数组的基本操作及应用

一、数组的基本操作

1. 查找

【问题描述】

有 10 个同学的学号及成绩如下表所示：

学号	23	12	38	25	9	36	10	16	33	28
成绩	89	96	75	88	69	100	56	98	74	90

要求：输入一个学号，输出该学号同学的成绩。

【分析】

用数组 A 和 B 分别存放同学的学号和成绩；

当输入一个学号后，只要在数组 A 中搜索一下 X 的位置，再输出该位置的数组 B 中的元素即可；

注意，输入的学号可能不存在。

【参考程序】

```
#include <iostream>
#include <cstdio>
using namespace std;

//int a[10+1]={0,23,12,38,25,9,36,10,16,33,28},  b[10+1], i, f, x;
// cin >> n; int a[n+1];

int a[10+1], b[10+1];

int main()
{
   int i, f, x;

   for (i=1; i<=10; i++)      // 读入每位同学的学号及成绩
      cin >> a[i] >> b[i];

   f = 0;            // f 为查找标记
   cin >> x;         // 输入学号: x=16

   for(i=1; i<=10; i++)     // 查找
      if (x==a[i])
      {
         cout << a[i] << "--" << b[i] << endl;
         f = 1;
         break;
```

```
      }

   if (f==0)   cout << "Not found!";
   return 0;
}
//data 23, 89, 12, 96, 38, 75, 25, 88, 9, 69
//data 36, 100, 56, 16, 98, 33, 74, 28, 90
```

2. 删除、移动

【问题描述】

有 10 个大小不同的数，次序凌乱地放在 A 数组中，请找出其中的最大数以及最大数所在的位置，并将该数删除，它后面的元素依次前移。

【参考程序】

```
#include <iostream>
#include <cstdio>
using namespace std;

int main()
{
   int  a[11],  i,  maxv,  t;

   for (i=1; i<=10; i++)  cin >> a[i];

   maxv = a[1];     // 假设第一个元素是最大数
   t = 1;

   for (i=2; i<=10; i++)      // 找最大数
      if (a[i] > maxv)
      {
         maxv = a[i];
         t = i;
      }

   cout << "max=" << maxv << "wei zhi:" << t << endl;

   for (i=t; i<=9; i++)    // 删除最大数后，后面的数依次前移
      a[i] = a[i+1];

   for (i=1; i<=9; i++)
      cout << a[i] << "  ";
   cout << endl;
   return 0;
}
```

3. 插入

【问题描述】

有 10 个数，已按从小到大的顺序排列好，并存放在数组 A 中。从键盘输入一个数 X，将它插入到数组 A 中，使插入元素后的数组仍然能按从小到大的顺序排列。

【参考程序一】{正向插入}

```
#include <iostream>
#include <cstdio>
using namespace std;

int main()
{
   int a[12], i, x, t;

   for (i=1; i<=10; i++)   cin >> a[i];   //scanf("%d", &a[i]);

   cin >> x;   //x=26

   if (x>a[10])
      a[11] = x;
   else
   {
      for (i=1; i<=10; i++)
         if (x<=a[i])
         {
            t = i;
            break;
         }
      for (i=10; i>=t; i--)
         a[i+1] = a[i];

      a[t] = x;
   }

   for (i=1; i<=11; i++)
      cout << a[i]<<' ';
   return 0;
}
//data 4, 6, 12, 34, 42, 51, 58, 67,79,88
```

【参考程序二】{反向插入}

```
#include <iostream>
#include <cstdio>
using namespace std;
```

```
int main()
{
   int a[12], i, x;

   for(i=1; i<=10; i++)  cin >> a[i];

   cin >> x;        // x=26

   for (i=10; i>=1; i--)
      if (x<a[i])
         a[i+1] = a[i];
      else
         break;

   a[i+1] = x;

   for (i=1; i<=11; i++)   cout << a[i] << " ";
   cout << endl;
   return 0;
}
//data 4,6,12,34,42,51,58,67,79,88
```

4. **统计：“学生成绩统计”**

【问题描述】

有 20 个学生的成绩存放在 DATA 语句中，要求编程统计得 100 分的有几个学生?90~99 分、80~89 分、70~79 分、…、 10~19 分、0~9 分各有多少学生?并按以下格式输出结果：

