

11 - 多维数组

C++ 程序设计基础

SOJ 信息学竞赛教练组

2024 年 6 月 5 日

1 复习回顾

2 二维数组

3 二维数组的遍历

4 二维数组的部分遍历

5 多维数组

6 总结

问题回顾 - 倒序输出数组元素

编程题

- 编写程序，输入一个整数 n ($1 \leq n \leq 100$)，表示有 n 个整数，接下来输入 n 个整数存储在数组中，要求倒序输出数组元素。

- 样例输入

6

1 4 2 8 5 7

- 样例输出

7 5 8 2 4 1

问题回顾 - 倒序输出数组元素

```
1 #include <iostream>
2
3 using namespace std;
4
5 int a[105];
6
7 int main() {
8     int n;
9     cin >> n;
10    // 按照下标 1 ~ n 的顺序输入数组元素
11    for (int i = 1; i <= n; i++) {
12        cin >> a[i];
13    }
14    // 按照下标 n ~ 1 的顺序输出数组元素
15    for (int i = n; i >= 1; i--) {
16        cout << a[i] << " ";
17    }
18    cout << endl;
19
20    return 0;
21 }
```

目录

1 复习回顾

2 二维数组

3 二维数组的遍历

4 二维数组的部分遍历

5 多维数组

6 总结

- 现实生活中有很多事物是以“二维”的方式组织起来的：

- 现实生活中有很多事物是以“二维”的方式组织起来的：
 - 电影院的每个座位会被编排为“第 i 排第 j 号”，其中 i 是第一维的编号， j 是第二维的编号。

- 现实生活中有很多事物是以“二维”的方式组织起来的：
 - 电影院的每个座位会被编排为“第 i 排第 j 号”，其中 i 是第一维的编号， j 是第二维的编号。
 - 你所在的班级往往编号为 x 年级 y 班，其中 x 可以看作是第一维的编号， y 可以看作是第二维的编号。

- 现实生活中有很多事物是以“二维”的方式组织起来的：
 - 电影院的每个座位会被编排为“第 i 排第 j 号”，其中 i 是第一维的编号， j 是第二维的编号。
 - 你所在的班级往往编号为 x 年级 y 班，其中 x 可以看作是第一维的编号， y 可以看作是第二维的编号。
- 程序是对现实世界的抽象，所以 C++ 提供了二维数组，让我们可以表示这类二维编号。

二维数组的声明

- 二维数组的声明
 - 元素类型 数组名 [第一维大小][第二维大小];

二维数组的声明

- 二维数组的声明
 - 元素类型 数组名 [第一维大小][第二维大小];
 - 数组的第一维通常称为“行”
 - 数组的第二维通常称为“列”

二维数组的声明

- 二维数组的声明
 - 元素类型 数组名 [第一维大小][第二维大小];
 - 数组的第一维通常称为“行”
 - 数组的第二维通常称为“列”
 - `int a[4][3];` // 定义了 4 行 3 列的存放整数的数组 a

```
int a[4][3];
```

	0	1	2
0			
1			
2			
3			

二维数组的初始化

- 二维数组的初始化
 - 二维数组可视为元素类型为一维数组的数组，故二维数组可按行分段赋值。

二维数组的初始化

- 二维数组的初始化
 - 二维数组可视为元素类型为一维数组的数组，故二维数组可按行分段赋值。
 - `// 定义了 2 行 4 列的存放整数的数组 a，并初始化`
`int a[2][4] = {{7, 8, 9, 1}, {2, 3, 4, 6}};`

	0	1	2	3
0				
1				

二维数组的初始化

- 二维数组的初始化
 - 二维数组可视作元素类型为一维数组的数组，故二维数组可按行分段赋值。
 - `// 定义了 2 行 4 列的存放整数的数组 a，并初始化`
`int a[2][4] = {{7, 8, 9, 1}, {2, 3, 4, 6}};`

	0	1	2	3
0	7	8	9	1
1				

二维数组的初始化

- 二维数组的初始化
 - 二维数组可视作元素类型为一维数组的数组，故二维数组可按行分段赋值。
 - `// 定义了 2 行 4 列的存放整数的数组 a，并初始化`
`int a[2][4] = {{7, 8, 9, 1}, {2, 3, 4, 6}};`

