**Практическая работа № 18**

**Тема: Объявление многомерных массивов в программе.**

**Цель: Разработка программ по объявлению и использованию многомерных массивов**

**Краткие теоретические сведения:**

Помимо одномерных массивов вам может понадобиться для работы использование многомерного массива (двумерного, трёхмерного…). В этом уроке будут рассмотрены двумерные массивы. Они самые распространенные, а остальные встречаются крайне редко. Чтобы обратиться к какому-либо элементу такого массива, достаточно указать его имя и индекс элемента. Первое отличие двумерного массива от одномерного – его элементы содержат два индекса: int arr [3][4];  Данные такого массива можно представить, как таблицу: 3 х 4.

[](https://purecodecpp.com/wp-content/uploads/2014/07/dvumernie-massivi-c--3.jpg)

Первый за именем массива индекс – это индекс строки, второй – индекс столбца.

[](https://purecodecpp.com/wp-content/uploads/2014/07/dvumernie-massivi-c--4.jpg)

Можно сказать о двумерном  массиве так – это массив, в котором каждый элемент также является массивом.  int arr [3][4];  – это массив из 3-х элементов, каждый из которых это массив из 4-х элементов.

**Инициализация двумерного массива.**

Записать данные в двумерный массив можно при его объявлении. Рассмотрим на примере с местами парковки. Допустим в паркинге 2 этажа по 4 места парковки на каждом. Объявим массив и инициализируем его:

**int floorsAndParkings[2][4] = { { 1, 2, 3, 4 }, { 1, 2, 3, 4 } };**

Чтобы такая инициализация выглядела более читабельно, оформим её так:

|  |  |
| --- | --- |
| 1  2  3  4  5 | int floorsAndParkings[2][4]  {  { 1, 2, 3, 4 }, // инициализация floorsAndParkings[0]  { 1, 2, 3, 4 }  // инициализация floorsAndParkings[1]  }; |

знак  **=**  можно опустить. Строки инициализируются по тому же принципу:

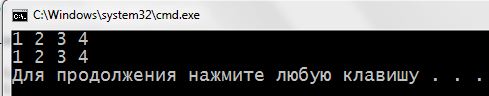
|  |  |
| --- | --- |
| 1  2  3  4  5  6 | char someStr[3][16]  {  "Двумерные ",  "массивы ",  "в С++!"  }; |

Как вывести на экран данные двумерного массива? Можно пойти длинным путём и обращаться к каждому элементу вручную:

вывод на экран данных двумерного массива

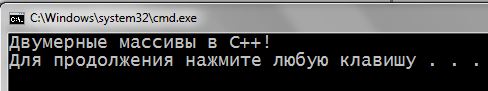
**Задача №1,№2,№3 (оценка 3)**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | #include <iostream>  using namespace std;    int main()  {  int floorsAndParkings[2][4]  {  { 1, 2, 3, 4 },  { 1, 2, 3, 4 }  };    cout << floorsAndParkings[0][0] << " ";  cout << floorsAndParkings[0][1] << " ";  cout << floorsAndParkings[0][2] << " ";  cout << floorsAndParkings[0][3] << " ";  cout << endl;    cout << floorsAndParkings[1][0] << " ";  cout << floorsAndParkings[1][1] << " ";  cout << floorsAndParkings[1][2] << " ";  cout << floorsAndParkings[1][3] << " ";  cout << endl;    return 0;  } |

[](https://purecodecpp.com/wp-content/uploads/2014/07/dvumernie-massivi-c--6.jpg)

**Задача № 2 символьный массив**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | #include <iostream>  using namespace std;    int main()  {  setlocale(LC\_ALL, "rus");    char someStr[3][16]  {  "Двумерные ",  "массивы ",  "в С++!\n"  };    cout << someStr[0];  cout << someStr[1];  cout << someStr[2];    return 0;  } |

[](https://purecodecpp.com/wp-content/uploads/2014/07/dvumernie-massivi-c--5.jpg)

А если нам надо заполнить и показать данные массива  **int floorsAndParkings[20][100]** или **char someStr[50][256]**? Эту неблагодарную работу можно в десятки раз облегчить, используя  циклы. Точнее [вложенные циклы](https://purecodecpp.com/archives/770" \o "Вложенные циклы в C++" \t "_blank).

**Задача № 3**

Рассмотрим пример с паркингом. Показать пользователю схему паркинга: этажи и места для парковки. Чтобы забронировать место он должен выбрать номер этажа и номер места. После бронирования – записать значение 0 в соответствующую ячейку, что будет означать “место занято”.