0~ 9:** (**代表人数)
10~19:**
20~29:**
……
90~99:**
100：**

【参考程序】

```
#include <iostream>
#include <cstdio>
#include <cstring>
using namespace std;

int main()
{
   int a[11], i, x;
```

```
   for (i=0; i<=10; i++)   a[i] = 0;   // memset(a, 0, sizeof(a));

   for (i=1; i<=20; i++) {
      cin >> x;
      a[x/10]++;
   }

   for (i=0; i<=9; i++) {           // 输出 0~90 分数段
      printf("%2d", i * 10);
      printf("~");
      printf("%2d", i*10+9);
      printf(":%d\n", a[i]);
   }

   printf("100:%d\n",a[10]);              // 输出 100 分的人数
   return 0;
}
//data 88, 76, 100, 57, 39, 67, 98
//data 100, 45, 97, 26, 8, 60, 85
//data  79, 86, 91, 83, 93, 62
```

二、排序

1. 选择排序

【参考程序】

```
#include <iostream>
#include <cstdio>
#include <cstring>
using namespace std;

int main()
{
   int a[11], i, j;

   for (i=1; i<=10; i++)   cin >> a[i];

   for (i=1; i<=9; i++)       // 一共要比九次
      for (j=i+1; j<=10; j++)
         if ( a[i] > a[j]) swap(a[i],a[j]);

   for (i=1; i<=10; i++)
      cout << a[i] << " ";

   return 0;
```

```
}
//data  25, 27, 31, 28, 29, 30, 29, 28, 31, 36
```

2. 冒泡排序

【参考程序】

```
#include <cstdio>
#include <iostream>
using namespace std;

int a[100], i, f, n, j;

int main()
{
   cin >> n;

   for (i=1; i<=n; i++)  cin >> a[i];

   j = 0;
   do {
      f = 0;
      j++;
      for (i=1; i<=n-j; i++)
         if ( a[i] > a[i+1])
         {
             swap(a[i], a[i+1]);
             f=1;
         }
   } while (f != 0);

   for (i = 1; i<=n; i++)  cout << a[i] << " ";
   return 0;
}
```

3. 计数排序

【问题描述】

输入若干个 100 以内的整数，将它们排成递增序。

【输入】

第 1 行：一个整数 n，表示待排序的数字的个数。

接下来的 n 行：每行一个整数。

【输出】

一行共 n 个整数，为排好序后的数列。

【输入样例】

4
2
1
3
2

【输出样例】

1 2 2 3

【程序清单】

```
#include <cstdio>
#include <iostream>
using namespace std;

int a[101], i, n, x;

int main()
{
   cin >> n;

   for (i = 1; i<=n; i++) {
      cin >> x;    //2 1 3 2
      a[x]++;
   }

   for (i=1; i<=100; i++)
      while (a[i] > 0)
      {
         cout << i << " ";
         a[i]--;
      }

   return 0;
}
```

4. 插入排序

【问题描述】

输入 n 个整数，采用插入排序的方法，将它们排成递增序。

【程序清单】

```
#include <cstdio>
#include <iostream>
using namespace std;

int a[100], i, f, n, x, j;
```

```
int main()
{
   cin >> n;
   cin >> a[1];

   for (i=2; i<=n; i++) {  //从左向右比较、插入
      cin >> x;
      j = 1 ;
      while (x > a[j] && j<=i-1)   j++;
      //循环结束时，下标停在 j 处——x 应该插入在 j 处
      for (k=i-1; k>=j; k--)      //移动后面的元素
         a[k+1] = a[k];

      a[j] = x;
   }

   for(i=1; i<=n; i++)
      cout << a[i] << " ";
   cout << endl;
   return 0;
}
```

5. 合并排序(merge.cpp/c/pas)

【问题描述】

将两个有序数列合并成一个有序数列：假设已有两个已排好次序（由小到大）的数列 A 和 B，要求合并为一个数列 C，使合并后的 C 也按从小到大的次序排列。A、B 数列的元素个数由键盘输入。如：

