195 平均成绩排序

有 n 位学生，每位学生修读的科目数不尽相同，已知所有学生的各科成绩，要求按学生平均成绩由高到低输出学生的学号、平均成绩；当平均成绩同时，按学号从低到高排序。对平均成绩，只取小数点后前 2 位，从第 3 位开始舍弃（无需舍入）。

输入格式

□□输入为 n+1 行，第一行为 n 表示学生人数。

□□从第二行开始的 n 行，每行为一名学生的成绩信息，包括：学号、科目数，各科成绩。其中 n、学号、成绩均为整数，它们的值域为：0≤n≤10000，1≤学号≤1000000，0≤成绩≤100。学生的科目数都不超过 100 门。

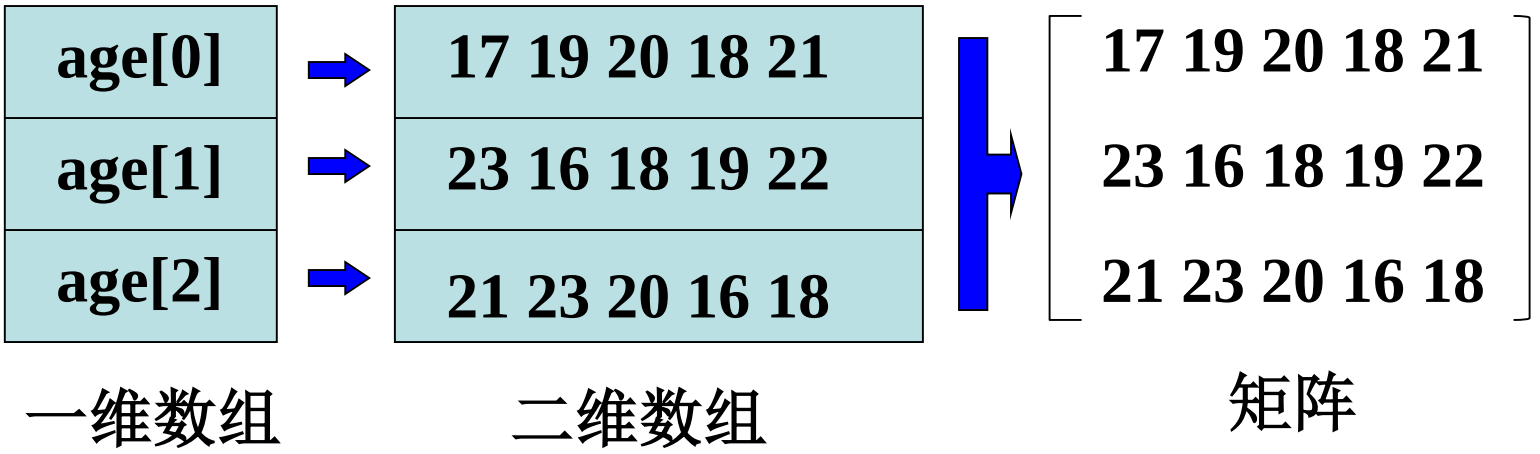
输出格式

□□最多 n 行，每行两个数，学号在前，后为平均成绩，空格分隔。若 n 为 0，输出 NO；若某学生所修科目不到 2 门，则不纳入排序，若无人修满 2 门，也输出 NO。

输入样例	输出样例
5	2003 93.75
1001 2 89 78	1001 83.50
2003 4 88 99 100 88	1004 72.66
4004 3 72 80 66	4004 72.66
1004 3 70 66 82	
3001 1 100	

二维数组的概念及其定义

- 当一维数组的每个元素是一个一维数组时，就构成了二维数组。
- 二维数组与数学中的矩阵概念相对应。



二维数组的定义

- 定义方式
  - 类型标识符 **数组名[常量表达式1] [常量表达式2]**
  - 类型标识符：数组中每个元素的数据类型。可以是C语言中所有的数据类型。
  - 数组名：合法的标识符，数组名就是变量名。
  - 常量表达式1：又称行下标，二维数组中一维数组元素个数。
  - 常量表达式2：又称列下标，表明每个一维数组元素个数。

多维数组的定义

- 二维数组的每一个元素又是相同的类型的一维数，就构成了三维数组，……依此类推，就可构成四维或更高维数组
- 定义一个n维数组：

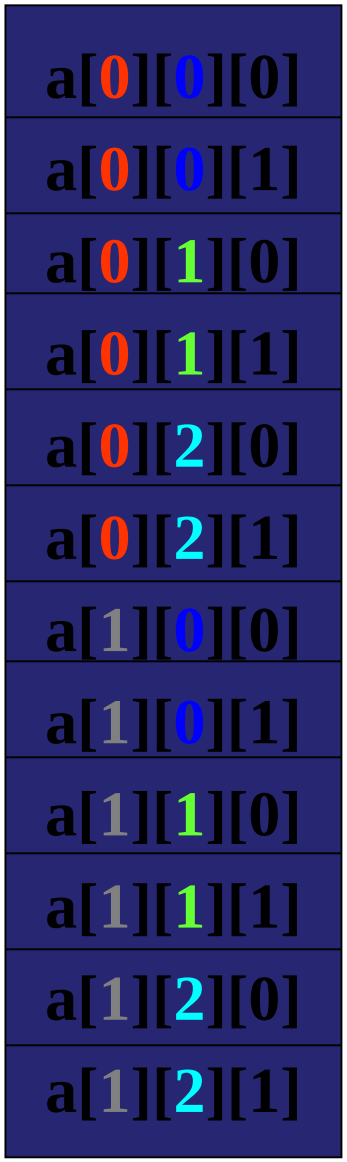
类型标识符 数组名 [常量1][常量2]…[常量n]

