

Ravesli [Ravesli](#)


- [Уроки по C++](#)
- [OpenGL](#)
- [SFML](#)
- [Qt5](#)
- [RegEx](#)
- [Ассемблер](#)
- [Купить .PDF](#)

Урок №78. Многомерные массивы

 [Юрий](#) |

- [Уроки C++](#)

|

 Обновл. 15 Сен 2020 |

 45836

[↑](#)  7

Элементы массива могут быть любого типа данных, даже [массива](#)!

Оглавление:

1. [Многомерные массивы](#)
2. [Инициализация двумерных массивов](#)
3. [Доступ к элементам в двумерном массиве](#)
4. [Многомерные массивы более двух измерений](#)
5. [Пример двумерного массива](#)

Многомерные массивы

Массив массивов называется **многомерным массивом**:

```
1 | int array[2][4]; // 2-элементный массив из 4-элементных массивов
```

Поскольку у нас есть 2 индекса, то это **двумерный массив**.

В двумерном массиве первый (левый) индекс принято читать как количество строк, а второй (правый) как количество столбцов. Массив выше можно представить следующим образом:

```
[0][0] [0][1] [0][2] [0][3] // строка №0  
[1][0] [1][1] [1][2] [1][3] // строка №1
```

Чтобы получить доступ к элементам двумерного массива, просто используйте два индекса:

```
1 array[1][3] = 7; // без приставки int (типа данных)
```

Инициализация двумерных массивов

Для инициализации двумерного массива проще всего использовать вложенные фигурные скобки, где каждый набор значений соответствует определенной строке:

```
1 int array[3][5] =  
2 {  
3 { 1, 2, 3, 4, 5 }, // строка №0  
4 { 6, 7, 8, 9, 10 }, // строка №1  
5 { 11, 12, 13, 14, 15 } // строка №2  
6 };
```

Хотя некоторые компиляторы могут позволить вам упустить внутренние фигурные скобки, все же рекомендуется указывать их в любом случае: улучшается читабельность и уменьшается вероятность получения незапланированных нулевых элементов массива из-за того, что C++ заменяет отсутствующие инициализаторы значением 0:

```
1 int array[3][5] =  
2 {  
3 { 2, 4 }, // строка №0 = 2, 4, 0, 0, 0  
4 { 1, 3, 7 }, // строка №1 = 1, 3, 7, 0, 0  
5 { 8, 9, 11, 12 } // строка №2 = 8, 9, 11, 12, 0  
6 };
```

В двумерном массиве со [списком инициализаторов](#) можно не указывать только левый индекс (длину массива):

```
1 int array[][5] =  
2 {  
3 { 1, 2, 3, 4, 5 },  
4 { 6, 7, 8, 9, 10 },  
5 { 11, 12, 13, 14, 15 }  
6 };
```

Компилятор может сам вычислить количество строк в массиве. Однако не указывать два индекса — это уже ошибка:

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

Подобно обычным массивам, многомерные массивы можно инициализировать значением 0 следующим образом:

```
1 | int array[3][5] = { 0 };
```

Обратите внимание, это работает только в том случае, если вы явно объявляете длину массива (указываете левый индекс)! В противном случае, вы получите двумерный массив с 1 строкой.

Доступ к элементам в двумерном массиве

Для доступа ко всем элементам двумерного массива требуется два цикла: один для строк и один для столбцов. Поскольку доступ к двумерным массивам обычно выполняется по строкам, то левый индекс используется в качестве внешнего цикла:

```
1 | for (int row = 0; row < numRows; ++row) // доступ по строкам
2 |     for (int col = 0; col < numCols; ++col) // доступ к каждому элементу в строке
3 |         std::cout << array[row][col];
```

Многомерные массивы более двух измерений

Многомерные массивы могут быть более двух измерений. Например, объявление трехмерного массива:

```
1 | int array[4][3][2];
```

Трехмерные массивы трудно инициализировать любым интуитивным способом с использованием списка инициализаторов, поэтому лучше инициализировать весь массив значением 0 и явно присваивать элементам значения с помощью вложенных циклов.

Доступ к элементам трехмерного массива осуществляется так же, как и к элементам двумерного массива:

```
1 | std::cout << array[3][2][1];
```

Пример двумерного массива

Рассмотрим пример использования двумерного массива:

```
1 | #include <iostream>
2 |
3 | int main()
4 | {
5 |     // Объявляем массив 10x10
6 |     const int numRows = 10;
7 |     const int numCols = 10;
8 |     int product[numRows][numCols] = { 0 };
9 |
10 |    // Создаем таблицу умножения
11 |    for (int row = 0; row < numRows; ++row)
12 |        for (int col = 0; col < numCols; ++col)
13 |            product[row][col] = row * col;
14 | }
```

```
15
16 // Выводим таблицу умножения
17 for (int row = 1; row < numRows; ++row)
18 {
19     for (int col = 1; col < numCols; ++col)
20         std::cout << product[row][col] << "\t";
21
22     std::cout << '\n';
23 }
24
25 return 0;
}
```

Эта программа вычисляет и выводит таблицу умножения от 1 до 9 (включительно). Обратите внимание, при выводе таблицы в цикле for мы начинаем с 1 вместо 0. Это делается с целью предотвращения вывода нулевой строки и нулевого столбца, содержащих одни нули!

