【输入格式】

4行，每行为一个自然数，分别表示M、S、N、K，值都不超过1000。

【输出格式】

一行包含N个正整数，之间用一个空格隔开，表示先后出圈的数的序列。

【输入样例】

8

1

3

2

【输出样例】

3 1 5 2 7 4 6 8

第7课　二维数组的定义和操作

学习目标

1. 理解二维数组及其存储结构。
2. 掌握二维数组的初始化、输入输出等基本操作。

知识讲解

由前面介绍可知，一维数组的元素可以是任何基本数据类型，也可以是结构体。那么，如果一维数组的每一个元素又是一个一维数组，则称这种数组为“二维数组”。

1. 二维数组的定义和初始化

定义二维数组的一般格式为：

```
类型标识符　数组名［常量表达式1］［常量表达式2］;
```

常量表达式1的值表示第一维大小，常量表达式2的值表示第二维大小，常量表达式1和常量表达式2的乘积就是二维数组的元素个数。例如：

```
int h[4][5];
```

表示数组h有4×5=20个元素，每个元素都是int型。可以把h[0]~h[3]作为一维数组的名字，数组h[0]又有5个元素h[0][0]、h[0][1]、h[0][2]、h[0][3]和h[0][4]。形式上可以把二维数组看作一张表格或一个矩阵。例如，h数组可以被看作下面这个表格：

h[0][0]	h[0][1]	h[0][2]	h[0][3]	h[0][4]
h[1][0]	h[1][1]	h[1][2]	h[1][3]	h[1][4]
h[2][0]	h[2][1]	h[2][2]	h[2][3]	h[2][4]
h[3][0]	h[3][1]	h[3][2]	h[3][3]	h[3][4]

在二维数组定义的同时，可以进行初始化赋值。例如：

```
int a[2][3] = {{1,2,3},{4,5,6}};// 分行初始化
int a[2][3] = {1,2,3,4,5,6};     // 不分行初始化
```

以上两种初始化都相当于下面 6 个语句：

```
a[0][0] = 1; a[0][1] = 2; a[0][2] = 3;
a[1][0] = 4; a[1][1] = 5; a[1][2] = 6;
```

也可以给数组中的部分元素初始化。例如：

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

第一行只有 2 个初值，按顺序分别赋值给 a[0][0]和 a[0][1]，第二行的初值 4 赋给 a[1][0]，其他元素默认为 0。

在定义二维数组时，可以省略第一维的大小，但是第二维的大小不能省略。例如，“int a[ ][5];”是允许的，被省略的第一维大小根据初值的个数由系统来确定。例如：

```
int a[][4] = {1,2,3,4,5,6,7,8,9,10,11,12};
```

系统根据{}中的元素个数，自动确定 a 数组的第一维大小为 3。

2. 二维数组的存储及元素引用

因为二维数组本质上是一维数组的每一个元素又是一个一维数组，而计算机内部存储一维数组采用的是连续存储单元。所以，二维数组的存储方式是“行优先”的连续存储，先逐个存储第 0 行上的所有元素，再逐个存储第 1 行上的所有元素，依此类推。

引用二维数组的某一个元素，格式为：

```
数组名[下标1][下标2]
```

例如：

```
cin >> h[3][1]; h[3][1] = h[3][1] * 2; cout << h[3][1];
```

3. 二维数组的输入输出

二维数组的输入、输出操作也是针对每一个元素进行，结合两个维度的下标变化，用循环嵌套实现。

例1 回型方阵。(hxfz,1s,256MB)

【问题描述】

输入一个正整数 n，输出 n×n 的回型方阵。例如，n=5 时，输出：

1 1 1 1 1
1 2 2 2 1
1 2 3 2 1
1 2 2 2 1
1 1 1 1 1

【输入格式】

一行一个正整数 n，2≤n≤9。

【输出格式】

共 n 行，每行包含 n 个正整数，之间用一个空格隔开。

【输入样例】

5

【输出样例】

1 1 1 1 1
1 2 2 2 1
1 2 3 2 1
1 2 2 2 1
1 1 1 1 1

【问题分析】

定义一个二维数组 a[n][n]存储回型方阵。

方法 1　先给左上角的 a[n/2][n/2]赋值，a[i][j]=min(i,j)，右上角、左下角、右下角三部分，通过下标的对称性复制过去即可。例如，a[i][n+1-j]=a[n+1-i][j]=a[n+1-i][n+1-j]=a[i][j]。

