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 6 日

目录

问题回顾 - 倒序输出数组元素

编程题

- 第一行输入一个整数 n ($1 \leq n \leq 100$)，表示有 n 个整数；第二行输入 n 个整数 x ($-10^9 \leq x \leq 10^9$)，表示数组中的每个元素。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

```cpp
#include <iostream>

using namespace std;

int a[105];

int main() {
  int n;
  cin >> n;
  // 按照下标 1 ~ n 的顺序输入数组元素
  for (int i = 1; i <= n; i++) {
    cin >> a[i];
  }
  // 按照下标 n ~ 1 的顺序输出数组元素
  for (int i = n; i >= 1; i--) {
    cout << a[i] << " ";
  }
  cout << endl;

  return 0;
}
```

目录

引入

- 现实生活中有很多事物是以“二维”的方式组织起来的：

引入

- 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。

引入

- 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。

引入

- 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];**

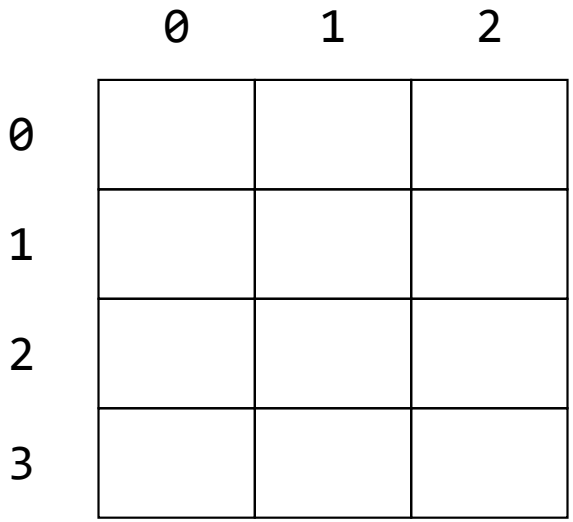
二维数组的声明

- 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 列存放整数的二维数组 a`

`int a[4][3];`



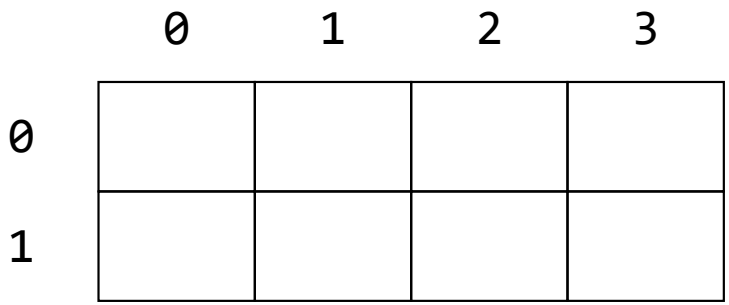
二维数组的初始化

- 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。

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 列存放整数的二维数组 a，并初始化`
    `int a[2][4] = {{7, 8, 9, 1}, {2, 3, 4, 6}};`

`int a[2][4];`



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 列存放整数的二维数组 a，并初始化`
    `int a[2][4] = {{7, 8, 9, 1}, {2, 3, 4, 6}};`

`int a[2][4];`

	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 列存放整数的二维数组 a，并初始化`
    `int a[2][4] = {{7, 8, 9, 1}, {2, 3, 4, 6}};`

`int a[2][4];`

	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：$0, 1, 2$
  - 可以使用的第 0 行元素：`a[0][0]`, `a[0][1]`, `a[0][2]`
  - 可以使用的第 1 行元素：`a[1][0]`, `a[1][1]`, `a[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：$0, 1, 2$
  - 可以使用的第 0 行元素：`a[0][0]`，`a[0][1]`，`a[0][2]`
  - 可以使用的第 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]` ($0 \le x \le 1, 0 \le y \le 2$)

`int a[2][3];`

	0	1	2
0	5	0	1
1	2	3	4

随堂练习

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：

随堂练习

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

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

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

  - 从下标 1 开始储存

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

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

```
for (int i = 0; i < n; i++) {
  for (int j = 0; j < m; j++) {
    cout << a[i][j] << " ";
  }
  cout << endl;
}
```

  - 从下标 1 开始储存

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

例 11.1：求二维数组元素的和

编程题

- 输入两个整数 n 和 m $(1 \le n, m \le 100)$；接下来输入 n 行 m 列的整数矩阵 a，其中 $-10^9 \le a_{i,j} \le 10^9$。求整数矩阵中元素的和。
- 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

```cpp
// #include ...

int a[105][105];

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];
    }
  }
  long long sum = 0;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      sum += a[i][j];
    }
  }
  cout << sum << endl;

  return 0;
}
```

例 11.1：求二维数组元素的和

```cpp
// #include ...

int a[105][105];

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];
    }
  }
  long long sum = 0;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      sum += a[i][j];
    }
  }
  cout << sum << endl;

  return 0;
}
```