三维数组的排列顺序

- 说明一个三维数组：

  int a[2][3][2];

- 12个元素在内存中排列顺序如右图：



2.2 访问二维数组和多维数组

- 访问二维数组中元素的形式：

  数组名[下标][下标]

- 其中：
  - 每一个下标写在一个方括号中；
  - 下标是整型表达式，如果为浮点型数据，C截去小数部分，自动取整。
  - 引用时下标不能超界，否则编译程序检查不出错误，但执行时出现不可知结果。

二维数组和多维数组的初始化

- 二维数组与多维数组的初始化本质上与一维数组初始化相同
  - 例:

    二维数组的初始化

    int a[2][3]={{1,2},{4,5,6}};

    或写成

    int a[2][3]={1,2,4,5,6};

    int a[2][3]={{1,2,4},{5,6}};

矩阵转置

```
#include <stdio.h>
int main()
 {  int a[2][3]={{1,2,3},{4,5,6}};
    int b[3][2],i,j;
    printf("array a:\n");
    for (i=0;i<=1;i++)
    { for (j=0;j<=2;j++)
      {  printf("%5d",a[i][j]);
         b[j][i]=a[i][j];     }
      printf("\n");
     }
    printf("array b:\n");
    for (i=0;i<=2;i++)
    {  for(j=0;j<=1;j++)
          {printf("%5d",b[i][j]);
          printf("\n");}
    }
    return 0;
}
```

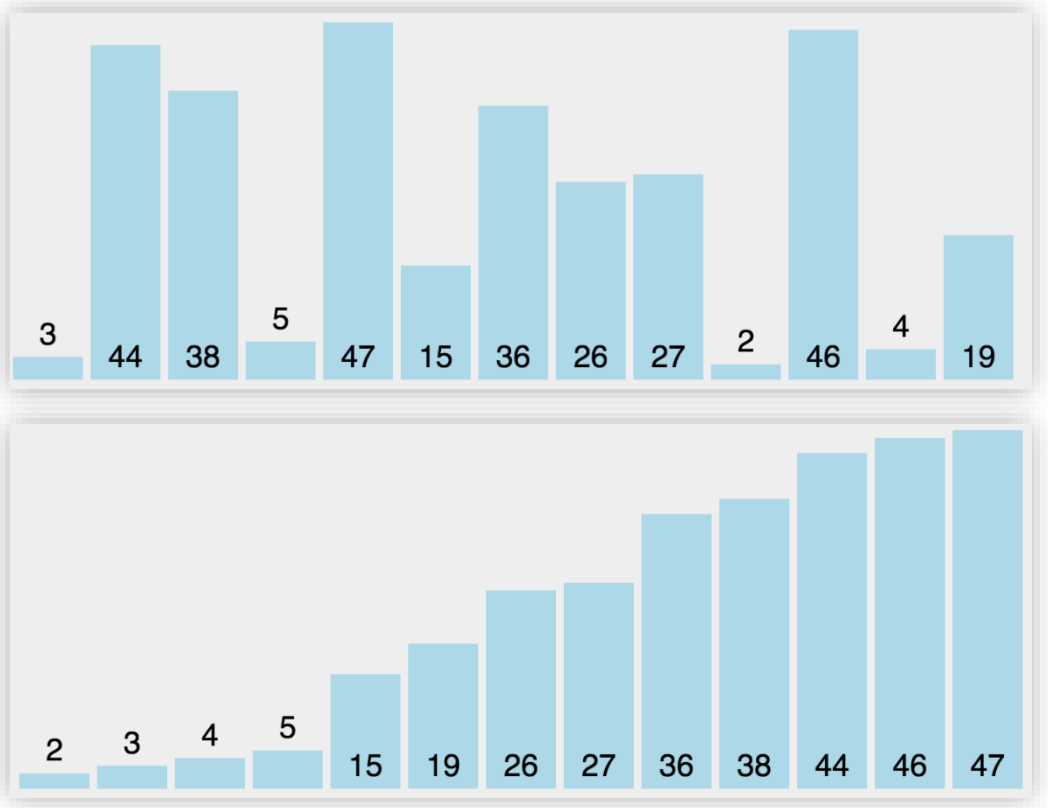
$$a=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix} \Rightarrow b=\begin{bmatrix}1 & 4\\ 2 & 5\\ 3 & 6\end{bmatrix}$$

