**Массивы. Класс Random, класс Array**

***Цель работы –*** получить навыки работы с массивами: правильно описывать массивы, вводить и выводить данные массива, различными способами обрабатывать элементы массива.

***Ключевые понятия:*** ссылочные типы данных, одномерные и многомерные массивы, размерность, диапазон.

**Теоретический материал**

**Массив — упорядоченная совокупность элементов одного типа.**

Тип «массив» относится к ссылочным типам данных, применяется в тех случаях, когда приходится иметь дело с наборами однотипных и однородных данных. Элементы массива пронумерованы, и обратиться к каждому из них можно, указав имя массива и индекс элемента в квадратных скобках. Индекс может быть буквальной константой или выражением.

Массив задает способ организации данных. Количество индексов характеризует размерность массива. Каждый индекс изменяется в некотором диапазоне от нуля до N. Индексы задаются целочисленным типом.

В программировании количество индексов массива называется его *размерностью* (рангом), количество допустимых значений каждого индекса – его *диапазоном*, а совокупность размерности и диапазонов – *формой* массива.

В математических вычислениях часто приходится иметь дело с векторами. Вектор – это пример массива, в котором элементы нумеруются одним индексом. Например, a (a0, a1, a3)

Если речь идёт о хранении в массиве таблицы значений (матрицы), его элементы нумеруются двумя индексами.

В роли базового класса для всех массивов в среде CLR выступает класс **Array**, он предоставляет методы для создания, изменения, поиска и сортировки массивов. Например, свойство **length** — позволяет определить длину массива, а метод **Sort** – отсортировать массив.

Заполнять массив можно «вручную» или случайными числами. Для этого нужно воспользоваться классом **Random.**

Класс**Random** имеет конструктор класса: для того, чтобы **вызывать методы класса**, нужно вначале **создавать экземпляр класса**.

Этим**Random** отличается от класса**Math,** у которого все поля и методы статические, что позволяет обойтись без создания экземпляров класса**Math**.

Методы класса**Random,** необходимые для генерирования последовательностей случайных чисел:

**конструктора класса** (две реализации)‏:

* **public Random()** позволяет генерировать неповторяющиеся при каждом запуске серии случайных чисел.
* **public Random(int)** обеспечивает возможность генерирования повторяющейся серии случайных чисел. Параметр используется для построения начального элемента серии, поэтому при задании одного и того же значения параметра серия будет повторяться.

**public int Next()** - выдает целые положительные числа во всем положительном диапазоне типа int;

**public int Next(int max)** — выдает целые положительные числа в диапазоне [0, max-1];

**public int Next(int min, int max)** — выдает целые положительные числа в диапазоне [min,max-1].

**public double NextDouble()** — при каждом вызове метода выдается новое случайное число, равномерно распределенное в интервале [0-1).

**public voidNextBytes (byte[ ] buffer)** — позволяет при одном обращении получать целую серию случайных чисел.

Параметр - массив, который и будет заполнен случайными числами в диапазоне [0, 255].

*Объявление одномерного массива.*

Общий вид: **<тип>[] <объявление>;**

*Варианты объявления:*

**тип[] имя;** объявление с отложенной инициализацией (инициализация должна произойти позже)

пример, int [] m1; (описана ссылка на массив с элементами целого типа, память не выделена)

**тип[] имя = new тип [ размерность ];** (типы должны совпадать)

пример, double [] m2 = newdouble [5] (объявлен и размещён в динамической памяти массив размера 5 с элементами вещественного типа)

**тип[] имя = { список\_инициализаторов };**

string[] weekDays = {"Mon", "Tue", "Wed", "Thu", "Fri", "Sat","Sun",} (объявлен и размещён в динамической памяти массив   
размера 7 с заданными элементами строкового типа)

**тип[] имя = new тип [] { список\_инициализаторов };**

пример, int [] m4 = new int [] {2, -5, 7, 1 -20} (объявлен и размещён в динамической памяти массив размера 5 с заданными элементами)

**тип[] имя = new тип [ размерность ] { список\_инициализаторов };**

пример, int [] m5 = new int [5] {2, -5, 7, 1 -20} (объявлен и размещён в динамической памяти массив размера 5 с заданными элементами)

m2 [1] = -100.25; m2 [15 % 4] = 10;

weekDays [0] = “Mon”;

m4 [2+2] = -23;

Console.WriteLine(m5[3]);

Обращение к элементам массива возможно только после инициализации, поэтому к элементам массива m1 обратиться нельзя.

*Объявление двумерного массива.*

int[,] m\_4x2 = new int[4, 2]; (объявлен и размещён в динамической памяти массив размера 4x2, то есть четыре строки и два столбца)

int[,] m\_3x4 = {

{1,10,15,-2},

{2,20,10,11},

{3,30,34,-67,}

};

(объявлен и размещён в динамической памяти массив размера 3x4 c заранее заданными элементами)

Варианты заполнения элементов массива:

* при объявлении (рассмотрено ранее)
* элементы вычисляются
* ввод элементов с клавиатуры
* заполнение массива случайными числами

**Элементы массива вычисляются:**

**Пример 1.**

int[] a = new int[5];

for (int i = 0; i < 5; i++)

a[i] = (i+1)\*2;

**Пример 2.**

for (int i=0; i < a.**Length**; i++) //вместо константы использовано свойство

a[i] = (i+1)\*2;