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

二维数组的访问

- 访问数组元素
 - 通过 **数组名 [i][j]** 访问存储在数组中第 i 行第 j 列的值
 - 下标 i 的范围: $0 \sim R - 1$ (R 为数组第一维大小)
 - 下标 j 的范围: $0 \sim C - 1$ (C 为数组第二维大小)
 - 要注意数组每一维都不能越界
 - 数组元素的用法与一般变量的用法相同

二维数组的访问 - 示例

- 例如：声明数组 `int a[2][3];`

```
int a[2][3];
```

	0	1	2
0	5	0	1
1	2	3	4

二维数组的访问 - 示例

- 例如：声明数组 `int a[2][3];`
 - 可以使用的行下标：0, 1
 - 可以使用的列下标：0, 1, 2

```
int a[2][3];
```

	0	1	2
0	5	0	1
1	2	3	4

二维数组的访问 - 示例

- 例如：声明数组 `int a[2][3];`
 - 可以使用的行下标：0, 1
 - 可以使用的列下标：0, 1, 2
 - 对应的可以使用的数第 0 行元素：
`a[0][0]`, `a[0][1]`, `a[0][2]`
 - 对应的可以使用的数第 1 行元素：
`a[1][0]`, `a[1][1]`, `a[1][2]`

		0	1	2
<code>int a[2][3];</code>	0	5	0	1
	1	2	3	4

二维数组的访问 - 示例

- 例如：声明数组 `int a[2][3];`
 - 可以使用的行下标：0, 1
 - 可以使用的列下标：0, 1, 2
 - 对应的可以使用的数第 0 行元素：
`a[0][0]`, `a[0][1]`, `a[0][2]`
 - 对应的可以使用的数第 1 行元素：
`a[1][0]`, `a[1][1]`, `a[1][2]`
 - 可以使用变量作为数组下标：`a[x][y]` ($0 \leq x \leq 1, 0 \leq y \leq 2$)

		0	1	2
<code>int a[2][3];</code>	0	5	0	1
	1	2	3	4

填空题

1. 阅读程序写结果

```
1 int a[3][4] = {{7, 8, 9, 1}, {2, 3, 4, 6}, {10, 11, 15, 19}};  
2  
3 int main() {  
4     int x, y;  
5     cin >> x >> y;  
6     cout << a[x][y] << endl;  
7  
8     return 0;  
9 }
```

输入: 2 1

输出:

填空题

1. 阅读程序写结果

```
1 int a[3][4] = {{7, 8, 9, 1}, {2, 3, 4, 6}, {10, 11, 15, 19}};  
2  
3 int main() {  
4     int x, y;  
5     cin >> x >> y;  
6     cout << a[x][y] << endl;  
7  
8     return 0;  
9 }
```

输入: 2 1

输出: 11

目录

1 复习回顾

2 二维数组

3 二维数组的遍历

4 二维数组的部分遍历

5 多维数组

6 总结

二维数组的遍历

- 通常使用二重循环来遍历二维数组
- 遍历输入二维数组

二维数组的遍历

- 通常使用二重循环来遍历二维数组
- 遍历输入二维数组
 - 从下标 0 开始储存

二维数组的遍历

- 通常使用二重循环来遍历二维数组
- 遍历输入二维数组
 - 从下标 0 开始储存

```
1 for (int i = 0; i < n; i++) {  
2     for (int j = 0; j < m; j++) {  
3         cin >> a[i][j];  
4     }  
5 }
```

二维数组的遍历

- 通常使用二重循环来遍历二维数组
- 遍历输入二维数组
 - 从下标 0 开始储存

```
1 for (int i = 0; i < n; i++) {  
2     for (int j = 0; j < m; j++) {  
3         cin >> a[i][j];  
4     }  
5 }
```

- 从下标 1 开始储存

二维数组的遍历

- 通常使用二重循环来遍历二维数组
- 遍历输入二维数组
 - 从下标 0 开始储存

```
1 for (int i = 0; i < n; i++) {  
2     for (int j = 0; j < m; j++) {  
3         cin >> a[i][j];  
4     }  
5 }
```

- 从下标 1 开始储存

```
1 for (int i = 1; i <= n; i++) {  
2     for (int j = 1; j <= m; j++) {  
3         cin >> a[i][j];  
4     }  
5 }
```

