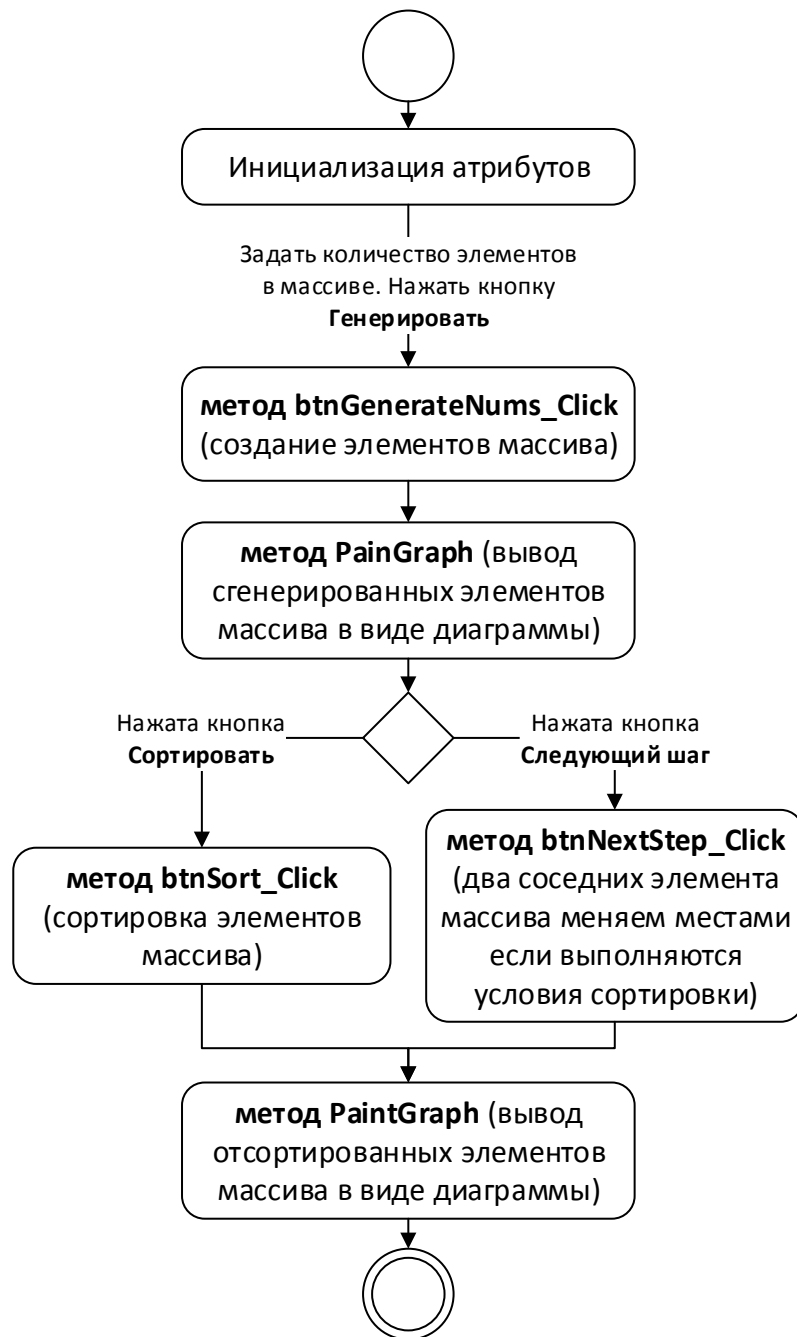


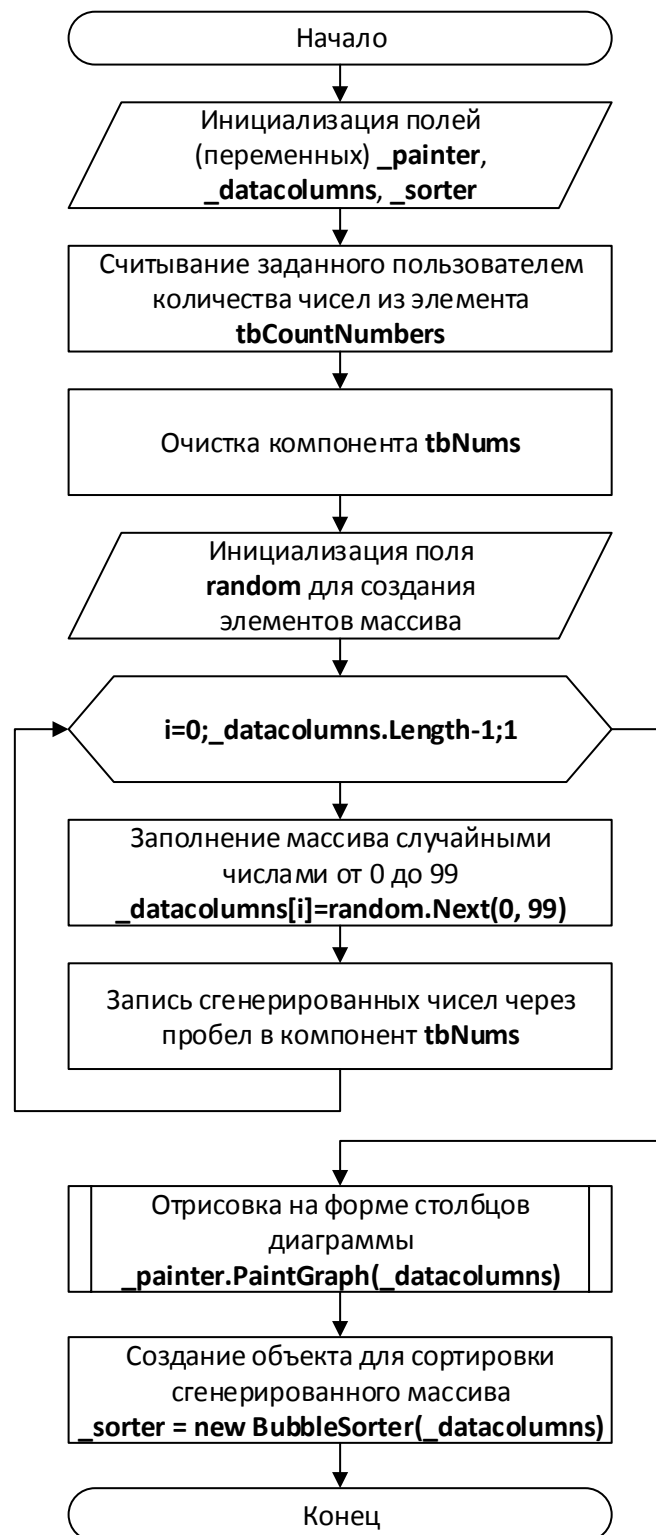
Лабораторная работа № 3

«Сортировка данных»

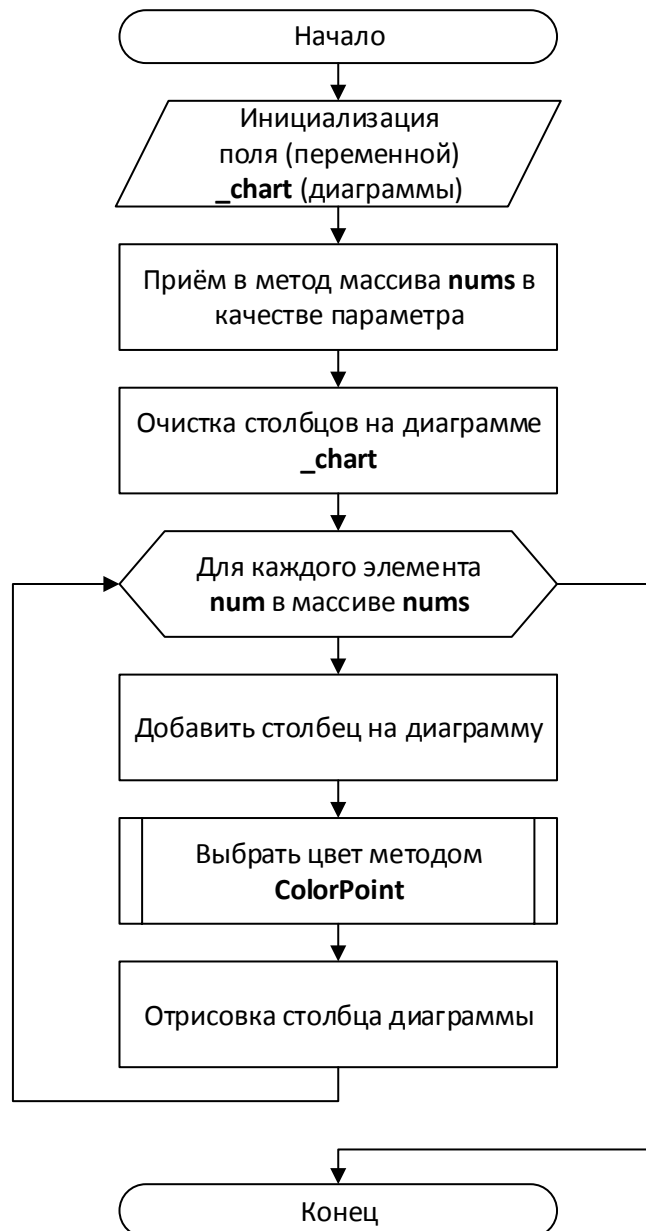
Общий алгоритм программы
(UML-диаграмма действий)



