**МАССИВЫ**

[**МАССИВЫ** 1](#_Toc337139393)

[**Цели обучения:** 2](#_Toc337139394)

[**0.** **Введение** 3](#_Toc337139395)

[1. Объявление массива 4](#_Toc337139396)

[2. Обращение к элементам массива 5](#_Toc337139397)

[**Пример(1)** программы поиска значения максимального элемента целочисленного массива из n элементов (n<=100): 5](#_Toc337139398)

[3. Типовые алгоритмы работы с массивами 7](#_Toc337139399)

[**Пример(2)** программы подсчета отрицательных чисел в массиве из n (n<=100) вещественных чисел: 7](#_Toc337139400)

[**Пример(3)** программы сортировки массива из n вещественных чисел (n<=100) по возрастанию методом прямого обмена (пузырьковым методом): 7](#_Toc337139401)

[**Пример(4)** программы удаления из массива, содержащего n целых чисел, последнего отрицательного элемента: 8](#_Toc337139402)

[**Пример(5)** программы формировния нового массива из отрицательных чисел целочисленного массива размера n (n<=100): 9](#_Toc337139403)

[4. Многомерные массивы 10](#_Toc337139404)

[Пример(6) программы вычисления суммы чисел, расположенных выше главной диагонали матрицы целых чисел из n строк и n столбцов (n<=100): 10](#_Toc337139405)

[5. Динамические массивы 12](#_Toc337139406)

[**6.** **Вопросы и задания** 13](#_Toc337139407)

[**7.** **Список литературы** 14](#_Toc337139408)

**Цели обучения:**

1. Определить, что такое массив.

2. Рассмотреть различные виды массивов.

1. Обучиться базовым алгоритмам работы с массивами.
2. **Введение**

Массив – это составной тип данных, используемый в программе для представления конечной последовательности данных одного типа. Под элементы массива выделяются смежные участки памяти. Доступ к элементу массива осуществляется по индексу.

[В начало лекции](#_top)

1. Объявление массива

Синтаксис объявления массива:

*Тип элементов имя массива [размер массива];*

Пример объявления массивов:

int a[100]; //целочисленный массив из 100 элементов

float b[10]; //массив из 10 вещественных чисел

При объявлении массива можно указать начальные значения для его элементов:

int a[4]={10,20,30,40};

или

int a[]={10,20,30,40};

В последнем случае размер памяти под массив определяется автоматически по количеству констант. Можно проинициализировать только часть первых элементов массива:

int a[4]={10,20};

Формирование массивов типа char отельный случай. Возможны 2 варианта задания массивов такого типа:

char str[] = {“H”, “e”, “l”, “l”, “o”};

char cstr[] = “Hello”;

Во втором случае в конец массива добавляется символ ‘\0’, что увеличивает размер массива.

[[1](#l1), [4](#l4)]

[В начало лекции](#_top)

1. Обращение к элементам массива

Чаще всего к элементу массива обращаются с помощью индекса (номера) элемента.

Синтаксис обращения:

*Имя\_массива [индекс]*

Индекс определяет положение элемента в массиве и задается выражением. Выражение должно быть целого типа.

Примеры обращения к элементам массива:

a[1]=2; a[i]=3;

a[i+2]=a[i];

Элементы массива в языке С++ нумеруются с 0. Поэтому в массиве, объявленном как массив из 100 элементов, первый элемент имеет индекс равный 0, а последний – 99. Контроль выхода индекса элемента массива за границы допустимых значений при выполнении программы в С++ не выполняется.

В языке С++ отсутствуют операции над массивом. Исключение составляет операция индексации []. Над элементами массива можно выполнять операции, которые допустимы для типа элементов массива. [[1](#l1)]

Существует и другой способ обращения к элементам массива.

Имя массива также является адресом, по которому в памяти находится первый элемент массива. Соответственно, при необходимости обратиться к какому-либо элементу массива, можно использовать запись вида (p+i)[0], где i – номер элемента, что равносильно записи p[i]. [[2](#l2), [3](#l3)]

**Пример(1)** программы поиска значения максимального элемента целочисленного массива из n элементов (n<=100):

#include <iostream.h>

#include <conio.h>

void main ()

{

int a[100]; //массив

int n, i; //количество элементов и индекс элемента

int max; //значение максимального элемента

cout<<”n? “;

cin>>n;

cout<<”a? “;

//Ввод массива

for (i=0; i<n; i++)

cin>>a[I];

max=a[0];

for (i=1; i<n; i++) //цикл поиска максимального элемента

if (a[i]<max

max=a[I];

cout<<”max=”<<max;

getch();

}

Тесты:

n=5

a: 1, 8, -9, 3, 5 max=8

a: 1, 8, 2, 8, 6 max=8

a: 1, 1, 1, 1, 1 max=1

[В начало лекции](#_top)

[В начало раздела](#Два)

1. Типовые алгоритмы работы с массивами

Массив используется для хранения последовательности однотипных данных и выполнения над этой последовательностью операций. Типовые операции над массивами:

* поиск в массиве;
* вычисление суммы, количества и произведения определенных элементов массива;
* сортировка массива;
* удаление определенных элементов из массива;
* добавление элементов в массив;
* формирование массива.