二维数组的遍历

- 通常使用二重循环来遍历二维数组
- 遍历输出二维数组

二维数组的遍历

- 通常使用二重循环来遍历二维数组
- 遍历输出二维数组
 - 从下标 0 开始储存

二维数组的遍历

- 通常使用二重循环来遍历二维数组
- 遍历输出二维数组
 - 从下标 0 开始储存

```
1 for (int i = 0; i < n; i++) {  
2     for (int j = 0; j < m; j++) {  
3         cout << a[i][j] << " ";  
4     }  
5     cout << endl;  
6 }
```


二维数组的遍历

- 通常使用二重循环来遍历二维数组
- 遍历输出二维数组
 - 从下标 0 开始储存

```
1 for (int i = 0; i < n; i++) {  
2     for (int j = 0; j < m; j++) {  
3         cout << a[i][j] << " ";  
4     }  
5     cout << endl;  
6 }
```

- 从下标 1 开始储存

二维数组的遍历

- 通常使用二重循环来遍历二维数组
- 遍历输出二维数组
 - 从下标 0 开始储存

```
1 for (int i = 0; i < n; i++) {  
2     for (int j = 0; j < m; j++) {  
3         cout << a[i][j] << " ";  
4     }  
5     cout << endl;  
6 }
```

- 从下标 1 开始储存

```
1 for (int i = 1; i <= n; i++) {  
2     for (int j = 1; j <= m; j++) {  
3         cout << a[i][j] << " ";  
4     }  
5     cout << endl;  
6 }
```

例 11.1：求二维数组元素的和

编程题

- 编写程序，输入两个整数 n 和 m ($1 \leq n, m \leq 100$)，接下来输入 n 行 m 列的整数 x ($-10^9 \leq x \leq 10^9$)，求这些整数的和。
- 样例输入
2 4
7 2 6 1
3 5 8 0
- 样例输出
32

例 11.1：求二维数组元素的和

```
1 // #include ...
2
3 int a[105][105];
4
5 int main() {
6     int n, m;
7     cin >> n >> m;
8     for (int i = 1; i <= n; i++) {
9         for (int j = 1; j <= m; j++) {
10             cin >> a[i][j];
11         }
12     }
13     int sum = 0;
14     for (int i = 1; i <= n; i++) {
15         for (int j = 1; j <= m; j++) {
16             sum += a[i][j];
17         }
18     }
19     cout << sum << endl;
20
21     return 0;
22 }
```

例 11.2: 求数组元素的最大值

编程题

- 编写程序，输入两个整数 n 和 m ($1 \leq n, m \leq 100$)，接下来输入 n 行 m 列的整数 x ($-10^9 \leq x \leq 10^9$)，求这些整数中的最大值。
- 样例输入
2 4
7 2 6 1
3 5 8 0
- 样例输出
8

例 11.2：求数组元素的最大值

```
1 // #include ...
2
3 int a[105][105];
4
5 int main() {
6     int n, m;
7     cin >> n >> m;
8     for (int i = 1; i <= n; i++) {
9         for (int j = 1; j <= m; j++) {
10             cin >> a[i][j];
11         }
12     }
13     int maxv = -1000000000;
14     for (int i = 1; i <= n; i++) {
15         for (int j = 1; j <= m; j++) {
16             if (a[i][j] > maxv) maxv = a[i][j];
17         }
18     }
19     cout << maxv << endl;
20
21     return 0;
22 }
```

例 11.3: 求数组元素的最大值及下标

编程题

- 编写程序，输入两个整数 n 和 m ($1 \leq n, m \leq 100$)，接下来输入 n 行 m 列的整数 x ($-10^9 \leq x \leq 10^9$)，求这些整数中的最大值及位置，保证数据中只有一个最大值。

- 样例输入

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

- 样例输出

```
8
2 3
```

例 11.3：求数组元素的最大值及下标

```
1 #include <iostream>
2
3 using namespace std;
4
5 int a[105][105];
6
7 int main() {
8     int n, m;
9     cin >> n >> m;
10    // 输入 a 数组, 从 1 开始储存
11    for (int i = 1; i <= n; i++) {
12        for (int j = 1; j <= m; j++) {
13            cin >> a[i][j];
14        }
15    }
```


