实验舱
青少年编程
走近科学 走进名校

挑战信息学奥林匹克

C++教程（301）
二维数组

二维数组定义和引用

- 定义数组：

类型名 数组名[下标1][下标2]

例：int a[4][6]

[0][0]	[0][1]	[0][2]	[0][3]	[0][4]	[0][5]
[1][0]	[1][1]	[1][2]	[1][3]	[1][4]	[1][5]
[2][0]	[2][1]	[2][2]	[2][3]	[2][4]	[2][5]
[3][0]	[3][1]	[3][2]	[3][3]	[3][4]	[3][5]

- 行下标：0，1，2，3
- 列下标：0，1，2，3，4，5
- 引用：a[i][j]

二维数组读取数据

```
int n, m;
cin >> n >> m;
for ( int i = 1; i <= n; i++ )
    for ( int j = 1; j <= m; j++ )
    {
        cin >> a[i][j];
    }
```

【输入样例】

3 5
64 17 18 51 81
39 21 99 81 54
36 58 42 37 25

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	64	17	18	51	81
2	0	39	21	99	81	54
3	0	36	58	42	37	25

二维数组输出（按行输出）

```
for ( int i = 1; i <= n; i++ )
{
    for ( int j = 1; j <= m; j++ )
    {
        cout << a[i][j] << ' ';
    }
    cout << endl;
}
```

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	64	17	18	51	81
2	0	39	21	99	81	54
3	0	36	58	42	37	25

二维数组输出（按列输出）

```
for ( int i = 1; i <= m; i++ )
{
    for ( int j = 1; j <= n; j++ )
    {
        cout << a[j][i] << ' ';
    }
    cout << endl;
}
```

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	64	17	18	51	81
2	0	39	21	99	81	54
3	0	36	58	42	37	25

	0	1	2	3
0	0	0	0	0
1	0	64	39	36
2	0	17	21	58
3	0	18	99	42
4	0	51	81	37
5	0	81	54	25

二维数组的初始化

- 例：

int a[4][2]={{1,0},{0,1},{-1,0},{0,-1}};

1	0
0	1
-1	0
0	-1

- 给数组赋初值0
  - 方法一

    int a[4][6]={0};
  - 方法二

    memset(a,0,sizeof(a))
  - 方法三

    在main()函数之前定义数组

例2:矩阵交换行

描述
给定一个n*m的矩阵（数学上，一个r×c的矩阵是一个由r行c列元素排列成的矩形阵列），将第x行和第y行交换，输出交换后的结果。
输入
第一行，2个整数n和m，n和m均不大于100。
接下来是n行m列的矩阵，每个元素均是整数。
最后一行是两个整数x和y，即需要交换的行号。（ x<=n，y <=m）
输出
输出交换之后的矩阵，矩阵的每一行元素占一行，元素之间以一个空格分开。

样例输入	样例输出
5 5	3 0 8 2 4
1 2 2 1 2	5 6 7 8 3
5 6 7 8 3	9 3 0 5 3
9 3 0 5 3	7 2 1 4 6
7 2 1 4 6	1 2 2 1 2
3 0 8 2 4	
1 5	

问题分析

1. 用二维数组将全部数据读入。
2. 读取x和y。
3. 一重循环，将x行和y行的元素逐个交换。
4. 输出二维数组的数据。

1	2	2	1	2
5	6	7	8	3
9	3	0	5	3
7	2	1	4	6
3	0	8	2	4

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

问题分析

1. 用二维数组将全部数据读入。
2. 读取x和y。
3. **一重循环，将x行和y行的元素逐个交换。**
4. 输出二维数组的数据。

1	2	2	1	2
5	6	7	8	3
9	3	0	5	3
7	2	1	4	6
3	0	8	2	4

```
cin>>x>>y;
for (int i=1;i<=m;i++)
{
    swap(a[x][i],a[y][i]);
}
```

问题分析

1. 用二维数组将全部数据读入。
2. 读取x和y。
3. 一重循环，将x行和y行的元素逐个交换。
4. 输出二维数组的数据。

1	2	2	1	2
5	6	7	8	3
9	3	0	5	3
7	2	1	4	6
3	0	8	2	4

```
for (int i=1;i<=n;i++)
{
    for (int j=1;j<=m;j++) cout<<a[i][j]<<" ";
    cout<<endl;
}
```

