

Модуль 2, практическое занятие 1-2 (нед. 1)

Многомерные массивы

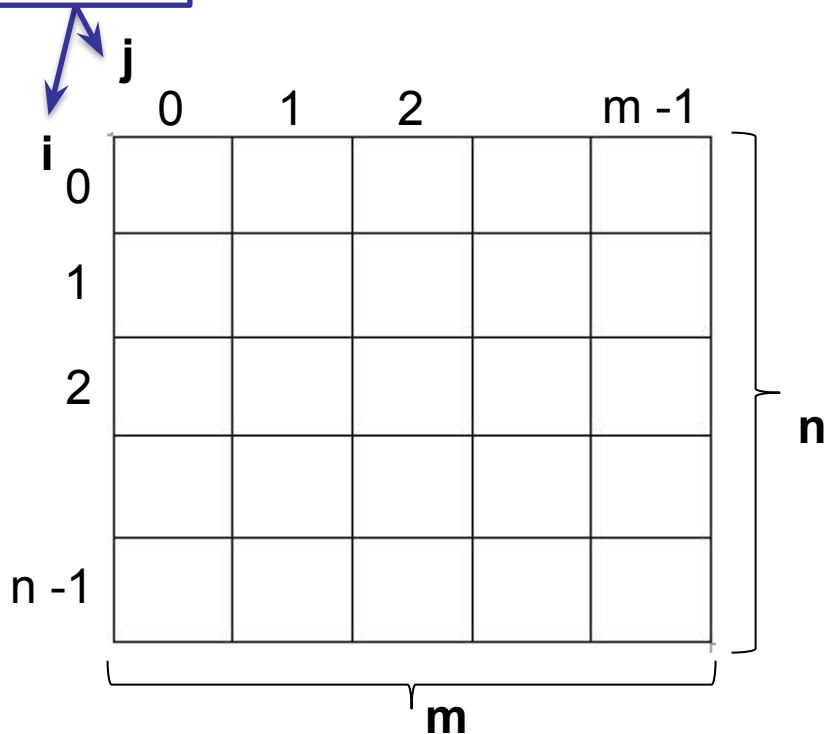
Массивы массивов

Отладка программы

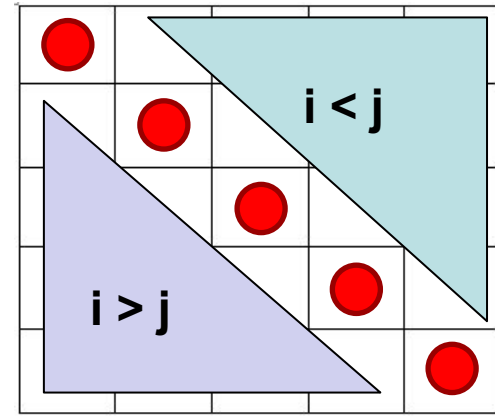
Матрицы. Индексы элементов матриц

Матрица как двумерный массив:
`int m = 5, n = 5`
`int[,] matr = new int[n, m]`

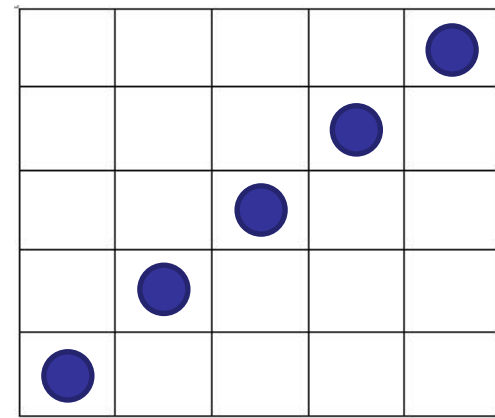
индексы



Главная диагональ $i = j$



Побочная диагональ



Задача 1

```
// Определить матрицу как двумерный массив, инициализировать ее  
// элементы.  
// Используя свойства и методы класса Array вывести ранг массива,  
// общее число его элементов, число элементов по разным  
// измерениям, предельные значения всех индексов, признак  
// фиксированных размеров...  
// Вывести элементы массива, используя цикл foreach...  
// Вывести элементы массива по строкам (в виде таблицы)...  
using System;  
class Program {  
    static void Main() {  
        int[,] matr = new int[3, 4] { { 0, 1, 3, 4 }, { 5, 6, 7, 8 },  
        { 9, -1, -2, -3 } };  

```

```
        Console.WriteLine("Для выхода из программы нажмите ENTER.");  
        Console.ReadLine();  
    }  
}
```

