Массивы в C#

Массивом называют **упорядоченную** совокупность элементов **одного типа**.

Каждый элемент массива имеет ***индексы***, которые определяют его расположение.

Число индексов характеризует **размерность** массива (одномерный, двумерный и т.д.).

Одномерные и двумерные массивы имеют наглядное практическое представление. Примером одномерного массива может служить линейная таблица или вектор:

Линейная таблица-одномерный массив с именем **A**:

2	-3	7	12	6	-35

Пример двумерного массива - это прямоугольная таблица или матрица:

Прямоугольная таблица - двумерный массив с именем **B:**

8	5	2	-5	0
3	11	-5	-7	4
12	45	9	10	23

При этом **1-й индекс показывает номер строки, а 2-й индекс - столбца.**

В языке C#, как и во многих других языках, индексы задаются целочисленным типом.

Каждый индекс изменяется в некотором диапазоне [0,n], то есть нумерация начинается с нуля!!!

Массивы в языке C# относятся к ссылочным типам, то есть являются динамическими. Поэтому память им отводится во время выполнения программы, в "куче".

При описании массива, сам массив не формируется, а создается только ссылка на него, имеющая неопределенное значение Null. Поэтому пока элементы массива не будут проинициализированы, использовать его в вычислениях нельзя.

Описание одномерного массива, без инициализации

При описании массива не указывается размер (количество элементов):

int [] Arr1;

Person [] Arr2;

Массив Arr1 будет содержать целые числа (то есть значения), а массив Arr2 – объекты класса Person (то есть ссылки на объекты).

Квадратные скобки в C# указываются после типа, перед именем массива.

Описание одномерного массива, с инициализацией

Существует два варианта инициализации.

В первом случае инициализация является явной и задается константным массивом.

Например:
double[] x= {5.5, 6.6, 7.7};

Элементы имеют индексы: 0, 1, 2.

Во втором случае создание и инициализация массива выполняется в объектном стиле с вызовом конструктора массива **new**. Это наиболее распространенная практика объявления массивов.

Пример: int[] d= new int[5];

Здесь 5 – количество элементов массива, а их индексы 0, 1, 2, 3, 4.

Во втором случае в динамической памяти создается сам массив, элементы которого инициализируются константами соответствующего типа (ноль для арифметики, пустая строка для строковых массивов), и ссылка связывается с этим массивом.

Если количество элементов массива заранее неизвестно, то сначала вводится количество элементов, а потом инициализируется массив.

Примеры описаний массива с инициализацией

Массив создается с помощью операции new:

```
//  все 4 элемента равны 0
int[] b = new int[4];
//  если указаны значения, new можно не писать
int[] c = { 61, 2, 5, -9 };
// размерность вычисляется
int[] d = new int[] { 61, 2, 5, -9 };
// избыточное описание
int[] e = new int[4] { 61, 2, 5, -9 };
```

Элементы массива нумеруются с нуля.

Цикл foreach

Цикл foreach – универсальный перечислитель для коллекций. Синтаксис:

foreach(<переменная> in <коллекция>)

Тут элемент <**переменная**> задает тип и имя переменной, которая при функционировании цикла foreach будет иметь значения элементов из коллекции.

Цикл имеет следующую семантику

«Для каждого элемента из коллекции делать».

Так как массив можно определить как коллекцию, то этот цикл может использоваться для перебора элементов массива. Значение элементов массива в этом цикле изменить нельзя!

Пример:

```
int[] a = new int[] { 61, 2, 55, -9 };
foreach (int i in a)
Console.WriteLine(i);
```