Chp05_矩阵转置.cpp

3. 数组排序问题

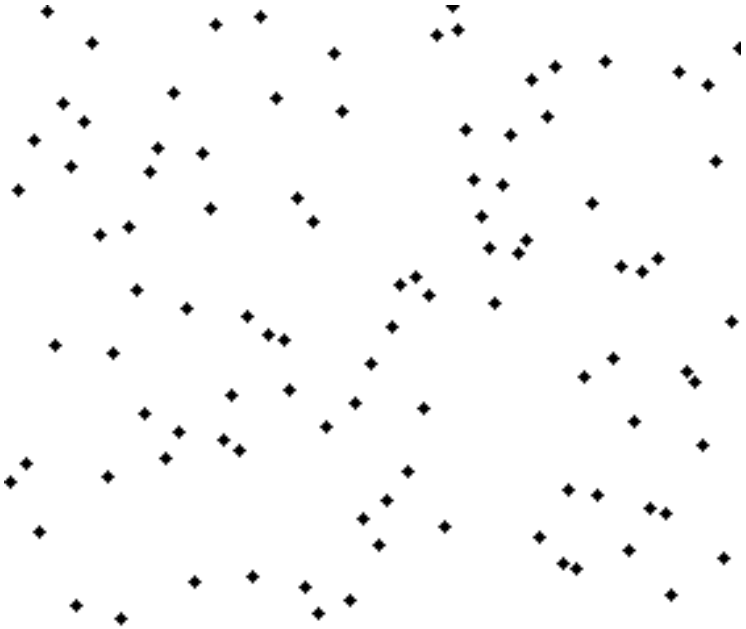
排序问题

- 问题定义
  - 将数组中的元素按照一定顺序重新排列



排序算法 – 冒泡排序 (1)

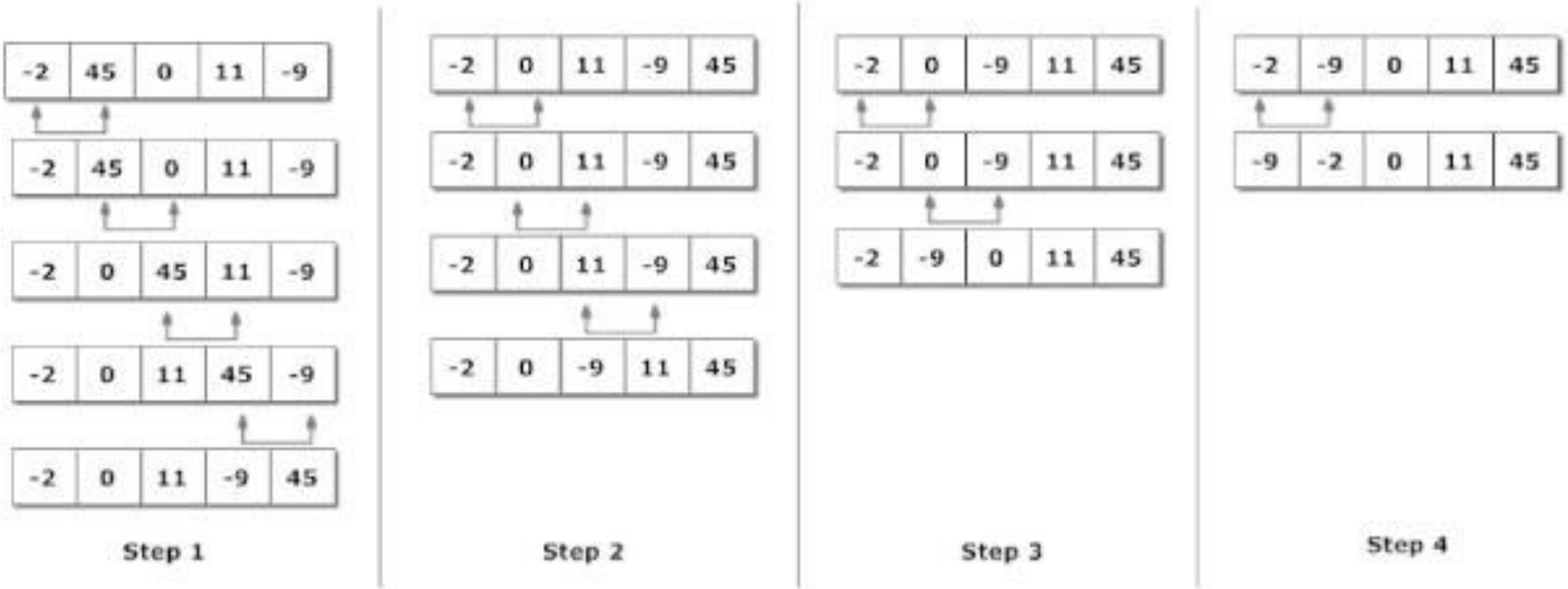
- 基本思路
  - 从头到尾依次访问数组
  - 如果发现顺序错误就交换过来
  - 直到没有再需要交换的元素



