Тема 8: Массивы. Строки как массив символов

- Структурированные типы данных
- Основные понятия массивов
- Одномерные массивы
- Двумерные массивы
- Виды строк в C++
- Строковые переменные и константы
- Одномерный массив строк
- Операции со строками

Структурированные (составные) типы данных

Структурированные (составные) типы данных - типы данных, состоящие из простых типов:

- **целых;**
- **вещественных;**
- **символьных;**
- **логических**

К таким типам данных относятся:

- Массивы и строки;
- Структуры и объединения;
- классы

Основные понятия массивов

Массив (табличная величина, таблица) – это упорядоченный набор фиксированного количества однотипных элементов, доступ к которым осуществляется с помощью индексов.

Массив – это последовательная группа ячеек памяти, имеющих одинаковое имя и одинаковый тип.

Имя массива – идентификатор.

Размер массива – количество элементов, т.е. число ячеек памяти.

Основные понятия массивов

Все элементы массива имеют порядковый номер – индекс.

Индекс может быть:

- переменной
- константой
- арифм. выражением целого типа

Индексация элементов с нуля.

По количеству индексов, которые нужны для доступа к конкретному элементу массива, разделяют одномерные, двумерные, ***n***-мерные массивы.

Основные понятия массивов

Одномерный массив (линейная таблица, вектор) – это массив, каждый элемент которого определяется одним индексом.

Элементы одномерного массива в памяти компьютера располагаются последовательно.

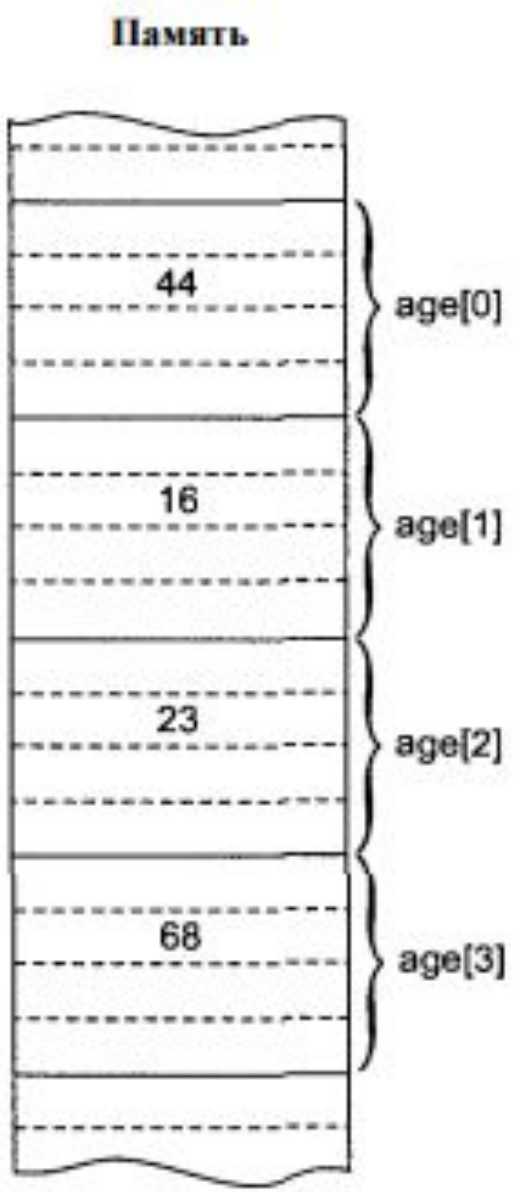
Основные понятия массивов

Двумерный массив (матрица) – это массив, каждый элемент которого определяется двумя индексами.

Первый индекс указывает на номер строки, второй – на номер столбца в этой строке.

Одномерные массивы

Массивы занимают область в памяти. Программист указывает тип каждого элемента, количество элементов, требуемое каждым массивом, и компилятор может зарезервировать соответствующий объем памяти.