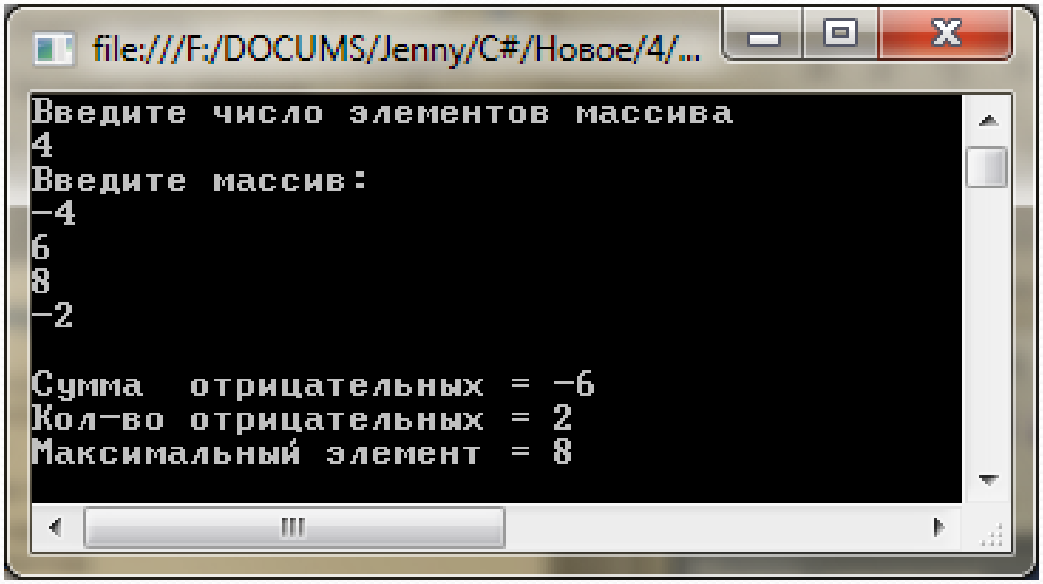
Результат работы: 61, 2, 55, -9

То есть выводятся не индексы элементов, а соответствующие им значения.

Пример 1

Ввести с клавиатуры количество элементов массива и сами элементы. Найти:

- сумму отрицательных элементов;
- количество отрицательных элементов;
- максимальный элемент.



Пример 1

```
// ввод количества элементов
int n =
Convert.ToInt32(Console.ReadLine());
//описание массива с инициализацией
нулями
int[] a = new int[n] ;
// ввод элементов массива в столбик
for (int i = 0; i < n; i++)
a[i]=Convert.ToInt32(Console.ReadLine()
);
```

```
long sum = 0; // сумма отрицательных
int num = 0; // количество отрицательных
// просмотр  массива и подсчет
for (int i = 0; i < n; i++)
    if (a[i] < 0)
        {
            sum = sum+a[i]; num=num+1;
        }
// вывод результата
Console.WriteLine("Сумма  отр = " + sum);
Console.WriteLine("Кол-во отр = " + num);
```

```
int max = a[0]; // начальное значение макс
//поиск максимального
foreach (int i in a) if (i > max) max = i;
Console.WriteLine("Максимальный элемент =
{0}" , max);
// запоминается не номер, а элемент с этим
//номером
//вывод максимального
Console.WriteLine("Максимальный элемент=“ +
max);
Console.ReadKey();
```

Случайные числа в C#. Ввод массива с помощью случайных чисел

Язык Си Шарп предоставляет большие возможности для генерации случайных величин.

Если вам нужно получить генерацию целых или дробных чисел, то можно это сделать следующим образом:

```
Random x = new Random(); // объявление переменной для генерации чисел
```

Генерация целых случайных чисел

C# имеет 3 перегруженных метода:

1.Возвращает значение большее нуля. Максимальное число больше 10 миллионов:

```
int n = x.Next();
```

2.Возвращает значение в промежутке (min,max). При этом min значение входит в случайное число, а max не входит:

```
int n= x.Next(-100,100);
```

3.Возвращает целое положительное число не больше максимального:

```
int n = x.Next(10);
```

При этом максимальное число также не входит в генерацию случайных чисел, то есть в данном случае от 0 до 9.

Генерация дробных случайных чисел

В языке C# нет методов для генерации дробных чисел, но если нам надо получить дробные числа, мы можем сделать это следующим образом:

```
double r = Convert.ToDouble(x.Next(-100, 100)/10.0);
```

В данном случае мы имеем числа от -10 до 9.9.

```
double r = Convert.ToDouble(x.Next(100)/10.0);
```