算法运行详解

https://visualgo.net/sorting

排序算法 – 冒泡排序 (2)



从小到大

排序算法 – 冒泡排序 (3)

a[0] – a[n-1] 从小到大

```
for (int i = 0; i < n - 1; i++)
    for (int j = 0; j < n - 1 - i; j++)
        if (a[j] > a[j + 1]) {
            int tmp = a[j];
            a[j] = a[j + 1];
            a[j + 1] = tmp;
        }
```

Chp05_冒泡排序.cpp

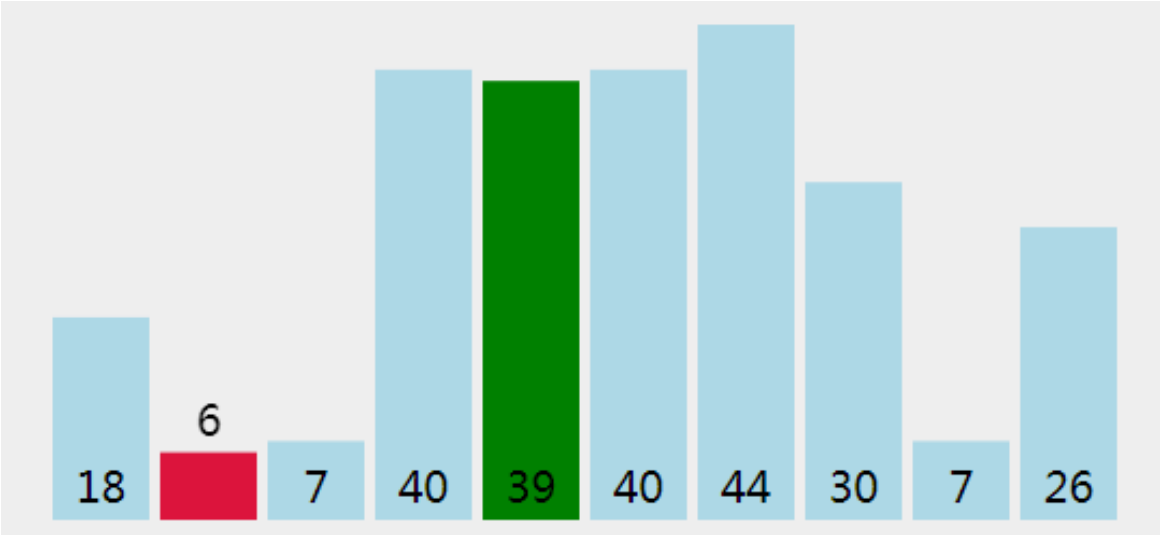
是否一定需要N轮？

冒泡排序 优化

```
for (int i = 0; i < count - 1; i ++) {
    int swap = 0;
    for (int j = 0; j < count - 1 - i; j ++) {
        if (arr[j] > arr[j + 1]) {
            int tmp = arr[j];
            arr[j] = arr[j + 1];
            arr[j + 1] = tmp;
            swap = 1;
        }
    }
    if (swap == 0)
        break;
}
```

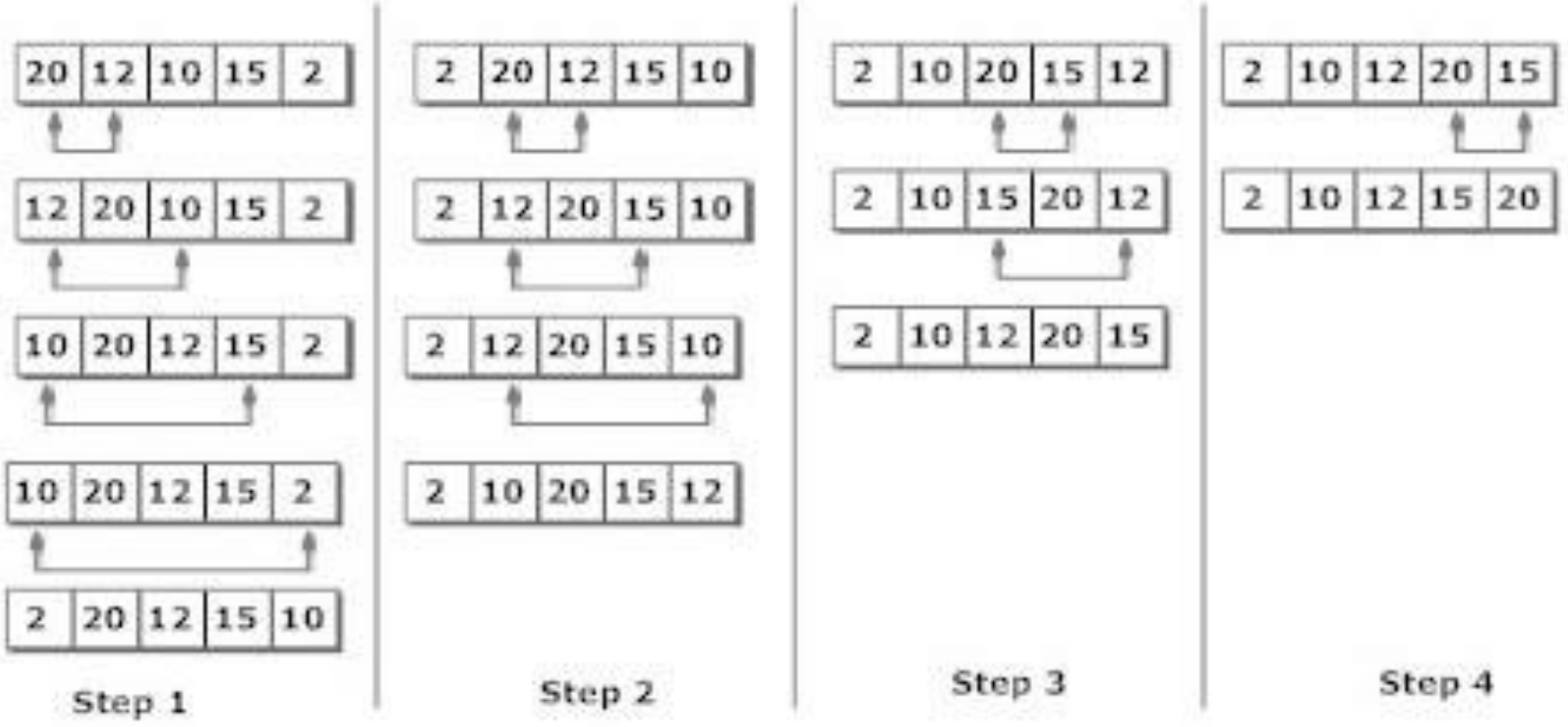
排序算法 – 选择排序 (1)

