**Тема 5:** Обработка одномерных массивов

**Цель работы:** овладеть основными приемами работы с таблицами для эффективной обработки массивов

**Задание 1.** Разработка класса по обработке массива

Разработать программу обработки одномерного массива по заданию своего варианта. Методы по обработке массива поместите в отдельный класс. Выполните тестирование созданных методов с помощью тест-кейсов.

*Пример оформления статического метода дополнительного класса:*

public class MyArray

{

/// <summary>

/// Метод вычисляет среднее арифметическое положительных элементов

/// </summary>

/// <param name="X">Исходный одномерный массив</param>

/// <returns>Возвращает среднее значение положительных элементов</returns>

/// <throw>Если в массиве нет положительных элементов, среднее получить нельзя</throw>

public static double AverangePositivOfArray(int[] X)

{

int n = X.Length; // количество элементов массива

int k = 0; // количество положительных

int s = 0; // сумма положительных

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

{

if (X[i] > 0)

{

s += X[i];

k++;

}

}

if (k > 0)

return (double)s / k;

else

throw new Exception("В массиве нет положительных элементов.");

}

}

*Пример создания тест-кейсов:*

[TestClass]

public class UnitTest1

{

[TestMethod]

public void TestMethodAvarangePositiv1()

{

int[] A = { -9, -7, 35, 10, -4, 15, 3 };

double expected = 15.75;

double actual = MyArray.AverangePositivOfArray(A);

Assert.AreEqual(expected, actual, 0.000000001, "Ожидаемое значение не получено.");

}

[TestMethod]

public void TestMethodAvarangePositiv2()

{

int[] A = { -3, -11, 0, -78, -1 };

try

{

double actual = MyArray.AverangePositivOfArray(A);

}

catch (Exception ex)

{

StringAssert.Contains(ex.Message, "В массиве нет положительных элементов.");

return;

}

Assert.Fail("Ожидаемое исключение не получено.");

}

**Задание 2.** Настройка пользовательского интерфейса

Ввод исходных данных должен осуществляться пользователем с клавиатуры или случайно. Размеры таблицы должны меняться интерактивно в определённых пределах.

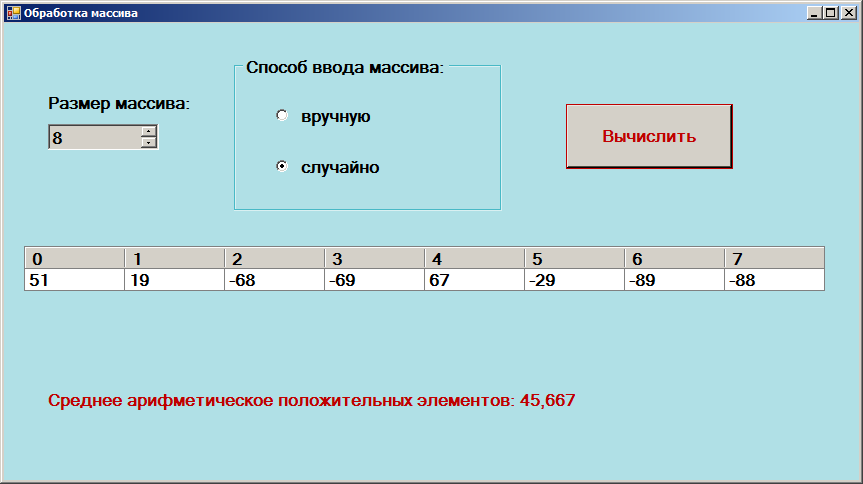
*Основные настройки пользовательского интерфейса:*

Таблица представляет собой специальный компонент, позволяющие отображать данные в виде строк и столбцов. Компонент **dataGridView** позволяет хранить и отображать текстовую и графическую информацию. Но хранение и отображение данных выполняется программистом. Есть возможность подключить источник данных, которые будут представляться в таблице.

radioButton1

groupBox1

radioButton2



button1

label1

dataGridView1

numbericUpDown1

*Рисунок 1 Примерный вид формы*

**Основные свойства и методы табличных компонентов**

**RowCount, ColumnCount** – задают число строк и столбцов таблицы (нумерация строк и столбцов начинается с нуля). Путем задания новых значений этих свойств можно динамически менять размерность таблицы.

