小马加编信息学教案(十二)

C++数组(三)

一. 课程内容

1. 二维数组的定义
2. 二维数组的使用

二. 知识讲解

1. 二维数组的定义

1.1 二维数组定义

> 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呢？我们称这种数组为“二维数组”。

- 二维数组的定义方法：

> 二维数组定义的一般形式是：
> `dataType arrayName[length1][length2];`
> 其中， `dataType` 为数据类型， `arrayName` 为数组名，$length1$ 为第一维下标的长度，$length2$ 为第二维下标的长度。
> 常量表达式 $length1$ 的值表示第一维大小，常量表达式 $length2$ 的值表示第二维大小
> **常量表达式 $length1$ 和常量表达式 $length2$ 的乘积就是二维数组的元素个数**

1.2 二维数组的理解

> 我们可以将二维数组看做一个 $Excel$ 表格，有行有列，$length1$ 表示行数，$length2$ 表示列数
> **要在二维数组中定位某个元素，必须同时指明行和列**。

> 例如：
> `int a[3][4];`
> 定义了一个3行4列的二维数组，共有$3 \times 4 = 12$个元素，数组名为 `a` ，即：
>
> $$\begin{matrix} a[0][0] & a[0][1] & a[0][2] \\ a[1][0] & a[1][1] & a[1][2] \\ a[2][0] & a[2][1] & a[2][2] \end{matrix}$$

> 如果想表示第2行第1列的元素，应该写作 `a[2][1]`

> 二维数组可以看作是由一维数组嵌套而成的；
> 如果一个数组的每个元素又是一个数组，那么它就是二维数组。
> 当然，前提是各个元素的类型必须相同。
> 根据这样的分析，一个二维数组也可以分解为多个一维数组， `C++` 语言允许这种分解。

> 例如，二维数组 `a[3][4]` 可分解为三个一维数组，它们的数组名分别为 `a[0]`、`a[1]`、`a[2]` 。
> 这三个一维数组可以直接拿来使用。这三个一维数组都有 4 个元素，比如，一维数组 `a[0]` 的元素为 `a[0][0]`、`a[0][1]`、`a[0][2]`、`a[0][3]` 。

1.3 二维数组的存储

> 二维数组在概念上是二维的，但在内存中是连续存放的
> 换句话说，二维数组的各个元素是相互挨着的，彼此之间没有缝隙。
> 那么，如何在线性内存中存放二维数组呢?

> 在 `C++` 语言中，**二维数组是按行排列的**。
> 也就是先存放 `a[0]` 行，再存放 `a[1]` 行，最后存放 `a[2]` 行；每行中的 4 个元素也是依次存放。
> 数组 `a` 为 `int` 类型，每个元素占用 4 个字节，**整个数组共占用 $4 \times (3 \times 4) = 48$ 个字节**。

1.4 二维数组的初始化

- 二维数组的初始化**可以按行分段赋值，也可按行连续赋值**。

> 例如，对于数组 `a[5][3]`

> 按行分段赋值应该写作：

```
int a[5][3] = { {80,75,92}, {61,65,71}, {59,63,70}, {85,87,90}, {76,77,85} };
```

> 按行连续赋值应该写作：

```
int a[5][3] = {80, 75, 92, 61, 65, 71, 59, 63, 70, 85, 87, 90, 76, 77, 85};
```

> 这两种赋初值的结果是完全相同的。

- 二维数组初始化注意要点

> (1) **可以只对部分元素赋值，未赋值的元素自动取“零”值**
> 例如:

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

> 是对每一行的第一列元素赋值，未赋值的元素的值为 0。赋值后各元素的值为：

$$\begin{matrix} 1 & 0 & 0 \\ 2 & 0 & 0 \\ 3 & 0 & 0 \end{matrix}$$

> (2) **如果对全部元素赋值，那么第一维的长度可以不给出**
> 例如:

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

> 两种方法是等价的

2. 二维数组的使用

2.1 二维数组元素的引用

> 引用二维数组的某一个元素，格式为：
> `数组名[下标1][下标2]`

- 理解

> **二维数组可以看作是由一维数组嵌套而成的；如果一个数组的每个元素又是一个数组，那么它就是二维数组。**当然，前提是各个元素的类型必须相同。根据这样的分析，**一个二维数组也可以分解为多个一维数组**， C++ 语言允许这种分解。

> 例如，二维数组 `a[3][4]` 可分解为三个一维数组，它们的数组名分别为 `a[0]`、`a[1]`、`a[2]` 。这三个一维数组可以直接拿来使用。这三个一维数组都有 4 个元素，比如，一维数组 `a[0]` 的元素为 `a[0][0]`、`a[0][1]`、`a[0][2]`、`a[0][3]` 。

2.2 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。**

> 例如，输入一个二维数组:

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

- 代码分析

> 上述代码中，通过两重循环来实现读入一个二维数组中的数据
> 外层 `i` 的循环枚举数组的第一维下标
> 内层 `j` 的循环枚举数组的第二维下标
> 根据 `C++` 语言循环结构的运行方式可知，该输入是循环每一行输入，输入完一行的"每一格"之后再继续输入下一行。

三. 经典例题

1. 数字方阵

> 输入一个正整数n，输出一个$n*n$的方阵。
> 该方阵的行编号为$1\sim n$,列编号为$1\sim n$。
> 第 i 行第 j 列的数字应该为$(i-1)*n+j$

> 输入格式
> 一个正整数n。

> 输出格式
> 输出一个满足以上条件的$n*n$的方阵

样例输入	样例输出
3	1 2 3 4 5 6 7 8 9

> 数据范围
> $n \leq 10$

2. 回型方阵

> 输入一个正整数 n，输出 $n \times n$ 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

> 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。

样例输入	样例输出
5	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

> 数据范围
> $2 \leq n \leq 9$

3. 杨辉三角形

> 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

样例输入	样例输出

样例输入	样例输出
5	1 1 1 1 2 1 1 3 3 1 1 4 6 4 1

> 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

> 数据范围
> $1 \leq n \leq 20$

四. 提高巩固

1. 数字三角形(请用二维数组完成)

> 输入一个正整数 n，输出 n 行的数字三角形。其中，第 1 行为数字 1，第 2 行为数字 23，第 3 行为数字 456，第 4 行为数字 7890，第 5 行为数字 $12345...$

> 输入格式
> 一行一个正整数 n。

> 输出格式
> n 行的数字三角形。

样例输入	样例输出
4	1 23 456 7890

> 数据范围
> $1 \leq n \leq 100$

2. 拐角方阵

> 输入一个正整数n,生成一个$n * n$的拐角矩阵
>
> $n = 7$时，拐角矩阵如下

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

输入格式
一个正整数n

输出格式
一个$n*n$的拐角矩阵，共n行，每行n个正整数，之间用一个空格隔开

样例输入	样例输出
7	1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 1 2 3 3 3 3 3 1 2 3 4 4 4 4 1 2 3 4 5 5 5 1 2 3 4 5 6 6 1 2 3 4 5 6 7

3. 轰炸

一个大小为$N*M$的城市遭到了X次轰炸，每次都炸了一个每条边都与边界平行的矩形。
在轰炸后，有Y个关键点，指挥官想知道，它们有没有受到过轰炸，如果有，被炸了几次，最后一次是第几轮。

输入格式
第一行，四个整数：n、m、x、y。
以下x行，每行四个整数：$x1$、$y1$、$x2$、$y2$，
表示被轰炸的矩形的左上角坐标和右下角坐标（比如13710就表示被轰炸的地方是从$(1,3)$到$(7,10)$的矩形）
再以下y行，每行两个整数，表示这个关键点的坐标。

输出格式
共y行，
每行第一个字符为Y或N，表示是否被轰炸，若为Y，在一个空格后为两个整数，表示被炸了几次和最后一次是第几轮。

输入样例	输出样例

输入样例	输出样例
10 10 2 3 1 1 5 5 5 5 10 10 3 2 5 5 7 1	Y 1 1 Y 2 2 N

> 数据范围
> $1 <= N, M <= 100$