Задача 1

```
Console.WriteLine("matr.GetType() = " + matr.GetType());
Console.WriteLine("matr.IsFixedSize = " + matr.IsFixedSize);
Console.WriteLine("matr.Rank = " + matr.Rank);
Console.WriteLine("matr.Length = " + matr.Length);
Console.WriteLine("matr.GetLength(1) = " + matr.GetLength(1));
Console.WriteLine("matr.GetUpperBound(1) = " + matr.GetUpperBound(1));

foreach (int memb in matr) // все элементы матрицы подряд
    Console.Write("{0,3}", memb);

Console.WriteLine("\n");

// ВЫВОД ПО СТРОКАМ!!!
for (int i = 0; i < matr.GetLength(0); i++, Console.WriteLine())
    for (int j = 0; j < matr.GetLength(1); j++)
        Console.Write("{0,3}", matr[i, j]);
```

Задача 2

- Получить от пользователя целое число N . Создать двумерный массив размера в $N \times N$ и заполнить его по правилу:

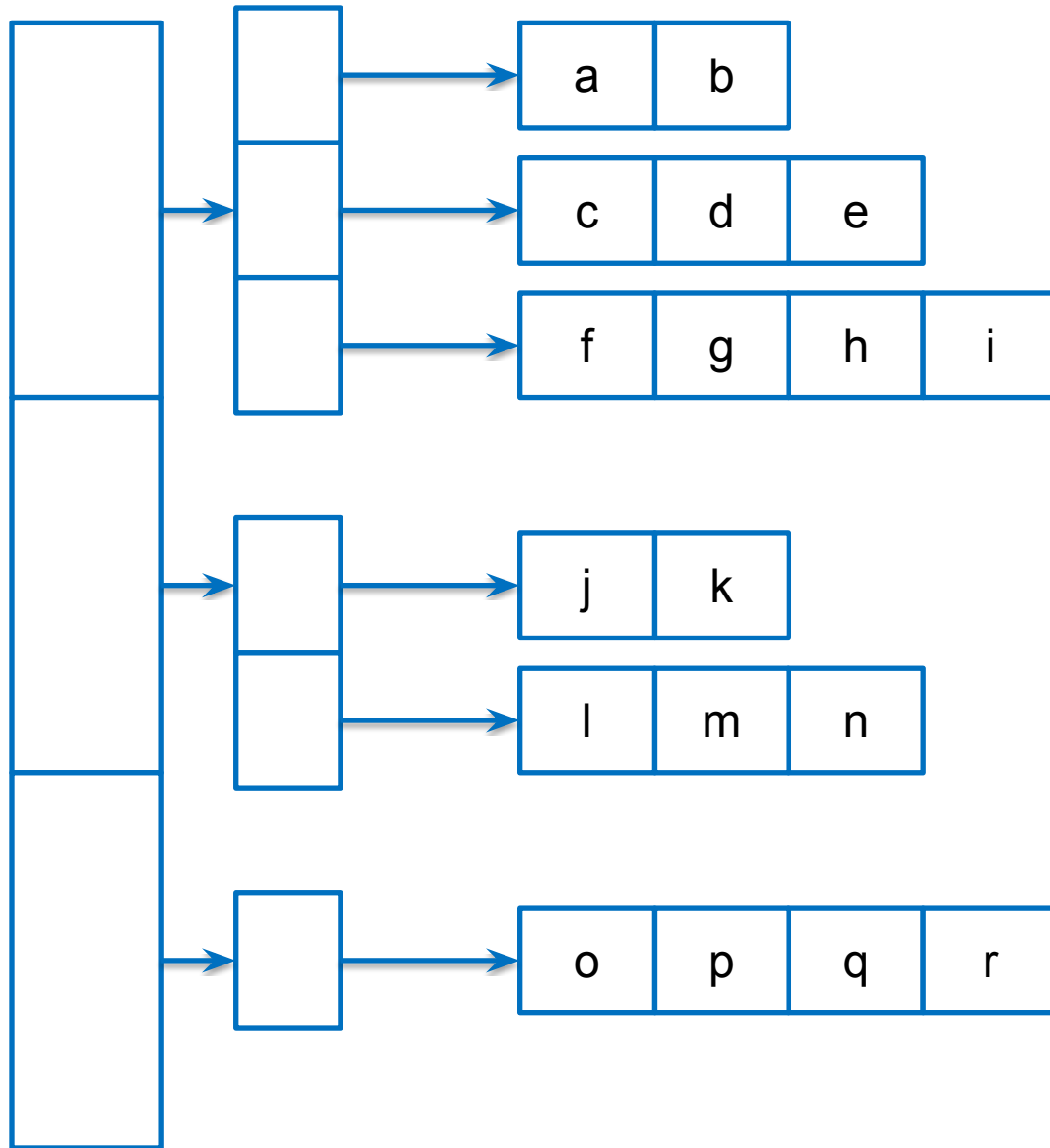
$$\begin{pmatrix} 1 & 2 & \dots & N-2 & N-1 & N \\ 2 & 3 & \dots & N-1 & N & 1 \\ 3 & 4 & \dots & N & 1 & 2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ N-1 & N & 1 & 2 & \dots & N-2 \\ N & 1 & 2 & \dots & \dots & N-1 \end{pmatrix}$$