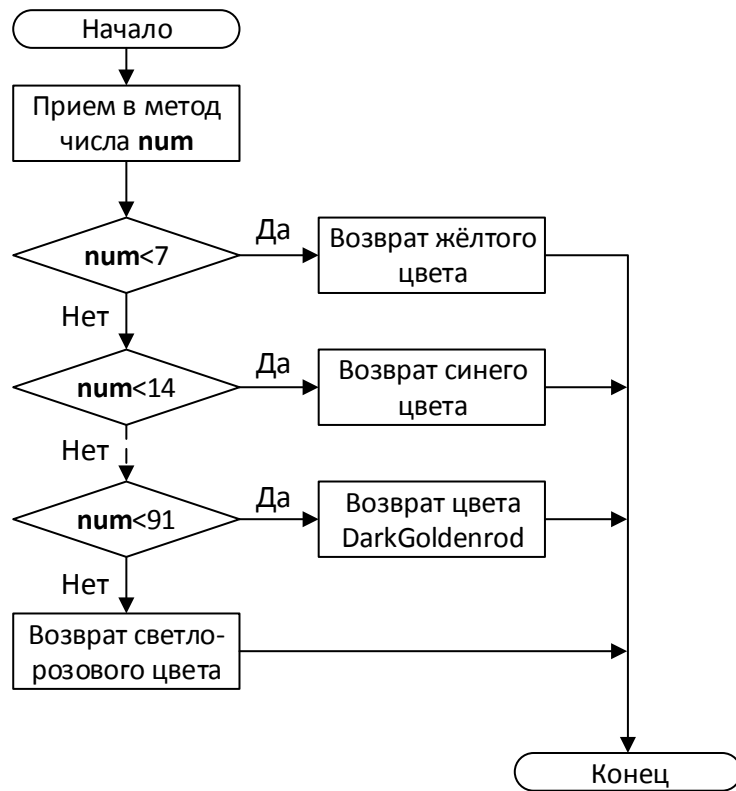
Метод btnGenerateNums_Click
(создание/генерация элементов массива)



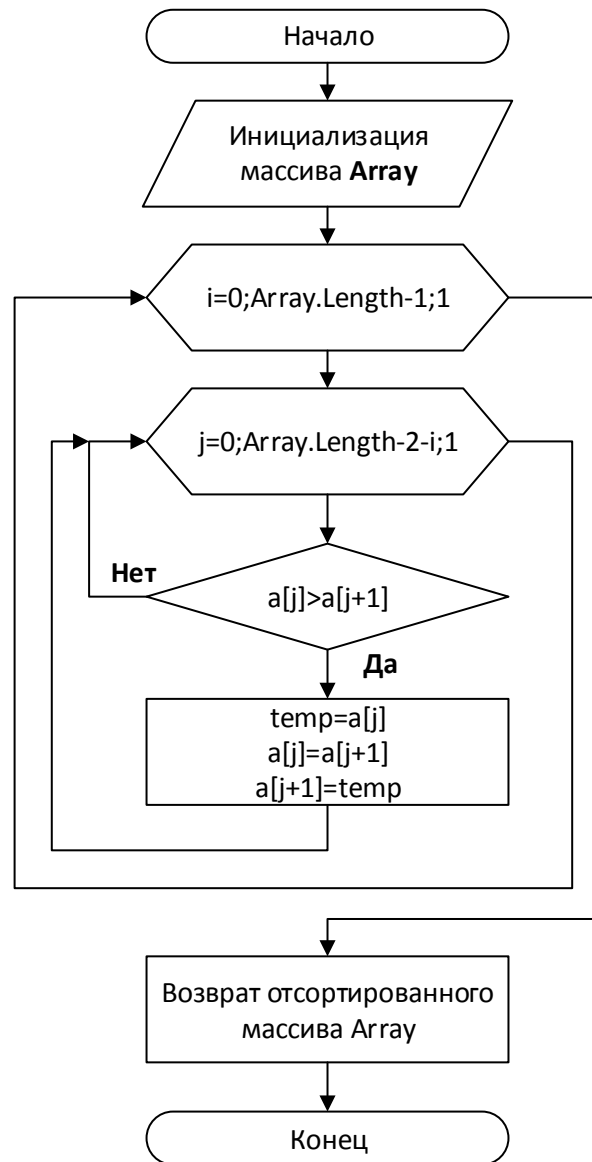
Метод PaintGraph (отрисовка столбцов диаграммы)



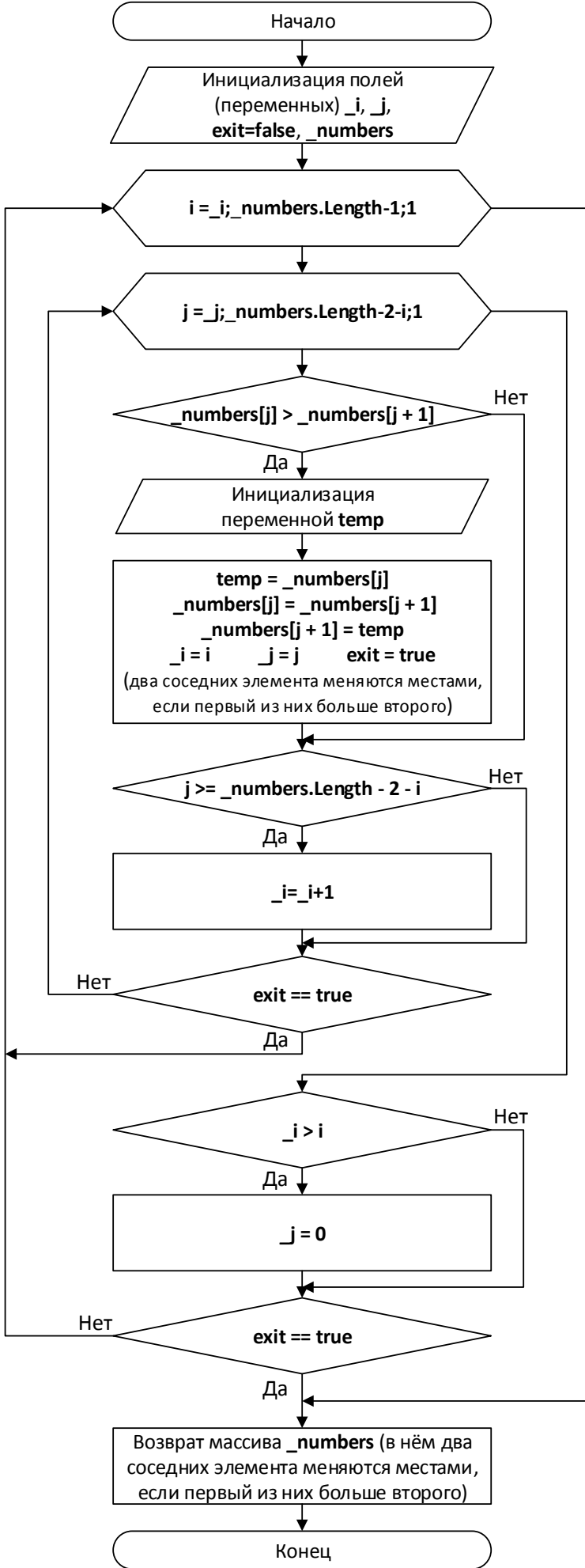
Метод ColorPoint
(выбор цвета для столбца диаграммы)



Метод btnSort_Click
(пузырьковая сортировка элементов массива)



Метод **StepByStepSort** (пошагового изменения массива)



i и j – целые числа.

exit – булева (логическая) переменная.

_numbers – массив целых чисел.

При каждом вызове этого метода (при каждом нажатии на кнопку пошаговой сортировки), меняется местами только одна пара чисел.

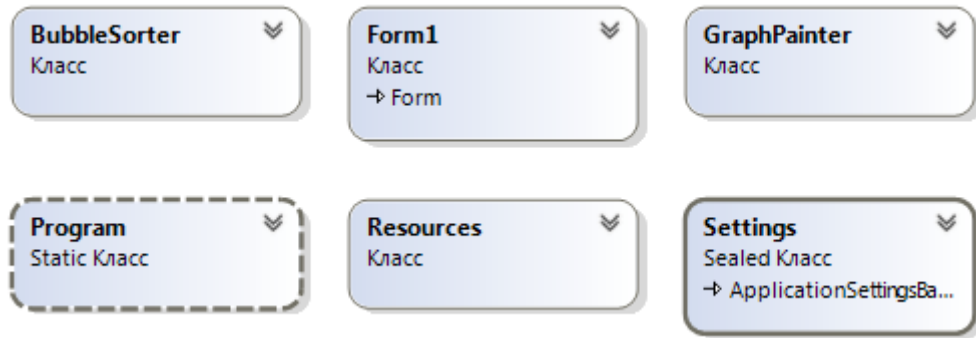
При первой инициализации переменных `_i` и `_j`, им присваиваются нулевые значения (т.е. `_i = 0`, `_j = 0`). В процессе выполнения метода значения переменных `_i` и `_j` будут меняться.

