Содержание

[ОБЩАЯ ИНФОРМАЦИЯ 2](#_heading=h.gjdgxs)

[Тема занятия](#_heading=h.30j0zll) 2

[Цели и задачи 2](#_heading=h.1fob9te)

[Ожидаемый результат 2](#_heading=h.3znysh7)

[Структура занятия 3](#_heading=h.tyjcwt)

[ОПИСАНИЕ ЗАНЯТИЯ 4](#_heading=h.3dy6vkm)

# ОБЩАЯ ИНФОРМАЦИЯ

### Тема занятия

Указатели. Массивы.

### Цели и задачи

* Познакомиться с понятием массива
* Познакомиться с понятием указателя
* Научиться работать с адресами памяти компьютера

### Ожидаемый результат

По результатам занятия Слушатель должен знать:

* Что такое массив
* Что такое указатель
* Что такое индекс в программировании

По результатам занятия Слушатель должен уметь:

* Понимать и разбираться в типах массивов
* Использовать указатели для работы с массивами
* Использовать цикл индексы для перемещения и получения данных из массива

### Структура занятия

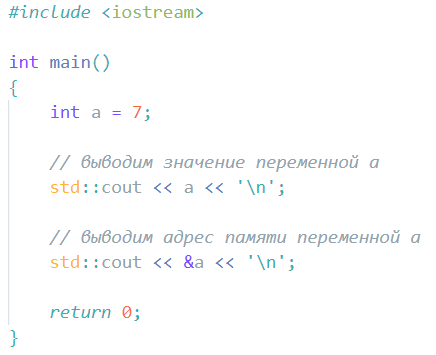
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Тайминг занятия** | |  |  |  | Таблица 1 |
| № | Этапы | Что делает преподаватель | Что делает Слушатель | Время | Общее время |
| 1 | Приветственное слово преподавателя |  |  | 5 мин. | 5 мин. |
| 2 | Повторение пройденного |  |  | 5 мин. | 40 мин. |
| 3 | Теоретическая часть |  |  | 30 мин. |
| 4 | Вопросы по теоретической части |  |  | 5 мин. |
|  | *Перерыв* |  |  | 15 мин. | 15 мин. |
| 5 | Практическая часть |  |  | 40 мин. | 45 мин. |
| 6 | Рефлексия и вопросы |  |  | 5 мин. |

# ОПИСАНИЕ ЗАНЯТИЯ

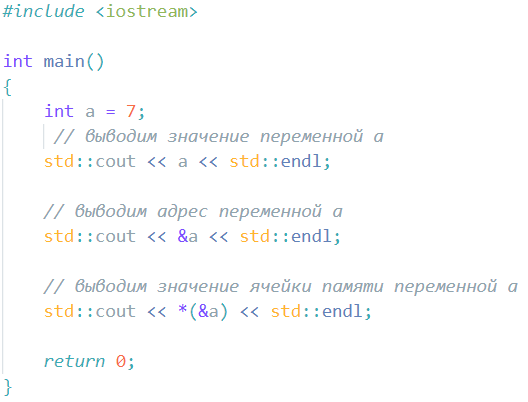
**Теоретическая часть**

**Оператор адреса (&)**

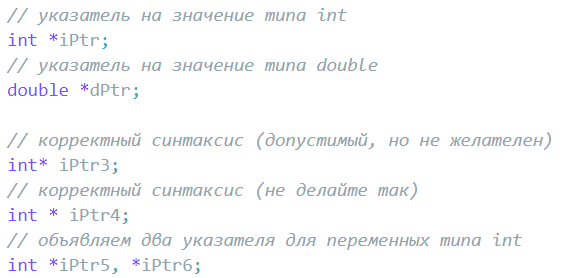
При выполнении инициализации переменной, ей автоматически присваивается свободный адрес памяти, и, любое значение, которое мы присваиваем переменной, сохраняется в этом адресе памяти



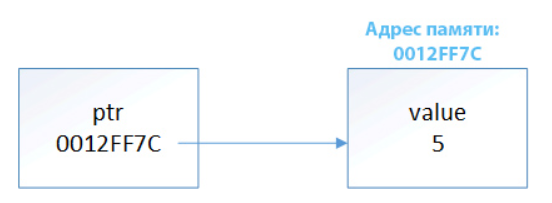
**Оператор разыменования**(**\***) позволяет получить значение по указанному адресу:

