Оператор цикла `foreach` для работы с одномерными массивами

служит для циклического обращения к элементам *коллекции*, представляющей собой группу объектов

Синтаксис оператора:

```
foreach (тип имя_переменной_цикла in коллекция)
{
   Оператор
};
```

Задача 7. Создать одномерный массив A(7), элементы массива рассчитать по формуле. Для накопления суммы всех элементов массива использовать оператор цикла `foreach`.

```
Массив A:
7 9 11 13 15 17 19

Сумма элементов массива sum = 91
```

```
const int n = 7;
int[] A = new int[n];
int i;
int sum = 0;

// Заполнение массива значениями и вывод на экран
Console.WriteLine(" Массив A:   ");
for (i = 0; i <= (n - 1); i++)
{
    A[i] = 2 * i + 7;
    Console.Write(" " + A[i]);
}
Console.WriteLine();
//цикл проходит по каждому целочисленному элементу в массиве
foreach (int k in A) sum = sum + k;

Console.WriteLine("\nСумма элементов массива sum = " + sum);
Console.ReadLine();
```

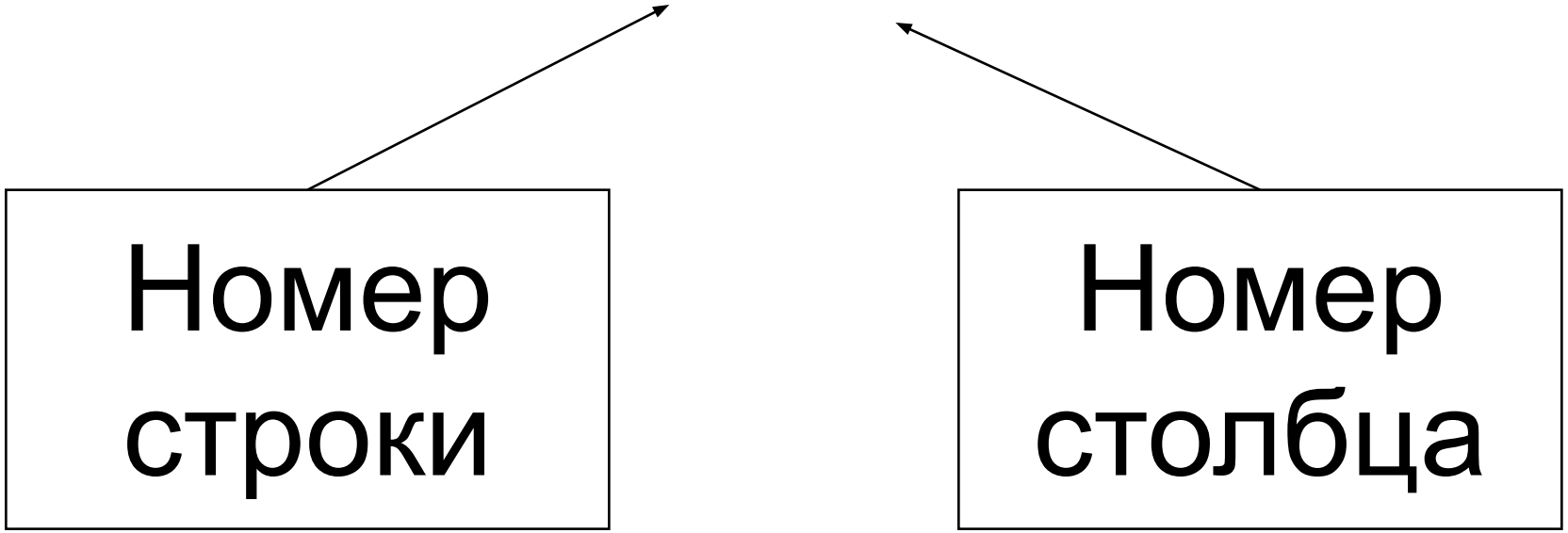
Тема 2.4.2 Многомерные массивы

i \ j	0	1	2	3
0	7,8	9	7	15
1	–8	4,5	2	–5
2	8,3	15,7	7,1	5,6
3	10	1,8	–3	–18
4	5	15	4,6	–9,2