При каждом следующем обращении к этому методу, переменные `_i` и `_j` сохраняют свои значения с момента предыдущего вызова метода (т.е. будут инициализированы предыдущими значениями).

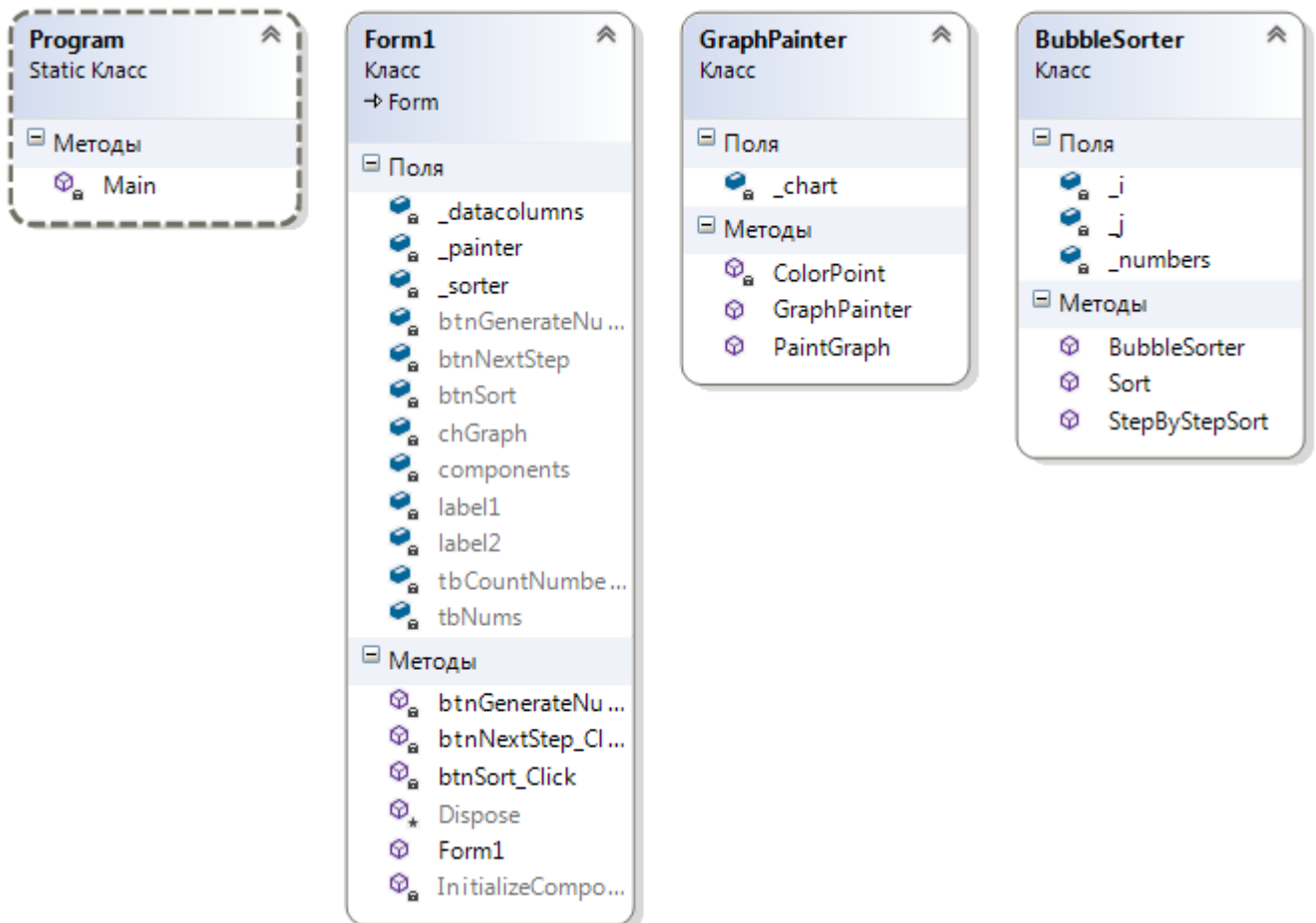
Этим будет достигаться
«отслеживание»
сортировки, продвижение
сортировки от начала
массива к его концу.

Таким образом программа «будет знать» какую соседнюю пару чисел в массиве сравнивать при вызове метода.

Диаграмма классов



Структура классов (только имя элементов класса)



Структура классов (имя и тип элементов класса)

Program
Static Класс

Методы

Main() : void

Form1
Класс
→ Form

Поля

_datacolumns : int[]
 _painter : GraphPainter
 _sorter : BubbleSorter
 btnGenerateNums : Button
 btnNextStep : Button
 btnSort : Button
 chGraph : Chart
 components : IContainer
 label1 : Label
 label2 : Label
 tbCountNumbers : TextBox
 tbNums : TextBox

Методы

btnGenerateNums_Click() : void
 btnNextStep_Click() : void
 btnSort_Click() : void
 Dispose() : void
 Form1()
 InitializeComponent() : void

GraphPainter
Класс

Поля

_chart : Chart

Методы

ColorPoint() : Color
 GraphPainter()
 PaintGraph() : void

BubbleSorter
Класс

Поля

_i : int
 _j : int
 _numbers : int[]

Методы

BubbleSorter()
 Sort() : int[]
 StepByStepSort() : int[]

Структура классов

(полная сигнатура элементов класса)

Program
Static Класс

Методы

Main() : void

GraphPainter
Класс

Поля

_chart : Chart

Методы

ColorPoint(int num) : Color

GraphPainter(Char chart)

PaintGraph(int[] nums) : void

BubbleSorter
Класс

Поля

_i : int

_j : int

_numbers : int[]

Методы

BubbleSorter(int[] numbers)

Sort() : int[]

StepByStepSort() : int[]

Form1
Класс
→ Form

Поля

_datacolumns : int[]

_painter : GraphPainter

_sorter : BubbleSorter

btnGenerateNums : Button

btnNextStep : Button

btnSort : Button

chGraph : Chart

components : IContainer

label1 : Label

label2 : Label

tbCountNumbers : TextBox

tbNums : TextBox

Методы

btnGenerateNums_Click(object sender, EventArgs e) : void

btnNextStep_Click(object sender, EventArgs e) : void

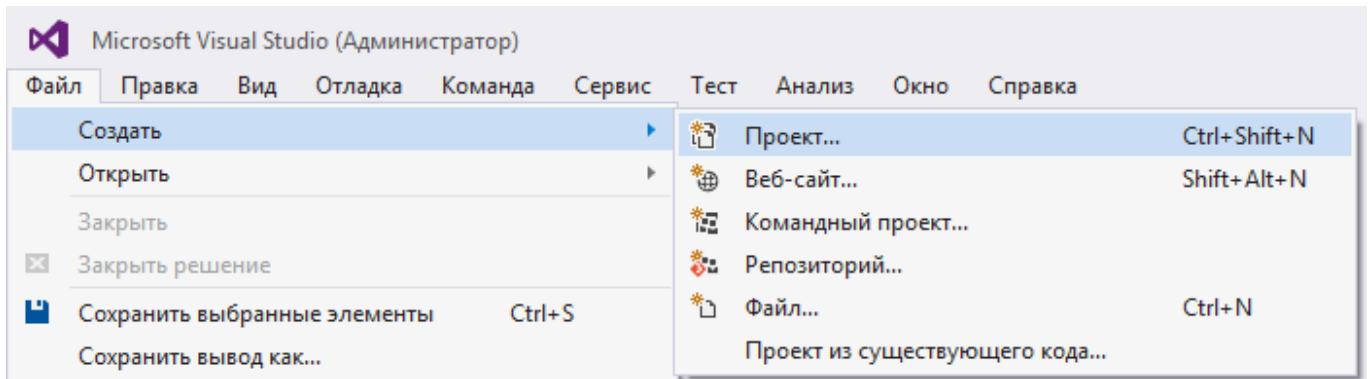
btnSort_Click(object sender, EventArgs e) : void

Dispose(bool disposing) : void

Form1()

InitializeComponent() : void

1. Создаём проект с названием **SortDiagram** в папке **d:\КомпОбрДанн\Csharp**.