**Пример(2)** программы подсчета отрицательных чисел в массиве из n (n<=100) вещественных чисел:

#include <iostream.h>

#include <conio.h>

void main ()

{

float a[100]; //массив чисел

int n, i; //количество элементов в массиве и индекc элемента

int k; //количество отрицательных чисел в массиве

cout<<”n? “;

cin>>n;

cout<<”a? “;

for (i=0; i<n; i++) //цикл ввода массива

cin>>a[I];

k=0;

for (i=0; i<n; i++) //цикл поиска и подсчета отрицательных чисел

if (a[i]<0

k++;

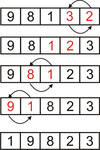
cout<<”k=”<<k;

getch();

}

**Пример(3)** программы сортировки массива из n вещественных чисел (n<=100) по возрастанию методом прямого обмена (пузырьковым методом):

Схема метода для массива из 5 элементов(первый прогон):



#include <iostream.h>

#include <conio.h>

void main ()

{

float a[100]; //массив чисел

int n, i; //количество элементов в массиве и индекc элемента

int k; //номер прохода массива при сортировке

cout<<”n? “; cin>>n;

cout<<”a? “;

for (i=0; i<n; i++) //цикл ввода массива

cin>>a[i];

//Сортировка пузырьковым методом

for (k=1; k<n; k++)

for (i=0; i<k-1; i++)

if (a[i]>a[i+1])

{int t; t=a[i]; a[i]=a[i+1]; a[i+1]=t;} //перестановка элементов

cout<<”a:”<<endl;

for (i=0; i<n; i++) //цикл вывода массива на экран

cout<<a[i]<<’ ‘;

getch();

}

**Пример(4)** программы удаления из массива, содержащего n целых чисел, последнего отрицательного элемента:

#include <iostream.h>

#include <conio.h>

void main ()

{

int a[100]; //массив чисел

int n, i ,j; //количество элементов массива, индекcы элементов

cout<<”n? “; cin>>n;

cout<<”a? “;

for (i=0; i<n; i++) //цикл ввода массива

cin>>a[i];

i=n-1; //начальное значение индекса для поиска с конца

while (i>=0 && a[i]>=0) //цикл поиска отрицательного числа

i--;

if (i>=0) //число найдено, номер числа i

{

for (j=i; j<n-1; j++) //удаление a[i], сжатие массива

a[j]=a[j+1];

n--; //количество элементов будет на 1 меньше

}

for (i=0; i<n; i++) //вывод массива

cout<<a[I]<<’ ‘;

getch();

}

Тесты:

n=5, a: 1 –2 –3 5 6 результат: a: 1 –2 5 6, n=4

n=5, a: -1 –2 –3 –4 -6 результат: a: -1 –2 –3 –4, n=4

n=5, a: 1 2 3 4 5 результат: a: 1 2 3 4 5, n=5

**Пример(5)** программы формировния нового массива из отрицательных чисел целочисленного массива размера n (n<=100):

#include <iostream.h>

#include <conio.h>

void main ()

{

int a[100], b[100]; //исходный и формируемый массивы

int n, i, j; //количество элементов в массиве и индекcы элементов

cout<<”n? “; cin>>n;

cout<<”a? “;

for (i=0; i<n; i++) //цикл ввода массива

cin>>a[i];

j=0; //индекс первого элемента в формируемом массиве

for (i=0; i<n; i++) //цикл поиска a[i]<0 и запись их в массив

if (a[i]<0)

{ b[j]=a[I]; j++;}

if (j==0) //новый массив пуст

cout<<”b is empty”;

else //вывод сформированного массива

{

cout<<”b:”;

for (i=0; i<j; i++)

cout<<b[i]<<’ ‘;

}

getch();

}

Тесты:

n=5, a: 1 -2 3 -4 -5 результат b: -2 -3 -5

n=5, a: -1 -2 -3 -4 -5 результат b: -1 -2 -3 -4 -5

n=5, a: 1 2 3 4 5 результат : b is empty

[В начало лекции](#_top)

[В начало раздела](#Три)

1. Многомерные массивы

Многомерный массив – это массив, компонентами которого являются массивы.