Двумерный массив – это одномерный массив, тип элементов которого также является массивом (массив массивов). Положение элементов в двумерных массивах описывается двумя индексами

```
double[,] A1 = new double[5, 4];
//объявляется и инициализируется
//массив A из 20 (5×4) вещественных чисел
```

```
A[1,2] = 2.0;
A[2,3] = 5.6;
```



Варианты описания двумерного массива:

1. тип[,] имя;
2. тип[,] имя = new тип [rows, columns];
3. тип[,] имя = { список_инициализаторов };
4. тип[,] имя = new тип [,] { список_инициализаторов };
5. тип[,] имя = new тип [rows, columns] { список_инициализаторов };

```
 // двумерный массив B из 2×2 целых чисел
int[,] B = new int[2, 2];


    // объявляется двумерный массив C из 2×3 целых чисел,
    // одновременно массив наполняется нужными значениями.
int[,] C = new int[2, 3] { { 1, 2, 4 }, { 3, 5, 7 } };
```

Ввод - вывод элементов двумерного массива

Ввод:

- поэлементно с помощью оператора присваивания;
- при объявлении, поместив значения массива в фигурные скобки;
- элементы массива можно рассчитать по формуле;
- элементы массива можно задать при помощи генератора случайных чисел.

Вывод:

- для консольного приложения на экран с помощью Console.WriteLine();
- для Windows- приложения – в поле списка ListBox, в текствое поле TextBox, на компонент DataGridView.

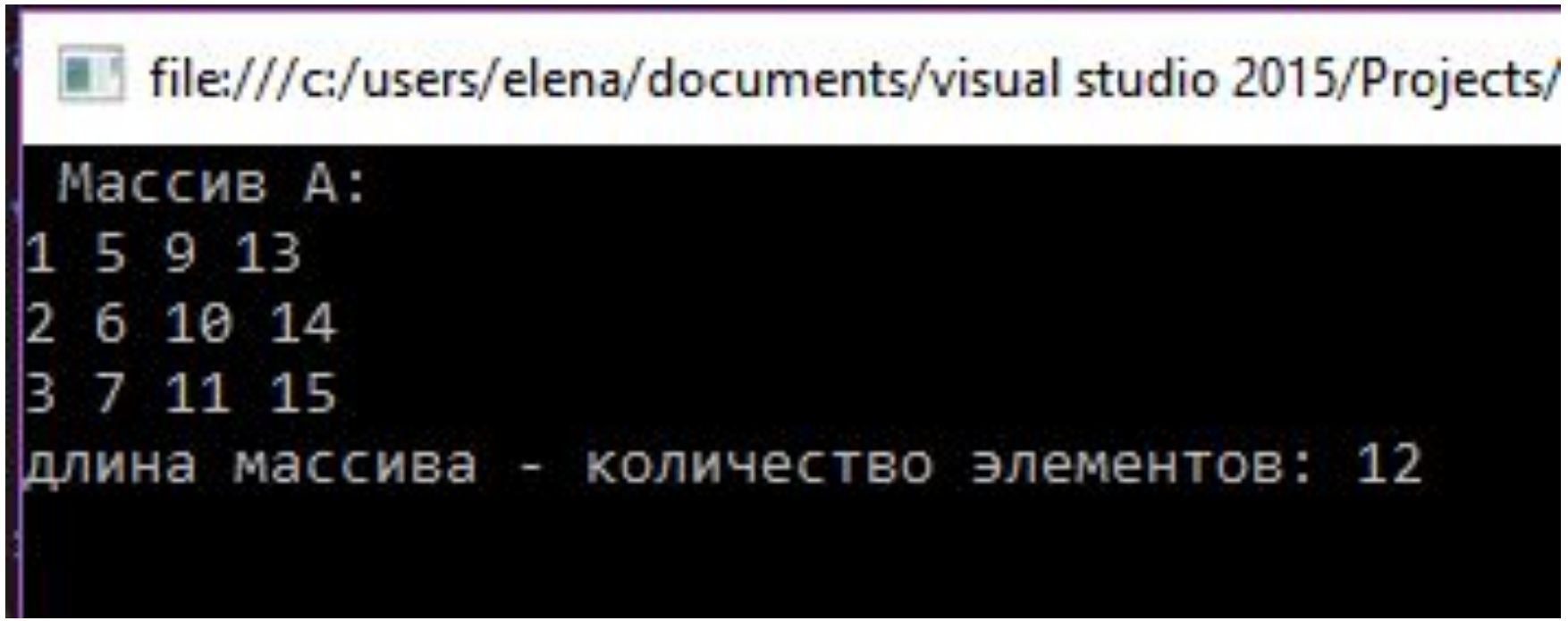
Задача 1. Двумерный массив A размером 3×4 сначала заполняется числами (элементы рассчитываются по формуле), а затем выводится его содержимое.

```
int i, j;
int[,] A = new int[3, 4];
Console.WriteLine(" Массив A: ");
for (i = 0; i <= 2; i++)
{
    for (j = 0; j <= 3; j++)
    {
        A[i, j] = i + 4 * j + 1;
        Console.Write(A[i, j] + " ");
    }
    Console.WriteLine();
}
Console.ReadLine();
```

```
 Массив A:
1 5 9 13
2 6 10 14
3 7 11 15
```