В результате в массиве будут храниться числа 2 4 6 8 10

**Ввод элементов с клавиатуры**

Console.WriteLine("введите размерность массива");

**int n =**  **int.Parse(Console.ReadLine());**

int[] a = new int[n];

for (int i = 0; i < n; ++i)‏

{

Console.Write("a[{0}]= ", i);

**a[i] = int.Parse(Console.ReadLine());**

}

**Заполнение массива случайными числами**

Random rnd = new Random();

// случайные числа в диапазоне [0,1)

double[] d = new double[4];

for (int i = 0; i < d.Length; i++)

**d[i] = rnd.NextDouble();**

// случайные числа в диапазоне[min,max)

int min = -100, max = 100

Random rnd = new Random();

int[] a = new int[5];

for (int i = 0; i < a.Length; i++)

**a[i] = rnd.Next(min, max)**;

// случайный массив байтов

const int initRnd = 25;

Random arrRnd = new Random(initRnd);

byte[] b = new byte[10];

**arrRnd.NextBytes(b);**

Console.WriteLine("Массив случайных чисел в диапазоне [0, 255]");

for (int i = 0; i < b.Length; i++)

Console.WriteLine("b[{0}] = {1}", i, b[i]);

***Задания для самостоятельной работы***

**Задание 1. (номер варианта = номер по списку % 16+1)**

n= …

int [] x = new int [n];

Вычислить для вектора x:

1. x[0]\*x[n-1] + x[1]\*x[n-2] + ... + x[n-1]\*x[0]
2. x[n-1]\*(x[n-1] + x[n-2])\*(x[n-1] + x[n-2] + x[n-3])\* ... \*(x[n-1] + ... + x[0])
3. Среднее геометрическое |x[0]\*x[1]\*...x[n-1]| (корень n степени из модуля произведения)
4. x[0] + x[0]\*x[1] + x[0]\*x[1]\*x[2] + ... + x[0]\*x[1]\*...\*x[n-1]
5. min (x[i]^i), i=0,2,4, …
6. x[0]\*1! + x[1]\*2! + ... + x[n-1]\*n!
7. (x[0] + x[1] + ... + x[n-1])/(x[0]\*x[1]\*...\*x[n-1])
8. x[n-1]^1 + x[n-2]^2 + x[n-3]^3 + ... + x[0]^n
9. x[1] + x[3]/3! + x[5]/5! + ... +
10. x[0]^1 – x[1]^2/2 + x[2]^3/3 - ... -
11. (5 + х[1]/x[0])(5 + x[2]/x[1]) +...
12. x[0]^0 + x[1]^2/3 + x[2]^4/5 + ...
13. x[0]^(0+1) + x[1]^(2+3)/3 + x[2]^(4+5)/5 + ...
14. 1 – x[1]^2/7! + x[2]^4/7! - ...
15. 7x[0] + 7x[1]^3/3 + 7x[2]^5/5 + ...
16. x[0] – x[2]^3/3! + x[4]^5/5! - ...

**Задание 2. (номер варианта = номер по списку % 16+1)**

1. Написать программу, которая печатает True, если элементы массива x упорядочены строго по возрастанию, False в противном случае
2. Написать программу, которая печатает True, если элементы массива x не содержат 0, False в противном случае
3. Написать программу, которая печатает True, если элементы массива x чередуются по знаку (положительные, отрицательные)
4. Написать программу, которая печатает True, если элементы массива x четные и положительные, False в противном случае
5. Написать программу, которая печатает True, если элементы массива x кратны 10, False в противном случае
6. Написать программу, которая печатает True, если элементы массива x чередуются по четности (четные, нечетные)
7. Написать программу, которая печатает True, если элементы массива x упорядочены строго по убыванию, False в противном случае
8. Написать программу, которая печатает True, если элементы массива x упорядочены строго по убыванию, False в противном случае
9. Написать программу, которая печатает True, если все элементы массива x заканчиваются на цифру 7, False в противном случае
10. Написать программу, которая печатает True, если число максимальных элементов массива х больше 2, False в противном случае
11. Написать программу, которая печатает True, если все элементы массива х являются полными квадратами, False в противном случае
12. Написать программу, которая печатает True, если все элементы массива х различные, False в противном случае
13. Написать программу, которая печатает True, если каждый элемент массива х отличается от соседнего не более, чем на 1, False в противном случае
14. Написать программу, которая печатает True, если все положительные элементы массива х различные, False в противном случае
15. Написать программу, которая печатает True, если модуль среднего арифметического всех отрицательных элементов массива х больше среднего арифметического всех положительных элементов массива х, False в противном случае
16. Написать программу, которая печатает True, если все элементы массива х отличаются от минимального и максимального не более, чем на 100, False в противном случае

**Задание 3. (номер варианта = номер по списку % 8+1)**

1. Номер первого отрицательного элемента массива х, или -1, если такого элемента нет
2. Номер первого нулевого элемента, стоящего после отрицательного элемента, или -1, если такого элемента нет.
3. Среднее арифметическое нулевых элементов, стоящих после отрицательных элементов.
4. Номер последнего положительного элемента массива х, или -1, если такого элемента нет
5. Среднее арифметическое положительных элементов массива х, кратных 7
6. Номер последнего нулевого элемента или -1, если такого элемента нет.
7. Среднее арифметическое нулевых элементов, стоящих после положительных элементов.
8. Сумму всех отрицательных элементов массива, стоящих на четных местах.