- 基本思路
  - 把数组分为已排序和未排序两部分
  - 每次遍历未排序部分，选择出其中最小元素，放到已排序部分
  - 直到所有元素都位于已排序部分



算法运行详解 **https://visualgo.net/sorting**

排序算法 – 选择排序 (2)


20 12 10 15 2
12 20 10 15 2
10 20 12 15 2
10 20 12 15 2
2 20 12 15 10
Step 1
2 20 12 15 10
2 12 20 15 10
2 12 20 15 10
2 10 20 15 12
Step 2
2 10 20 15 12
2 10 15 20 12
2 10 12 20 15
Step 3
2 10 12 20 15
2 10 12 15 20
Step 4

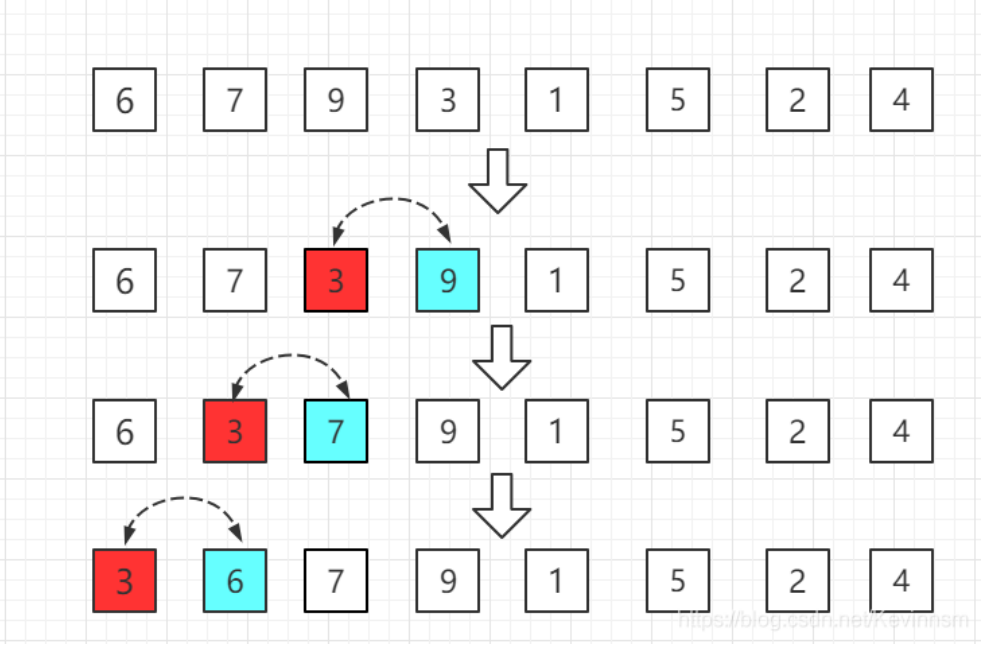
排序算法 – 选择排序 (3)

```
for (int i = 0; i < n - 1; i++) {
    int min_index = i;
    for (int j = i + 1; j < n; j++)
        if (a[min_index] > a[j]) {
            min_index = j;
        }
    if (min_index != i) {
        int tmp = a[min_index];
        a[min_index] = a[i];
        a[i] = tmp;
    }
}
```