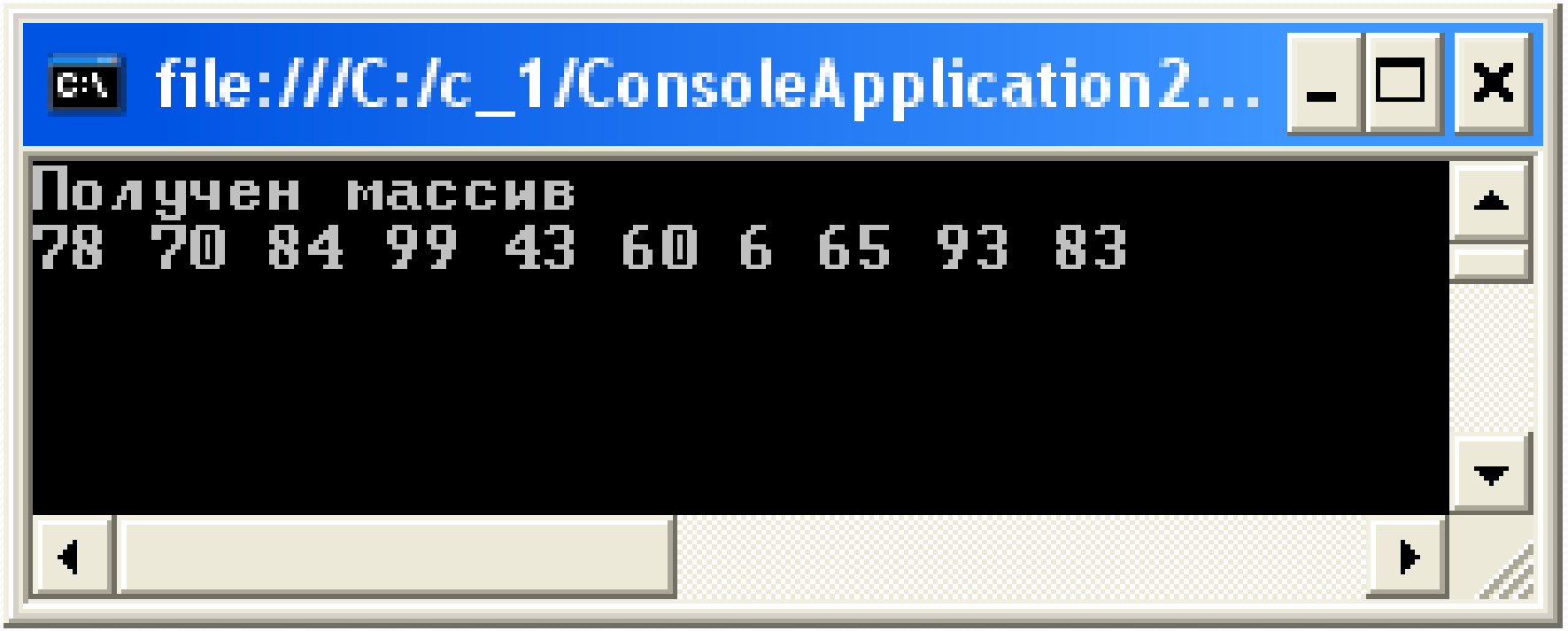
В данном случае мы имеем генерацию от 0 до 9.9.

Пример
Ввод массива с помощью случайных чисел

```
int x;
int[] a;
a = new int[10];
 // Генерация массива
Random r = new Random();
for (int i = 0; i < 10; i++)
{

    a[i] = r.Next(0, 100);

}
```

```
 // Вывод на экран
Console.WriteLine("Получен массив ");
for (int i = 0; i < 10; i++)
{
    Console.Write(a[i] + " ");

}
Console.WriteLine();
Console.ReadLine();
```

Протокол:


file:///C:/c_1/ConsoleApplication2...
Получен массив
78 70 84 99 43 60 6 65 93 83

НЕКОТОРЫЕ МЕТОДЫ КЛАССА Array

Методы IndexOf, LastIndexOf

IndexOf, LastIndexOf - определяют индексы первого и последнего вхождения образца в массив, возвращая -1, если такового вхождения не обнаружено.