! Свойство Length возвращает общее количество элементов в массиве.

Console.WriteLine($"длина массива - количество элементов: {A.Length}“);



Накопление суммы элементов двумерного массива. Сумма элементов главной и побочной диагонали

Сумму всех элементов двумерного массива записывают как $\sum A$ или с указанием индекса строк и столбцов $\sum_{i=0}^{n-1}\sum_{j=0}^{m-1} A_{ij}$. В записи заданы два цикла – внешний и внутренний. Внутренний цикл считает суммы элементов строк, внешний – суммирует их в общую сумму.

Задача 2. Вычислить сумму элементов двумерного массива

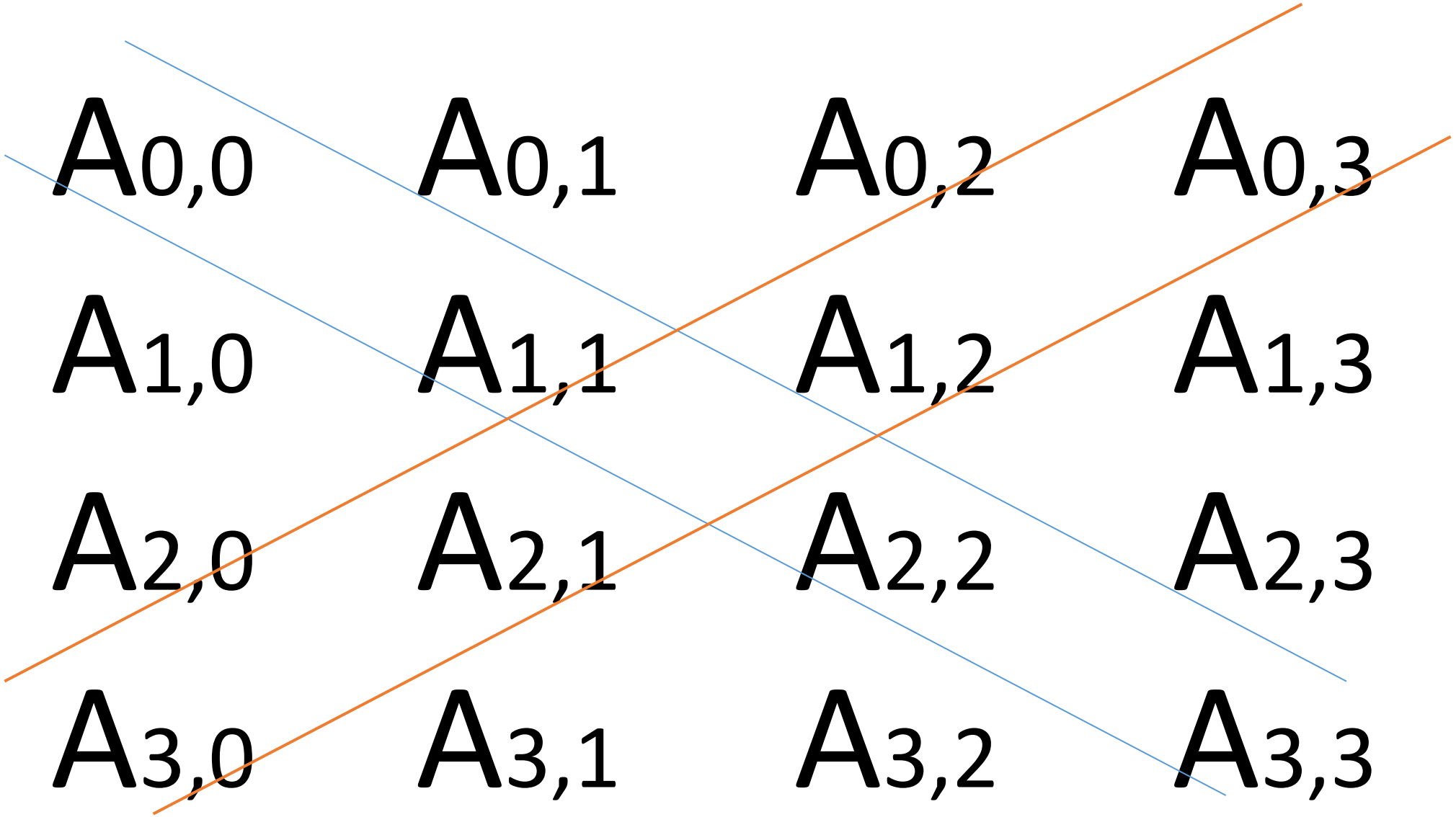
```
int i, j; //индексы массива
int[,] A = new int[3, 4];
Console.WriteLine(" Массив A: ");
Random rnd = new Random();
for (i = 0; i <= 2; i++) //сначала строки
{
    for (j = 0; j <= 3; j++) //потом столбцы
    {
        A[i, j] = rnd.Next(0, 100); //заполняется случайными числами от 0 до 99 вкл
        Console.Write($"A[{i},{j}] = {A[i,j]}\t"); // вывод значений элементов массива
    }
    Console.WriteLine(); //переход на новую строку
}
int sum = 0; //переменная суммы, вначале = 0
foreach (int x in A) // проходим по всем элементам массива A
{
    sum += x; //увеличиваем сумму на значение текущего элемента
}

Console.WriteLine($"сумма всех элементов массива = {sum}");
Console.ReadLine();
```

```
 Массив A:
A[0,0] = 97     A[0,1] = 15     A[0,2] = 26     A[0,3] = 25
A[1,0] = 37     A[1,1] = 70     A[1,2] = 0      A[1,3] = 16
A[2,0] = 54     A[2,1] = 87     A[2,2] = 77     A[2,3] = 93
сумма всех элементов массива = 597
```

Главная диагональ
Побочная диагональ

Пример матрицы



- У элементов главной диагонали индексы строк и столбцов равны (i==j).
- Если элемент располагается ниже главной диагонали, то у него (i < j), а выше (i > j).
- Для описания расположения элементов на побочной диагонали можно привести условие ($i == n - j - 1$).

Задача 3. Создать и вывести векторы из диагоналей матрицы и вычислить их суммы.