Пример объявления многомерного массива (двумерного массива) из двух элементов, каждый из которых является массивом из трех вещественных чисел:

float b[2][3];

Пример объявления трехмерного массива из четырех элементов, каждый из которых является массивом из 10 двумерных массивов целых чисел:

int a[4][10][15];

При обращении к элементу многомерного массива надо указывать индекс по каждому измерению. Каждый индекс заключается в квадратные скобки. Нумерация индексов по каждому измерению начинается с нуля, например, обращение к элементу, объявленного выше массива:

a[0][9][4]=25;

Многомерные массивы можно инициализировать. Примеры допустимых вариантов инициализации массива:

float b[2][3]={1.2, 1.3, 1.4, 1.6,1.7, 1.8};

float b[2][3]={{1.2, 1.3, 1.4},{1.6, 1.7, 1.8}};

float b[][]={{1.2, 1.3, 1.4},{1.6, 1.7, 1.8};

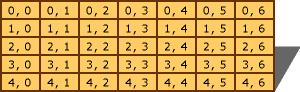
Многомерные массивы располагаются в памяти так, что быстрее всего меняется последний индекс. Так данные двумерного массива, объявленного как

float b[2][3];

располагаются в памяти в следующем порядке:

b[0][0], b[0][1], b[0][2], b[1][0], b[1][1], b[1][2]

На рисунке ниже представлена концептуальная модель двумерного массива с пятью строками и семью столбцами:



### Пример(6) программы вычисления суммы чисел, расположенных выше главной диагонали матрицы целых чисел из n строк и n столбцов (n<=100):

#include <iostream.h>

#include <conio.h>

void main()

{

int a[100][100]; //матрица целых чисел

int s; //сумма элементов матрицы

int i, j; //номера строк и столбцов

int n; //размер матрицы

cout<<”n? “; cin>>n;

//Ввод матрицы

cout<<”a? “;

for(i=0; i<n; i++)

for(j=0; j<n; j++)

cin>>a[i][j];

//Вычисление суммы элементов выше главной диагонали

s=0;

for(i=0; i<n-1; i++)

for(j=i+1; j<n; j++)

s+=a[i][j];

cout<<”s=”<<s;

getch();

}

[[1](#l1), [4](#l4), [5](#l5), [6](#l6)]

[В начало лекции](#_top)

[В начало раздела](#Четыре)

1. Динамические массивы

Рассмотренные ранее массивы называются статическими. Но что же делать, если мы не знаем заранее размеров массива? В этом случае, мы можем изначально задать массив в виде указателя на нулевой элемент ( int \*arr ), не выделяя памяти, а затем, узнав число элементов, инициализировать его:

arr1 = new int[n];

Двумерные динамические массивы задаются сложнее:

int \*\*arr2 = new int\*[dim1];

for(int i = 0; i < dim1; i++) arr2 = new int[dim2];

После создания динамического массива и работы с ним необходимо очищать память:

delete[] arr1; //в случае одномерного массива

for(int i = 0; i < dim1; i++) delete[]arr2[i]; // последовательное уничтожение строк

delete[] arr2; //в случае двумерного массива

[[2](#l2), [5](#l5), [7](#l7)]

[В начало лекции](#_top)

1. **Вопросы и задания**
2. С какого числа начинается нумерация элементов массива?[Отв1](#Отв1)
3. Сколько элементов будет содержать массив?[Отв2](#Отв2)

int a []= {3,6,2,8,-2}

char city[] = “Москва”

char city1[] = {“М”, “о”, “с”, “к”, “в”, “а”}

1. Создан массив указателей на целые числа. Какой элемент будет хранить адрес числа 7?[Отв3](#Отв3)

int \* mas [4];

\*(mas +1) = 7;

1. Напишите программу, вычисляющую суммы положительных и отрицательных элементов массива.
2. Напишите программу, вычисляющую произведение всех чисел двумерного массива 5\*8 элементов, которые делятся на 3 и 7 без остатка.
3. Напишите программу циклического сдвига одномерного массива на n позиций влево.

[В начало лекции](#_top)

1. **Список литературы**
2. Андреева Л.П. «Программирование. Курс лекций»
3. http://msdn.microsoft.com

1. <http://do.rksi.ru/library/courses/demo/>

1. <http://citforum.ru/programming/cpp_march/>

1. [http://code-live.ru](http://code-live.ru/)

1. <http://ru.wikipedia.org>
2. Шилдт Г. «С++ Базовый курс (3-е издание, 2010)»