例 11.3：求数组元素的最大值及下标

```
16  int pos1 = 0, pos2 = 0;
17  for (int i = 1; i <= n; i++) {
18      for (int j = 1; j <= m; j++) {
19          if (pos1 == 0 && pos2 == 0 || a[i][j] > a[pos1][pos2]) {
20              pos1 = i;
21              pos2 = j;
22          }
23      }
24  }
25  cout << a[pos1][pos2] << endl;
26  cout << pos1 << " " << pos2 << endl;
27
28  return 0;
29 }
```

例 11.3：求数组元素的最大值及下标

```
16  int pos1 = 0, pos2 = 0;
17  for (int i = 1; i <= n; i++) {
18      for (int j = 1; j <= m; j++) {
19          if (pos1 == 0 && pos2 == 0 || a[i][j] > a[pos1][pos2]) {
20              pos1 = i;
21              pos2 = j;
22          }
23      }
24  }
25  cout << a[pos1][pos2] << endl;
26  cout << pos1 << " " << pos2 << endl;
27
28  return 0;
29 }
```

目录

- 1 复习回顾
- 2 二维数组
- 3 二维数组的遍历
- 4 二维数组的部分遍历**
- 5 多维数组
- 6 总结

例 11.5：求矩阵第 x 行元素的和

编程题

- 编写程序，输入两个整数 n 和 m ($1 \leq n, m \leq 100$)，接下来输入 n 行 m 列的整数 x ($1 \leq x \leq n$)，求这个整数矩阵中第 x 行元素的和。

- 样例输入

```
2 4
4 5 9 2
1 7 0 3
2
```

- 样例输出
11

例 11.5: 求矩阵第 x 行元素的和

编程题

- 编写程序，输入两个整数 n 和 m ($1 \leq n, m \leq 100$)，接下来输入 n 行 m 列的整数 x ($1 \leq x \leq n$)，求这个整数矩阵中第 x 行元素的和。

- 样例输入

```
2 4
4 5 9 2
1 7 0 3
2
```

- 样例输出
11

例 11.5：求矩阵第 x 行元素的和

```
1 #include <iostream>
2
3 using namespace std;
4
5 int a[105][105];
6
7 int main() {
8     int n, m;
9     cin >> n >> m;
10    // 输入 a 数组, 从 1 开始储存
11    for (int i = 1; i <= n; i++) {
12        for (int j = 1; j <= m; j++) {
13            cin >> a[i][j];
14        }
15    }
16    int x, sum = 0;
17    cin >> x;
18    for (int j = 1; j <= m; j++) sum += a[x][j];
19    cout << sum << endl;
20
21    return 0;
22 }
```

例 11.5：求矩阵第 x 行元素的和

```
1 // #include ...
2
3 int a[105][105];
4
5 int main() {
6     int n, m, x, sum = 0;
7     cin >> n >> m;
8     for (int i = 1; i <= n; i++) {
9         for (int j = 1; j <= m; j++) {
10             cin >> a[i][j];
11         }
12     }
13     cin >> x;
14     for (int i = 1; i <= n; i++) {
15         for (int j = 1; j <= m; j++) {
16             if (i == x) sum += a[i][j];
17         }
18     }
19     cout << sum << endl;
20
21     return 0;
22 }
```

例 11.5: 求矩阵第 x 行元素的和

```
1 // #include ...
2
3 int a[105][105];
4
5 int main() {
6     int n, m, x, sum = 0;
7     cin >> n >> m;
8     for (int i = 1; i <= n; i++) {
9         for (int j = 1; j <= m; j++) {
10             cin >> a[i][j];
11         }
12     }
13     cin >> x;
14     for (int i = 1; i <= n; i++) {
15         for (int j = 1; j <= m; j++) {
16             if (i == x) sum += a[i][j];
17         }
18     }
19     cout << sum << endl;
20
21     return 0;
22 }
```


例 11.6: 求矩阵第 y 列元素的和

编程题

- 编写程序，输入两个整数 n 和 m ($1 \leq n, m \leq 100$)，接下来输入 n 行 m 列的整数 y ($1 \leq y \leq m$)，求这个整数矩阵中第 y 列元素的和。

- 样例输入

```
2 4
4 5 9 2
1 7 0 3
2
```

- 样例输出
12

例 11.6: 求矩阵第 y 列元素的和

编程题

- 编写程序，输入两个整数 n 和 m ($1 \leq n, m \leq 100$)，接下来输入 n 行 m 列的整数 y ($1 \leq y \leq m$)，求这个整数矩阵中第 y 列元素的和。