Задача 3

```
// Определить массив из трех элементов – ссылок на массивы разной
// длины.
//     1-й элемент - массив из 3-х элементов – ссылок на массивы,
// соответственно, из 2-х, 3-х и 4-х элементов типа char.
//     2-й элемент - массив из 2-х элементов ссылок на массивы,
// соответственно, из 2-х и 3-х элементов типа char.
//     3-й элемент - массив из ОДНОГО элемента – ссылки на массив из 4-х
// элементов типа char.
// Используя свойства и методы класса Array вывести ранг массива,
// общее число его элементов, число элементов по разным
// измерениям, предельные значения всех индексов.
// Вывести элементы массива с помощью циклов foreach, размещая
// значения элементов каждого массива нижнего уровня по строкам...
using System;
class Program {
    static void Main() {
```

```
        Console.WriteLine("Для выхода из программы нажмите ENTER.");
        Console.ReadLine();
    }
}
```

Задача 3



Задача 3

```
char[][][] ch = { // элементы массива - массивы ссылок на массивы
    new char [][] { new char [] {'a', 'b'}, // у массива
        new char [] {'c', 'd', 'e'}, // элемент - ссылка на массив
        new char [] {'f', 'g', 'h', 'i'}
    },
    new char [][] { new char [] {'j', 'k'},
        new char [] {'l', 'm', 'n'}
    },
    new char [][] { new char [] {'o', 'p', 'q', 'r'}, }
};

Console.WriteLine("ch.Rank = " + ch.Rank);
Console.WriteLine("ch[0].Rank = " + ch[0].Rank);
Console.WriteLine("ch[0][0].Rank = " + ch[0][0].Rank);
Console.WriteLine("ch.GetType() = " + ch.GetType());
Console.WriteLine("ch[1][1][2] = " + ch[1][1][2]);
Console.WriteLine("ch.Length = " + ch.Length);
Console.WriteLine("ch.GetLength(0) = " + ch.GetLength(0));
Console.WriteLine("ch[1].GetLength(0) = " + ch[1].GetLength(0));
```


Задача 3

```
foreach (char[][] memb1 in ch) {  
    Console.WriteLine("Уровень 1:");  
    foreach (char[] memb2 in memb1) {  
        Console.Write("\tУровень 2:\t");  
        foreach (char memb3 in memb2)  
            Console.Write("{0,3}", memb3);  
        Console.WriteLine();  
    }  
}
```

Задача 4

```
// Ввести положительные значения N и M. Построить двумерный  
// целочисленный массив (матрицу) с размерами N на M, элементы  
// которого  $a[i, j] = (i+1)*(2*j+1)$ , для  $i$  от 0 до (N-1),  
//  $j$  от 0 до (M-1).  
// Вывести матрицу в виде таблицы, а также значения свойств  
// Rank и Length.
```

```
using System;  
class Program {  
    static void Main() {  
        uint N, M;           // размеры массива  
        int[,] ar;           // ссылка на массив  
        do {                 // цикл для повторения решения
```

```
Console.WriteLine("Для выхода нажмите клавишу ESC");  
        } while (Console.ReadKey(true).Key !=  
ConsoleKey.Escape);  
    }  
}
```

Задача 4

```
do Console.Write("Введите число строк (N>0): ");
while (!uint.TryParse(Console.ReadLine(), out N) || N == 0);

do Console.Write("Введите число столбцов (M>0): ");
while (!uint.TryParse(Console.ReadLine(), out M) || M == 0);

ar = new int[N, M];
for (int i = 0; i < N; i++)
    for (int j = 0; j < M; j++)
        ar[i, j] = (i + 1) * (2 * j + 1); // заполняем матрицу по правилу

Console.WriteLine("ar.Length: " + ar.Length);
Console.WriteLine("ar.Rank: " + ar.Rank);

Console.WriteLine("Элементы массива:");
for (int i = 0; i < N; i++, Console.WriteLine())
    for (int j = 0; j < M; j++)
        Console.Write("{0,3}", ar[i, j]);
```

Задание к задаче 4

1. Формирование матрицы вынести в отдельный метод.
2. Написать метод, заменяющий в квадратной матрице, переданной в качестве параметра, все элементы, стоящие под побочной диагональю нулями. Если матрица не квадратная, никаких действий не производить.
3. Применить метод к матрице, сформированной в задаче 2. Результат вывести на экран.

