第 08 课 二维数组(二)

学习目标

编程知识

掌握：二维数组应用

了解：笛卡尔坐标系*

了解：多维数组*

拓展知识

三维坐标系

矩阵的旋转

重难点解析

场景重现：

你还记得在 C++中二维数组的含义吗，如何去定义一个二维数组吗？

技能小贴士：

在编程中，我们可以使用二维数组去解决矩阵的一些问题。

当然在定义二维数组的时候，我们要确定数组存储的数据类型。

比如下段代码中，定义了一个 5 行 6 列的二维数组 a，行下标 0~4，列下标 0~5。其中 int 是整数类型，a 是二维数组名称。

```
int a[5][6];
```

小码术语箱

对角线的概念：

1. 定义为连接多边形任意两个不相邻顶点的线段。
2. 在二维数组中，从左上至右下的数归为主对角线。
3. 在二维数组中，从左下至右上的数归为副对角线。

【例 1】

小码星球最近进行了一次大测验，看看大家对自己的星球到底有没有很了解，此次测验一共有N个人参加，每个人都有M个测试。现在把他们的成绩都放到一起，要你求出每个人的平均成绩是多少?

输入格式：第一行输入两个整数N、M。接下来N行，每行有M列，表示每个人M个测试的成绩。N，M≤100。

输出格式：输出一行有N个用空格隔开的数，表示每个人的平均成绩。其中每个人的成绩都是正整数，如果结果有小数部分则保留两位小数。

输入样例：

3 4

12 22 32 18

20 24 30 28

18 20 30 24

输出样例：21 25.50 23

【思路分析】:

需要一个二维数组将所有数据保存起来; 因为每个人的成绩都是一行的，所以很容易将每个人的总分加起来; 对于每个人的成绩计算完成后，判断它的平均分是小数还是整数使用模(%) 运算得知，输出对应内容。

【代码】:

```
for (int i = 1; i <= n; i++)  {
    sum = 0;
    for (int j = 1; j <= m; j++) {
        sum += a[i][j];
    }
    if (sum % m == 0) cout << sum / m << " ";
    else printf("%.2f ",(sum * 1.0) / m);
}
```

【例 2】

小码君看着这个 N*N 的矩阵在想，我要是知道这个矩阵对角线上所有数字的和就好了。你能帮助他吗?

输入格式： 输入一个 N，接下来 N 行输入一个 N*N 的矩阵。

输出格式： 输出一个数，表示这个矩阵对角线的和。

输入样例:

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

输出样例：36

【思路分析】:

用一个二维数组存储初始矩阵。

第一条对角线，横坐标和纵坐标相等。

第二条对角线，横坐标加上纵坐标之和为 N+1。

【代码】:

```
int sum = 0;

for(int i = 1;i <= n;i++) {

    for(int j = 1;j <= n;j++) {

        if(i == j || i+j == n+1)

            sum += a[i][j];

    }

}
```

【例 3】

小码君有一个 N*N 的矩阵，现在他想输出中间 M*M 的矩阵，你可以帮他吗?

输入格式： 输入 N，M(N，M 都是偶数)，接下来 N 行输入一个 N*N 的矩阵。

输出格式： 输出中间大小为 M*M 的矩阵。

输入样例:

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

输出样例:

```
2 2 2 2
2 3 3 2
2 3 3 2
2 2 2 2
```

【思路分析】:

首先需要定义一个二维数组用来存储。

由于要取中间 M 行 M 列矩阵，上下一共需要留出(N-M)行，左右一共需要留出(N-M)列。

可以发现上下，左右需要留出的行和列都是一样的，所以上、下、左、右分别留出 (N-M)/2 ，记为 t。

定义循环变量 i，j。i 表示行，j 表示列，循环变量 i 和 j 范围 [t+1,N-t]。

【代码】:

```cpp
#include<iostream>
using namespace std;
const int N = 110;
int a[N][N];
int main()
{
    int m,n;
    cin >> n >> m;
    for(int i = 1;i <= n;i++)
        for(int j = 1;j <= n;j++)
            cin >> a[i][j];
    int t = (n-m)/2;
    for(int i = t+1;i<=n-t;i++){
        for(int j = t+1;j<=n-t;j++){
            cout << a[i][j] <<" ";
        } cout << endl;
    }
    return 0;
}
```

【例 4】

在 n 行 m 列的雷区中有一些格子含有地雷（称之为地雷格），其他格子不含地雷（称之为非地雷格）。玩家翻开一个非地雷格时，该格将会出现一个数字——提示周围格子中有多少个是地雷格。游戏的目标是在不翻出任何地雷格的条件下，找出所有的非地雷格。

现在给出 n 行 m 列的雷区中的地雷分布，要求计算出每个非地雷格周围的地雷格数。

注意：一个格子的周围格子包括其上、下、左、右、左上、右上、左下、右下八个方向上与之直接相邻的格子。

输入样例：	输出样例：
2 3	2*1
?*?	*21
*??	

【思路分析】:

1. 定义字符数组 a 存雷区，整型数组 b 存地雷个数，两个方向数组 x 和 y 存点 (i,j) 周围 8 个方向的坐标。 2. 输入雷区，如果 a[i][j] 等于 *，遍历方向数组，将 (i,j) 周围 8 个格子的地雷个数加 1。3. 最后遍历整个 a 数组，如果 a[i][j] 等于 *，输出 *，否则输出 (i,j) 的地雷个数。

【代码】:

```cpp
char a[105][105];
int b[105][105];
int x[10] = { -1,-1,-1,0,0,1,1,1 };
int y[10] = { -1,0,1,-1,1,-1,0,1 };
int main() {
    int n, m; cin >> n >> m;
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= m; j++) {
            cin >> a[i][j];
            if (a[i][j] == '*') {
                for (int k = 0; k <= 7; k++)
                    b[i + x[k]][j + y[k]]++; }}}
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= m; j++) {
            if (a[i][j] == '*') cout << "*";
            else cout << b[i][j];}
        cout << endl;
    }
    return 0;
}
```

小码挑战状 ——每天挑战一点点，人生将有大不同！

挑　战　状

本人_____

决心通过查找资料，独立完成下列挑战并理解题目。

完成挑战，绝不抄袭。特立此状，以示决心。

计算机病毒是指（　）
A、能传染给用户的磁盘病毒
B、已感染病毒的磁盘
C、具有破坏性的特制程序
D、已感染病毒程序

电子邮件的英文缩写为（　）
A、WWW
B、TELENT
C、E-mail
D、BBS

下列软件中不是计算机操作系统的是（　）
A、Windows
B、Linux
C、MacOS X
D、Dev-C++

鼠标是一种（　）设备
A、输出
B、输入
C、存储
D、运算

监督人：

Tips

- **单词：**

matrix	/ˈmeɪtrɪks/	-矩阵
diagonal	/daɪˈægənl/	-对角线
average	/ˈævərɪdʒ/	-平均
spiral	/ˈspaɪrəl/	-螺旋