- 样例输入

```
2 4
4 5 9 2
1 7 0 3
2
```

- 样例输出
12

例 11.6: 求矩阵第 y 列元素的和

```
1 // #include ...
2
3 int a[105][105];
4
5 int main() {
6     int n, m, y, sum = 0;
7     cin >> n >> m;
8     for (int i = 1; i <= n; i++) {
9         for (int j = 1; j <= m; j++) {
10             cin >> a[i][j];
11         }
12     }
13     cin >> y;
14     for (int i = 1; i <= n; i++) {
15         for (int j = 1; j <= m; j++) {
16             if (j == y) sum += a[i][j];
17         }
18     }
19     cout << sum << endl;
20
21     return 0;
22 }
```

例 11.6: 求矩阵第 y 列元素的和

```
1 // #include ...
2
3 int a[105][105];
4
5 int main() {
6     int n, m, y, sum = 0;
7     cin >> n >> m;
8     for (int i = 1; i <= n; i++) {
9         for (int j = 1; j <= m; j++) {
10             cin >> a[i][j];
11         }
12     }
13     cin >> y;
14     for (int i = 1; i <= n; i++) {
15         for (int j = 1; j <= m; j++) {
16             if (j == y) sum += a[i][j];
17         }
18     }
19     cout << sum << endl;
20
21     return 0;
22 }
```

例 11.7：求方阵正对角线元素的和

编程题

- 编写程序，输入一个整数 n ($1 \leq n \leq 100$)，接下来输入 n 行 n 列的整数方阵，求这个整数方阵正对角线（左上到右下）元素的和。

- 样例输入

```
3
4 5 9
1 7 5
3 4 6
```

- 样例输出
17

例 11.7：求方阵正对角线元素的和

编程题

- 编写程序，输入一个整数 n ($1 \leq n \leq 100$)，接下来输入 n 行 n 列的整数方阵，求这个整数方阵正对角线（左上到右下）元素的和。
- 样例输入

```
3
4 5 9
1 7 5
3 4 6
```

- 样例输出
17

例 11.7：求方阵正对角线元素的和

```
1 // #include ...
2
3 int a[105][105];
4
5 int main() {
6     int n, sum = 0;
7     cin >> n;
8     for (int i = 1; i <= n; i++) {
9         for (int j = 1; j <= n; j++) {
10             cin >> a[i][j];
11         }
12     }
13     for (int i = 1; i <= n; i++) {
14         for (int j = 1; j <= n; j++) {
15             if (i == j) sum += a[i][j];
16         }
17     }
18     cout << sum << endl;
19
20     return 0;
21 }
```

例 11.7：求方阵正对角线元素的和

```
1 // #include ...
2
3 int a[105][105];
4
5 int main() {
6     int n, sum = 0;
7     cin >> n;
8     for (int i = 1; i <= n; i++) {
9         for (int j = 1; j <= n; j++) {
10             cin >> a[i][j];
11         }
12     }
13     for (int i = 1; i <= n; i++) {
14         for (int j = 1; j <= n; j++) {
15             if (i == j) sum += a[i][j];
16         }
17     }
18     cout << sum << endl;
19
20     return 0;
21 }
```


例 11.8：求方阵副对角线元素的和

编程题

- 编写程序，输入一个整数 n ($1 \leq n \leq 100$)，接下来输入 n 行 n 列的整数方阵，求这个整数方阵副对角线（右上到左下）元素的和。

- 样例输入

```
3
4 5 9
1 7 5
3 4 6
```

- 样例输出
19

例 11.8：求方阵副对角线元素的和

编程题

- 编写程序，输入一个整数 n ($1 \leq n \leq 100$)，接下来输入 n 行 n 列的整数方阵，求这个整数方阵副对角线（右上到左下）元素的和。
- 样例输入

```
3
4 5 9
1 7 5
3 4 6
```

- 样例输出
19

例 11.8：求方阵副对角线元素的和

```
1 // #include ...
2
3 int a[105][105];
4
5 int main() {
6     int n, sum = 0;
7     cin >> n;
8     for (int i = 1; i <= n; i++) {
9         for (int j = 1; j <= n; j++) {
10             cin >> a[i][j];
11         }
12     }
13     for (int i = 1; i <= n; i++) {
14         for (int j = 1; j <= n; j++) {
15             if (i + j == n + 1) sum += a[i][j];
16         }
17     }
18     cout << sum << endl;
19
20     return 0;
21 }
```