Одномерные массивы

Объявление одномерного массива

```
<тип элементов> <имя массива>[<размерность>];
```

Способы объявления:

```
1)   const int n = 10;
     int A[n];
```

где n - его размер (именованная константа),
число ячеек,
A – имя переменной, в которой хранится массив.

```
2) double b[100], x[27];
   char s[30];//строка символов
```

Одномерные массивы

Доступ к элементам одномерного массива (индексация)

<имя_массива>[индекс]

Например:

```
int mas[10], a, b;
b = 20;
mas[5] = 25;
a = mas[5] + b;
```

Одномерные массивы

Инициализация одномерного массива

1 способ: при объявлении массива после знака = в фигурных скобках через запятую указать начальные значения

```
int n[5] = {1, 3, 5, 7, 11};
```

Если начальных значений больше, чем размерность массива - синтаксическая ошибка.

```
int n[5] = {32, 27, 64, 18, 95, 14};
```

Если начальных значений меньше, чем элементов в массиве, оставшиеся элементы автоматически получают нулевые начальные значения:

```
int n[10] = {1, 3, 5, 7, 11};
```

Одномерные массивы

Обнуление элементов массива:

```
int n[10] = {0};
```

ЗАМЕЧАНИЕ:

1) явно присваивает нулевое начальное значение первому элементу и неявно – остальным элементам.

2) автоматически массивы не получают нулевые начальные значения неявно: нужно присвоить нулевое начальное значение по крайней мере первому элементу для того, чтобы автоматически были обнулены оставшиеся элементы.

Одномерные массивы

Если размер массива не указан в объявлении со списком инициализации, то количество элементов массива будет равно количеству элементов в списке начальных значений.

Например:

```
int n[ ] = {1, 2, 3, 4, 5};
```

Одномерные массивы

2 способ: используя индексацию, присвоить начальные значения уже в программе в цикле **for**:

- ввести значения элементов с клавиатуры
- вычислить по формуле
- задавать с помощью генератора псевдослучайных чисел

Одномерные массивы

Ввод и вывод элементов одномерного массива

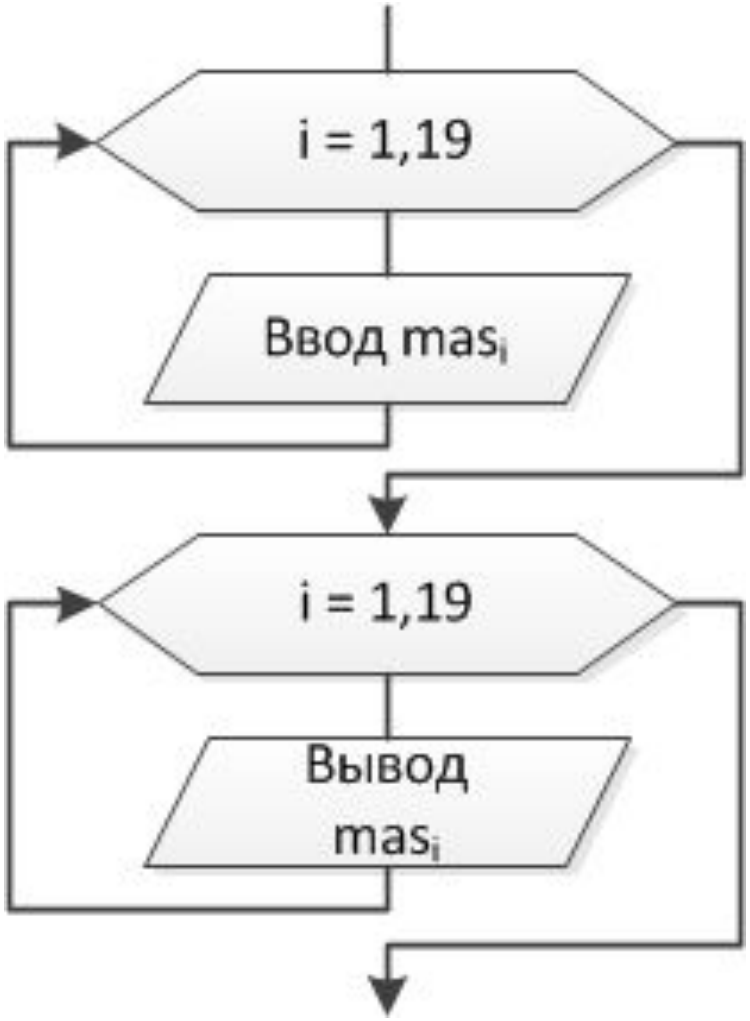
При работе с массивами необходимо обязательно вывести на экран исходный и результирующий массив.