A 数列：　　5　12　18　72

B 数列：　　1　9　26

则 C 数列为：　1　5　9　12　18　26　72

【参考程序】

```
#include <cstdio>
#include <iostream>
using namespace std;

int main()
{
   int m, n ;

   cout<<"shu ru a, b shu lie de ge shu:";
   cin >> m >> n;
   int a[m+1], b[n+1], c[m+n+1], i, j, k, x;
```

```
  cout << "shu ru a shu lie: "
  for (i=1; i<=m; i++)  cin >> a[i];

  cout << "shu ru b shu lie: " ;
  for (i=1; i<=n; i++)   cin >> b[i];

  i = 1;   j = 1;    k = 1;
  do {
    if (a[i] <= b[j])
    {
        c[k] = a[i];
        i++;
        k++;
    }
    else {
       c[k] = b[j];
       j++;
       k++;
    }
  } while (i <= m && j <= n);

  if (i > m)
    for (x=j; x<=n; x++)
    {
      c[k] = b[x];
      k++;
    }
  else
    for (x=i; x<=m; x++) {
      c[k] = a[x];
      k++;
    }

  for (i=1; i<=m + n; i++)   cout << c[i] << ' ';
  cout << endl;
  return 0;
}
```

三、二分查找

1. 二分查找

【参考程序】

```
#include <cstdio>
#include <iostream>
```

```
using namespace std;

int main()
{
   int  i, f, n, j, x, m, L, r;
   cin >> n;
   int a[n+1];

   for(i=1; i<=n; i++)  cin >> a[i];

   for(i=1; i<=n-1; i++)
      for(j=i+1; j<=n; j++)
         if (a[i] > a[j])  swap(a[i], a[j]);

   L = 1; r = n;
   cin >> x;

   while (L <= r)
   {
      m = (L+r) / 2;
      if (x==a[m])
      {
         cout <<   "Found" << endl;
         return 0;
      }
      if (x < a[m])
         r = m - 1;
      else
         L = m + 1;
   }

   cout << "Not found" << endl;
   return 0;
}
```

输入：3
20
30
10
35

输出：Not found

带二分查找的插入排序

四、**其他应用问题**

1. 灯的开与关问题

【问题描述】

10 个灯排成一排，开始时处于奇数位置的灯是亮的。现进行如下操作：

所有电灯的按钮按动一次；

1~9 号电灯的按钮按动一次；

1~8 号电灯的按钮按动一次；

1~7 号电灯的按钮按动一次；

……

1~2 号电灯的按钮按动一次；

1 号电灯的按钮按动一次；

问：最后哪几只电灯是亮的？(假设：按动一次按钮即对电灯进行一次反操作，即原来亮的变暗，原来暗的变亮)

【参考程序】

```
#include <cstdio>
#include <iostream>
using namespace std;

int main()
{
   int a[11], i, j;

   for (i=1; i<=10; i++)
      if (i%2==1)
         a[i]=1;
      else
         a[i]= -1;

   for (i=10; i>=1; i--)
      for (j=1; j<=i; j++)
          a[j] = a[j] * (-1);

   for (i=1; i<=10; i++)
      if (a[i]==1)  cout << i << " ";

   return 0;
}
```

2. 筛法求素数程序

【参考程序】

```
#include <cstdio>
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
   int n;
   cin >> n;
   int i, j, a[n+1];

   for (i=2; i<=n; i++)   a[i]=1;

   for (i=2; i<=int(sqrt(n)); i++)   //ceil(x),floor(x)
      if (a[i]==1)
          for (j=i+i; j<=n; j+=i) a[j] = 0;

   for (i=2; i<=n; i++)
      if (a[i]==1)  cout << i << " ";

   return 0;
}
```