例 11.8：求方阵副对角线元素的和

```
1 // #include ...
2
3 int a[105][105];
4
5 int main() {
6     int n, sum = 0;
7     cin >> n;
8     for (int i = 1; i <= n; i++) {
9         for (int j = 1; j <= n; j++) {
10             cin >> a[i][j];
11         }
12     }
13     for (int i = 1; i <= n; i++) {
14         for (int j = 1; j <= n; j++) {
15             if (i + j == n + 1) sum += a[i][j];
16         }
17     }
18     cout << sum << endl;
19
20     return 0;
21 }
```

二维数组的部分遍历

- 对于二维数组遍历特定元素的问题，都可以使用 **二重循环 + 分支求解**

```
1 for (int i = 1; i <= n; i++) {    // i 行号
2     for (int j = 1; j <= m; j++) { // j 列号
3         if (/*如果 A[i][j] 是需要处理的特定元素*/) {
4             // 对 A[i][j] 进行处理 (比如输出、求和、计算等)
5         }
6     }
7 }
```

目录

1 复习回顾

2 二维数组

3 二维数组的遍历

4 二维数组的部分遍历

5 多维数组

6 总结

三维数组

- 三维数组的声明
 - 元素类型 数组名 [第一维大小][第二维大小][第三维大小];
 - `int a[100][100][100];`

三维数组

- 三维数组的声明
 - 元素类型 数组名 [第一维大小][第二维大小][第三维大小];
 - `int a[100][100][100];`
- 三维数组的元素访问
 - 数组名称 [第一维下标][第二维下标][第三维下标];

三维数组

- 三维数组的声明
 - 元素类型 数组名 [第一维大小][第二维大小][第三维大小];
 - `int a[100][100][100];`
- 三维数组的元素访问
 - 数组名称 [第一维下标][第二维下标][第三维下标];
- 三维数组的遍历
 - 使用三重循环遍历三维数组

三维数组的遍历

- 三重循环遍历输入三维数组

三维数组的遍历

- 三重循环遍历输入三维数组
 - 从下标 0 开始储存

三维数组的遍历

- 三重循环遍历输入三维数组
 - 从下标 0 开始储存

```
1 for (int i = 0; i < n; i++) {  
2     for (int j = 0; j < m; j++) {  
3         for (int k = 0; k < h; k++) {  
4             cin >> a[i][j][k];  
5         }  
6     }  
7 }
```

三维数组的遍历

- 三重循环遍历输入三维数组
 - 从下标 0 开始储存

```
1 for (int i = 0; i < n; i++) {  
2     for (int j = 0; j < m; j++) {  
3         for (int k = 0; k < h; k++) {  
4             cin >> a[i][j][k];  
5         }  
6     }  
7 }
```

- 从下标 1 开始储存

三维数组的遍历

- 三重循环遍历输入三维数组
 - 从下标 0 开始储存

```
1 for (int i = 0; i < n; i++) {  
2     for (int j = 0; j < m; j++) {  
3         for (int k = 0; k < h; k++) {  
4             cin >> a[i][j][k];  
5         }  
6     }  
7 }
```

- 从下标 1 开始储存

```
1 for (int i = 1; i <= n; i++) {  
2     for (int j = 1; j <= m; j++) {  
3         for (int k = 1; k <= h; k++) {  
4             cin >> a[i][j][k];  
5         }  
6     }  
7 }
```

三维数组的遍历

- 三重循环遍历输出三维数组

三维数组的遍历

- 三重循环遍历输出三维数组
 - 从下标 0 开始储存

三维数组的遍历

- 三重循环遍历输出三维数组
 - 从下标 0 开始储存

```
1 for (int i = 0; i < n; i++) {  
2     for (int j = 0; j < m; j++) {  
3         for (int k = 0; k < h; k++) {  
4             cout << a[i][j][k] << " ";  
5         }  
6         cout << endl;  
7     }  
8 }
```

三维数组的遍历

- 三重循环遍历输出三维数组
 - 从下标 0 开始储存

```
1 for (int i = 0; i < n; i++) {  
2     for (int j = 0; j < m; j++) {  
3         for (int k = 0; k < h; k++) {  
4             cout << a[i][j][k] << " ";  
5         }  
6         cout << endl;  
7     }  
8 }
```