В открывшемся окне:

Шаблоны – **Visual C#**

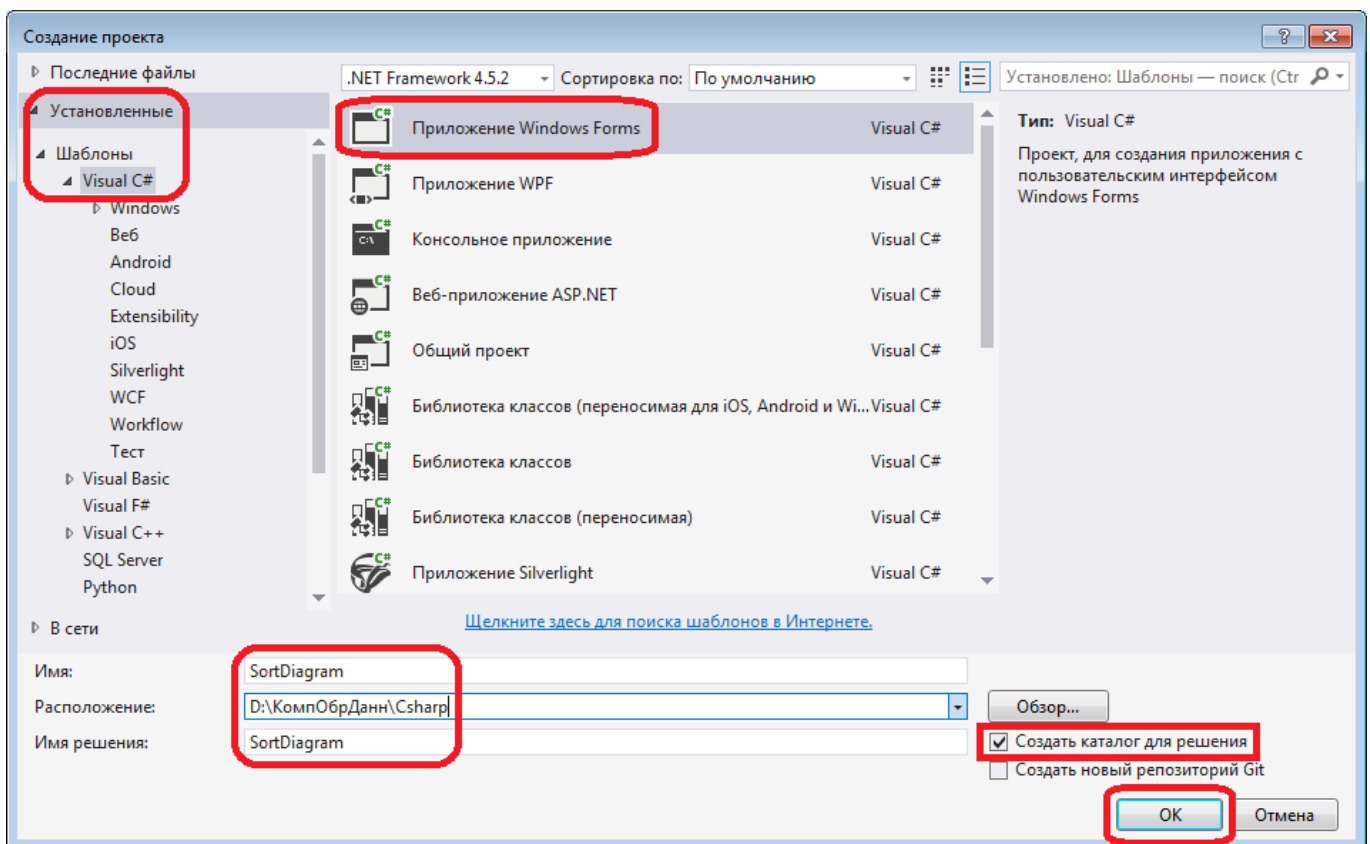
Шаблон – **Приложение Windows Forms**

Имя – **SortDiagram**

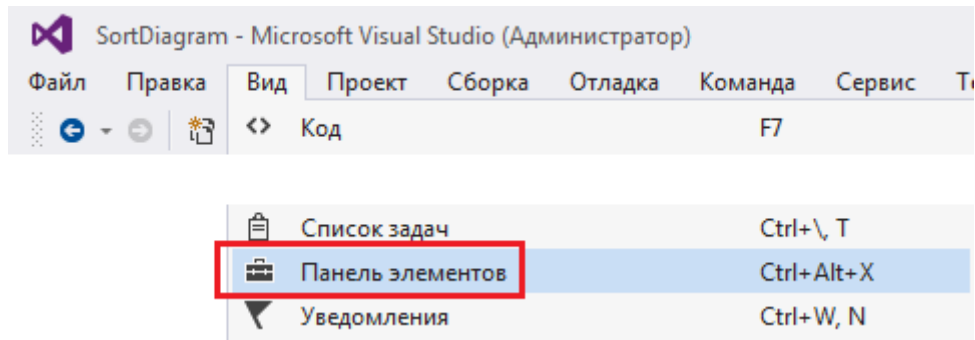
Расположение - **d:\КомпОбрДанн\Csharp**

Создать каталог для решения – поставить галочку

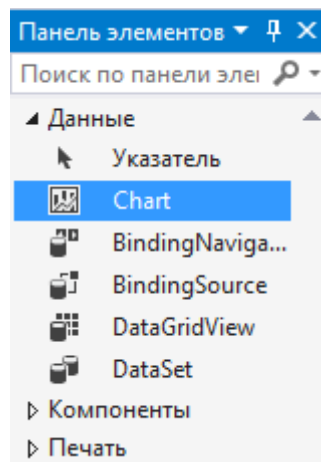
Нажать **ОК**.



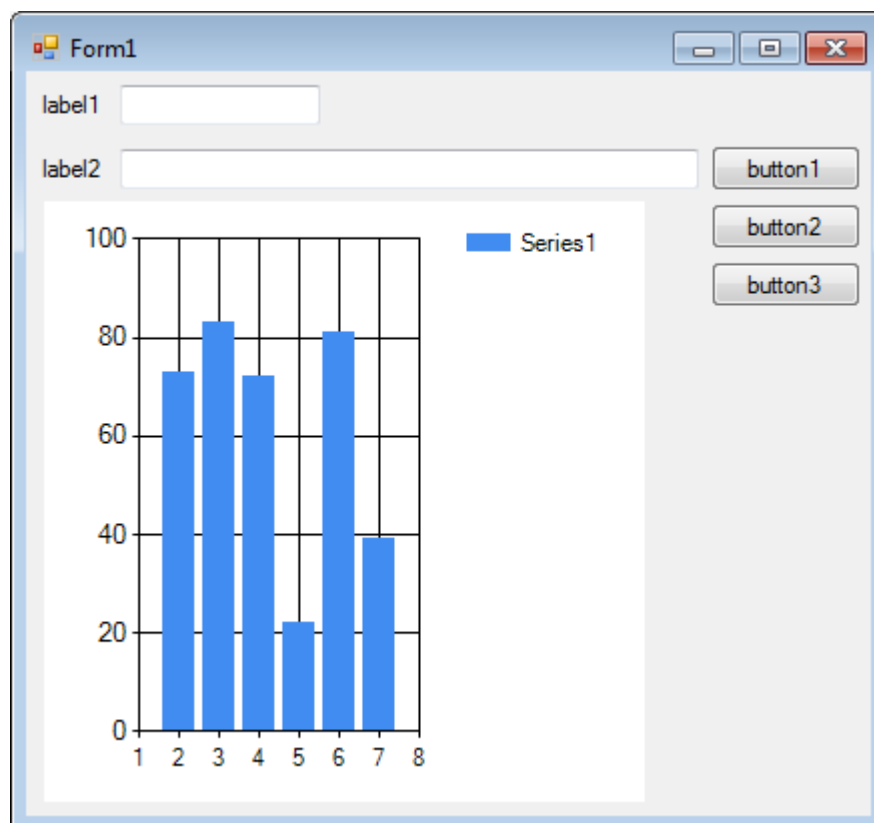
2. В панели меню выбрать **Вид – Панель элементов**.