Chp05_选择排序.cpp

排序算法 – 插入排序 (1)

- 基本思路
  - 把数组分为已排序和未排序两部分
  - 依次访问未排序元素，将它插入到已排序部分的相应位置
  - 为了支持插入，需要将元素向后移位
  - 直到所有元素都位于已排序部分



算法运行详解

https://visualgo.net/sorting

排序算法 – 插入排序 (2)

```
for (int i = 1; i < n; i++) {
    int key = a[i];
    int index = i;
    for (int j = i - 1; j >= 0; j--)
        if (key < a[j]) {
            a[j + 1] = a[j];
            index = j;
        }
    a[index] = key;
}
```

Key往前移动一步

Chp05_插入排序.cpp

时间复杂度是 O(n^2)

#195 平均成绩排序

有 n 位学生，每位学生修读的科目数不尽相同，已知所有学生的各科成绩，要求按学生平均成绩由高到低输出学生的学号、平均成绩；当平均成绩同时，按学号从低到高排序。对平均成绩，只取小数点后前 2 位，从第 3 位开始舍弃（无需舍入）。

输入格式

□□输入为 n+1 行，第一行为 n 表示学生人数。

□□从第二行开始的 n 行，每行为一名学生的成绩信息，包括：学号、科目数，各科成绩。其中 n、学号、成绩均为整数，它们的值域为：0≤n≤10000，1≤学号≤1000000，0≤成绩≤100。学生的科目数都不超过 100 门。

输出格式

□□最多 n 行，每行两个数，学号在前，后为平均成绩，空格分隔。若 n 为 0，输出 NO；若某学生所修科目不到 2 门，则不纳入排序，若无人修满 2 门，也输出 NO。

输入样例	输出样例
5	2003 93.75
1001 2 89 78	1001 83.50
2003 4 88 99 100 88	1004 72.66
4004 3 72 80 66	4004 72.66
1004 3 70 66 82	
3001 1 100	

```
int main() {
    int id[10000];
    double aver[10000], tmpf;
    int n, i, j, StuNo, KemuNum, tmp, sum, count = 0;

    scanf("%d", &n);
    // 输入 n 位学生信息
    for (i = 0; i < n; i++) {
        scanf("%d %d", &StuNo, &KemuNum);
        for (sum = 0, j = 0; j < KemuNum; j++) {
            scanf("%d", &tmp);
            sum = sum + tmp; }
        if (KemuNum >= 2) {
            id[count] = StuNo;
            aver[count] = sum * 1.0 / KemuNum;
            count++; }
    }
```

```
if (count == 0) {
    printf("NO");
    return 0; }
else {
    for (i = 0; i < count - 1; i++)
        for (j = 0; j < count – i-1; j++)
            if ((fabs(aver[j] - aver[j + 1]) < 1e-7 && id[j] > id[j + 1]) || aver[j] < aver[j + 1]) {
                tmp = id[j]; id[j] = id[j + 1]; id[j + 1] = tmp;
                tmpf = aver[j]; aver[j] = aver[j + 1]; aver[j + 1] = tmpf; }
    for (i = 0; i < count; i++)
        printf("%d %.2lf\n", id[i], int(aver[i] * 100) / 100.0 );
    return 0;
    }
```

Chp05_平均成绩排序(#195).cpp

```
scanf ("%d", &n);
for (i = 0; i < n; i++) {// 输入n个信息
    scanf ("%d %d", &StuNo, &KemuNum);
    for (sum = 0, j = 0; j < KemuNum; j++) {
        scanf ("%d", &tmp);
        sum = sum + tmp;  }
    if ( ** ) { id[count]; aver[count];count++;}
}
```

```
// 输出结果
if ( count  == 0 )
        printf ("NO");
else {
        for (i = 0; i <count-1; i++)
        for (j = 0; j <count-1-i; j++)
                {                       }

        for (i = 0; i < count; i++)
            printf(    );
    }
```

典型

统计排序

程序框架

a[0]---a[n-1] n 个数排序

```
for (i = 0; i < n - 1; i++)
    for (j = 0; j < n - 1 - i; j++)
        if (a[j] < a[j + 1])
            { tmp = a[j]; a[j] = a[j + 1]; a[j + 1] = tmp; }
```

i	0	n-1
j	0	n-1-i

a[1]---a[n] n 个数排序

```
for (i = 1; i < n; i++)
    for (j = 1; j < n - i; j++)
        if (a[j] < a[j + 1])
            { tmp = a[j]; a[j] = a[j + 1]; a[j + 1] = tmp; }
```

多排序依据

```
for (i = 0; i < n - 1; i++)
    for (j = 0; j < n - 1 - i; j++)
        if (aver[j] < aver[j + 1] || (fabs(aver[j] - aver[j + 1]) < 1e-7 && id[j] > id[j + 1] ) {
            tmp = id[j]; id[j] = id[j + 1]; id[j + 1] = tmp;
            tmpf = aver[j]; aver[j] = aver[j + 1]; aver[j + 1] = tmpf; }
```