例 11.2：求数组元素的最大值

编程题

- 输入两个整数 n 和 m $(1 \le n, m \le 100)$；接下来输入 n 行 m 列的整数矩阵 a，其中 $-10^9 \le a_{i,j} \le 10^9$。求整数矩阵中元素的最大值。
- 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

```cpp
// #include ...

int a[105][105];

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];
    }
  }
  int maxv = -1000000000;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      if (a[i][j] > maxv) maxv = a[i][j];
    }
  }
  cout << maxv << endl;

  return 0;
}
```

例 11.3：求数组元素的最大值及下标

编程题

- 输入两个整数 n 和 m $(1 \leq n, m \leq 100)$；接下来输入 n 行 m 列的整数矩阵 a，其中 $-10^9 \leq a_{i,j} \leq 10^9$。求整数矩阵中元素的最大值及位置（保证最大值是唯一的）。
- 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

```cpp
#include <iostream>

using namespace std;

int a[105][105];

int main() {
  int n, m;
  cin >> n >> m;
  // 输入 a 数组，从 1 开始储存
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      cin >> a[i][j];
    }
  }
```

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

例 11.4：求矩阵第 x 行元素的和

编程题

- 输入两个整数 n 和 m $(1 \leq n, m \leq 100)$；接下来输入 n 行 m 列的整数矩阵 a，其中 $-10^9 \leq a_{i,j} \leq 10^9$；最后输入一个整数 x $(1 \leq x \leq n)$。求整数矩阵中第 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 11.4：求矩阵第 x 行元素的和

编程题

- 输入两个整数 n 和 m $(1 \leq n, m \leq 100)$；接下来输入 n 行 m 列的整数矩阵 a，其中 $-10^9 \leq a_{i,j} \leq 10^9$；最后输入一个整数 x $(1 \leq x \leq n)$。求整数矩阵中第 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 11.4：求矩阵第 x 行元素的和

```cpp
#include <iostream>

using namespace std;

int a[105][105];

int main() {
  int n, m, x;
  cin >> n >> m;
  // 输入 a 数组，从 1 开始储存
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      cin >> a[i][j];
    }
  }
  cin >> x;
  long long sum = 0;
  for (int j = 1; j <= m; j++) sum += a[x][j];
  cout << sum << endl;

  return 0;
}
```

例 11.4：求矩阵第 x 行元素的和

```cpp
// #include ...

int a[105][105];

int main() {
  int n, m, x;
  cin >> n >> m;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      cin >> a[i][j];
    }
  }
  cin >> x;
  long long sum = 0;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      if (i == x) sum += a[i][j];
    }
  }
  cout << sum << endl;

  return 0;
}
```

例 11.4：求矩阵第 x 行元素的和

```cpp
// #include ...

int a[105][105];

int main() {
  int n, m, x;
  cin >> n >> m;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      cin >> a[i][j];
    }
  }
  cin >> x;
  long long sum = 0;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      if (i == x) sum += a[i][j];
    }
  }
  cout << sum << endl;

  return 0;
}
```

例 11.5：求矩阵第 y 列元素的和

编程题

- 输入两个整数 n 和 m $(1 \leq n, m \leq 100)$；接下来输入 n 行 m 列的整数矩阵 a，其中 $-10^9 \leq a_{i,j} \leq 10^9$；最后输入一个整数 y $(1 \leq y \leq m)$。求整数矩阵中第 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 11.5：求矩阵第 y 列元素的和

编程题

- 输入两个整数 n 和 m $(1 \le n, m \le 100)$；接下来输入 n 行 m 列的整数矩阵 a，其中 $-10^9 \le a_{i,j} \le 10^9$；最后输入一个整数 y $(1 \le y \le m)$。求整数矩阵中第 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 11.5：求矩阵第 y 列元素的和

```cpp
// #include ...

int a[105][105];

int main() {
  int n, m, y;
  cin >> n >> m;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      cin >> a[i][j];
    }
  }
  cin >> y;
  long long sum = 0;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      if (j == y) sum += a[i][j];
    }
  }
  cout << sum << endl;

  return 0;
}
```

例 11.5：求矩阵第 y 列元素的和

```cpp
// #include ...

int a[105][105];

int main() {
  int n, m, y;
  cin >> n >> m;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      cin >> a[i][j];
    }
  }
  cin >> y;
  long long sum = 0;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      if (j == y) sum += a[i][j];
    }
  }
  cout << sum << endl;

  return 0;
}
```

例 11.6：求方阵正对角线元素的和

编程题

- 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 a，其中 $-10^9 \leq a_{i,j} \leq 10^9$。求整数方阵中正对角线（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 11.6：求方阵正对角线元素的和