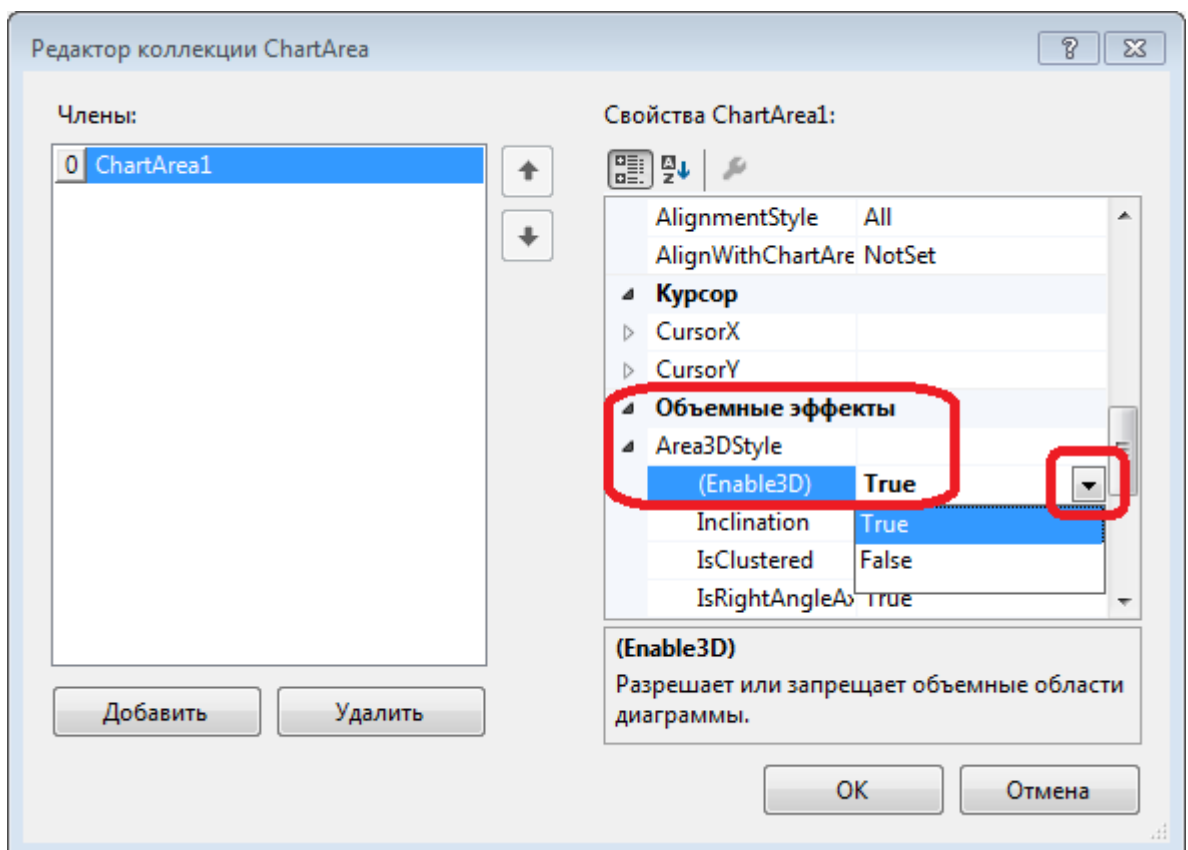
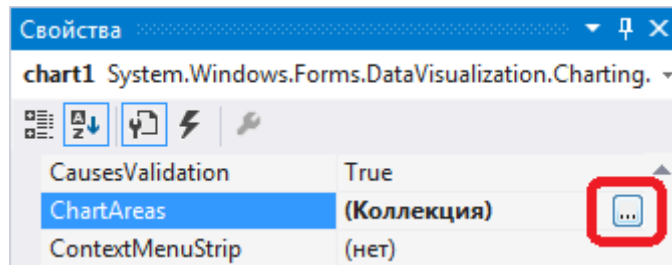
3. В **Панели элементов** раскрыть группу **Данные**, и выбрав элемент **Chart**, поместить его на форму.



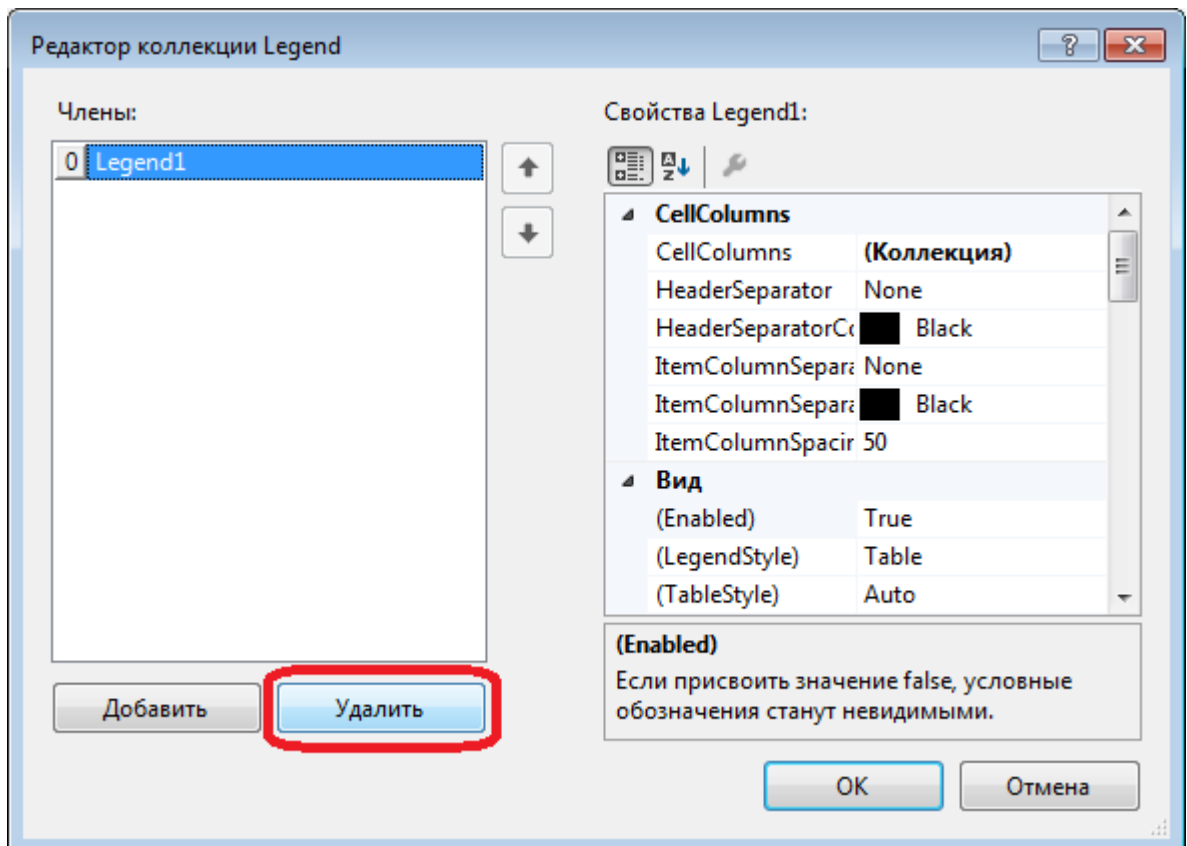
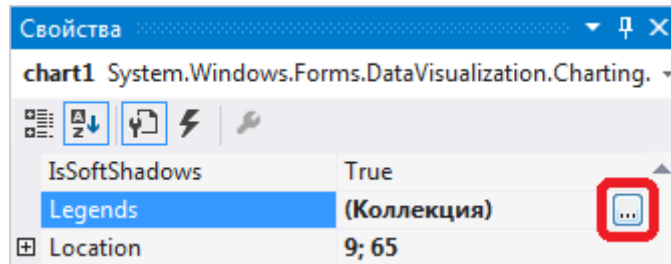
4. В **Панели элементов** раскрыть группу **Все формы Windows Forms** и поместить на форму 2 компонента **label**, 2 компонента **textBox**, 3 компонента **button**.



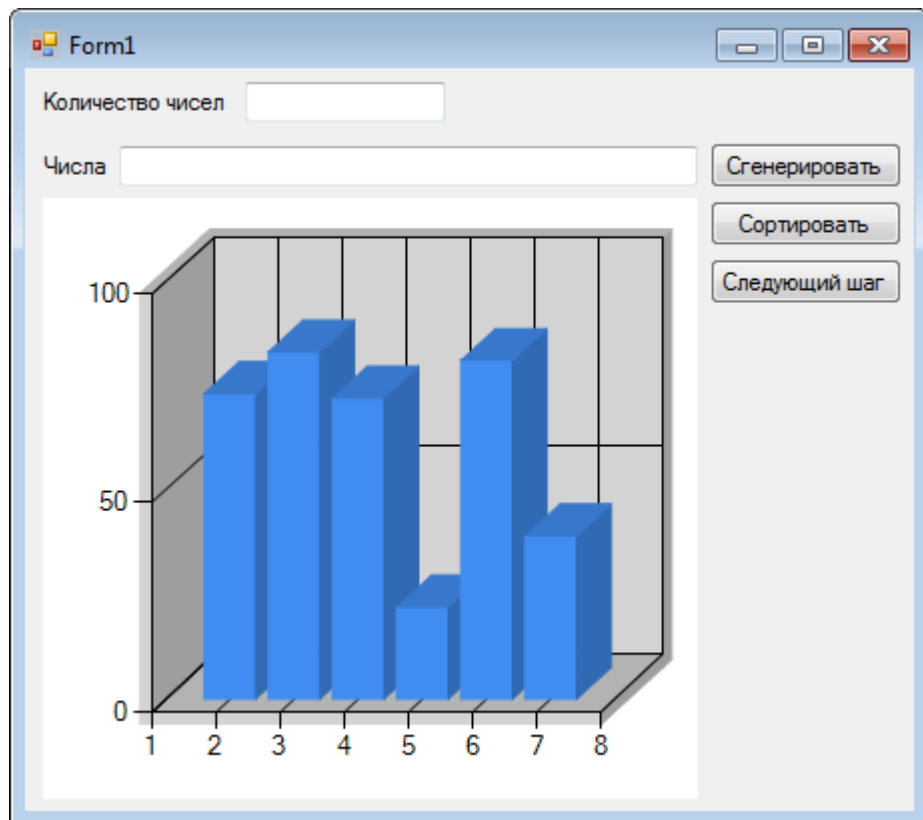
5. Клик правой кнопкой по компоненту **Chart** – выбрать **Свойства**, в окне **Свойства** перейти к пункту **ChartAreas**, щёлкнуть на кнопку справа от значения **(Коллекция)**, в открывшемся окне в правой части найти пункт **Объёмные эффекты** – **Area3DStyle** – **(Enable3D)** – из выпадающего меню выбрать **True** – нажать **OK**.