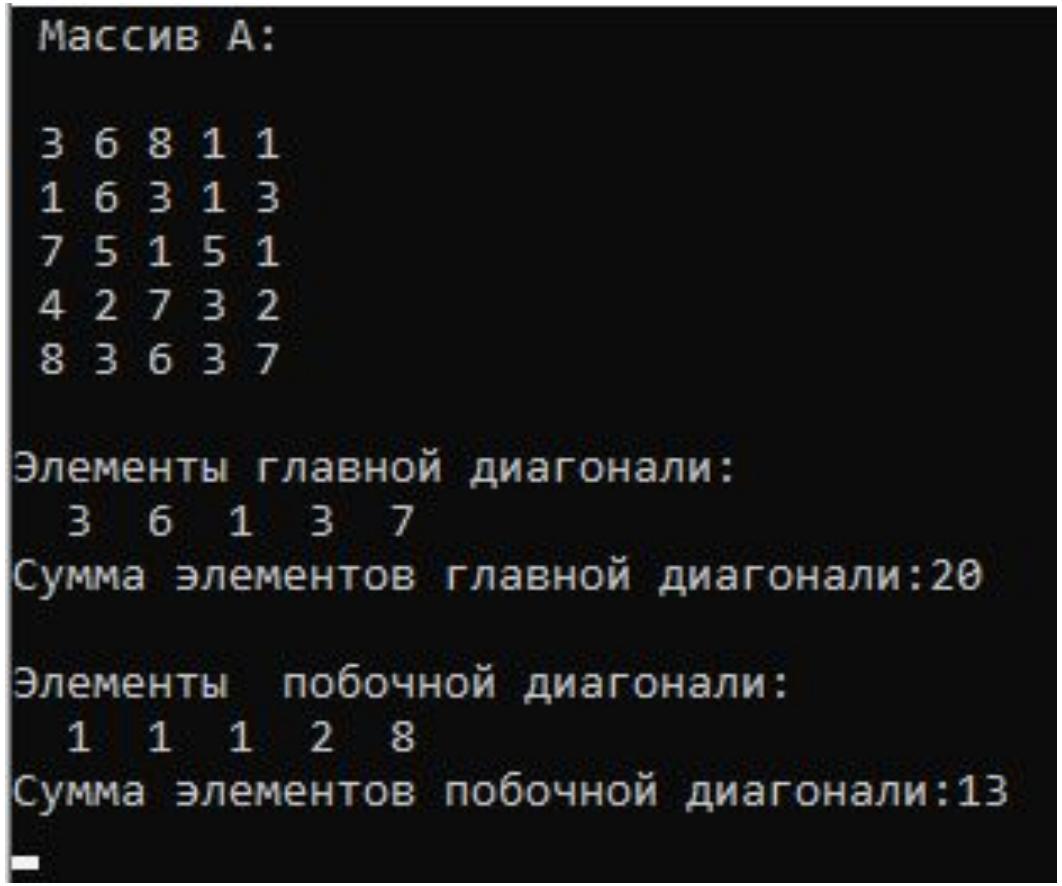
```
const int n = 5; //константа размера матрицы
int[,] A = new int[n, n];
int[] C = new int[n];          //элементы главной диагонали
int[] S = new int[n];          //элементы побочной диагонали
Random rnd = new Random();
int sumC = 0;       // сумма элементов главной диагонали
int sumS = 0;       // сумма элементов побочной диагонали
int i, j;
```

```
// Накопление и вывод элементов массива
Console.WriteLine(" Массив A: \n ");
for (i = 0; i <= (n - 1); i++)
{
    for (j = 0; j <= (n - 1); j++)
    {
        A[i, j] = rnd.Next(1, 10);
        Console.Write(" " + A[i, j]);
    }
    Console.WriteLine();
}
```

```
//Создание одномерных массивов C[n] и S[n] и накопление их сумм
for (i = 0; i <= (n - 1); i++)
{
    for (j = 0; j <= (n - 1); j++)
    {
        if (i == j)
        {
            C[i] = A[i, j]; sumC = sumC + C[i];
        }
        if (i == n - j - 1)
        {
            S[i] = A[i, j]; sumS = sumS + S[i];
        }
    }
}
```

```
Console.WriteLine("Элементы главной диагонали: ");
for (i = 0; i <= (n - 1); i++)
    Console.Write("  " + C[i]);
Console.WriteLine("\nСумма элементов главной диагонали:" + sumC);
Console.WriteLine("\nЭлементы  побочной диагонали: ");
for (i = 0; i <= (n - 1); i++)
    Console.Write("  " + S[i]);
Console.WriteLine("\nСумма элементов побочной диагонали:" + sumS);
```

