第 07 课 二维数组的应用(一)

学习目标

编程知识

掌握：二维数组的应用

掌握：二维字符数组的使用

拓展知识

图像相似度

【例 1】
输入 n 和 m (0 < n, m ≤ 100)，代表矩阵的行和列，然后输入 01 矩阵(即输入 n 行，每行有 m 个数字，每个数字不是 0 就是 1)，统计这个矩阵有多少个 0。
输入格式：
第 1 行输入 n 和 m (0 < n, m ≤ 100) ，代表矩阵的行和列。
然后输入 01 矩阵(即输入 n 行，每行有 m 个数字，每个数字不是 0 就是 1)。
输出格式：
输出这个 01 矩阵有多少个 0。
输入样例：
3 4
1 0 0 0
1 0 1 1
0 1 1 1
输出样例：
5
【思路分析】:
1、定义整型二维数组 a ，长度都设置为 110 用来存储待输入的数据；
2、定义 cnt 变量初值设为 0 ，用于统计 0 的个数；
3、双层循环输入数据，接着遍历数据，使用 if 语句判断当前位置的数据是否为 0 ，满足条件就令 cnt++ ；
4、最后输出 cnt 即可。

【代码】:

```
#include<iostream>
using namespace std;
int main() {
    int n, m, a[110][110], cnt = 0;
    cin >> n >> m;
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= m; j++)
            cin >> a[i][j];
    for (int i = 1; i <= n; i++) {//遍历行
        for (int j = 1; j <= m; j++) {//遍历列
            if (a[i][j] == 0) {//判断是否为 0
                cnt++; //进行计数
            }
        }
     }
     cout << cnt; //输出结果
     return 0;
}
```

【例 2】
输入两个 n 行 m 列的矩阵 A 和 B，输出它们的和 A+B，矩阵加法的规则是两个矩阵中对应位置的值进行加和，具体参照样例。
输入格式：
第一行包含两个整数 n 和 m，表示矩阵的行数和列数(1≤n≤100，1≤m≤100)。
接下来 n 行，每行 m 个整数，表示矩阵 A 的元素。
接下来 n 行，每行 m 个整数，表示矩阵 B 的元素。
相邻两个整数之间用单个空格隔开，每个元素均在 1~1000 之间。
输出格式：
n 行，每行 m 个整数，表示矩阵加法的结果。相邻两个整数之间用单个空格隔开。
输入样例：
3 3
1 2 3
1 2 3
1 2 3
1 2 3
4 5 6
7 8 9
输出样例：
2 4 6
5 7 9
8 10 12
【思路分析】：
1、定义整型二维数组 a 和 b，长度都设置为 110 用来存储待输入的数据；
2、使用双层循环分别录入数组 a 和数组 b；
3、使用双层循环输出数组 a 和数组 b 对应位置的和，注意输出格式，需要加空格和换行。

【代码】:

```
#include<iostream>
using namespace std;
int main() {
    int n, m, a[110][110], b[110][110];
    cin >> n >> m;
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= m; j++)
            cin >> a[i][j];
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= m; j++)
            cin >> b[i][j];
    for (int i = 1; i <= n; i++) {//遍历行
        for (int j = 1; j <= m; j++) {//遍历列
            //输出两个数组对应位置上的数字之和
            cout << a[i][j] + b[i][j] << " ";
        }
         cout << endl;
     }
     return 0;
}
```

【例 3】

A 海域遗留了许多的宝藏，小码君幸运的收集到了 A 海域的地图。现在他准备前往探险， A 海域可以看成是一个 n×m 的格点组成，每个格点有三种状态，分别是 #、S、T，其中 # 是礁石，S 陷阱，T 是宝藏。现在小码君想要确定宝藏和陷阱的位置，给出海域的地图，请帮小码君找出宝藏和陷阱的坐标位置。

输入格式：

输入有 n+1 行。第 1 行输入两个整数 n 和 m（0<n≤100，0<m≤1000），代表 A 海域 的行和列。接下来输入 n 行，每一行有 m 个字符，有可能包括的三种字符： #，S，T。

输出格式：

输出有两行，第一行为宝藏的坐标，第二行为陷阱的坐标，

输出格式为(x,y),两个坐标之间空一个空格。

注意，输出的坐标样式（x,y）起始下标从 0 开始。

输入样例：

3 6

#TT#SS

T###ST

####TS

输出样例：

(0,1) (0,2) (1,0) (1,5) (2,4)

(0,4) (0,5) (1,4) (2,5)

【思路分析】:

1、定义字符二维数组 mp，行列设置为 110 和 1100；

2、使用循环将地图信息存入到数组 mp 中；

3、使用双层循环遍历数组，先输出宝藏所在的位置，编写判断语句，如果当前位置的字符为 T 则按照格式输出该位置的坐标；

4、接着使用双层循环遍历数组，输出陷阱所在的位置，编写判断语句，如果当前位置的字符为 S 则按照格式输出该位置的坐标。

【代码】:

```
char mp[110][1010]; //用二维数组存地图
int n, m;
cin >> n >> m;
for(int i = 0; i < n; i++) cin >> mp[i]; //输入地图
for(int i = 0; i < n; i++){
   for(int j = 0; j < m; j++){
       if(mp[i][j] == 'T'){ //如果是宝藏
          cout << '(' << i << ',' << j << ')'<<" ";
       }
    }
}
cout << endl;
for(int i = 0; i < n; i++){
   for(int j = 0; j < m; j++){
       if(mp[i][j] == 'S'){ //如果是陷阱
          cout << '(' << i << ',' << j << ')'<<" ";
       }
    }
}
```