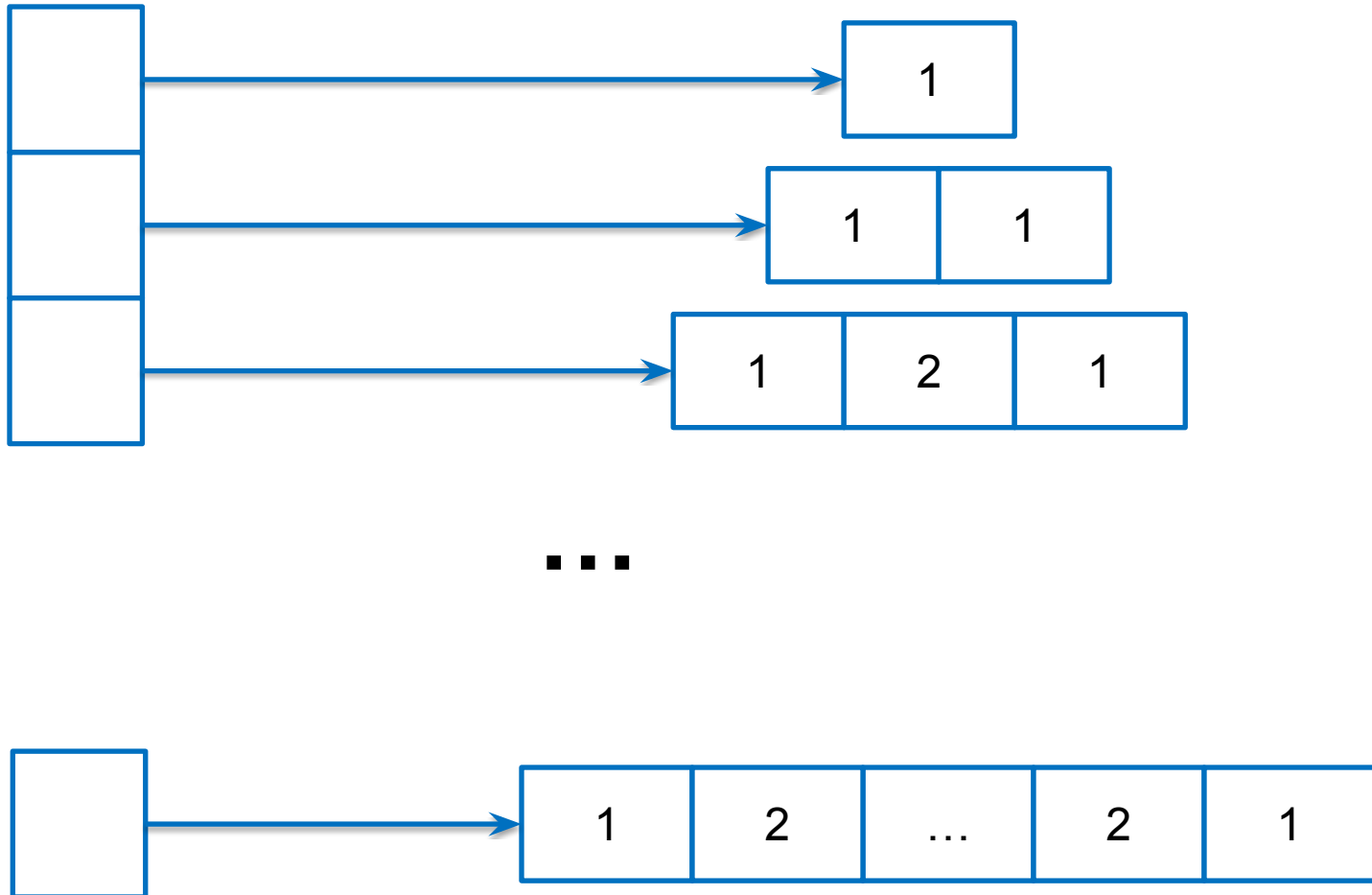
Задача 5

```
// Определить массив массивов для представления треугольника Паскаля.  
// 0-й элемент - массив из одного элемента со значением  $C(0,0)=1$ ,  
// 1-й элемент - массив из 2-х элементов  $C(1,0)=C(1,1)=1$ .  
// 2-й элемент - массив из 3-х элементов  $C(2,0)=C(2,2)=1$ ,  $C(2,1)=2...$   
// n-й элемент - массив из n+1 элементов:  $C(n,0)=C(n,n)=1$ ,  
//  $C(n,k)=C(n-1,k-1)+C(n-1,k)$ .  
// Вводя неотрицательные значение n, построить массив-массивов  
// со значениями биномиальных коэффициентов и вывести его на экран  
// с помощью циклов foreach, размещая значения элементов каждого  
// массива нижнего уровня по на отдельной строке...
```

```
using System;  
class Program {  
    static void Main() {  
        int[][] paskal; // ссылка на массив ссылок на массив  
        int n;  
        do // цикл для повторения решения  
        {
```

```
            Console.WriteLine("Для выхода нажмите клавишу ESC");  
        } while (Console.ReadKey(true).Key != ConsoleKey.Escape);  
        Console.WriteLine();  
    }  
}
```