3. 十进制整数转换成二进制数

【问题描述】

输入一个十进制整数 X(0≤X ≤ 32767)，将它转换成二进制数后输出。

【分析】

十进制整数转换成二进制数，用的是“除以 2 反序取余法”。

因为要将余数倒过来输出，又不知道转换后有多少个二进位，因此，我们用数组将每次除以 2 后的余数存放起来。

最后，将数组按反序输出即可。

【参考程序】

```
#include <cstdio>
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
   int a[32+1];      // why  32?
   int t, p, x, i;
```

```
  t = 0;
  cin >> x;

  if (x==0) {
    cout << 0;
    return 0;
  }

  while (x!=0) {
    p = x%2;        // 求余数
    x /= 2;         // 求整数商
    t++;            // 产生下标
    a[t] = p;       // 余数存入数组 a
  }

  for (i=t; i>=1; i--)     // 将余数按反序输出
    cout << a[i];
  cout << endl;
  return 0;
}
```

4. **明明的随机数**（random.cpp/c/pas）

【问题描述】

明明想在学校中请一些同学一起做一项问卷调查，为了实验的客观性，他先用计算机生成了 N 个 1 到 1000 之间的随机整数（N≤100），对于其中重复的数字，只保留一个，把其余相同的数去掉，不同的数对应着不同的学生的学号。然后再把这些数从小到大排序，按照排好的顺序去找同学做调查。请你协助明明完成“去重”与“排序”的工作。

【输入】

输入文件 random.in 有 2 行，第 1 行为 1 个正整数，表示所生成的随机数的个数：

N

第 2 行有 N 个用空格隔开的正整数，为所产生的随机数。

【输出】

输出文件 random.out 也是 2 行，第 1 行为 1 个正整数 M，表示不相同的随机数的个数。第 2 行为 M 个用空格隔开的正整数，为从小到大排好序的不相同的随机数。

【输入样例】

```
10
20 40 32 67 40 20 89 300 400 15
```

【输出样例】

```
8
15 20 32 40 67 89 300 400
```

5． **集合元素问题**(setelem.cpp/c/pas)

【问题描述】

有这样的一种集合 m ，其中数据的特点如下：

（1） 1∈m;

（2） 假设 x∈m，则有 2x+1∈m，3x+1∈m;

（3） 再无别的元素属于集合 m。

要求：输出集合 m 的第 n 小的数。

【输入】

一个整数 n。

【输出】

一个整数，即题中所要求的、集合 m 中第 n 小的数。

五、二维数组

1． 数字图形打印(print.cpp/c/pas)

【问题描述】

给一个二维数组 A 赋如下数据：

```
 0   1   1   1   1
-1   0   1   1   1
-1  -1   0   1   1
-1  -1  -1   0   1
-1  -1  -1  -1   0
```

【解题分析】

这是一张行与列的数据个数相等的数据表，有 5 行 5 列组成，我们称之为 5×5 的方阵或 5 阶矩阵。

本矩阵以主对角线为分界，主对角线右上方(即上三角)元素为 1，主对角线左下方(即下三角)元素为-1 。

根据观察，可以找出以下规律：

- 主对角线的特征是行下标=列下标；
- 下三角的特征是行下标>列下标；
- 上三角的特征是行下标<列下标。

【参考程序】

```
#include <cstdio>
#include <iostream>
using namespace std;

int main()
{
   int a[6][6], i, j;
```

```
   for (i=1; i<=5; i++)
      for (j=1; j<=5; j++)
          if (i==j)
              a[i][j]= 0;
          else if (i>j)
              a[i][j] = -1;
          else
              a[i][j] = 1;

   for (i=1; i<=5; i++)
   {
      for (j=1; j<=5; j++)
          cout << a[i][j] << " ";  // printf("%d  ", a[i][j]);
      cout << endl;
   }

   return 0;
}
```

2. 二维数组转化成一维数组(convert.cpp/c/pas)

【问题描述】

实际上，二维数组的所有元素在计算机内存中是占一列连续的存储单元的。例如，对于下面的二维数组：

x[0][0]	x[0][1]	x[0][2]
x[1][0]	x[1][1]	x[1][2]
x[2][0]	x[2][1]	x[2][2]

Int x[3][3];

其实，该数组在内存中这样存储的：

1	2	3	4	5	6	7	8	9
x[0][0]	x[0][1]	x[0][2]	x[1][0]	x[1][1]	x[1][2]	x[2][0]	x[2][1]	x[2][2]