ЗАМЕЧАНИЕ: В случае ввода элементов с клавиатуры используются два цикла **for** для ввода и вывода исходного массива, а третий **for** - для вывода результирующего массива.

```
cout << "Исходный массив: " << endl;
for (int i = 0; i < 10; i++)
```

Одномерные массивы

Программа 1 заполняет с клавиатуры массив из 20 элементов целыми числами и выводит его на экран

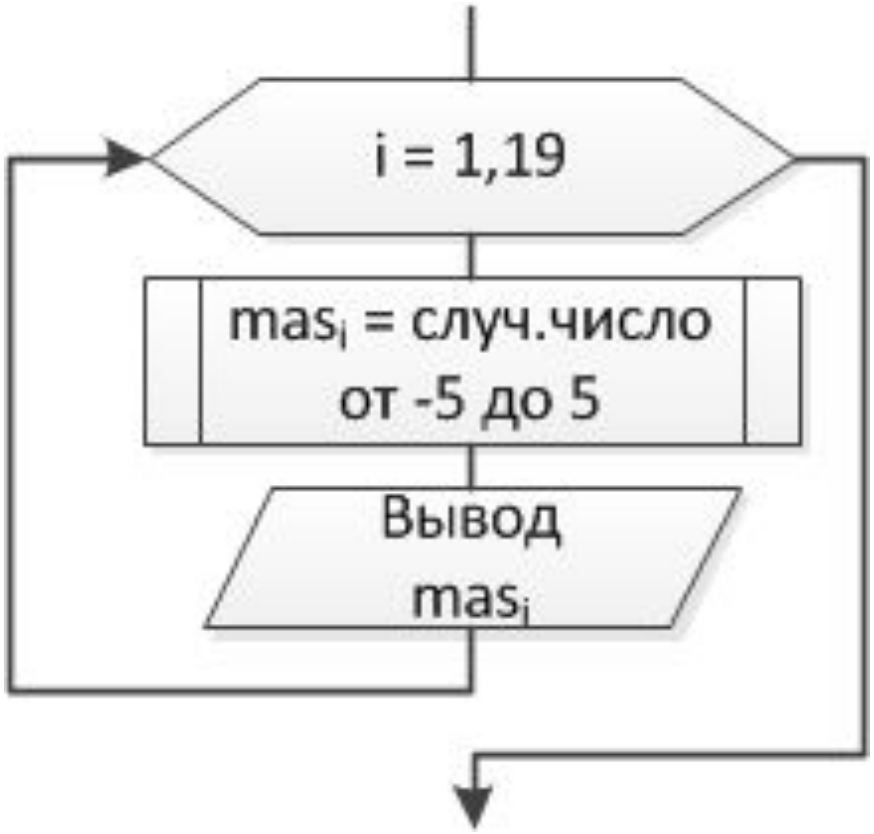


Одномерные массивы

```
int mas[20]; //описываем массив
cout << “Введите элементы: ”;
for (int i = 0;i < 20;i++)
{
    cin >> mas[i];
}
cout << “Исходный массив: ” << endl;
for (int i = 0;i < 20;i++)
{
  cout << mas[i] << "\t”;
}
```

Одномерные массивы

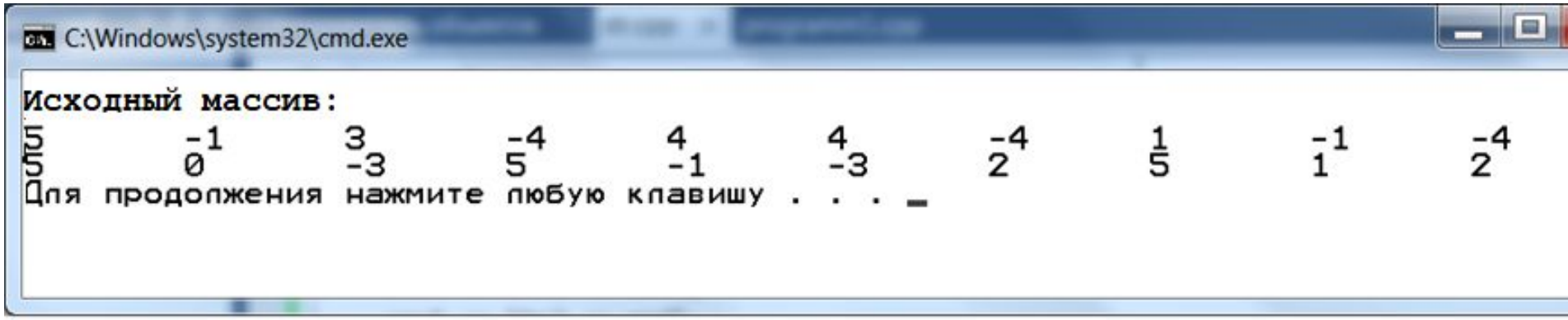
Программа 2 заполняет массив из 20 элементов целыми случайными числами в диапазоне от -5 до 5 и выводит его на экран