Задача 5



Задача 5

```
do Console.Write("Введите n: ");
while (!int.TryParse(Console.ReadLine(), out n) || n < 0);
paskal = new int[n + 1][]; // объект - массив пустых ссылок

Console.WriteLine("paskal.GetType()=" + paskal.GetType());
for (int i = 0; i < paskal.Length; i++) {
    paskal[i] = new int[i + 1]; // объект - массив элементов типа int
    paskal[i][0] = paskal[i][i] = 1;
    for (int j = 1; j < i; j++)
        paskal[i][j] = paskal[i - 1][j - 1] + paskal[i - 1][j];
}

foreach (int[] ar in paskal) // перебор ссылок типа int[]
{
    foreach (int cnk in ar) // перебор элементов типа int
        Console.Write("{0,4}", cnk);
    Console.WriteLine();
}
```

Задание к задаче 5

1. Код, формирующий массив с элементами треугольника паскаля оформить в отдельный метод.

Задача 6

```
// Определите и инициализируйте массив строк.  
// Выведите строки в порядке возрастания их длин.  
// Порядок элементов в исходном массиве строк не менять.  
using System;  
class Program {  
    static void Main() {  
        string[] lines = new string[] { "нуль", "один", "два",  
                                         "три", "четыре",  
                                         "пять", "шесть", "семь", "восемь",  
                                         "девять", "десять" };  
    }
```

```
        Console.WriteLine("\nИсходный массив:");  
        foreach (string s in lines)  
            Console.Write(s + " ");  
  
        Console.WriteLine("\nДля выхода нажмите любую клавишу.");  
        Console.ReadKey(true);  
    } // конец Main()  
} // конец Program
```

Задача 6

```
int len = lines.Length;
int[] index = new int[len]; // Массив индексов
for (int i = 0; i < len; i++)
    index[i] = i;

// Сортировка массива индексов:
for (int i = 0; i < len - 1; i++)
    for (int j = i + 1; j < len; j++)
        if (lines[index[i]].Length > lines[index[j]].Length) {
            int temp = index[i];
            index[i] = index[j];
            index[j] = temp;
        }

Console.WriteLine("Результат перебора:");
foreach (int n in index)
    Console.Write(lines[n] + " ");
```

Задача 7

- Результаты продаж автомобилей филиалами компании по кварталам 2009 года отображены в таблице:

Квартал \ Филиал	Западный	Центральный	Восточный
I	20	24	25
II	21	20	18
III	23	27	24
IV	22	19	20

- Написать программу, решающую следующие задачи (исходные данные задать в коде программы):
 - Подсчитать общее количество автомобилей, проданных всеми филиалами компании за год.
 - Вывести максимальное количество автомобилей, проданных филиалом за квартал, а также название филиала и номер квартала.
 - Вывести название филиала, который продал максимальное количество автомобилей по результатам года, а также проданное филиалом количество автомобилей.
 - Вывести наиболее успешный квартал, в котором компания показала наилучший результат по продажам (учитываются все филиалы), а также количество автомобилей проданное в нем.

Задача 7

- Статические поля класса Program

```
static string[] Filials = { "Западный", "Центральный", "Восточный"
};
static string[] Kvartal = { "I", "II", "III", "IV" };
static int[,] auto = { { 20, 24, 25 }, // I
                        { 21, 20, 18 }, // II
                        { 23, 27, 24 }, // III
                        { 22, 19, 20 }  // IV
};
```

Задача 7

```
static void Main() {  
    string s, input;  
    Console.Write(PrintSrc()); // печать исходных данных  
    do {  
        Console.Write(Print()); // вывод текстового меню  
        // обработка выбранного пункта меню + вывод результата  
        s = PrintResults(input = Console.ReadLine());  
        Console.WriteLine(s);  
    } while (input != "0"); // выход из меню по нулю  
    Console.ReadLine();  
}
```