二维数组的这种存放顺序称为按行存放。

【分析】

那么，二维数组 x 中的某一个元素(如第二行第三列)应该存放在哪个单元中呢？

把这个问题推广一下：如果一个二维数 t 组有 i 行、j 列(起始行都为 1)，那么怎样计算 t[x][y]的数据存放在哪一个单元中呢？

第 x 行的前面有 x-1 行，共(x-1)*j 个元素；

在第 x 行中，在第 y 列前共有 y-1 个元素；

因此，t[x][y]之前共有元素(x-1)*j+(y-1)个；

如果第一个元素存放在 1 号位置的话，t[x][y]就应该存放在(x-1)*j+(y-1)+1 位

置上；

将一个 4 行 5 列的二维数组 t 转化成一维数组 b 后输出。该二维数组的数据如下：

4	1	5	9	8
2	7	9	5	12
51	25	8	4	10
7	6	21	48	9

程序的结构应分为三个部分：

- 读入该二维数组 t；
- 二维数组 t 转化成一维数组 b；
- 输出一维数组 b。

【参考程序一】

```
#include <cstdio>
#include <iostream>
using namespace std;

int main()
{
   int t[5][6],  b[21], i, j;

   for (i=1; i<=4; i++)
      for (j=1; j<=5; j++)
         cin >> t[i][j];

   for (i=1; i<=4; i++)
      for (j=1; j<=5; j++)
         b[(i-1) * 5 + j-1  +  1]  =  t[i][j];

  for (i=1; i<=20; i++)   cout << b[i]<<' ';
  cout << endl;
  return 0;
}
//data  4, 1, 5, 9, 8, 2, 7, 9, 5, 12
//data 51, 25, 8, 4, 10, 7, 6, 21, 48, 9
```

【参考程序二】

```
#include <cstdio>
#include <iostream>
using namespace std;

int main()
{
   int t[5][6], b[21];
   int  p, i, j;
```

```
   for (i=1; i<=4; i++)
      for (j=1; j<=5; j++)
          cin >> t[i][j];

   p = 0;
   for (i=1; i<=4; i++)
      for (j=1; j<=5; j++) {
          p++;
          b[p] = t[i][j];
      }

     for (i=1; i<=20; i++)   cout << b[i]<<' ';
     cout << endl;
     return 0;
}
```

3. 二维数组的排序问题

【问题描述】

有一个 3 行 4 列的数组 A，将它的元素按从小到大的次序排列，然后将排列好的元素按行的顺序存放以后输出。例如，排序前的数据为：

```
 5  7  3  4
12  1 25  9
 6 19 34  7
```

排序后的数据为：

```
 1  3   4   5
 6  7   7   9
12  19 25  34
```

二维数组的排序可以借助于一维数组来实现。因此，解决该问题可以分三步：

1) 二维数组转一维数组；
2) 一维数组元素排序；
3) 一维数组转回二维数组。

【参考程序】

```
#include <cstdio>
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
   int b[13], a[4][5];
   int i, j, q;
```

```
   for (i=1; i<=3; i++)
      for (j=1; j<=4; j++)
          cin >> a[i][j];

   q = 0;
   for (i=1; i<=3; i++)
      for (j=1; j<=4; j++) {
           q++;
           b[q] = a[i][j];
      }

   for (i=1; i<=q-1; i++)
      for (j=i+1; j<=q; k++)
          if (b[i] > b[j]) swap(b[i],b[j]);

   for (x=1; x<=q; x++) {
      i = (x - 1) / 4;
      j = (x - 1) % 4;
      a[i+1][j+1] = b[x];
   }

   for (i=1; i<=3; i++) {
      for (j=1; j<=4; j++)
          cout << setw(4)<<a[i][j]<< ' ';
      cout << endl;
   }

   return 0;
}
//data 5,7,3,4,12,1,25,9,6,19,34,7
```