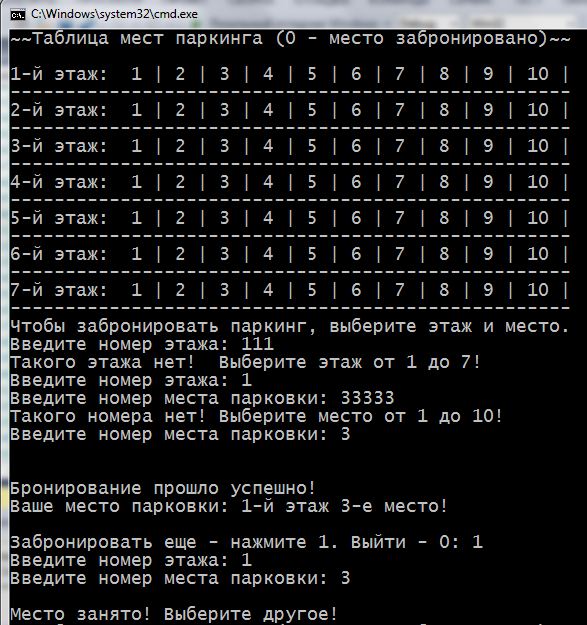
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85 | #include <iostream>  using namespace std;    int main()  {  setlocale(LC\_ALL, "rus");    const int AMOUNT\_FLOORS = 7; // к-во этажей  const int AMOUNT\_PARKINGS = 10; // к-во парковочных мест на этаже  int floorsAndParkings[AMOUNT\_FLOORS][AMOUNT\_PARKINGS]; // объявление двумерного массива    // присвоение значений и отображение  cout << "~~Таблица мест паркинга (0 - место забронировано)~~" << endl << endl;    for (int f = 0; f < AMOUNT\_FLOORS; f++) // используем встроенные циклы  {  cout << f + 1 << "-й этаж:  ";  for (int p = 0; p < AMOUNT\_PARKINGS; p++)  {  floorsAndParkings[f][p] = p + 1; // присвоить значение  cout << floorsAndParkings[f][p] << " | "; // сразу показать  }  cout << endl << "---------------------------------------------------" << endl;  }    int floor = 0; // этаж  int parkingPlace = 0; // парковочное место  char exit = '1'; // для выхода из do while    cout << "Чтобы забронировать паркинг, выберите этаж и место.\n";    do // внешний do while  {  do // встроенный do while для выбора этажа  {  cout << "Введите номер этажа: ";  cin >> floor;    if (floor < 1 || floor > 7) // если такого этажа нет  {  cout << "Такого этажа нет!  Выберите этаж от 1 до 7!\n";  }  } while (floor < 1 || floor > 7);      do // встроенный do while для выбора места  {  cout << "Введите номер места парковки: ";  cin >> parkingPlace;    if (parkingPlace < 1 || parkingPlace > 10)  {  cout << "Такого номера нет! Выберите место от 1 до 10!\n";  }  } while (parkingPlace < 1 || parkingPlace > 10);    if (floorsAndParkings[floor - 1][parkingPlace - 1] != 0) // если место свободно  {  floorsAndParkings[floor - 1][parkingPlace - 1] = 0; //  отметить, как забронированное  cout << "\n\nБронирование прошло успешно!\n";  cout << "Ваше место парковки: " << floor << "-й этаж " << parkingPlace << "-е место!\n\n";  cout << "Забронировать еще - нажмите 1. Выйти - 0: ";  cin >> exit;  }  else // если место занято (хранит значение 0)  {  cout << "\nМесто занято! Выберите другое!\n";    // отобразить таблицу, чтобы было видно какие места свободны  cout << "~~Таблица мест паркинга (0 - место забронировано)~~" << endl << endl;    for (int f = 0; f < AMOUNT\_FLOORS; f++)  {  cout << f + 1 << "-й этаж:  ";  for (int p = 0; p < AMOUNT\_PARKINGS; p++)  {  cout << floorsAndParkings[f][p] << " | ";  }  cout << endl << "---------------------------------------------------" << endl;  }  }  } while (exit != '0');    return 0;  } |

