

6.5 二维数组

任务1: 湖泊测量

测量湖泊的水深，湖中各处的水深是不一样的。如下页图所示。可以给湖面打上格子，测量每个格子处水的深度，就可以从整体上描述湖的情况。

图中的数字0表示地面，非0的数字1，2，3，4，5等表示水的深度，单位为米。每一格的大小为5米X5米（25平方米）。

请你编写程序，计算湖面大小及平均水深。

湖泊的水文情况



二维数组的定义和初始化

类型名 数组名[第一维的元素数目][第二维的元素数目];

或:

类型名 数组名[行数][列数];

例: **int Lake[5][9];**

```
int Lake[5][9] =  
    {{0,0,1,2,2,3,0,0,0},{  
0,2,3,5,5,3,2,0,0},{0,1  
,4,3,4,2,2,1,0},{0,0,1,  
1,0,0,1,1,0},{0,0,0,0,0  
,0,0,0,0}};
```

请注意代码格式

```
int Lake[5][9]= {
```

```
    { 0, 0, 1, 2, 2, 3, 0, 0, 0 },
```

```
    { 0, 2, 3, 5, 5, 3, 2, 0, 0 },
```

```
    { 0, 1, 4, 3, 4, 2, 2, 1, 0 },
```

```
    { 0, 0, 1, 1, 0, 0, 1, 1, 0 },
```

```
    { 0, 0, 0, 0, 0, 0, 0, 0, 0 } };
```

请注意代码格式

0	0	1	2	2	3	0	0	0
0	2	3	5	5	3	2	0	0
0	1	4	3	4	2	2	1	0
0	0	1	1	0	0	1	1	0
0	0	0	0	0	0	0	0	0

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
Lake[0]	0	0	1	2	2	3	0	0	0
Lake[1]	0	2	3	5	5	3	2	0	0
Lake[2]	0	1	4	3	4	2	2	1	0
Lake[3]	0	0	1	1	0	0	1	1	0
Lake[4]	0	0	0	0	0	0	0	0	0

元素访问方法举例：Lake[2][1]，读取或写入该数组单元。
如：cout << Lake[2][1]; cin >> Lake[2][1];

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

```
int main() {
```

```
    // 为节省PPT的空间，此处删除了对Lake数组的定义和初始化
```

```
    int depth_sum = 0, block_num = 0;
```

```
    for (int i=0; i<5; i++) {
```

```
        for (int j=0; j<9; j++) {
```

```
            if (Lake[i][j] > 0) {
```

```
                block_num = block_num + 1;
```

```
                depth_sum = depth_sum + Lake[i][j];
```

```
            }
```

```
        }
```

```
    }
```

```
    cout << "湖面大小为: " << block_num * 25 << endl;
```

```
    cout << "平均水深为: " << depth_sum / block_num << endl;
```

```
    return 0;
```

```
}
```


运行结果

```
d:\20101102>g++ lake.cpp  
d:\20101102>a  
湖面大小为： 525  
平均水深为： 2
```

第一次修改，但仍不正确

```
cout << "湖面大小为: " << block_num * 25 << endl;  
cout << "平均水深为: " << depth_sum / block_num << endl;  
  
float avg;  
avg = depth_sum / block_num;  
cout << "平均水深为: " << avg << endl;
```

```
d:\20101102>g++ lake.cpp
```

```
d:\20101102>a
```

```
湖面大小为: 525
```

```
平均水深为: 2
```

```
平均水深为: 2
```

正确的实现方式

```
cout << "平均水深为: "      // 方法一
    << 1.0 * depth_sum / block_num
    << endl;

cout << "平均水深为: "      // 方法二
    << float(depth_sum) / block_num
    << endl;
```

```
平均水深为: 2.33333
平均水深为: 2.33333
```

任务2：矩阵相乘运算

- 矩阵是代数课程中的重要概念，在各类工程学科中应用极广。矩阵的运算是工程问题中的常见运算。
- 请编程实现任意两个 3×3 大小矩阵的相加和相乘运算。矩阵内容用键盘输入，逐行逐列输入。输出要求排列成 3×3 的矩阵形式。

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

int main() {
    cout << "Input m[][]:" << endl;
    int m[3][3];
    for (int i=0; i<3; i++)
        for (int j=0; j<3; j++)
            cin >> m[i][j];

    cout << "Input n[][]:" << endl;
    int n[3][3];
    for (int i=0; i<3; i++)
        for (int j=0; j<3; j++)
            cin >> n[i][j];
```

```

int r[3][3];
for (int i=0; i<3; i++)
    for (int j=0; j<3; j++)
        r[i][j] = m[i][0] * n[0][j] +
                    m[i][1] * n[1][j] +
                    m[i][2] * n[2][j];

cout << "m[][] * n[][] = " << endl;
for (int i=0; i<3; i++) {
    for (int j=0; j<3; j++)
        cout << r[i][j] << ' ';
    cout << endl;
}

return 0;
}

```

```

d:\20101102>a
Input m[][]:
1 1 1 1 1 1 1 1
Input n[][]:
2 2 2 2 2 2 2 2
m[][] * n[][] =
6 6 6
6 6 6
6 6 6

```

任务 3：姓名排序

- 电视歌手大奖赛开赛报名时，由于人数较多，一些参赛信息需要及时录入计算机并用计算机进行管理。
- 其中一个很重要的工作就是：要按选手姓名（汉语拼音）排序后编号，以决定选手比赛的顺序。
- 请你编程实现对姓名拼音串按英文字典顺序排序的程序。
- 为测试程序，假定共有10名选手，选手姓名拼音最长不超过20个英文字符，且中间无空格。

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

```
int main() {
```

```
return 0;
```

```
}
```




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

int main() {
    char namelist[10][20];    // 这是一个二维字符数组!
    for (int i=0; i<10; i++) {
        cout << i << " singer name: ";
        cin >> namelist[i];
    }

    return 0;
}
```

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

```
int main() {
    char namelist[10][20];    // 这是一个二维字符数组!
    for (int i=0; i<10; i++) {
        cout << i << " singer name: ";
        cin >> namelist[i];
    }
```



```
    for (int i=0; i<10; i++)
        cout << i << ' ' << namelist[i] << endl;
    return 0;
}
```

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

int main() {
    char namelist[10][20];    // 这是一个二维字符数组!
    for (int i=0; i<10; i++) {
        cout << i << " singer name: ";
        cin >> namelist[i];
    }

    // 此处代码省略（见后）—用冒泡排序法对姓名进行排序
    for (int i=0; i<10; i++)
        cout << i << ' ' << namelist[i] << endl;
    return 0;
}
```

```
for (int i=0; i<9; i++) //轮(遍,趟)数=元素数目-1
```

```
{
```

```
    for (int j=0; j<9-i; j++) //比较数=总轮数-当前轮次
```

```
{
```


```
    if (
```

```
{
```

```
}
```

```
}
```

```
}
```

```
for (int i=0; i<9; i++)  
{  
    for (int j=0; j<9-i; j++)  
    {  
        if (strcmp(namelist[j], namelist[j+1]) > 0)  
        {  
  
        }  
    }  
}
```

```
for (int i=0; i<9; i++)  
{  
    for (int j=0; j<9-i; j++)  
    {  
        if (strcmp(namelist[j], namelist[j+1]) > 0)  
        {  
            char tmp[20];  
            strcpy(tmp, namelist[j]);  
            strcpy(namelist[j], namelist[j+1]);  
            strcpy(namelist[j+1], tmp);  
        }  
    }  
}
```

字符串元素的特殊交换算法

结 束