Одномерные массивы

```
//подключение и инициализация ГСЧ
int mas[20]; //описываем массив
cout << “Исходный массив: ” << endl;
for (int i = 0;i < 20;i++)
{
  mas[i]=rand()%11-5; //от -5 до 5
cout << mas[i] << "\t”;
}
```

Одномерные массивы



Двумерные массивы

Объявление двумерного массива

```
<тип элементов> <имя массива>[<количество строк>][<количество столбцов>];
```

Например:

```
int a[2][3];
double b[50][50], c[10][10];
char Trio[100][30];// одномерный массив
        // строк, состоящий  из 100
// строк, каждая длинной 30.
```

ЗАМЕЧАНИЕ: размер строк и столбцов можно указывать через именованные константы

Двумерные массивы

Доступ к элементам двумерного массива

<имя_масс>[номер стр.][номер стб.]

Например:

```
int mas[10][5], a, b;
b = 20;
mas[5][1] = 25;
a = mas[5][1] + b;
```

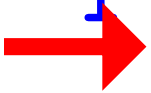
ЗАМЕЧАНИЕ: индексация строк и столбцов начинается с нуля

Двумерные массивы

Инициализация двумерного массива

1 способ: при объявлении массива после знака = в фигурных скобках через запятую сгруппировать в фигурных скобках строки массива

```
int B[2][2]={{1,2},{3,4}};
```