4. 鞍点问题

【问题描述】

若一个 M×N 矩阵中的某一项 A(I,J)是第 I 行中的最小值，也是第 J 列中的最大值，则称 A(I,J)为一个鞍点。编程求出一个矩阵的鞍点。例如：

20 18 14 17
9 5 22 8
63 61 41 20

的鞍点为第三行第四列的元素 20。

【分析】

1. 首先必须找到每一行的最小值，并记录下它的列位置 L；
2. 然后，以此列位置为基础，从第一行开始搜索该列中有没有比它大的数；若无，则表示找到了鞍点。

【参考程序】

```
#include <cstdio>
#include <iostream>
using namespace std;

int main()
{
   int m, n;

   cin >> m >> n;
   int  a[m+1][n+1];

   for (i=1; i<=m; i++)
      for (j=1; j<=n; j++)
          cin >> a[i][j];

   for (i=1; i<=m; i++)
   {
      minv = a[i][1]; L=1;
      for (j=2; j<=n; j++)
         if (a[i][j] < minv)
         {
            minv = a[i][j];
            L=j;
         }
      f  = 0;
      for (x=1; x<=m; x++)
         if (a[x][L] > minv)  {
            f = 1;
            break;
         }
      if (f==0)
         cout << "i=" << i << "j=" << L << "鞍点是:" << minv << endl;
   }

   return 0;
}
```

六、循环数组

1. **智得奖品**(award.cpp)

【问题描述】

12 个人围成一圈，希望得到一份奖品。奖品得主按下面的要求产生：从第 1 个人开始，1~3 报数，报到 3 的人走出圈；从他后面的人开始，重新做 1~3 报数，报到 3 的人再走出

圈。如此重复，直到圈中只剩 1 人，他就是获奖者。小红根据规则动了一下脑筋，最后如愿以偿地获得了奖品。请问她开始时站在哪个位置？

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;

int a[13], i, p, k;

int main()
{
   for (i=1; i<=12; i++)  a[i] = 1;

   i = 0;
   do {
      p = 0;
      do {
         i++;
         if (i==13) i=1;
         p += a[i];

      } while (p!=3);
      a[i] = 0;

      k++;
   } while (k<12);

   cout << i << endl;
   return 0;
}
```

2. **狐狸找兔子**(hide.cpp)

【问题描述】

围绕着山顶有 10 个洞，一只兔子和一只狐狸各住一个洞，狐狸总想吃掉兔子。一天，兔子对狐狸说：你想吃我有一个条件，就是第一次隔一个洞找我，第二次隔两个洞找我，以后依此类推，次数不限。若能找到我，你就可以饱餐一顿，在没有找到我之前不能停止。狐狸一想只有 10 个洞，寻找的次数又不限，哪有找不到的道理，就答应了条件。结果就是没找着。现请你编写一个程序，假定狐狸找了 1000 次，兔子躲在哪个洞里才安全。

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;
```

```
int a[11], k, i;

int main()
{
   for (i=1; i<=10; i++)  a[i]=0;

   k=0;
   for (i=1; i<=1000; i++) {
      k =(k+i-1) % 10  +1;
      a[k]=1;
   }

   for (i=1; i<=10; i++)
      if (a[i]==0)  cout << i << "  ";
   cout << endl;

   return 0;      // 2 4 7 9 是安全的
}
```

3. **循环报数**(baoshu.cpp)

【问题描述】

现有 N 个人围成一圈(N 为输入的数据),按 1~M 的间隔报数(M 也是一个输入的数据)。根据报数的结果发现,第一个出列的人是 1 号,第二个出列的人是 2 号,第三个出列的人是 3 号,……,最后一个出列的人是 N 号。问原来这 N 个人的排列位置是怎样的?

【输入】

两个整数 N 和 M,表示人数和报数的间隔。

【输出】

一行共 N 个数,即这 N 个人原来的排列顺序。

【参考程序】

```
#include <iostream>
#include <cstdio>
using namespace std;

int main()
{
   int n, m, i, p, k=0;
   cin >> n >> m;        // n=12, m=3

   int a[n+1];

   for (i=1; i<=n; i++)  a[i] = 0;

   i = 0;
```

```
   do {
      p =0;
      do {
         i++;
         if (i > n) i = i - n;
         if (a[i]==0)  p++;

      } while (p!=m);
      k++;
      a[i]= k;
   } while (k!=n);

   for (i=1; i<=n; i++)  cout << a[i] << " ";
   cout << endl;

   return 0;
}
//7 9 1 5 11 2 8 6 3 12 10 4
```