(aver[j] < aver[j + 1] || (fabs(aver[j] -

aver[j + 1]) < 1e-7 && id[j] > id[j + 1])

类似题目

- #195 平均成绩排序（数组、排序）
- #307 生辰八字（数组、排序）
- #91 相似乐曲
- #308 猪场分配
- #177 膜拜大神
- #225 名次查询（填空题）

#307 生辰八字

大富商杨家有一女儿到了出阁的年纪，杨老爷决定面向全城适龄男士征婚。杨老爷遵循传统，决定按生辰八字作为选女婿的依据。每个应征的男士须提供自己的生辰八字，亦即八个正整数，每个数的取值范围均在[1,24]。杨老爷特意聘请了黄半仙来算命，选择和女儿最契合的前 k 个男士作为候选。黄半仙采用的算命方法是西洋传入的余弦相似度，具体做法如下：

给定两个生辰八字 A = [a_1, a_2,...,a_8]，B=[b_1, b_2,...,b_8]，则

$$Similarity(A,B) = \sum_{i=1}^{8} a_i * b_i / (\sqrt{\sum_{i=1}^{8}(a_i)^2} * \sqrt{\sum_{i=1}^{8}(b_i)^2})$$

例如，若 A = [10,1,1,1,1,1,1,1] ，B = [1,1,1,1,1,1,1,1] ，则 Similarity(A,B) = (10*1+1*1+...1*1)/(sqrt(10^2+1^2+...1^2)+sqrt(1^2+1^2+...+1^2)) = 1.29

黄半仙认为一个男士和小姐两人的生辰八字的余弦相似度越大，两人就越契合。

【输入格式】

第 1 行两个整数，n 和 k，(1 ≤ n ≤ 100, 1 ≤ k ≤n)，表示有 n 个男士应征，以及需要选择 k 人进入候选名单。

第 2 行 8 个整数，表示杨小姐的生辰八字。

后面 n 行，每行 9 个整数，第一个数字是某男士的身份证号，8 位整数；后面 8 个数字是该男士的生辰八字。整数之间均以空格隔开。

【输出格式】

k 个整数，表示契合度最好的前 k 位男士的身份证号，按相似度从高到低排列。

注意：若两个男士和小姐的相似度相同，则身份证号码大的一个排在前面。当相似度相差小于 1e-10 时，认为相同。

#91 相似乐曲

一个音乐搜索引擎，用户输入一首乐曲 Q、一个正整数 k。从音乐库中查找与 Q 最相似的 k 首乐曲。乐曲由一组正整数组成，每个正整数表示声音的频率。计算乐曲相似度的方法是**直方图方法**。下面是详细的计算过程。

第一步，将整个频率的取值区域（[0,255]）均分为 16 个子区域，即[0,15]、[16,31]、……、[240,255]。扫描一首乐曲，计算组成该乐曲的所有频率值出现在每个子区域的次数。例如一首乐曲 M1 由 4 个频率组成 10、12、245、245，则它的直方图就是

[0,15]:2

[16,31]:0

……

[240,255]:2

【输入格式】

- 第 1 行，表示查询乐曲，一个正整数 n0（1≤n0≤100），表示乐曲长度，后面有 n0 个整数，每个整数在[0,255]内，表示一个频率。
- 第 2 行，两个整数 n 和 k，用空格隔开。表示有 n 首乐曲，1≤n≤100，查找最相似的 k(1≤k≤n)首乐曲。
- 第 3 行到 n+2 行，表示编号从 0 到 n-1 的 n 首乐曲。每行一个正整数 ni(1≤ni≤100)，表示该乐曲长度，后面 ni 个整数，每个整数在[0,255]内，表示一个频率。

【输出格式】

输出 k 个整数，与查询乐曲最相似的乐曲的编号，注意乐曲编号范围是[0,n-1]。按相似度从高到低（即：欧式距离从小到大）的顺序输出。若两首乐曲与 Q 的距离相同，则编号小的排名靠前。

#308 猪场分配

老赵经营着一家现代化畜牧养殖企业，在全国各地建有 n 家专业养猪场。但是由于受非洲猪瘟的影响，2019 年损失较为惨重。在国家政策和市场需求的激励下，老赵决定分三个批次购入仔猪，恢复生产。为了杜绝疫病交叉感染的潜在风险，同一个批次的只能放在同一个养殖场。由于不同场地的养殖成本与容量不同，现在他需要考虑如何安排养殖场，以实现最佳的经济效益。假定各批次出栏时，市场预期价格一致，根据输入的养殖场现状信息，编程找出一种总成本最少的猪场分配方案（不是成本最少的所有方案，见输出说明）。

【输入格式】

第 1 行 3 个整数，分别表示三个批次的仔猪数量。

第 2 行 1 个整数，表示老赵经营的养殖场数 n。

第 3 行开始，共 n 行，每行 5 个整数，依次表示：养殖场的编号、运营状态、最大养殖容量、运营基础成本和每头仔猪的出栏养殖成本。其中运营状态用 0、1 表示，1 表示已在运营，不能安排其它批次仔猪进场。比如：112 1 **3000** 5000 300。