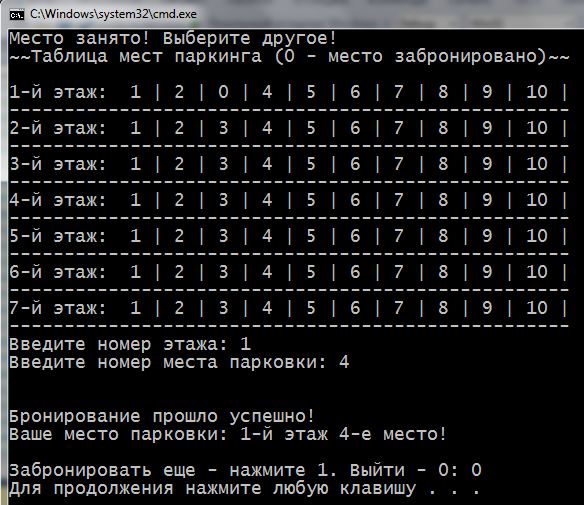
**Мы использовали [цикл for](https://purecodecpp.com/archives/623" \o "Цикл for в C++" \t "_blank) ,в строках 15 – 24, для записи данных в массив и одновременно отображения их на экране. Если представлять этот двумерный массив как таблицу – то внешний цикл for проходит по индексам строк  – от 0-й до 6-й. Вложенный цикл – по индексам столбцов (по ячейкам строк таблицы) – от 0-й до 9-й.**

**В строках 32 – 82 находится цикл [do while](https://purecodecpp.com/archives/754" \o "Циклы while и do while в C++" \t "_blank). Его роль в том, чтобы снова и снова предлагать забронировать место для автомобиля, пока это необходимо пользователю. В нем находятся два вложенных цикла do while. Они реализовывают выбор этажа и места для парковки с защитой от некорректного ввода значений.**

**Строки 57 – 81 содержат блок if else , который, в случае корректного выбора пользователя выводит сообщение об успешном бронировании. Если же место занято (ячейка содержит значение 0) – сообщает об этом, предлагает повторить выбор этажа и места и отображает обновлённую схему паркинга, где отмечены забронированные места.**

Работает это так:

[](https://purecodecpp.com/wp-content/uploads/2014/07/dvumernie-massivi-c--7.jpg)

[](https://purecodecpp.com/wp-content/uploads/2014/07/dvumernie-massivi-c--8.jpg)

**Многомерные массивы**

**Заполнение массива значениями**

**Многомерный массив заполняется значениями с помощью вложенных циклов. Причём, как правило, количество циклов совпадает с размерностью массива:**

**Задача № 4,№5 (оценка 4)**

#include <iostream>

#include <iomanip>

using namespace std;

const unsigned int DIM1 = 3;

const unsigned int DIM2 = 5;

int ary[DIM1][DIM2];

int main() {

**for (int i = 0; i < DIM1; i++) {**

**for (int j = 0; j < DIM2; j++) {**

**ary[i][j] = (i + 1) \* 10 + (j + 1);**

**}**

**}**

// ...

В этом примере каждому элементу массива присваивается значение, первая цифра которого указывает номер строки, а вторая цифра — номер столбца для этого значения (нумерация с 1).

**Вывод значений массива на консоль**

В продолжение предыдущего примера можно написать:

**for (int i = 0; i < DIM1; i++) {**

**for (int j = 0; j < DIM2; j++) {**

**cout << setw(4) << ary[i][j];**

**}**

**cout << endl;**

**}**

**return 0;**

**}**

В результате получим следующий вывод на консоль:

**11 12 13 14 15**

**21 22 23 24 25**

**31 32 33 34 35**

Для трёхмерного массива можно написать код, использующий те же приёмы:

**Задача № 5**

#include <iostream>

#include <iomanip>

using namespace std;

const unsigned int DIM1 = 3;

const unsigned int DIM2 = 5;

const unsigned int DIM3 = 2;

int ary[DIM1][DIM2][DIM3];

int main() {

for (int i = 0; i < DIM1; i++) {

for (int j = 0; j < DIM2; j++) {

for (int k = 0; k < DIM3; k++) {

ary[i][j][k] = (i + 1) \* 100 + (j + 1) \* 10 + (k + 1);

cout << **setw(4**) << ary[i][j][k];

}

cout << endl;

}

cout << endl;

}

return 0;

}

**Дополнительная задача (на 5)**

**Объявить, заполнить 4-х мерный целочисленный массив, вывести на консоль результат**

**Контрольные вопросы (для всех)**

**1.Как оценить эффективность того или иного метода сортировки?**

**2. Для чего используется сортировка?**

**3. В каких случаях затраты на сортировку оправданы?**

**4. Как добавляются новые данные в отсортированный массив?**

**5. Для чего используется цикл [DO WHILE](https://purecodecpp.com/archives/754" \o "Циклы while и do while в C++" \t "_blank) и IF ELSE и FOR в задаче № 3**