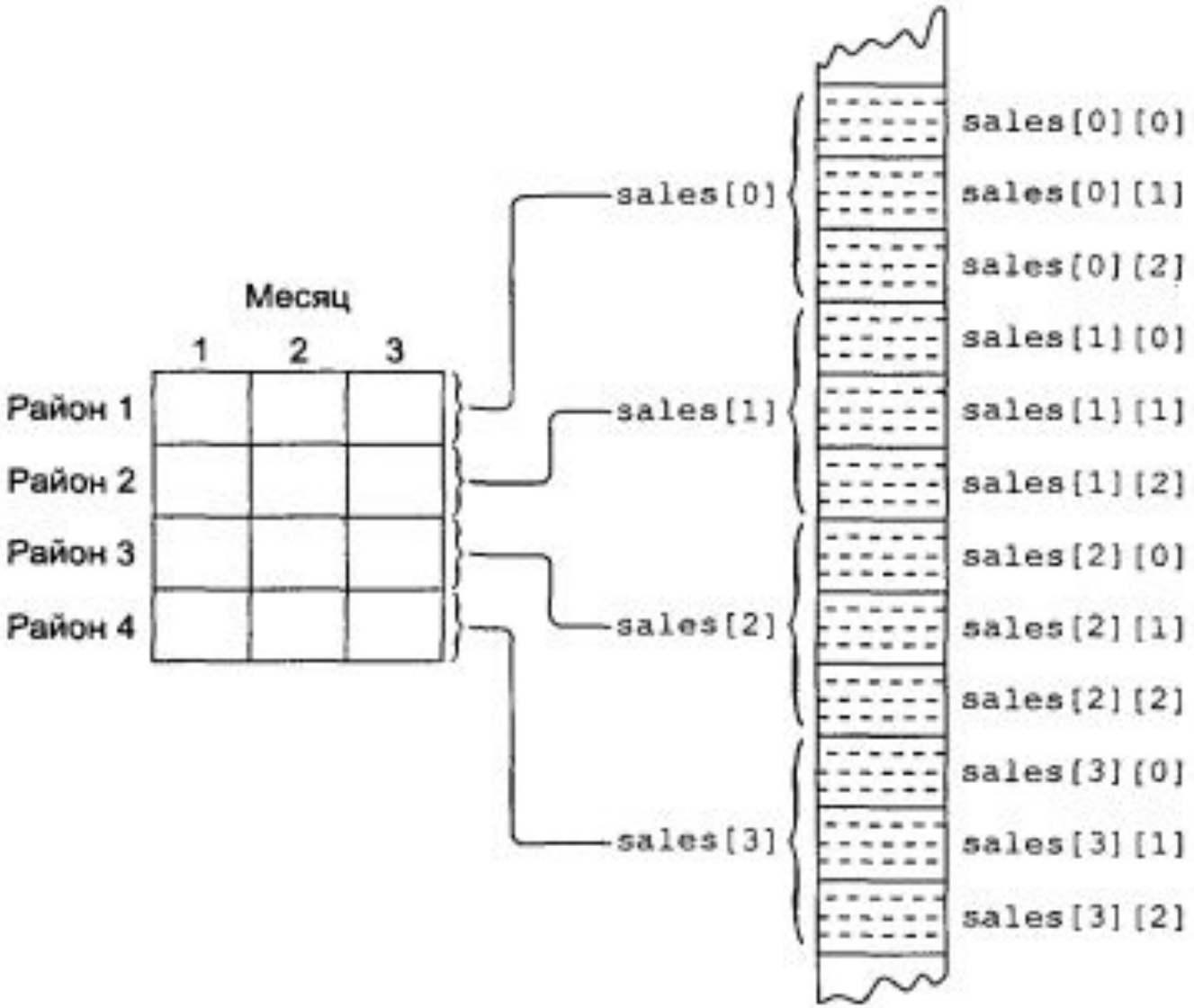


```
1 2
3 4
```

Если начальных значений больше, чем размерность массива - синтаксическая ошибка.

```
int B[2][2] = {{1,0,1},{3,4,-5}};
//это не массив B[2][3]
```

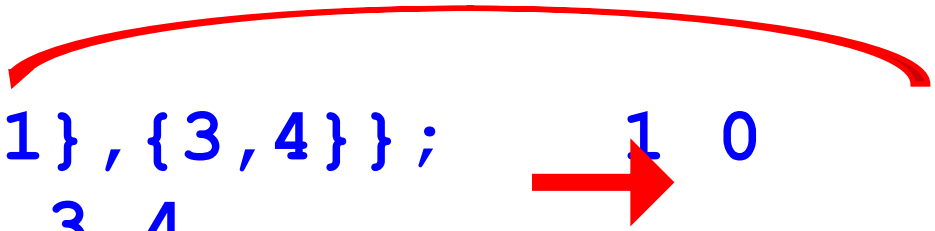
Двумерные массивы



Двумерные массивы

Если начальных значений меньше, чем элементов в массиве, оставшиеся элементы автоматически получают нулевые начальные значения:

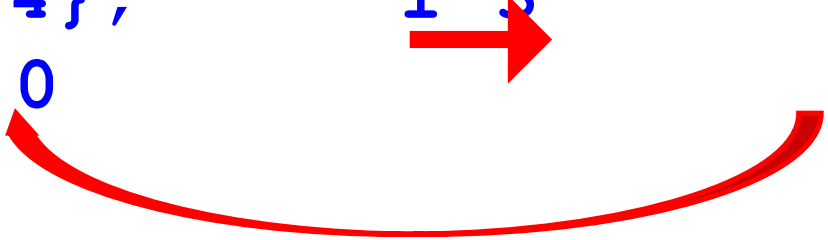
```
int B[2][2]={{1},{3,4}};
```



```
1 0
3 4
```

Отсутствие внутренних скобок:

```
int B[2][2]={1,3,4};
```



```
1 3
4 0
```

Обнуление элементов массива:

```
int n[10][5] = {0};
```

Двумерные массивы

2 способ: используя индексацию, присвоить начальные значения уже в программе в циклах **for**:

- значения элементов ввести с клавиатуры
- вычислять по формуле
- задать с помощью генератора псевдослучайных целых чисел.

Двумерные массивы

Вывод элементов двумерного массива

производится во вложенных циклах **for:**

- внешний цикл – строки
- внутренний - столбцы.

ЗАМЕЧАНИЕ: В случае ввода элементов с клавиатуры исполь-зуются два вложенных цикла **for** для ввода и вывода исходного массива, а третий вложенный **for** - для вывода результирующего массива.

Двумерные массивы

Программа 3 заполняет с клавиатуры двумерный целочисленный массив размерностью 5x5
и выводит его на экран

```
int mas[5][5];//объявляем матрицу
cout << "Введите элементы массива: "
   << endl;
for (int i = 0;i < 5;i++)
{
   cout << "Введите элементы " <<
      i << "-й строки:" << endl;
   for(int j = 0;j < 5;j++)
      cin >> mas[i][j];
}
```

Двумерные массивы

```
cout << “Исходный массив: ” << endl;
for (int i = 0;i < 5;i++)
{ //вывод на экран элементов матрицы построчно
for(int j = 0;j < 5;j++)
     cout<<mas[i][j]<<"\t";//выводим
                    //на экран
  cout<<“endl"; //закончив строку - ENTER
}
```

Обработка элементов массива

- Поиск элементов, удовлетворяющих условию (например, макс или мин)
- Вычисление суммы, произведения и т.д. элементов
- Упорядочивание элементов массива