**RowHeadersVisible, ColumnHeadersVisible** – задают будут ли отображаться заголовки строк и столбцов. Для обращения к заголовочным ячейкам используется свойство Name: dataGridView1.Rows[i].Name и dataGridView1.Columns[j].Name.

Необходимо помнить, что фиксированные (заголовочные) строки и столбцы редактировать пользователь не может. Они доступны для редактирования только программисту.

**ColumnHeadersHeights, RowHeadersWidths** – задают высоту и ширину ячеек таблицы. Размеры таблицы и её ячеек задаются в пикселях и могут изменяться во время работы программы по содержимому ячеек.

**ColumnHeadersHeightSizeMode** – определяет поведение при изменении высоты заголовков столбцов (установите это свойство в значение EnableResizing, в этом случае высота заголовков столбцов не будет изменяться по содержимому, и мы сможем в конструкторе задать высоту строки заголовка).

**AutoSizeColumnMode** – определяет режим автоматической установки размера для видимых столбцов (установите значение Fill).

**AllowUserToAddRows** – определяет, есть ли возможность добавления строк пользователем (установите значение false).

**AllowUserToDeleteRows** – определяет, есть ли возможность удаления строк пользователем (установите значение false).

**dataGridView1.Rows[i].Cells[j].Value** – свойство используется для доступа к отдельной ячейке с указанием номера столбца и номера строки. Используя цикл для перебора номеров строк и столбцов таблицы, можно поочередно обращаться к каждой ячейке.

Чтобы пользователь мог редактировать содержимое ячеек таблицы, необходимо свойству **ReadOnly** присвоить значение false.

*Например, при запуске формы требуется заполнить таблицу случайными значениями:*

Random r = new Random(); // Инициализация генератора случайных чисел.

/// <summary>

/// Метод обработки события загрузки формы.

/// </summary>

/// <param name="sender">указатель на форму</param>

/// <param name="e">доп.аргумент</param>

private void Form1\_Load(object sender, EventArgs e)

{

int n = Convert.ToInt32(numericUpDown1.Value);

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

{

dataGridView1.Columns[i].Name = i.ToString();

if (radioButton1.Checked)

dataGridView1.Rows[0].Cells[i].Value = 0;

else

dataGridView1.Rows[0].Cells[i].Value = r.Next(-100, 100);

}

}

/// <summary>

/// Кнопка "Вычислить"- запускает метод вычисления среднего значения положительных элементов.

/// </summary>

/// <param name="sender">указатель на button1</param>

/// <param name="e">дополнительный аргумент</param>

private void button1\_Click(object sender, EventArgs e)

{

int n = dataGridView1.ColumnCount;

dataGridView1.RowCount = 1;

dataGridView1.ColumnCount = n;

label2.Text = ""; // Очистка поля результата.

int[] A = new int[n]; // Выделение памяти под массив.

try

{

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

{

A[i] = Convert.ToInt32(dataGridView1.Rows[0].Cells[i].Value);

// Значения из таблицы помещаем в массив.

}

double avarange = MyArray.AverangePositivOfArray(A);

label2.Text = "Среднее арифметическое положительных элементов: " + avarange.ToString("F3"); // Вывод результатов.

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

*Сервисные функции:*

Добавьте в проект проверку нажатых клавиш в процессе редактирования каждой ячейки таблицы. Запретить ввод недопустимых символов при редактировании в ручном режиме.

Если пользователь вручную вводит данные в таблицу, то велика вероятность ошибок при вводе. Поэтому желательно заблокировать недопустимые символы и добавить проверку вводимых в каждую ячейку данных, что сведёт количество ошибок ввода к минимуму. Для этого воспользуемся событием **KeyPress** для таблицы. Событие проверки нажимаемых клавиш должно вызываться при редактировании каждой ячейки таблицы, из события **EditingControlShowing:**

private void dataGridView1\_EditingControlShowing(object sender, DataGridViewEditingControlShowingEventArgs e)

{

e.Control.KeyPress += new KeyPressEventHandler(dataGridView1\_KeyPress);