【例 4】
给出两幅相同大小的黑白图像（用 0-1 矩阵）表示，求它们的相似度。
说明：若两幅图像在相同位置上的像素点颜色相同，则称它们在该位置具有相同的像素点。两幅图像的相似度定义为相同像素点数占总像素点数的百分比。
输入格式：
第一行包含两个整数 m 和 n，表示图像的行数和列数，中间用单个空格隔开。1≤m≤100,1≤n≤100。
之后 m 行，每行 n 个整数 0 或 1，表示第一幅黑白图像上各像素点的颜色。相邻两个数之间用单个空格隔开。
之后 m 行，每行 n 个整数 0 或 1，表示第二幅黑白图像上各像素点的颜色。相邻两个数之间用单个空格隔开。
输出格式：
一个实数，表示相似度（以百分比的形式给出），精确到小数点后两位。
输入样例：
3 3
1 0 1
0 0 1
1 1 0
1 1 0
0 0 1
0 0 1
输出样例：
44.44
【思路分析】:
1、定义整型二维数组 a 和 b，长度都设置为 110 用来存储待输入的数据；
2、定义浮点类型的变量 s 初值设为 0 ,用于计算相似的个数；
3、使用双层循环分别录入数组 a 和数组 b；
4、使用双层循环遍历两个数组的每一位，如果数组 a 和数组 b 对应位置的值相同，则令 s++ ；
5、计算 s/(n*m)*100 的结果并保留两位小数，则为图像相似度

```
int m, n,a[110][110], b[110][110];
double s = 0;
cin >> m >> n;
for (int i = 1; i <= m; i++) {
   for (int j = 1; j <= n; j++) {
       cin >> a[i][j]; //输入第一幅黑白图像
   }
}
for (int i = 1; i <= m; i++) {
    for (int j = 1; j <= n; j++) {
        cin >> b[i][j]; //输入第二幅黑白图像
   }
}
for (int i = 1; i <= m; i++) {
     for (int j = 1; j <= n; j++) {
        if (a[i][j] == b[i][j]) s++;
     }
}
//计算图像相似度
printf("%.2f", s / (n * m) * 100);
```

【例 5】

输入三个自然数 N，i，j（1≤i≤n，1≤j≤n），输出在一个 N×N 格的棋盘中（行列均从 1 开始编号），与格子（i，j）同行、同列、同一对角线的所有格子的位置。

如：n=4，i=2，j=3 表示了棋盘中的第二行第三列的格子，

当 n=4，i=2，j=3 时，输出的结果是：

(2,1)(2,2)(2,3)(2,4) 同一行上格子的位置。

(1,3)(2,3)(3,3)(4,3) 同一列上格子的位置。

(1,2)(2,3)(3,4) 左上到右下对角线上的格子的位置。

(4,1)(3,2)(2,3)(1,4) 左下到右上对角线上的格子的位置。

输入格式：

一行，三个自然数 N，i，j，相邻两个数之间用单个空格隔开（1≤N≤10）。

输出格式：

第一行：从左到右输出同一行格子位置；

第二行：从上到下输出同一列格子位置；

第三行：从左上到右下输出同一对角线格子位置；

第四行：从左下到右上输出同一对角线格子位置。

其中每个格子位置用如下格式输出：(x,y)，x 为行号，y 为列号，

采用英文标点，中间无空格。相邻两个格子位置之间用单个空格隔开。

输入样例：

4 2 3

输出样例：

(2,1) (2,2) (2,3) (2,4)

(1,3) (2,3) (3,3) (4,3)

(1,2) (2,3) (3,4)

(4,1) (3,2) (2,3) (1,4)

【思路分析】：

1、定义变量 n，x，y 分别表示棋盘的行列值以及要计算格子的坐标;

2、先输出同行的坐标，同一行的坐标保持行坐标为 x 不变，列坐标从 1 变化到 n；

3、再输出同列的坐标，同一列的坐标保持列坐标为 y 不变，行坐标从 1 变化到 n；

4、再输出左上到右下对角线的坐标，满足行列坐标的差值相等的条件，即：i-j==x-y；

5、再输出左下到右上对角线的坐标，满足行列坐标的之和相等的条件，即：i+j==x+y 。

【代码】：

```
int n, x, y;
cin >> n >> x >> y;
for (int i = 1; i <= n; i++) {  //同行
   cout << "(" << x << "," << i << ")" << " ";
}
cout << endl;
for (int i = 1; i <= n; i++) { //同列
   cout << "(" << i << "," << y << ")" << " ";
}
cout << endl;
```

```
for (int i = 1; i <= n; i++) {  //左上到右下
   for (int j = 1; j <= n; j++) {
       if (i - j == x - y) {
           cout <<"("<< i <<","<< j << ")" <<" ";
       }
   }
}
cout << endl;
for (int i = n; i >= 1; i--) {  //左下到右上
   for (int j = 1; j <= n; j++) {
       if (i + j == x + y) {
           cout <<"("<< i <<","<< j << ")" <<" ";
       }
   }
}
```

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。

下列软件中是计算机操作系统的是（　）
A、Windows
B、Word
C、PowerPoint
D、Dev-C++

键盘是一种（　）设备
A、输出
B、输入
C、存储
D、运算

监督人：

Tips

- **单词：**

  precious　　/ˈpreʃəs/　　-珍贵的