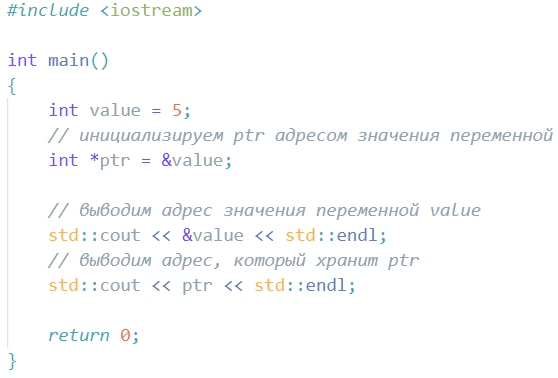


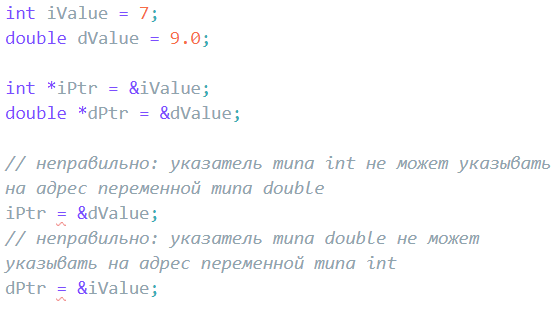
**Указатель** — это переменная, значением которой является адрес (ячейка) памяти. Указатели объявляются точно так же, как и обычные переменные, только со звёздочкой между типом данных и идентификатором:

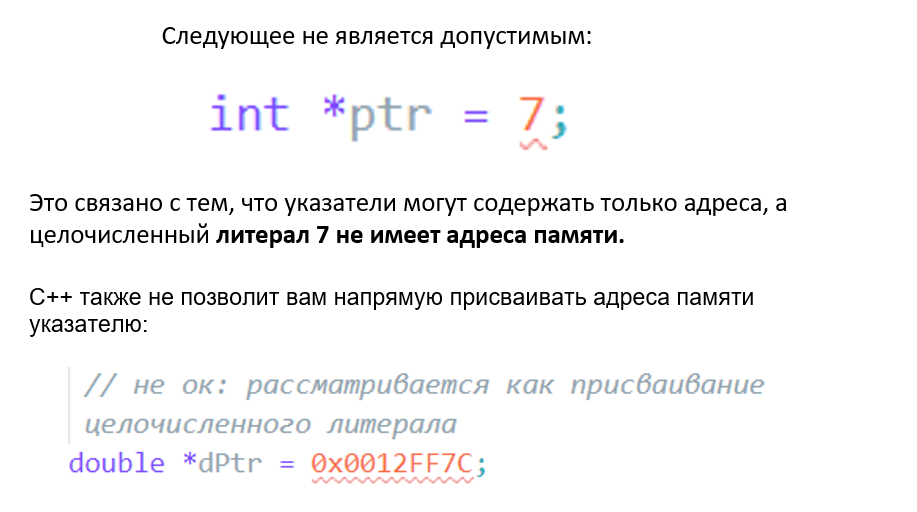


Поскольку указатели содержат только адреса, то при присваивании указателю значения — это значение должно быть адресом. Для получения адреса переменной используется оператор адреса:

****

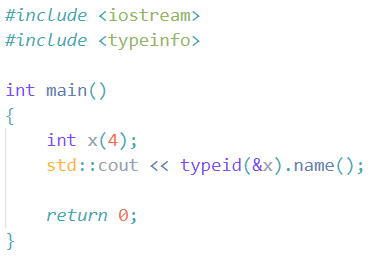
****

****

****

Оператор адреса возвращает указатель

Стоит отметить, что оператор адреса (&) не возвращает адрес своего операнда в качестве литерала. Вместо этого он возвращает указатель, содержащий адрес операнда, тип которого получен из аргумента (например, адрес переменной типа int передаётся как адрес указателя на значение типа int):