编程题

- 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 a，其中 $-10^9 \leq a_{i,j} \leq 10^9$。求整数方阵中正对角线（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 11.6：求方阵正对角线元素的和

```cpp
// #include ...

int a[105][105];

int main() {
  int n
  cin >> n;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j++) {
      cin >> a[i][j];
    }
  }
  long long sum = 0;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j++) {
      if (i == j) sum += a[i][j];
    }
  }
  cout << sum << endl;

  return 0;
}
```

例 11.6：求方阵正对角线元素的和

```cpp
// #include ...

int a[105][105];

int main() {
  int n
  cin >> n;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j++) {
      cin >> a[i][j];
    }
  }
  long long sum = 0;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j++) {
      if (i == j) sum += a[i][j];
    }
  }
  cout << sum << endl;

  return 0;
}
```

例 11.7：求方阵副对角线元素的和

编程题

- 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 a，其中 $-10^9 \leq a_{i,j} \leq 10^9$。求整数方阵中副对角线（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 11.7：求方阵副对角线元素的和

编程题

- 输入一个整数 n ($1 \le n \le 100$)，接下来输入 n 行 n 列的整数方阵 a，其中 $-10^9 \le a_{i,j} \le 10^9$。求整数方阵中副对角线（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 11.7：求方阵副对角线元素的和

```cpp
// #include ...

int a[105][105];

int main() {
  int n
  cin >> n;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j++) {
      cin >> a[i][j];
    }
  }
  long long sum = 0;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j++) {
      if (i + j == n + 1) sum += a[i][j];
    }
  }
  cout << sum << endl;

  return 0;
}
```

例 11.7：求方阵副对角线元素的和

```cpp
// #include ...

int a[105][105];

int main() {
  int n
  cin >> n;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j++) {
      cin >> a[i][j];
    }
  }
  long long sum = 0;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j++) {
      if (i + j == n + 1) sum += a[i][j];
    }
  }
  cout << sum << endl;

  return 0;
}
```

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

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 < c; 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 < c; 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

```
for (int i = 0; i < n; i++) {
  for (int j = 0; j < m; j++) {
    for (int k = 0; k < c; k++) {
      cin >> a[i][j][k];
    }
  }
}
```

  - 从下标 1 开始储存

```
for (int i = 1; i <= n; i++) {
  for (int j = 1; j <= m; j++) {
    for (int k = 1; k <= c; k++) {
      cin >> a[i][j][k];
    }
  }
}
```

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 < c; 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 < c; 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 < c; 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 <= c; k++) {
4       cout << a[i][j][k] << " ";
5     }
6     cout << endl;
7   }
8 }
```

例 11.8：查询指定学生的成绩

编程题

- 输入三个整数 n, m, c $(1 \le n, m, c \le 100)$，表示有 n 个年级，每个年级有 m 个班，每个班有 c 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，其中元素数值不超过 100。
  最后输入三个整数 x, y, z $(1 \le x \le n, 1 \le y \le m, 1 \le z \le c)$，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 11.8：查询指定学生的成绩

编程题

- 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 11.8：查询指定学生的成绩

```cpp
#include <iostream>

using namespace std;

int a[105][105][105];

int main() {
  int n, m, c;
  cin >> n >> m >> c;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      for (int k = 1; k <= c; k++) {
        cin >> a[i][j][k];
      }
    }
  }
  int x, y, z;
  cin >> x >> y >> z;
  cout << a[x][y][z] << endl;

  return 0;
}
```

例 11.9：查询学生成绩的最高分

编程题

- 输入三个整数 n, m, c $(1 \leq n, m, c \leq 100)$，表示有 n 个年级，每个年级有 m 个班，每个班有 c 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，其中元素数值不超过 100。
  求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 11.9：查询学生成绩的最高分

```cpp
#include <iostream>

using namespace std;

int a[105][105][105];

int main() {
  int n, m, c;
  cin >> n >> m >> c;
  for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= m; j++) {
      for (int k = 1; k <= c; k++) {
        cin >> a[i][j][k];
      }
    }
  }
```

例 11.9：查询学生成绩的最高分

```
17    int maxv = 0;
18    for (int i = 1; i <= n; i++) {
19      for (int j = 1; j <= m; j++) {
20        for (int k = 1; k <= c; 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

总结

- 二维数组的声明、初始化
- 二维数组的遍历、部分遍历
- 二维数组的应用
  - 二维数组求和
  - 二维数组求最值及其下标
  - 方阵的正对角线、副对角线

总结

- 二维数组的声明、初始化
- 二维数组的遍历、部分遍历
- 二维数组的应用
  - 二维数组求和
  - 二维数组求最值及其下标
  - 方阵的正对角线、副对角线
- 多维数组的声明、遍历
- 多维数组的应用
  - 多维数组求最值

Thank you!