Результаты решения задачи 3



Зубчатый (ступенчатый) массив

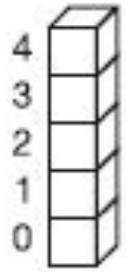
- это массив, в котором строки могут содержать произвольное количество элементов.

```
int[][] nums = new int[3][];
nums[0] = new int[2] { 1, 2 };          // выделяем память для первого подмассива
nums[1] = new int[3] { 1, 2, 3 };       // выделяем память для второго подмассива
nums[2] = new int[5] { 1, 2, 3, 4, 5 }; // выделяем память для третьего подмассива
```

1	2			
1	2	3		
1	2	3	4	5

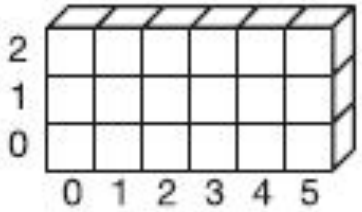
Примеры массивов:

Одномерный массив

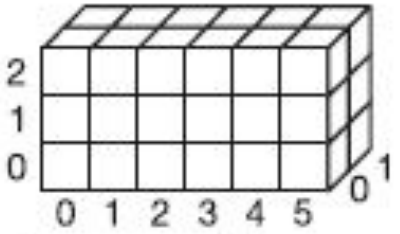


Одномерный массив
int[5]

Многомерные массивы

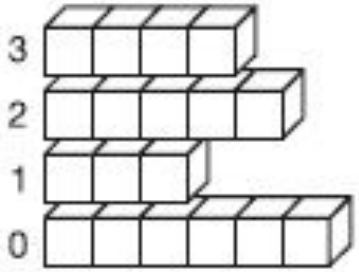


Двухмерный массив
int[3,6]



Трехмерный массив
int[3,6,2]

Зубчатый массив



Зубчатый массив
int[4][]

Вложенные многомерные массивы (пример трехмерного ступенчатого массива)

```
int[][,] nums = new int[3][,]
            {
                new int[,] { {1,2}, {3,4} },
                new int[,] { {1,2}, {3,6} },
                new int[,] { {1,2}, {3,5}, {8, 13} }
            };
```

Перебор зубчатых массивов

```
int[][] numbers = new int[3][];
numbers[0] = new int[] { 1, 2 };
numbers[1] = new int[] { 1, 2, 3 };
numbers[2] = new int[] { 1, 2, 3, 4, 5 };
foreach (int[] row in numbers) //проход по вложенным массивам
{
    foreach (int number in row) //проход по каждому элементу текущего вложенного массива
    {
        Console.Write($"{number} \t");
    }
    Console.WriteLine();
}
```

Перебор зубчатых массивов

```
// перебор с помощью цикла for
for (int i = 0; i < numbers.Length; i++)//проход по вложенным массивам
{
    for (int j = 0; j < numbers[i].Length; j++)//проход по каждому элементу текущего вложенного массива
    {
        Console.Write($"{numbers[i][j]} \t");
    }
    Console.WriteLine();
}
```

Основные понятия

Ранг (rank): количество измерений массива

Длина измерения (dimension length): длина отдельного измерения массива

Длина массива (array length): количество всех элементов массива

```
int[,] numbers = new int[3, 4];
```

Ранг:2.

Длина первого измерения - 3,

Длина второго измерения - 4.

Длина массива - 12.