方法 2　通过“一圈一圈”赋值的方法做，先给 a[1][1]~a[n][n]全部赋值 1，然后给 a[2][2]~a[n-1][n-1]全部赋值 2，……共 n/2 圈（如果 n 是奇数，则最后一圈就是一个数）。

```
//p5-7-1a
#include<iostream>
using namespace std;
int n,i,j,k,mi,ma,a[10][10];
int main(){
    cin >> n;
```

```
    for(i = 1; i <= (n+1)/2; i++)
        for(j = 1; j <= (n+1)/2; j++){
            a[i][j] = min(i,j);
            a[i][n+1-j]=a[n+1-i][j]=a[n+1-i][n+1-j]=a[i][j];
        }
    for(i = 1; i <= n; i++){
        for(j = 1; j <= n-1; j++){
            cout << a[i][j] << " ";
        }
        cout << a[i][n] << endl;
    }
    return 0;
}
```

```
//p5-7-1b
#include<iostream>
using namespace std;
int n,i,j,k,a[10][10];
int main(){
    cin >> n;
    for(k = 1; k <= (n+1)/2; k++)
        for(i = k; i <= n+1-k; i++)
            for(j = k; j <= n+1-k; j++)
                a[i][j] = k;
    for(i = 1; i <= n; i++){
        for(j = 1; j < n; j++)
            cout << a[i][j] << " ";
        cout << a[i][n] << endl;
    }
    return 0;
}
```

实践巩固

1. 对于定义“int a[3][4];”,则对 a 数组元素的非法引用是(　　)。

   A. a[0][2*1]　　B. a[1][3]　　C. a[4-2][0]　　D. a[0][4]

2. 对二维数组进行定义,正确的语句是(　　)。

   A. int a[3][ ]　　B. float a[3,2]　　C. double a[3][4]　　D. float a(3)(4)

3. 对二维数组进行初始化,正确的语句是(　　)。

   A. int c[3][ ]={{3},{3},{4}};　　B. int c[ ][3]={{3},{3},{4}};

   C. int c[3][2]={{3},{3},{4},{5}};　　D. int c[ ][3]={{3},{1},{1,2,3,4}};

4. 数字三角形。(numtri,1s,256MB)

【问题描述】

读入一个正整数 n,输出一个 n 行的数字三角形(见输出样例)。

【输入格式】

一行一个正整数 n,2≤n<10。

【输出格式】

共 n 行,第 i 行包含 i 个正整数,每个整数占 5 列。

【输入样例】

5

【输出样例】

```
    1
    2    3
    4    5    6
    7    8    9   10
   11   12   13   14   15
```

第 8 课　二维数组应用举例

学习目标

综合应用二维数组的基本操作解决一些实际问题。

知识讲解

例 1　杨辉三角形。(triangle,1s,256MB)

【问题描述】

输入正整数 n,输出杨辉三角形的前 n 行。例如,n=5 时,杨辉三角形如下:

```
      1
    1   1
  1   2   1
1   3   3   1
1   4   6   4   1
```

【输入格式】

一行一个正整数 n,1≤n≤20。

【输出格式】

共 n 行,第 i 行包含 i 个正整数,之间用一个空格隔开。

【输入样例】

5

【输出样例】

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

【问题分析】

定义一个二维数组 tri 存储杨辉三角形(其实只用到二维数组的左下部分)。对于第 i 行($1 \leqslant i \leqslant n$),共有 i 个数,其中第一个数和最后一个数都是 1,其他数 tri[i][j]=tri[i-1][j-1]+ tri[i-1][j]。具体实现,采用"递推法",逐行逐列给每个数组元素赋值。

```
//p5-8-1
#include<iostream>
#include<cstring>
#include<iomanip>
using namespace std;
int n,i,j,tri[21][21];
int main(){
    cin >> n;
    for(i = 1; i <= n; i++){
        tri[i][1] = 1;
        tri[i][i] = 1;
        for(j = 2; j < i; j++)
            tri[i][j] = tri[i-1][j-1] + tri[i-1][j];
    }
    for(i = 1; i <= n; i++){
        for(j = 1; j < i; j++) cout << setw(6) << tri[i][j] << " ";
        cout << setw(6) << tri[i][i] << endl;
    }
    return 0;
}
```