- 从下标 1 开始储存

三维数组的遍历

- 三重循环遍历输出三维数组
 - 从下标 0 开始储存

```
1 for (int i = 0; i < n; i++) {  
2     for (int j = 0; j < m; j++) {  
3         for (int k = 0; k < h; k++) {  
4             cout << a[i][j][k] << " ";  
5         }  
6         cout << endl;  
7     }  
8 }
```

- 从下标 1 开始储存

```
1 for (int i = 1; i <= n; i++) {  
2     for (int j = 1; j <= m; j++) {  
3         for (int k = 1; k <= h; k++) {  
4             cout << a[i][j][k] << " ";  
5         }  
6         cout << endl;  
7     }  
8 }
```

例 11.9: 查询指定学生的成绩

编程题

- 编写程序，输入三个整数 n, m, h ($1 \leq n, m, h \leq 100$)，表示有 n 个年级，每个年级有 m 个班，每个班有 h 名学生。现用一个三维整数数组登记整个学校的学生成绩，输入这个三维数组的元素（最高成绩不超过 100）。
最后输入三个整数 x, y, z ($1 \leq x \leq n, 1 \leq y \leq m, 1 \leq z \leq h$)，求这名 x 年级 y 班 z 号学生的成绩。

- 样例输入

```
2 2 4
88 85 86 98
70 93 96 90
78 87 89 90
66 99 76 60
1 2 3
```

- 样例输出

```
96
```

例 11.9：查询指定学生的成绩

- 样例输入

```
2 2 4
88 85 86 98
70 93 96 90
78 87 89 90
66 99 76 60
1 2 3
```

- 样例输出

```
96
```

1 年级	1	2	3	4
1 班	88	85	86	98
2 班	70	93	96	90
2 年级	1	2	3	4
1 班	78	87	89	90
2 班	66	99	76	60

例 11.9：查询指定学生的成绩

```
1 #include <iostream>
2
3 using namespace std;
4
5 int a[105][105][105];
6
7 int main() {
8     int n, m, h;
9     cin >> n >> m >> h;
10    for (int i = 1; i <= n; i++) {
11        for (int j = 1; j <= m; j++) {
12            for (int k = 1; k <= h; k++) {
13                cin >> a[i][j][k];
14            }
15        }
16    }
17    int x, y, z;
18    cin >> x >> y >> z;
19    cout << a[x][y][z] << endl;
20
21    return 0;
22 }
```

例 11.10: 查询学生成绩的最高分

编程题

- 编写程序，输入三个整数 n, m, h ($1 \leq n, m, h \leq 100$)，表示有 n 个年级，每个年级有 m 个班，每个班有 h 名学生。现用一个三维整数数组登记整个学校的学生成绩，输入这个三维数组的元素（最高成绩不超过 100）。
输出全校学生成绩的最高分。

- 样例输入

```
2 2 4
88 85 86 98
70 93 96 90
78 87 89 90
66 99 76 60
```

- 样例输出

```
99
```

例 11.10: 查询学生成绩的最高分

```
1 #include <iostream>
2
3 using namespace std;
4
5 int a[105][105][105];
6
7 int main() {
8     int n, m, h;
9     cin >> n >> m >> h;
10    for (int i = 1; i <= n; i++) {
11        for (int j = 1; j <= m; j++) {
12            for (int k = 1; k <= h; k++) {
13                cin >> a[i][j][k];
14            }
15        }
16    }
```


例 11.10: 查询学生成绩的最高分

```
17  int maxv = -1000000000;
18  for (int i = 1; i <= n; i++) {
19      for (int j = 1; j <= m; j++) {
20          for (int k = 1; k <= h; k++) {
21              if (a[i][j][k] > maxv) {
22                  maxv = a[i][j][k];
23              }
24          }
25      }
26  }
27  cout << maxv << endl;
28
29  return 0;
30 }
```

目录

1 复习回顾

2 二维数组

3 二维数组的遍历

4 二维数组的部分遍历

5 多维数组

6 总结

- 二维数组的声明和遍历
- 数组的应用
 - 二维数组求和
 - 二维数组求最值及其下标
- 多维数组的声明和遍历
 - 多维数组求最值

Thank you!