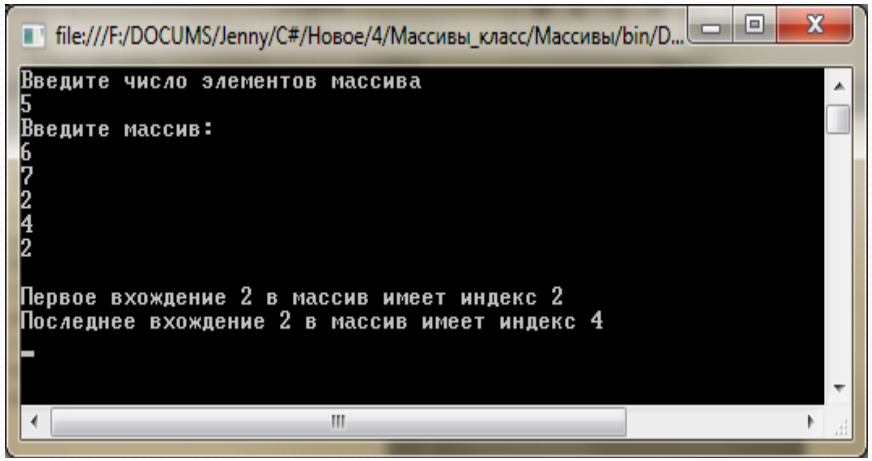
Пример.

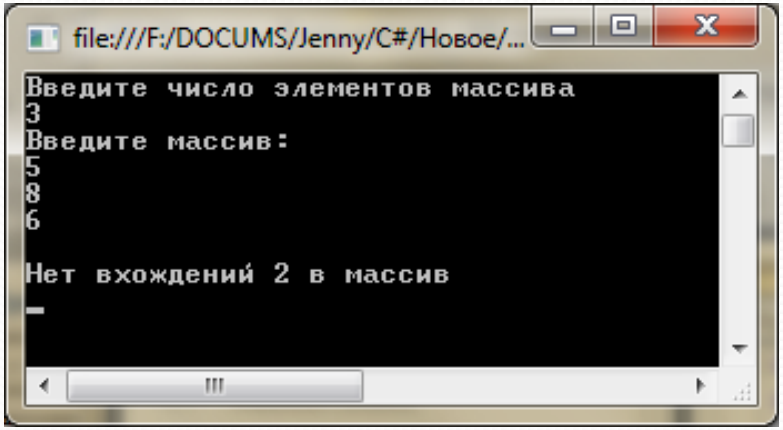
```
//в массиве a находим индекс первой 2
// и присваиваем его переменной first
int first = Array.IndexOf(a, 2);
//в массиве a находим индекс последней 2
// и присваиваем его переменной last
 int last = Array.LastIndexOf(a, 2);
```

Пример 2

Ввести с клавиатуры количество элементов массива и сами элементы. Найти:

- Индекс первой 2;
- Индекс последней 2;
- Выдать сообщение, если 2 нет.





```
// ввод количества элементов
int n =
Convert.ToInt32(Console.ReadLine());
//описание массива с инициализацией
нулями
int[] a = new int[n] ;
// ввод элементов массива в столбик
for (int i = 0; i < n; i++)
a[i]=Convert.ToInt32(Console.ReadLine())
;
```

```
int first = Array.IndexOf(a, 2);
int last = Array.LastIndexOf(a, 2);
if (first == -1)
Console.WriteLine("Нет вхождений 2 в массив");
else if (first == last)
Console.WriteLine("Одно вхождение 2 в массив имеет индекс {0}", first);
else
{
  Console.WriteLine
  ("Первое вхождение 2 в массив имеет индекс {0}", first);
  Console.WriteLine
  ("Последнее вхождение 2 в массив имеет индекс {0}", last);
}
```

Метод Reverse класса Array

Reverse - выполняет **обращение всего** массива, переставляя элементы в обратном порядке

Пример.

//в массиве **a** числа 4 5 6 7

// применяем к этому массиву метод Reverse

Array.Reverse(a);

//в массиве **a** будут числа 7 6 5 4

Метод Reverse класса Array

Reverse - выполняет **обращение части** массива, переставляя элементы этой части в обратном порядке

Пример.

//в массиве **a** числа 5 9 **4 2 3** 0

// применяем к этому массиву метод Reverse

Array.Reverse(a,2,3);

//в массиве **a** будут числа 5 9 **3 2 4** 0

Метод Sort класса Array

Sort - выполняет **сортировку всего массива по возрастанию**

Пример 1.

//в массиве **a** числа 3 1 6 2

// применяем к этому массиву метод Sort
Array.Sort(a);

//в массиве **a** будут числа 1 2 3 6

Метод Sort класса Array

Sort - выполняет **сортировку части массива по возрастанию**

Пример 1.

//в массиве **a** числа 3 1 **6 2 7** 4

// отсортируем 3 элемента, начиная со второго

Array.Sort(a,2,3);

//в массиве **a** будут числа 3 1 2 6 7 4