例 2　数字三角形。(numtri,1s,256MB)

【问题描述】

读入一个正整数 n,输出如下形式的数字三角形(具体见样例)。

【输入格式】

一行一个正整数 n,$1 \leqslant n < 100$。

【输出格式】

共 n 行,第 i 行包含 i 个正整数,每个正整数占 5 列。

【输入样例】

5

【输出样例】

```
1  2  3  4  5
   1  2  3  4
      1  2  3
         1  2
            1
```

【问题分析】

定义二维数组 a 存储所求的数字三角形，初始化为 0。对于右上角的每一个元素 a[i][j]，分析发现：a[i][j]=j-i+1。具体实现采用“赋值法”。

```
//p5-8-2
#include<iostream>
#include<iomanip>
using namespace std;
int a[101][101];
int main(){
    int n,i,j;
    cin >> n;
    for(i = 1; i <= n; i++)
       for(j = i; j <= n; j++) a[i][j] = j - i + 1;
    for(i = 1; i <= n; i++){
       for(j = 1; j <= n; j++)
          if(!a[i][j]) cout << setw(5) << "";
          else cout << setw(5) << a[i][j];
       cout << endl;
    }
    return 0;
}
```

例 3　奖学金。(NOIP2007 普及组复赛，scholar，1s，256MB)

【问题描述】

学校得到了一笔赞助，打算拿出其中一部分为学习成绩优秀的前 5 名学生发奖学金。期末每个学生都有语文、数学、英语 3 门课的成绩。先按总分从高到低排序，如果两个同学总分相同，再按语文成绩从高到低排序；如果两个同学总分和语文成绩都相同，那么规定学号小的同学排在前面。这样，每个学生的排序是唯一确定的。

任务：先根据输入的 3 门课的成绩计算总分，然后按上述规则排序，最后按排名顺序输出前 5 名学生的学号和总分。注意，在前 5 名同学中，每个人的奖学金都不相同，因此必须严格按上述规则排序。例如，在某个正确答案中，如果前两行的输出数据(每行输出两个数：学号、总分)是：

7 279

5 279

这两行数据的含义是:总分最高的两个同学的学号依次是 7 号、5 号。这两名同学的总分都是 279(总分等于输入的语文、数学、英语三科成绩之和),但学号为 7 的学生语文成绩更高一些。如果前两名的输出数据是:

5 279

7 279

则按输出错误处理,不能得分。

【输入格式】

第 1 行为一个正整数 n,表示该校参加评选的学生人数。

第 2~n+1 行,每行有 3 个用一个空格隔开的数字,每个数字都在 0~100 之间。第 j 行的 3 个数字依次表示学号为 j-1 的学生的语文、数学、英语的成绩。每个学生的学号按照输入顺序编号为 1~n(恰好是输入数据的行号减 1)。