Результат выполнения программы:

1	2	3	4	5	6	7	8	9
2	4	6	8	10	12	14	16	18
3	6	9	12	15	18	21	24	27
4	8	12	16	20	24	28	32	36
5	10	15	20	25	30	35	40	45
6	12	18	24	30	36	42	48	54
7	14	21	28	35	42	49	56	63
8	16	24	32	40	48	56	64	72
9	18	27	36	45	54	63	72	81

Двумерные массивы обычно используются в играх типа *tile-based*, где каждый элемент массива представляет собой один фрагмент/плитку. Они также используются в компьютерной 3D-графике (в виде *матриц*) для вращения, масштабирования и отражения фигур.

Оценить статью:

★★★★★ (240 оценок, среднее: 4,88 из 5)



← [Урок №77. Сортировка массивов методом выбора](#)

[Урок №79. Строки C-style](#) →



Комментариев: 7



1. Анон:

[4 октября 2020 в 18:12](#)

Как задать двумерный массив, в котором количество строк и столбцов задаёт пользователь?

[Ответить](#)



1. Юрий:

[28 ноября 2020 в 21:41](#)

ТИПО, ВОТ ТАК:

```
1  #include <iostream>
2  #include<ctime>
3  using namespace std;
4  int** arrayMem(int** parray, int row, int col);
5  void fillArray(int** parray, int row, int col);
6  void drawArray(int** parray, int row, int col);
7  void drawArray(int** parray, int row);
8  int main()
9  {
10     srand(time(NULL));
11     setlocale(LC_ALL, "rus");
12     int row = 0;
13     int col = 0;
14     int** array = 0;
15     cout << "Количество строк:\n";
16     cin >> row;
17     cout << "количество столбцов:\n";
18     cin >> col;
19     array = arrayMem(array, row, col);
20     fillArray(array, row, col);
21     drawArray(array, row, col);
22     return 0;
23 }
24
25 int** arrayMem(int** parray, int row, int col)
26 {
27     parray = new int* [row];
28     for (int i = 0; i < row; i++)
29         parray[i] = new int[col];
30     return parray;
31 }
32 void fillArray(int** parray, int row, int col)
33 {
34     for (int i = 0; i < row; i++)
35     {
```

```

36         for (int j = 0; j < col; j++)
37         {
38             parray[i][j] = 10 + rand() % 41;
39         }
40     }
41 }
42
43 void drawArray(int** parray, int row, int col)
44 {
45     for (int i = 0; i < row; i++)
46     {
47         for (int j = 0; j < col; j++)
48         {
49             cout << parray[i][j] << " | ";
50         }cout << endl;
51     }
52 }
53 void drawArray(int** parray, int row)
54 {
55     for (int i = 0; i < row; i++)
56     {
57         delete[]parray[i];
58         delete[]parray;
59     }
60 }
61 }

```

[Ответить](#)



2. *Piter:*

[1 января 2020 в 23:52](#)

3 новым роком!

```

1  #include <iostream>
2  #include <windows.h> //для задержки Sleep
3  #include <cstdlib> //для system
4  #include <ctime>
5
6  int main()
7  {
8      srand(static_cast<unsigned int>(time(0)));
9
10     //Огодошуємо масив 10x10
11     const int numRows = 10;
12     const int numCols = 10;
13     int product[numRows][numCols] = {0};
14

```

```
15 char c[8] = {'c','o','l','o','r',' ',' ',0,0};
16
17 //Створюємо таблицю множення
18 for(int row = 0; row < numRows; ++row)
19 {
20     for(int col = 0; col < numCols; ++col)
21         product[row][col] = row * col;
22 }
23
24 //Виводимо таблицю множення
25 for(int row = 1; row < numRows; ++row)
26 {
27     for(int col = 1; col < numCols; ++col)
28         std::cout << product[row][col] << "\t";
29     std::cout << '\n' << '\n';
30
31 }
32 //Змінюємо колір фону і тексту в командному рядку
33 for(int i = 0; i < 16; ++i)
34 {
35     int j = rand() % 16;
36     for(int a = 0; a < j; ++a)
37     {
38         int k = rand() % 6;
39         if(j < 10)
40             c[6] = 48 + j;
41         else
42             c[6] = 55 + j;
43         c[7] = 97 + k;
44         system(c);
45         Sleep(400);
46     }
47     if(i == 15)
48     {
49         c[6] = 70;
50         c[7] = 48;
51         system(c);
52     }
53 }
54
55 return 0;
56 }
```

[Ответить](#)



1. Юрий:

[5 января 2020 в 19:13](#)

Дякую, і тебе з Новим роком!