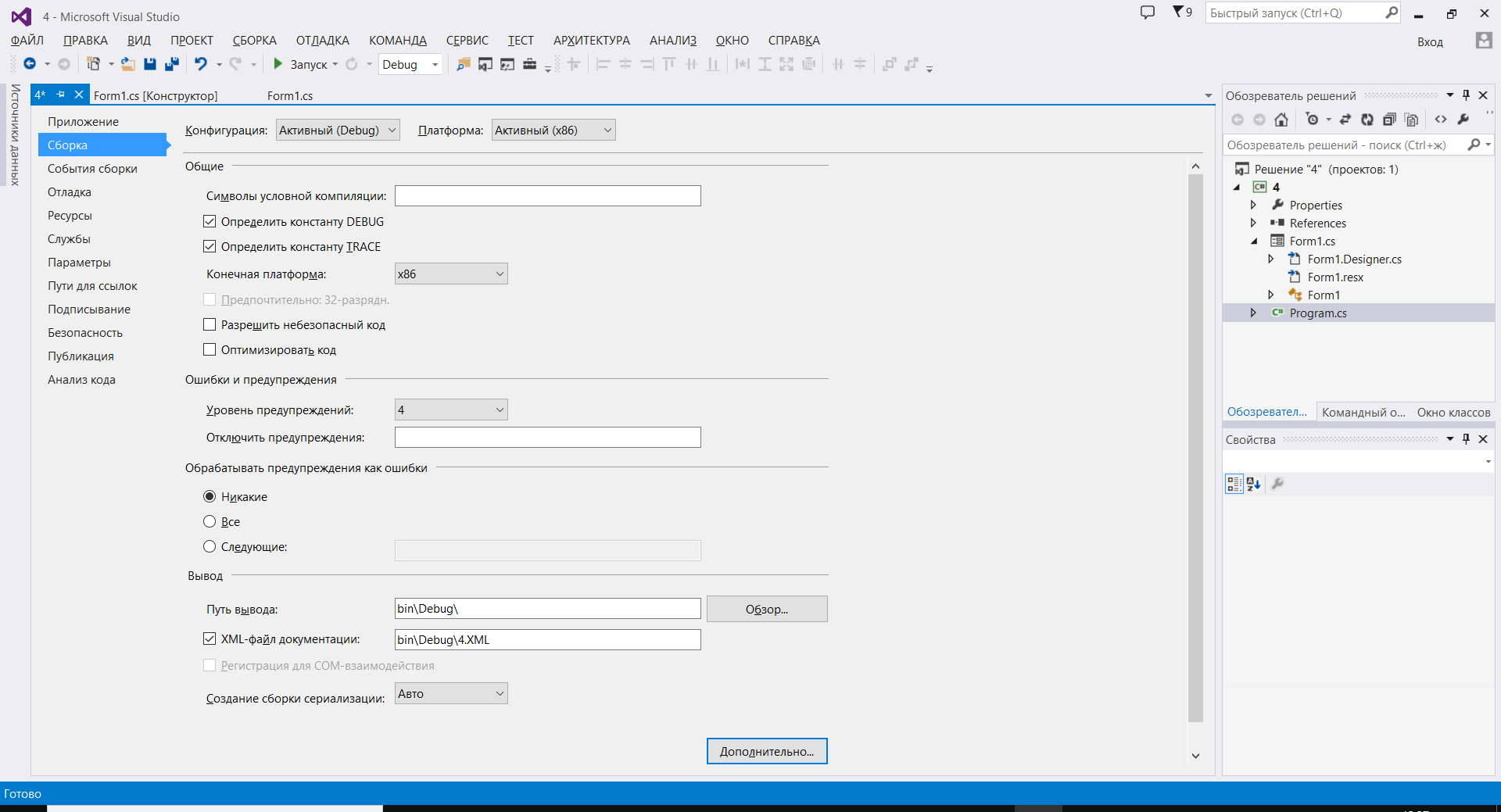
}

**Задание 3.** Обеспечение качества кода

# 3.1 Самодокументируемый код

С# <https://docs.microsoft.com/ru-ru/dotnet/csharp/codedoc>

* Запустить проект
* Добавить XML-комментарии для всех функций
* Продемонстрировать всплывающую подсказку при вводе данной функции
* Выполнить выгрузку комментариев в XML-файл



*Рисунок 2 Выгрузка комментариев в XML-файл*

# 3.2 Соглашение о кодировании

С#: <https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions>

* Отредактировать код в соответствии с соглашением о кодировании

**ВАРИАНТЫ:**

**Вариант 1**

Заполнить одномерный массив целыми числами. Вычислить количество перемен знака с минуса на плюс или с плюса на минус. Нуль знака не имеет.