Задача 7

```
/// <summary>
    /// все результаты
    /// </summary>
    /// <returns>строка, сформированная по результатам работы
    методов</returns>
    public static string PrintResults(string mode) {
        string st = "";
        int Nstroki; // номер строки
        int Nstolbca; // номер столбца
        int SumFilial; // продано филиалом
        int NFiliala_MaxAutoYear; // номер лучшего филиала
        int MaxAutoFilialZaGod; // продано лучшим филиалом за год
        int SumKvartal; // продано за квартал
        int NKvartal_MaxAuto; // номер квартала с максимальной продажей
        int MaxAutoKvartal; // максимальная продажа в квартал
        // переключатель
    }
```

Задача 7

```
switch (mode) {
    case "0": st += "Спасибо за работу!\r\n"; break;
    case "1": st += "Ответ 1. Общее количество автомобилей = " +
        GrandTotal() + "\r\n"; break;
    case "2": GetMax4Kvartal(out Nstroki, out Nstolbca);
        st += "Ответ 2. Максимальное количество автомобилей = " +
            auto[Nstroki, Nstolbca] +
            ",
Квартал = " + Kvartal[Nstroki] + ", Филиал = " + Filials[Nstolbca] + "\r\n";
break;
    case "3": maxAutoFilialZaGod(out SumFilial, out NFiliala_MaxAutoYear, out
MaxAutoFilialZaGod);
        t += "Ответ 3. Название филиала, который продал максимальное
количество автомобилей по результатам года = " +
            Filials[NFiliala_MaxAutoYear] +
            ", проданное количество автомобилей = " + MaxAutoFilialZaGod +
            "\r\n"; break;
    case "4": maxAutoKvartal(out SumKvartal, out NKvartal_MaxAuto, out
MaxAutoKvartal);
        st += "Ответ 4. Наиболее успешный квартал = " +
            Kvartal[NKvartal_MaxAuto] + ", проданное количество автомобилей = " +
            MaxAutoKvartal + "\r\n"; break;
    default: st += "Неизвестный режим. Введите число [0..4]\r\n";
}
return st;
```

Задача 7

```
/// <summary>
/// ВЫВОД МАССИВА
/// </summary>
/// <returns></returns>
private static string PrintSrc() {
    string st = "Исходные данные:\r\n\\t";
    foreach (var item in Filials) {
        st += item + "\\t";
    }
    st += "\\r\n";
    for (int i = 0; i < auto.GetLength(0); i++) {
        st += Kvartal[i] + "\\t";
        for (int j = 0; j < auto.GetLength(1); j++)
            st += auto[i, j] + "\\t\\t";
        st += "\\r\n";
    }
    return st;
}
```


Задача 7

```
/// <summary>Вывод меню </summary>
private static string Print() {
    return @"Выберите, что вы желаете сделать:
    1. Вычислить общее количество автомобилей;
    2. Вывести максимальное количество автомобилей, проданных
    филиалом за квартал (название филиала и номер квартала);
    3. Найти название филиала, который продал максимальное
    количество автомобилей по результатам года (и число проданных);
    4. Найти наиболее успешный квартал (номер квартала и число
    проданных);
    0. Завершить работу.
    Ваш выбор: ";
}
```

```
/// <summary>
/// 1) Подсчитать общее количество автомобилей, проданных всеми
/// филиалами компании за год.
/// </summary>
/// <returns>общее количество автомобилей</returns>
private static int GrandTotal() { // TODO: реализовать метод }
```

Задача 7

```
/// <summary>
/// 2) Вывести максимальное количество автомобилей, проданных
/// филиалом за квартал, а также название филиала и номер квартала.
/// </summary>
/// <param name="Nstroki"></param>
/// <param name="Nstolbca"></param>
private static void GetMax4Kvartal(out int Nstroki, out int Nstolbca) {
    Nstroki = 0;
    Nstolbca = 0;
    for (int i = 0; i < auto.GetLength(0); i++)
        for (int j = 0; j < auto.GetLength(1); j++)
            if (auto[Nstroki, Nstolbca] < auto[i, j]) {
                Nstroki = i;
                Nstolbca = j;
            }
}
```

Задача 7

```
/// <summary>
/// 3) Вывести название филиала, который продал максимальное количество
автомобилей по результатам года, а также их количество
/// </summary>
/// <param name="SumFilial"></param>
/// <param name="NFiliala_MaxAutoYear"></param>
/// <param name="MaxAutoFilialZaGod"></param>
private static void maxAutoFilialZaGod(out int SumFilial, out int
NFiliala_MaxAutoYear, out int MaxAutoFilialZaGod) {// TODO: реализовать}
```

```
/// <summary>
/// 4) Вывести наиболее успешный квартал, в котором компания показала
наилучший результат по продажам(учитываются все филиалы),
/// а также количество автомобилей проданное в нем.
/// </summary>
/// <param name="SumKvartal"></param>
/// <param name="NKvartal_MaxAuto"></param>
/// <param name="MaxAutoKvartal"></param>
private static void maxAutoKvartal(out int SumKvartal, out int
NKvartal_MaxAuto, out int MaxAutoKvartal) {// TODO: реализовать метод}
```