[Ответить](#)



3. Константин:

[28 июня 2019 в 04:49](#)

Здорового времени суток, Юр! Процитирую тебя:

"Трёхмерные массивы трудно инициализировать любым интуитивным способом с использованием списка инициализаторов, поэтому лучше инициализировать весь массив значением 0 и явно присваивать значения с помощью вложенных циклов."

Кинь, пожал-ста, примерчик каким обрезом явно присваивать значения с помощью вложенных циклов, а я вот свою ответку на первую половину мэсседжа шлю:

```
1 #include <iostream>
2 #include <Windows.h>
3
4 int main()
5
6 {   SetConsoleCP(1251); SetConsoleOutputCP(1251); using std::cout; using std::cin;
7     cout << "Трёхмерный массив:\n";
8     const int x{4}, y{3}, z{3};
9     double f[z][y][x] =
10     {
11         {
12             {0.01, 0.02, 0.03, 0.04},
13             {0.11, 0.12, 0.13, 0.14},
14             {0.21, 0.22, 0.23, 0.24}
15         },
16         {
17             {1.01, 1.02, 1.03, 1.04},
18             {1.11, 1.12, 1.13, 1.14},
19             {1.21, 1.22, 1.23, 1.24}
20         },
21         {
22             {25.4, 36.6, 69.1, 22.2},
23             {11.7, 77.5, 12.2, 54.2},
24             {90.4, 85.2, 81.9, 38.6}
25         }
26     };
27     for(int i{}; i < z; ++i)
28     {
29         cout << "\nЭтаж " << i << "\n";
30         for(int k{}; k < y; ++k)
31         {
32             cout << "\n";
33             for(int l{}; l < x; ++l)
34                 cout << f[i][k][l] << " |";
```



```

35 |         }
36 |     }
37 |     cout << endl;
38 |     return 0;
39 | }

```

[Ответить](#)



4. *Алексей:*

[21 марта 2019 в 11:57](#)

Добрый день, Юрий.

Прочитав комментарий к этому уроку и информацию на других сайтах я понял, что если надо получить многомерный массив `int array[][5]` заполненный нулями, то можно инициализировать все элементы массива следующим образом.

```

1 | int array[ ][5] = { { } };

```

Насколько это корректно?

Можно ли использовать цикл

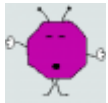
```

1 | for (i = 0; i < SIZE; i++)
2 | {
3 |     for (j = 0; j < SIZE; j++)
4 |     {
5 |         int array[i][j] = 0;
6 |     }
7 | }

```

Или этот цикл будет лишними строчками кода?

[Ответить](#)



5. *Александр:*

[15 февраля 2019 в 10:11](#)

инициализировать массивы через `= {0};` довольно плохая идея. Так возникает иллюзия (особенно у совсем начинающих), что число в скобках задает значение для ВСЕХ элементов массива. Лучше вообще не указывать ничего в скобках и писать `= {};`

А еще лучше подчеркивать уровень массива и писать `= {{{}}};` для двумерных.

Ну и стоит отметить, что в двумерных массивах не совсем очевидная адресация памяти, что часто приводит к выносящим мозг новичкам ошибкам, типа подхвата чисел из соседних строк при обращении к текущей.

Как по мне, лучше пользоваться одномерными массивами вместо двумерных везде, где это возможно (и быстрее, и проще, и меньше ошибок)

[Ответить](#)

Добавить комментарий

Ваш E-mail не будет опубликован. Обязательные поля помечены *

Имя *

Email *

Комментарий






☐ Сохранить моё Имя и E-mail. Видеть комментарии, отправленные на модерацию

☐ Получать уведомления о новых комментариях по электронной почте. Вы можете [подписаться](#) без комментирования.

[TELEGRAM](#)  [КАНАЛ](#)

[ПАБЛИК](#) 

ТОП СТАТЬИ

-  [Словарь программиста. Сленг, который должен знать каждый кодер](#)
-  [Урок №1. Введение в программирование](#)
-  [70+ бесплатных ресурсов для изучения программирования](#)
-  [Урок №1: Введение в создание игры «SameGame» на C++/MFC](#)
-  [Урок №4. Установка IDE \(Интегрированной Среды Разработки\)](#)

- [Ravesli](#)
- - [О проекте/Контакты](#) -
- - [Пользовательское Соглашение](#) -
- - [Все статьи](#) -
- Copyright © 2015 - 2020