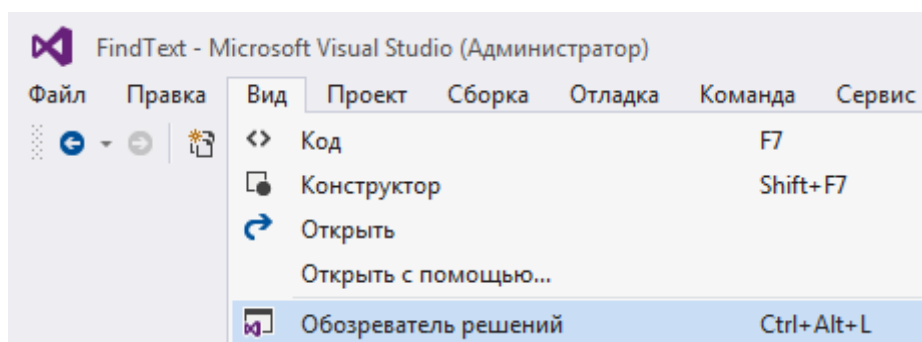
6. В окне **Свойства** перейти к пункту **Legends**, щёлкнуть на кнопку справа от значения **(Коллекция)**, в левой части появившегося окна нажать кнопку **Удалить** – нажать **ОК**.



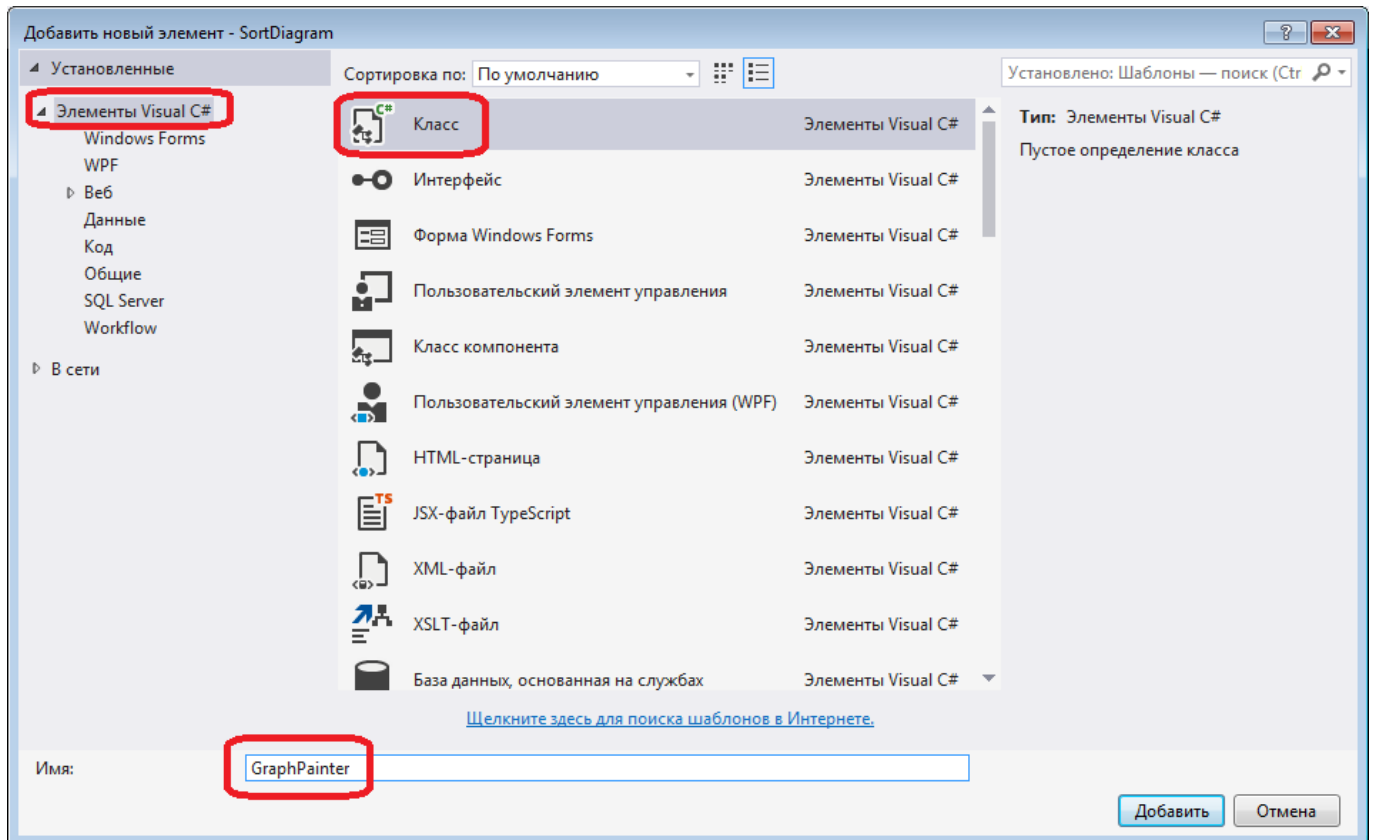
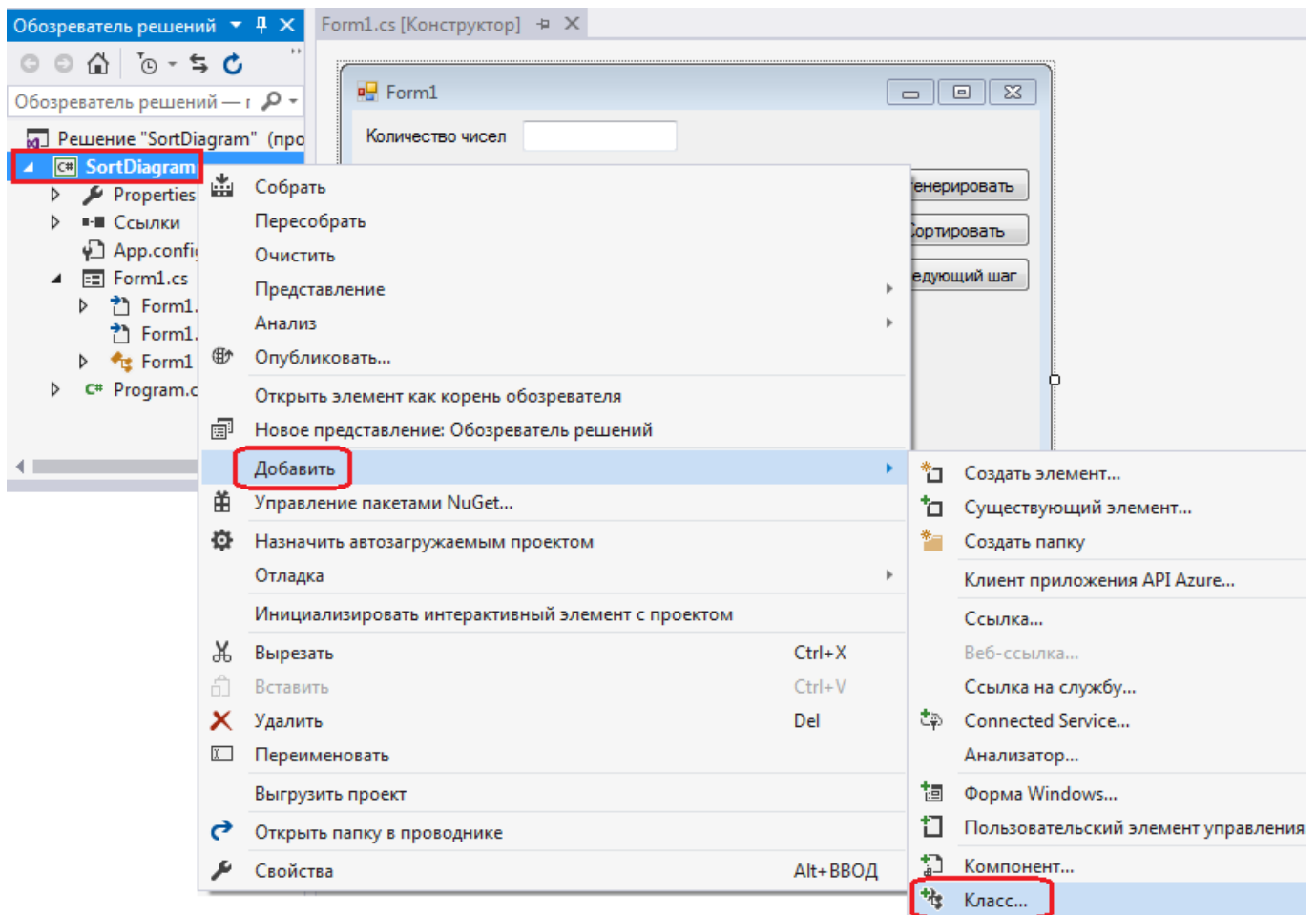
7. У компонента **label1** поменять свойство **Text** на **Количество чисел**.
У компонента **label2** поменять свойство **Text** на **Числа**.
У компонента **textBox1** поменять свойство **(Name)** на **tbCountNumbers**.
У компонента **textBox2** поменять свойство **(Name)** на **tbNums**.
У компонента **chart1** поменять свойство **(Name)** на **chGraph**.
У компонента **button1** поменять свойство **(Name)** на **btnGenerateNums**, свойство **Text** на **Сгенерировать**.
У компонента **button2** поменять свойство **(Name)** на **btnSort**, свойство **Text** на **Сортировать**.
У компонента **button3** поменять свойство **(Name)** на **btnNextStep**, свойство **Text** на **Следующий шаг**.