Задача 8

- В классе **Program**, размещённом в файле **Program.cs** написать:
 1. Метод **CreateMatrix()** возвращает целочисленную матрицу размера **M**, **N**, заполненную случайными значениями из диапазона [1;10]. **M**, **N** – целочисленные параметры метода.
 2. Метод **MatrixMult()** возвращает целочисленную матрицу представляющую произведение матриц **A** и **B**, переданных в качестве параметров. Если **A** и **B** не могут быть перемножены, метод возвращает значение **null**.
 3. Метод **MatrixToString()** возвращает строку с табличным представлением матрицы (каждая строка матрицы должна при выводе отображаться на новой строке)
- В том же классе разместить код метода **Main()**, который:
 1. Получает от пользователя значения размеры двух матриц **A** и **B** и формирует их при помощи метода **CreateMatrix()**;
 2. При помощи метода **MatrixMult()** формирует матрицу **C** произведения **AxB**, если это возможно, в противном случае вывести понятное сообщение.
 3. Формирует строки-представления матриц **A**, **B** и **C** при помощи метода **MatrixToString()** и выводит их на экран.
 4. Если матрицы перемножить невозможно, выводит на экран только строки-представления матриц **A**, **B** и сообщение о невозможности их перемножения.

Отладка программы

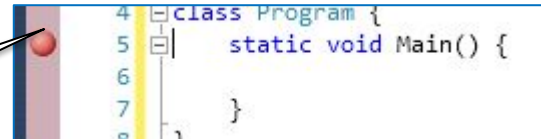
Установка простых точек останова

Установить курсор в строку кода, в которой необходимо установить точку останова.

Debug -> Break point
(Отладка -> Точка останова)

Щелкнуть левой кнопкой мыши по полю редактора слева

F9



```
4 class Program {  
5     static void Main() {  
6  
7     }  
8 }
```

The screenshot shows a code editor with a yellow vertical bar on the left margin. A red dot, representing a break point, is placed on the left margin next to line 5 of the code. The code is for a C# class named Program with a static void Main() method.

Точка останова

Запуск в отладочном режиме:

F5 или Debug -> Start Debugging

Выход из отладочного режима и прекращение выполнения программы:

Shift+F5 или Debug->Stop Debugging (Отладка -> Остановить отладку)

Пошаговое исполнение

С заходом в методы

F11 или Debug -> Step Into (Отладка -> Шаг с заходом)