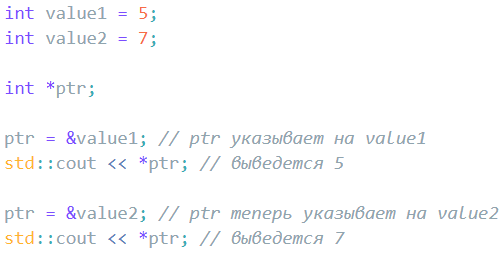
****

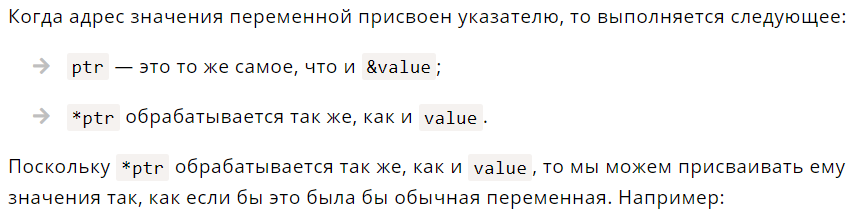
**Разыменование указателей**

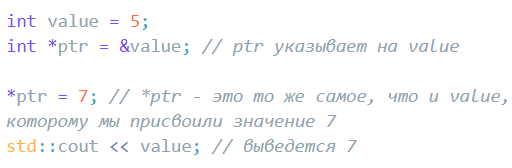
Как только у нас есть указатель, указывающий на что-либо, мы можем его разыменовать, чтобы получить значение, на которое он указывает. Разыменованный указатель — это содержимое ячейки памяти, на которую он указывает:

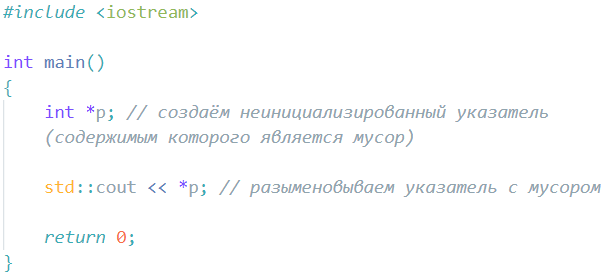
Вот почему указатели должны иметь тип данных. Без типа указатель не знал бы, как интерпретировать содержимое, на которое он указывает (при разыменовании). Также, поэтому и должны совпадать тип указателя с типом переменной. Если они не совпадают, то указатель при разыменовании может неправильно интерпретировать биты (например, вместо типа double использовать тип int).

Одному указателю можно присваивать разные значения:





****

****

**Массивы**

Массивы используются для хранения коллекций информации, но может быть полезным представлять массив как коллекцию переменных одинакового типа.

Вместо объявления множества переменных и хранения в них индивидуальных значений вы можете объявить один массив для хранения всех этих значений.

При объявлении массива укажите тип его элементов, а также и количество хранимых им элементов.

В примере, переменная a объявлена как массив пяти значений целочисленного типа [указанных в квадратных скобках]

Вы можете инициализировать массив указав все его значения.

Значения представлены в форме списка, разделены запятыми, закрыты внутри {фигурных скобок}.

Количество значений между фигурных скобок { } не должно превышать число элементов, объявленных в квадратных скобках [ ].

**Инициализация массивов**

Если вы опустите размер массива, то будет создан массив достаточно большого размера для хранения инициализации.

Таким образом создается массив идентичный созданному в прошлом примере.

**Индексация**

Каждый элемент, или член массива имеет свой индекс, который отмечает конкретную позицию каждого элемента.

Первый элемент массива имеет индекс равный 0, второй имеет индекс равный 1.

Для доступа к элементам массива, проиндексируйте имя массива путем подстановки индекса элемента в квадратные скобки после имени массива.

**Получение доступа к элементам массива**

Индексы могут быть также использованы для присвоения нового значения элементу.

В этой программе присваивается число 100 третьему элементу массива.

В этой программе присваивается число 100 третьему элементу массива.

**Массивы в циклах**

Иногда необходимо перебирать элементы массива, присваивая им значения, основанные на некоторых вычислениях.

Обычно это выполняется с использованием циклов.

**Массивы в циклах**

Каждому элементу массива присвоено значение 42.

Переменная x в цикле используется как индекс массива.

Последний индекс массива равен 4, поэтому условием цикла for является x<5.

**Массивы в циклах. Практика.**

Давайте выведем каждый индекс и соответствующее значение в массиве.

**Массивы в циклах. Практика.**

Давайте выведем каждый индекс и соответствующее значение в массиве.

Объявим массив и переменную sum, в которой будет храниться сумма всех элементов.