【输出格式】

如果找到最少成本的方案，输出两行，第 1 行只 1 个数，为最少成本；第 2 行 3 个数，为对应该成本的分配方案，即依仔猪批次顺序，输出养殖场的编号。这些编号是在所有可能最佳方案中，各批次可能分配的猪场最小编号。

如果找不到，输出 NO。

#177 膜拜大神

在 303 寝室里有一个大神，他的名字叫冯神。冯神是一个很厉害的人物，冯-诺依曼都和他都有某种联系。这一天有 n 个同学想要去拜访他，但是冯神是一个很有规矩的人，他规定，想要去拜访他的人的学号，必须满足条件：学号各个位上的数字之和必须可以被 k 整除。

□□请问想要拜访冯神的 n 个同学里面，有多少个可以去拜访他。

□□注：如需下载本题测试数据，请用 notepad++或者其他高级编辑器打开，否则看不到换行

输入格式

□□包含 n+1 行。

□□第 1 行 2 个整数 n（1≤n≤180）和 k(3≤k≤9)，其中 n 表示有 n 个同学想要去拜访。

□□接下来的 n 行，每行一个整数 a，表示一个学号。输入数据保证学号是信息学院大一新生的正规学号，且不存在相同的学号。

输出格式

□□包含 m+1 行，第 1 行一个整数 m，表示可以去拜访冯神的学生人数；接下来 m 行，每行一个整数，表示可以去膜拜冯神的同学的学号。注意：按照从小大的顺序输出学号。

输入样例

```
3 3
2013202476
2013202477
2013202478
```

输出样例

```
1
2013202476
```

#225 名次查询

信息学院一年级学生期末考试后，需对成绩进行评估排名。排名规则是：1）按成绩从高到低排序；2）同分名次相同；3）名次从 1 开始，某分数所在的名次由超过该分数的总人数决定。e.g. 若超过 90 分的总人数是 30，则 90 分对应名次是 31。请编程完成具体的学号和成绩输入后，查询指定学号的名次。注意，本题中学号以字符串表示，最长不超过 20 个字符，且不会重复。

输入格式

- 共有 n+1 行，第 1 行一个整数 n 和一个学号；**前者表示学生人数 n < 200，后者为待查的学号**，空格分隔，在主函数中完成输入；
- 第 2 行起，每行两个数据，第 1 个为学号（字符串），第 2 个为整型的分数，空格分隔。这 n 行数据在子函数中完成输入。

输出格式

1 个整数，为名次。

输入样例

【输入样例 1】

```
7 112
112 89
```

```
// Query 函数，它本身可再调用其它子函数。
//void Query(int n, char *id, int *prk);
//@{你的代码}

int main(){

    char stu_id[21];        //学生学号，以字符表示
    int rank = 0;           //名次
    int n;                  //总人数

    scanf("%d%s", &n, stu_id);
    Query(n, stu_id, &rank);

    printf("%d ", rank);
    return 0;
}
```

中国人民大学
RENMIN UNIVERSITY OF CHINA

4. 筛法求素数

求素数

```
1  int prime(int x)
2  {
3    for(int i=2;i*i<=x;i++)
4    {
5      if(x%i==0)
6        return 0;
7    }
8    return 1;
9  }
```

筛法思路

1不是素数，除1以外的自然数，当然只有素数与合数。筛法实际上是筛去合数，留下素数。

0	1	2	3	...
2	3	5	7	...
2x2				...
3x2	3x3			
4x2			...	
5x2	5x3	5x5		...
6x2				...
7x2	7x3	7x5	7x7	...
...	...	...	...	...

代码

```
int main() {
    int n, prime[10000];
    scanf("%d", &n);
    prime[1] = 0;
    for (int i = 2; i <= n; i++)
        prime[i] = 1;
    for (int i = 2; i <= n; i++) {
        if (prime[i] == 1)
            for (int j = 2; j * i <= n; j++)
                prime[i * j] = 0;

    }
    for (int i = 2; i <= n; i++)
        if (prime[i])
            printf("%d ", i);
    return 0;
}
```

优化?

Chp05_筛法求素数.cpp

效率的考虑

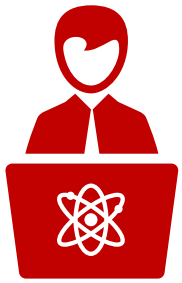
- 令 n 为合数（这里是100），c 为 n 的最小正因数

$$1 < c \leq \sqrt{n}$$

只要找到 c 就可以确认 n 为合数，将其筛去

- **注意：要进行“筛”的1—100的数字是与数组prime[101]的下标相对应的，而每个数组元素的取值只有2个：是0或1，分别代表（标志）与下标相对应的数字是素数或不是素数**

中国人民大学
RENMIN UNIVERSITY OF CHINA



谢谢大家！