例3:计算矩阵边缘元素之和

描述

输入一个整数矩阵，计算位于矩阵边缘的元素之和。所谓矩阵边缘的元素，就是第一行和最后一行的元素以及第一列和最后一列的元素。

输入

第一行分别为矩阵的行数m和列数n（m < 100，n < 100），两者之间以一个空格分开。

接下来输入的m行数据中，每行包含n个整数，整数之间以一个空格分开。

输出

输出对应矩阵的边缘元素和

样例输入

3 3

3 4 1

3 7 1

2 0 1

样例输出

15

问题分析

- 方法一
  - 枚举二维数组
  - 判断元素是否边缘元素
    - 第1行：i = 1
    - 第n行：i = m
    - 第1列：j = 1
    - 第m列：j = n

3	4	1
3	7	1
2	0	1

```
for (int i=1;i<=m;i++)
    for (int j=1;j<=n;j++)
    {
        if (j==1 || j==n || i==1||i==m) s+=a[i][j];
    }
```

问题分析

- 方法二
  - 用2个一重循环分别枚举第1行、第n行、第1列、第m列

[0][0]	[0][1]	[0][2]	[0][3]	[0][4]	[0][5]
[1][0]	[1][1]	[1][2]	[1][3]	[1][4]	[1][5]
[2][0]	[2][1]	[2][2]	[2][3]	[2][4]	[2][5]
[3][0]	[3][1]	[3][2]	[3][3]	[3][4]	[3][5]

```
for (int i=1;i<=n;i++)
{
    s+=a[1][i];
    s+=a[m][i];
}
```

```
for (int i=2;i<m;i++)
{
    s+=a[i][1];
    s+=a[i][n];
}
```

例4：填数

【描述】

将1到n×n排成一个正方形方阵，用一个小正方形框出m×m个数字，然后求和。例如，将连续自然数1到7×7排成方阵，求出2 2 3即：起点下标是：2 ,2，3*3的一个矩形数据区：9、10、11、16、17、18、23、24、25的和。

【输入说明】

第一行是n，表示以下是n行n列数字矩阵，第二行是数据块起点的下标及m。以上所有数字均为整数。

【输出说明】

一个整数。

【输入样例】

4

3 1 2

【输出样例】

46

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	32	33	34	35
36	37	38	39	40	41	42
43	44	45	46	47	48	49

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

算法分析

1. 二维数组填数
2. 数据块求和

```
cin>>n>>x>>y>>t;
for (int i=1;i<=n;i++)
    for (int j=1;j<=n;j++)
    {
        s++;
        a[i][j]=s;
    };
```

算法分析

1. 二维数组填数
2. 数据块求和

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

```
for (int i=x;i<x+t;i++)
{
    for (int j=y;j<y+t;j++) s+=a[i][j];
}
```

数组清零

题目描述

有一个 $n \times m$ 的矩阵，请你把 0 所在的行和列清零。

```
1 2 3             1 0 3
4 0 6     -->     0 0 0
7 8 9             7 0 9
```

输入描述

第一行两个数, $n(1 \leq n \leq 100)$ 和 $m(1 \leq m \leq 100)$

接下来是一个 $n \times m$ 的矩阵

输出描述

输出清零后的矩阵

输入样例

```
3 3
1 2 3
4 0 6
7 8 9
```

输出样例

```
1 0 3
0 0 0
7 0 9
```

算法分析

5	9	3	4
5	0	7	8
9	10	0	12
13	0	15	16

a

5	0	3	4
0	0	0	0
9	0	0	12
13	0	15	16

b

算法分析

- 定义2个数组a、b，a数组存储原始数据，b数组用来清零操作。
- 算法过程
  1. 读取数据，并将a数组数据赋值给b数组
  2. 枚举a数组，查找等于0的数组元素的位置[i, j]，将b数组中i行j列的元素值置0
  3. 输出b数组

算法分析

1. 读取数据，并将a数组数据赋值给b数组

```
for (int i=1;i<=n;i++)
    for (int j=1;j<=m;j++)
    {
        cin >> a[i][j];
        b[i][j] = a[i][j];
    }
```

算法分析

- 在a数组中查找0，在b数组中清0

```
for (int i=1;i<=n;i++)
    for (int j=1;j<=m;j++)
    {
        if ( a[i][j] == 0 ) clear0(i, j);
    }
```

算法分析

- 清0函数

```
void clear0(int x, int y)
{
    for ( int i = 1; i <= m; i++ ) b[x][i] = 0;
    for ( int i = 1; i <= n; i++ ) b[i][y] = 0;
}
```