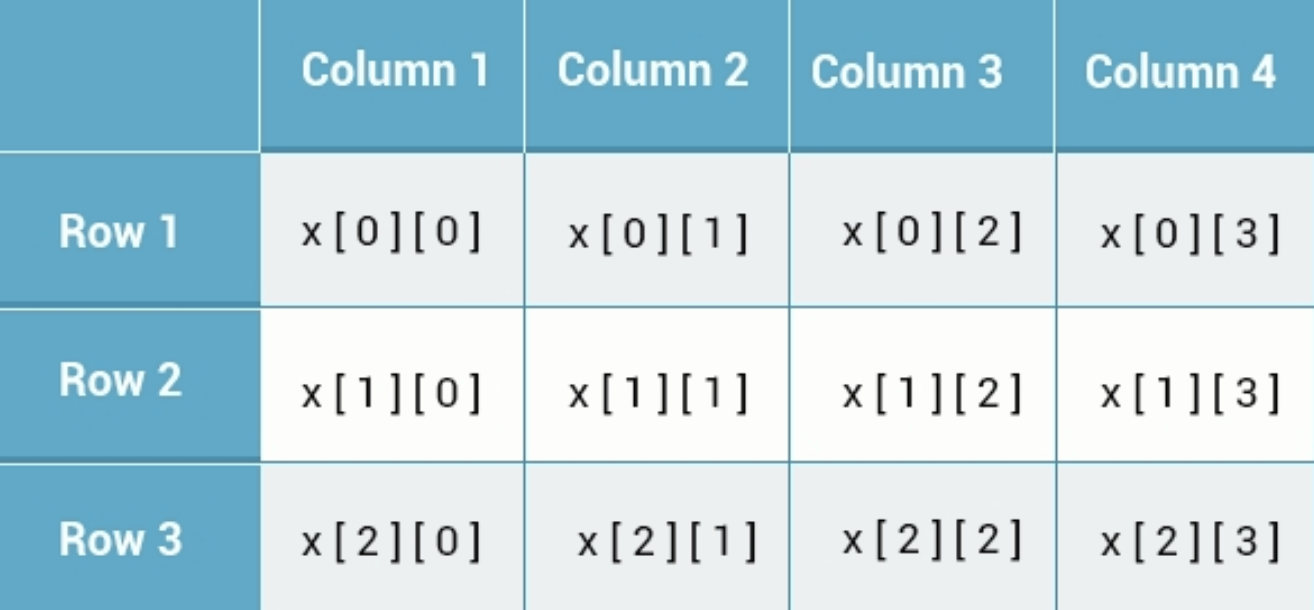
Затем, применим цикл for для прохождения по всем элементам массива и добавим соответствующие значения элементов к переменной sum.

**Многомерные массивы**

Многомерный массив хранит один или более массивов.

Таким образом мы создали двумерный массив целых чисел 3x4



Представляйте этот массив как таблицу, состоящую из 3 строк и 4 столбцов.

**Двумерные массивы**

Многомерные массивы могут быть инициализированы с использованием заключения значений внутри специальных скобок для каждой строки.

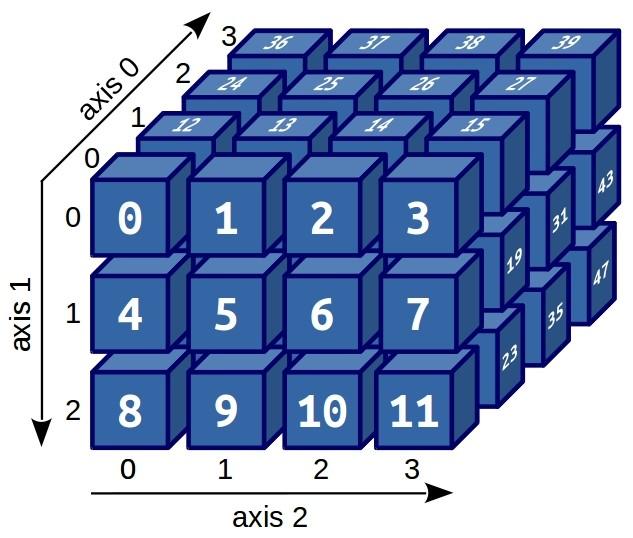
Далее следует массив из трёх строк и 3 столбцов:

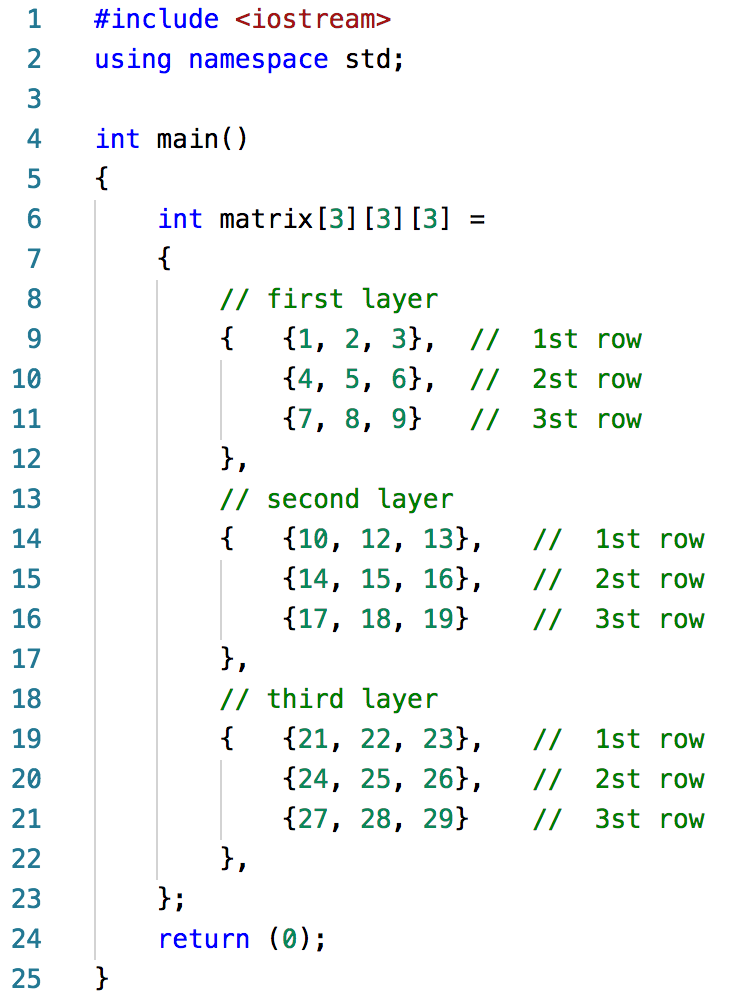
Вы также можете использовать такую же инициализацию в одной строке.

**Многомерные массивы**

Можно ли создать трёхмерный массив, четырёхмерный, пятимерный … n-мерный массив?

**Здесь главное рассказать ученикам о применении таких структур на практике.**

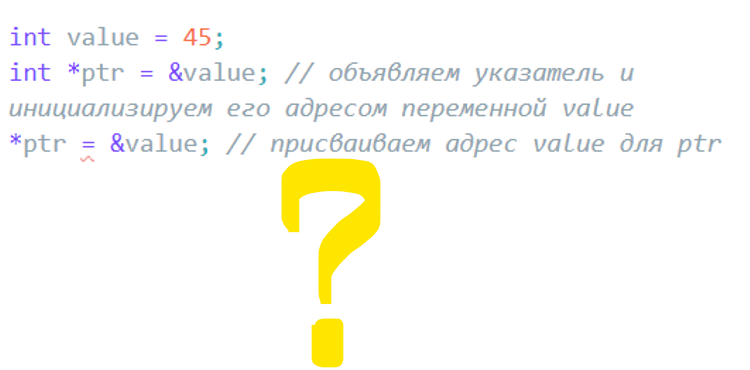
****

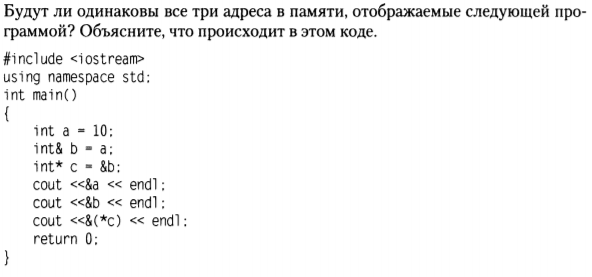


В приведенном выше примере объявлен трехмерный массив строк. Как мы делали ранее, можно использовать номера индексов для доступа и модификации элементов.



**Практическая часть**

****

**