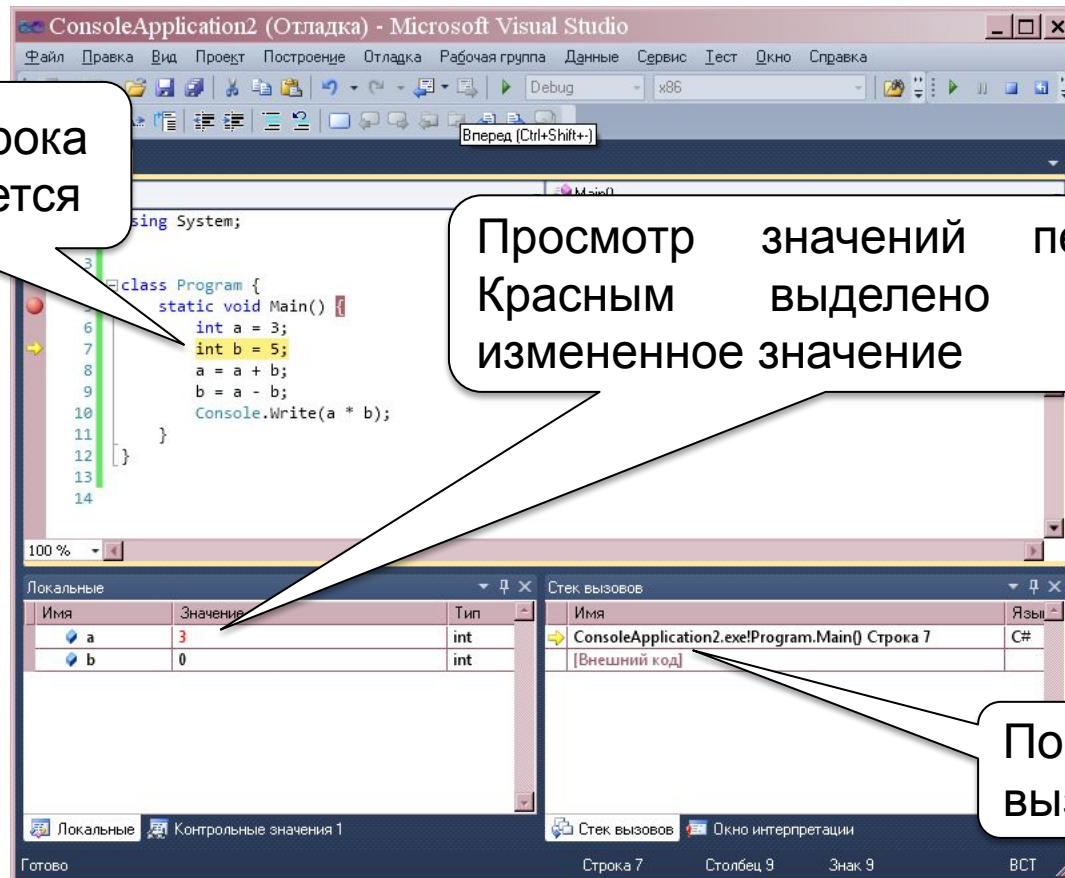
Без захода в методы

F10 или Debug -> Step Over (Отладка -> Шаг с обходом)

Активная строка
подсвечивается

Просмотр значений переменных.
Красным выделено последнее
измененное значение

Последовательность
вызовов

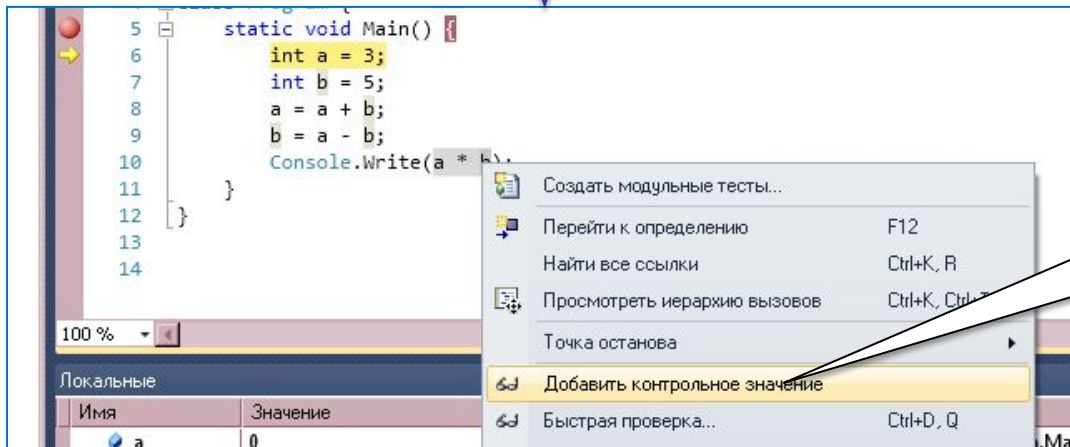


Добавление контролируемого значения

Отладчик VS 2010 позволяет просматривать контролируемые значения. Например, значения выражений.

Выделить интересующее значение

Правой кнопкой мыши вызвать контекстное меню



Задания для самостоятельного решения

Формирование массивов массивов и многомерных массивов

1. Сформировать и заполнить случайными значениями целочисленную матрицу размером $M \times N$ (M и N задаются с клавиатуры). На экран вывести сумму и произведение элементов k -ой строки (k – задается с клавиатуры).
2. Сформировать и заполнить случайными значениями вещественную матрицу размером $M \times N$ (M и N задаются с клавиатуры). На экран вывести сумму элементов для каждого столбца.
3. Написать метод, формирующий по целочисленной матрице $M \times N$ (M и N задаются с клавиатуры) одномерный массив индексов A . В A хранятся индексы столбцов матрицы в отсортированном виде, в порядке возрастания сумм элементов столбцов. В основной программе сформировать матрицу, получить индексный массив, использовать его для вывода столбцов матрицы в порядке возрастания сумм их элементов.

Задания для самостоятельного решения

Работа с элементами массивов массивов и многомерных массивов

- Получить от пользователя целые числа n и m . Сформировать двумерный массив размера $n \times m$ и заполнить его случайными числами $[-100; 100)$. Выполнить следующие преобразования:
 - Заменить максимальный по модулю элемент каждой строки на противоположный по знаку;
 - Вставить после каждой строки с чётным индексом нулевую строку;
 - Удалить все строки, содержащие хотя бы одно нулевое значение;
 - Поменять местами средние столбцы.
- После каждого преобразования матрицу выводить на экран.
- Все преобразования оформить методами.
- Предложить реализацию задачи с использованием массива массивов.