所给的数据都是正确的,不必检验。

【输出格式】

输出共有 5 行,每行是两个用一个空格隔开的正整数,依次表示前 5 名学生的学号和总分。

【输入样例】

6

90 67 80

87 66 91

78 89 91

88 99 77

67 89 64

78 89 98

【输出样例】

6 265

4 264

3 258

2 244

1 237

【数据规模】

对于 50% 的数据满足:各学生的总成绩各不相同。

对于 100% 的数据满足:6≤n≤300。

【问题分析】

本题涉及“多关键字”排序。用一个二维数组 stu 保存 n 位同学的学号、语文、数学 、英语、总分成绩,然后以总分成绩为第一关键字(降序)、语文成绩为第二关键字(降序)、学号为第三关键字(升序)排序。最后输出排序后的前 5 个同学的学号和总分。

```
//p5-8-3
#include <iostream>
#include<cstdio>
using namespace std;
int n,i,j,temp,stu[350][5];//stu[i][0]、stu[i][1]、stu[i][2]、stu[i][3]、
```

```
stu[i][4] 分别表示第 i 个同学的学号，语文、数学、英语、总分成绩
int main(){
    cin >> n;
    for(i = 1; i <= n; i++){
        cin >> stu[i][1] >> stu[i][2] >> stu[i][3];
        stu[i][0] = i;
        for(j = 1; j <= 3; j++) stu[i][4] += stu[i][j];
    }
    for(i = 1; i < n; i++)(// 排序
        for(j = i + 1; j <= n; j++)
            if (stu[i][4] < stu[j][4] || stu[i][4] == stu[j][4] &&
                stu[i][1] < stu[j][1] || stu[i][4] == stu[j][4] &&
                stu[i][1] == stu[j][1] && stu[i][0] > stu[j][0]){
                temp = stu[i][0],stu[i][0] = stu[j][0],stu[j][0] = temp;
                temp = stu[i][1],stu[i][1] = stu[j][1],stu[j][1] = temp;
                temp = stu[i][2],stu[i][2] = stu[j][2],stu[j][2] = temp;
                temp = stu[i][3],stu[i][3] = stu[j][3],stu[j][3] = temp;
                temp = stu[i][4], stu[i][4] = stu[j][4],stu[j][4] = temp;
            }
    }
    for(i = 1; i <= 5; i++)
        cout << stu[i][0] << " " << stu[i][4] << endl;
    return 0;
}
```

实践巩固

1. 蛇形数字三角形。(snaketri,1s,256MB)

【问题描述】

输入一个正整数 N,输出 N 行的蛇形数字三角形(具体见样例)。

【输入格式】

一行一个正整数 N,3≤N≤30。

【输出格式】

N 行,第 1 行 N 个数,第 2 行 N-1 个数,……第 N 行一个数。每个数占 5 列。

【输入样例】

9

【输出样例】

```
 1   2   4   7  11  16  22  29  37
 3   5   8  12  17  23  30  38
 6   9  13  18  24  31  39
10  14  19  25  32  40
15  20  26  33  41
```

21 27 34 42
28 35 43
36 44
45

2. 铺地毯。(NOIP2011 提高组复赛,carpet,1s,256MB)

【问题描述】

为了准备一个独特的颁奖典礼,组织者在会场的一片矩形区域(可看作是平面直角坐标系的第一象限)铺上一些矩形地毯。一共有 n 张地毯,编号从 1~n。现在将这些地毯按照编号从小到大的顺序平行于坐标轴先后铺设,后铺的地毯覆盖在前面已经铺好的地毯之上。

地毯铺设完成后,组织者想知道覆盖地面某个点的最上面的那张地毯的编号。注意:在矩形地毯边界和 4 个顶点上的点也算被地毯覆盖。

【输入格式】

第 1 行一个正整数 n,表示总共有 n 张地毯。

接下来的 n 行中,第 i+1 行表示编号 i 的地毯信息,包含 4 个正整数 a、b、g、k,每两个整数之间用一个空格隔开,分别表示铺设地毯的左下角坐标(a,b)以及地毯在 x 轴和 y 轴方向的长度。

第 n+2 行包含两个正整数 x 和 y,表示所求的地面的点的坐标(x,y)。

【输出格式】

输出一行一个整数,表示所求的地毯的编号;若此处没有被地毯覆盖,则输出 -1。

【输入样例】

3
1 0 2 3
0 2 3 3
2 1 3 3
2 2

【输出样例】

3

【样例解释】

如图 5.8-1,1 号地毯用实线表示,2 号地毯用虚线表示,3 号用双实线表示,覆盖点(2,2)的最上面一张地毯是 3 号地毯。

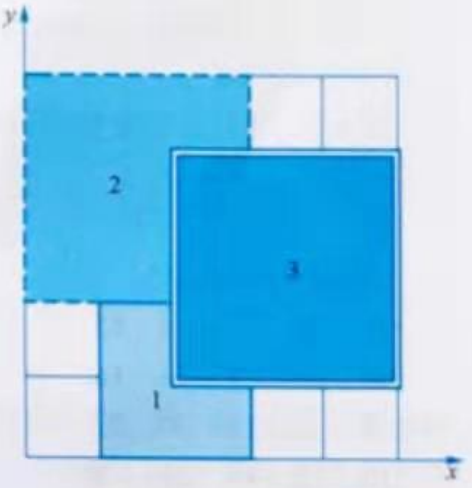

图 5.8-1 铺地毯的样例解释

【数据规模】

对于 30% 的数据满足:n≤2。

对于 50% 的数据满足:0≤a,b,g,k≤100。

对于 100% 的数据满足:0≤n≤10000,0≤a,b,g,k≤100000。