**Вариант 2**

Заполнить одномерный массив целыми числами. Вычислить произведение элементов массива, кратных своим порядковым номерам, а затем удалить эти элементы из массива. Удаление элементов производиться путем сдвига вперёд. Если таких элементов нет, то вывести сообщение об этом.

**Вариант 3**

Заполнить одномерный массив целыми числами. Вычислить сумму и произведение элементов массива, индексы которых кратны заданному числу, а затем вставить после каждого такого элемента значение 1.

**Вариант 4**

Заполнить одномерный массив целыми числами. Вычислить произведение ненулевых элементов массива, стоящих после первого отрицательного элемента, и поменять порядок следования этих элементов в массиве на обратный. Если отрицательных элементов нет, то вывести сообщение.

**Вариант 5**

Заполнить одномерный массив целыми числами. Вычислить номер первого кратного 3 и последнего кратного 3 элемента массива и поменять их в массиве местами. Вычислить также сумму элементов, стоящих между ними. Если кратных трём нет или только один, то вывести сообщение об этом.

**Вариант 6**

Заполнить одномерный массив целыми числами. Вычислить номер первого отрицательного и последнего отрицательного элемента массива. Вычислить также произведение элементов, стоящих между ними. Если отрицательных элементов нет или только один, то вывести сообщение об этом.

**Вариант 7**

Заполнить одномерный массив целыми числами. Поменять местами первый максимальный и первый минимальный элемент массива.

**Вариант 8**

Заполнить одномерный массив целыми числами. Вычислить сумму и произведение элементов массива, значения которых кратны заданному числу, а затем удалить эти элементы из массива. Удаление элементов производиться путем сдвига вперёд. Если таких чисел нет, то вывести соответствующее сообщение.

**Вариант 9**

Заполнить одномерный массив целыми числами. Вычислить произведение и среднее арифметическое элементов массива, значения которых кратны заданному числу. Если таких чисел нет, то вывести соответствующее сообщение.

**Вариант 10**

Заполнить одномерный массив целыми числами. Вычислить произведение чётных элементов в первой и во второй половине массива. Если в какой-либо половине массива нет чётных элементов, то вывести сообщение об этом.

**Вариант 11**

Заполнить одномерный массив целыми числами. Найти номер первого и последнего минимального элемента массива. Удвоить элементы массива, стоящие до первого минимального.

**Вариант 12**

Заполнить одномерный массив целыми числами. Найти в массиве элементы, кратные заданному числу, и вывести их номера, а затем удалить эти элементы из массива. Удаление элементов производиться путем сдвига вперёд. Если таких элементов нет, то вывести сообщение об этом.

**Вариант 13**

Заполнить одномерный массив целыми числами. Вычислить количество четных и количество нечетных по значениям элементов массива. Если чётных элементов меньше, чем нечётных, или наоборот, вставить в массив значения, равные 1 или 2, в таком количестве, чтобы чётных и нечётных стало одинаково. Вставку осуществлять в начало массива.

**Вариант 14**

Заполнить одномерный массив целыми числами. Вычислить произведение элементов массива, стоящих после последнего максимального элемента, а затем удалить из массива элементы, стоящие после последнего максимального.

**Вариант 15**

Заполнить одномерный массив целыми числами. Вычислить произведение элементов массива, стоящих между максимальным и минимальным элементами, и поменять порядок следования этих элементов в массиве на обратный.

**Вариант 16**

Заполнить одномерный массив целыми числами. Найти номер первого четного и последнего четного по значению элементов массива. Вычислить также сумму элементов, стоящих между ними, и поменять порядок следования этих элементов в массиве на обратный. Если чётных нет или только один, то вывести сообщение об этом.

**Вариант 17**

Заполнить одномерный массив целыми числами. Поменять местами первый положительный и последний положительный элемент массива, и вычислить среднее арифметическое элементов, стоящих между ними. Если положительных нет или только один, то вывести сообщение об этом.

**Вариант 18**

Заполнить одномерный массив целыми числами. Вычислить номер первого кратного 3 и последнего кратного 3 элемента массива и отсортировать в порядке возрастания элементы, стоящие между ними. Если кратных 3 элементов нет, то вывести сообщение об этом.

**Вариант 19**

Заполнить одномерный массив целыми числами. Найти максимальное значение и значение, второе по величине после максимума, и поменять их местами.

**Вариант 20**

Заполнить одномерный массив целыми числами. Найти минимальное значение в массиве и поменять все минимальные элементы местами с первым элементом.