8. В меню **Вид** выбрать **Обозреватель решений**.



В **Обозревателе решений** кликнуть правой кнопкой мыши на проекте **SortDiagram** – **Добавить** – **Класс** – в открывшемся окне задать имя класса **GraphPainter** – **Добавить**.

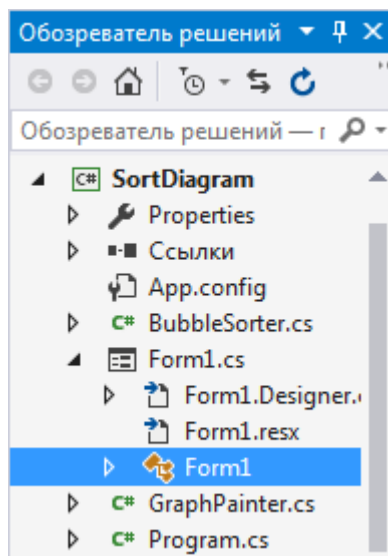


9. Аналогично создать класс **BubbleSorter**.

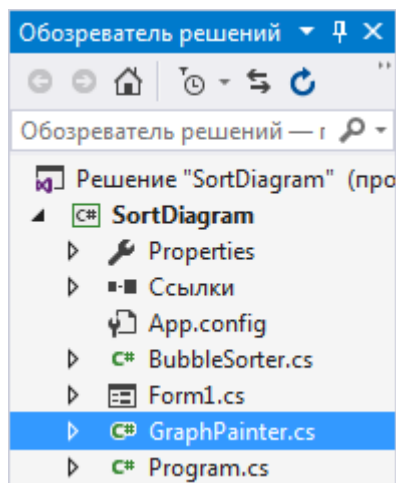
10. Методы обработки событий нажатий кнопок размещаются в классе **Form1**.

Окно формы (на которую помещали компоненты **button**, **chart**, **label**) находится в **Обозревателе решений** в **SortDiagram – Form1.cs**.

Окно с кодом, описывающим класс формы **Form1**, находится в **SortDiagram – Form1.cs – Form1**.



11. В **Обозревателе решений** двойной клик по классу **GraphPainter**.



12.B классе **GraphPainter** заменить используемые пространства имён на **System.Windows.Forms.DataVisualization.Charting, System.Drawing**

```
1 using System.Drawing;
2 using System.Windows.Forms.DataVisualization.Charting;
```

и добавить следующий код:

```
4 namespace SortDiagram
5 {
6     class GraphPainter
7     {
8         // объявляем переменную
9         private readonly Chart _chart;
10
11         // метод-конструктор класса
12         public GraphPainter(Chart chart)
13         {
14             // переменная _chart принимает данные chart,
15             // переданные из другого класса
16             _chart = chart;
17         }
18
19         // метод отрисовки данных на диаграмме
20         public void PaintGraph(int[] nums)
21         {
22             // очищаем точки (столбцы) на диаграмме
23             _chart.Series[0].Points.Clear();
24             // для каждого элемента в массиве данных
25             foreach (int num in nums)
26             {
27                 // добавляем точку (столбец) на диаграмму
28                 _chart.Series[0].Points.Add(num);
29                 // узнаём порядковый номер точки (столбца)
30                 int count = _chart.Series[0].Points.Count;
31                 // задаём точке (столбцу) цвет
32                 _chart.Series[0].Points[--count].Color = ColorPoint(num);
33             }
34         }
35
36         // метод задания цвета точкам (столбцам) на диаграмме
37         private Color ColorPoint(int num)
38         {
39             if (num < 7) return Color.Yellow;
40             if (num < 14) return Color.Blue;
41             if (num < 21) return Color.Green;
42             if (num < 28) return Color.LightBlue;
43             if (num < 35) return Color.MediumPurple;
44             if (num < 42) return Color.AntiqueWhite;
45             if (num < 49) return Color.Aquamarine;
46             if (num < 56) return Color.Bisque;
47             if (num < 63) return Color.Indigo;
48             if (num < 70) return Color.Brown;
49             if (num < 77) return Color.Coral;
50             if (num < 84) return Color.DarkBlue;
51             if (num < 91) return Color.DarkGoldenrod;
52             return Color.LightPink;
53         }
54     }
55 }
```

13. В классе **BubbleSorter** добавить следующий код:

```
1 namespace SortDiagram
2 {
3     class BubbleSorter
4     {
5         // объявление переменных
6         private int _i;
7         private int _j;
8         // объявление массива
9         private readonly int[] _numbers;
10
11         // метод-конструктор класса
12         public BubbleSorter(int[] numbers)
13         {
14             // переменная _numbers принимает данные numbers,
15             // переданные из другого класса
16             _numbers = numbers;
17         }
18
19         // метод пузырьковой сортировки по возрастанию
20         public int[] Sort()
21         {
22             for (int i = 0; i < _numbers.Length; i++)
23                 for (int j = 0; j < _numbers.Length - 1 - i; j++)
24                 {
25                     if (_numbers[j] <= _numbers[j + 1]) continue;
26                     int temp = _numbers[j];
27                     _numbers[j] = _numbers[j + 1];
28                     _numbers[j + 1] = temp;
29                 }
30             // возврат отсортированных данных
31             return _numbers;
32         }
33     }
34 }
```

```

34 // метод пошагового изменения массива: при каждом вызове
35 // этого метода, меняется местами только одна пара чисел
36 public int[] StepByStepSort()
37 {
38     bool exit = false;
39     for (int i = _i; i < _numbers.Length; i++)
40     {
41         for (int j = _j; j < _numbers.Length - 1 - i; j++)
42         {
43             if (_numbers[j] > _numbers[j + 1])
44             {
45                 int temp = _numbers[j];
46                 _numbers[j] = _numbers[j + 1];
47                 _numbers[j + 1] = temp;
48                 _i = i;
49                 _j = j;
50                 exit = true;
51             }
52             if (j >= _numbers.Length - 2 - i)
53                 _i++;
54             if (exit)
55                 break;
56         }
57         if (_i > i) _j = 0;
58         if (exit)
59             break;
60     }
61     return _numbers;
62 }
63
64 }
65
66 }

```

14. Изначально класс формы (**Обозреватель решений - SortDiagram – Form1.cs – Form1**) содержит автоматически сгенерированный код:

```

11 namespace SortDiagram
12 {
13     public partial class Form1 : Form
14     {
15     }
16 }

```

15.Перейти на форму (**Обозреватель решений - SortDiagram – Form1.cs**), и дважды кликнуть на каждой из трёх кнопок, размещённых на форме. В класс формы автоматически добавятся методы обработки событий нажатий на кнопки:

```
11 namespace SortDiagram
12 {
13     public partial class Form1 : Form
14     {
15         public Form1()
16         {
17             InitializeComponent();
18         }
19
20         private void btnGenerateNums_Click(object sender, EventArgs e)
21         {
22         }
23
24         private void btnSort_Click(object sender, EventArgs e)
25         {
26         }
27
28         private void btnNextStep_Click(object sender, EventArgs e)
29         {
30         }
31     }
32 }
33
34
35
```

16. В классе **Form1** заменить пространство имён, сгенерированное по умолчанию на **System** и **System.Windows.Forms**:

```
1 using System;
2 using System.Windows.Forms;
```

и добавить следующий код:

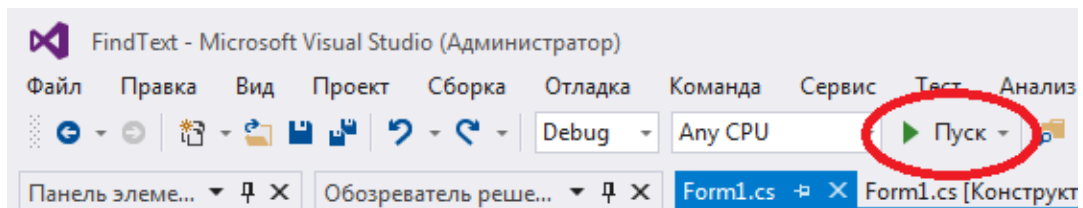
```
4 namespace SortDiagram
5 {
6     public partial class Form1 : Form
7     {
8         // переменная для работы с диаграммой
9         private readonly GraphPainter _painter;
10        // переменные для работы с данными
11        private int[] _datacolumns;
12        private BubbleSorter _sorter;
13
14        // метод-конструктор класса
15        public Form1()
16        {
17            // инициализация компонентов формы при её создании
18            InitializeComponent();
19            // создание объекта для диаграммы
20            _painter = new GraphPainter(chGraph);
21        }
22
23        // метод генерации данных
24        private void btnGenerateNums_Click(object sender, EventArgs e)
25        {
26            // объявляем массив с количеством элементов, заданным в tbCountNumbers
27            _datacolumns = new int[Convert.ToInt32(tbCountNumbers.Text)];
28            // очищаем текст в компоненте tbNums
29            tbNums.Text = "";
30            // создаём объект для генерации чисел
31            Random random = new Random();
32            // заполняем массив _datacolumns случайными числами
33            for (int i = 0; i < _datacolumns.Length; i++)
34            {
35                // случайные числа из диапазона от 0 до 99
36                int num = random.Next(0, 99);
37                _datacolumns[i] = num;
38                // записываем сгенерированные числа через пробел в tbNums
39                tbNums.Text += num + " ";
40            }
41            // необходимо убрать лишний пробел в конце строки
42            tbNums.Text = tbNums.Text.Trim();
43            // вызываем метод рисования столбцов на диаграмме
44            _painter.PaintGraph(_datacolumns);
45            // создаём объект для сортировки в дальнейшем
46            _sorter = new BubbleSorter(_datacolumns);
47        }
48    }
49 }
```

```

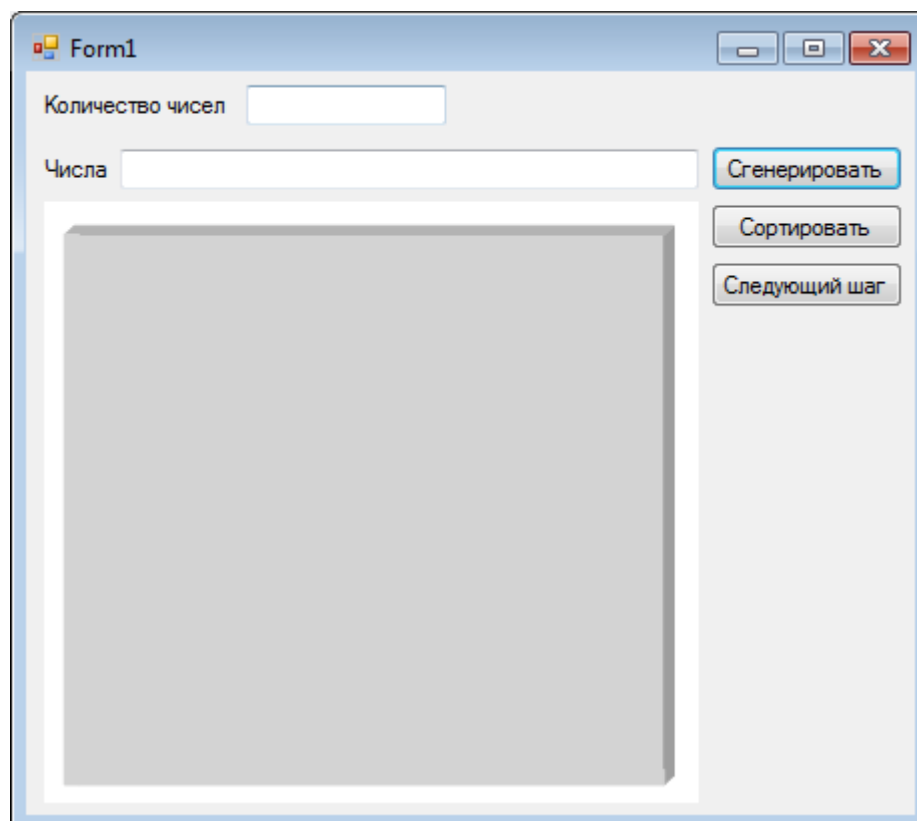
49 // метод сортировки данных
50 private void btnSort_Click(object sender, EventArgs e)
51 {
52     // если данных нет - то выходим из метода
53     if (tbNums.Text.Length == 0 || _sorter == null) return;
54     // если данные есть - сортируем их
55     _datacolumns = _sorter.Sort();
56     // и выводим отсортированные данные на диаграмму
57     _painter.PaintGraph(_datacolumns);
58 }
59
60 // метод пошаговой сортировки данных
61 private void btnNextStep_Click(object sender, EventArgs e)
62 {
63     // если данных нет - то выходим из метода
64     if (tbNums.Text.Length == 0 || _sorter == null) return;
65     // если данные есть - изменяем их пошагово
66     _datacolumns = _sorter.StepByStepSort();
67     // и каждый шаг отображаем на диаграмме
68     _painter.PaintGraph(_datacolumns);
69 }
70 }
71 }

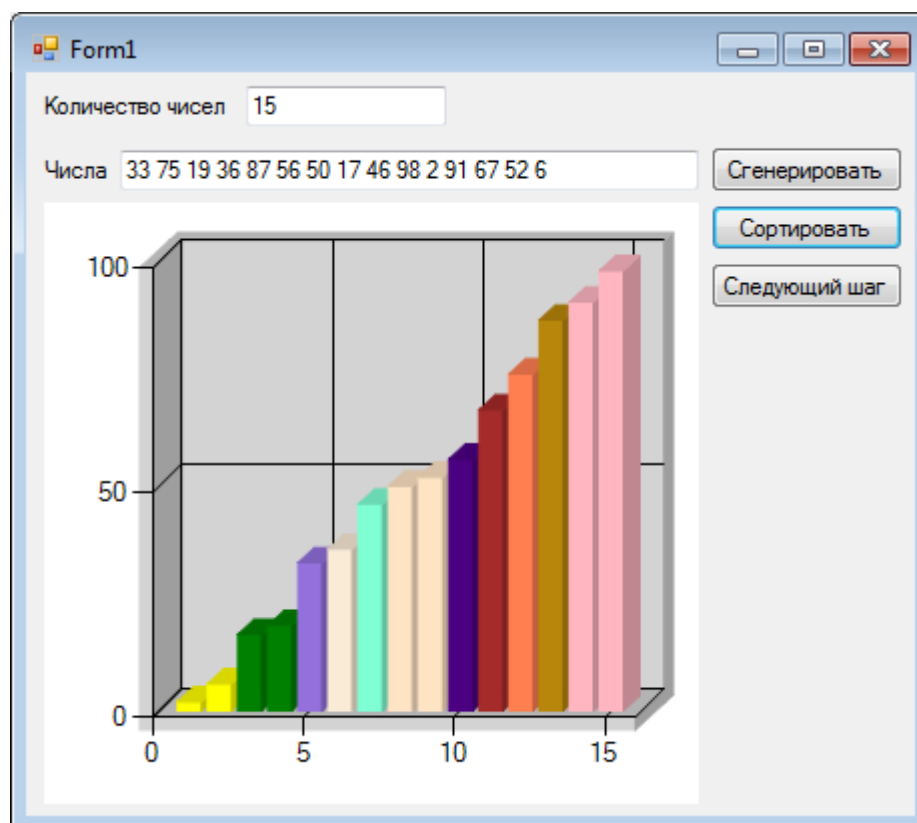
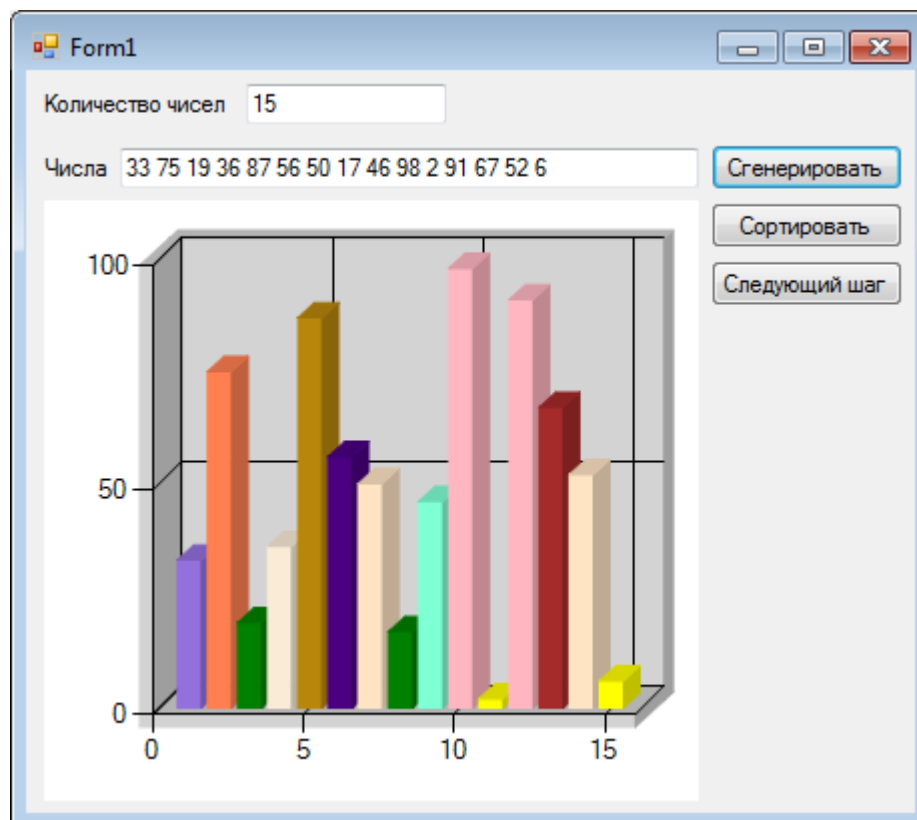
```

17. Запустить программу



18. Результат выполнения

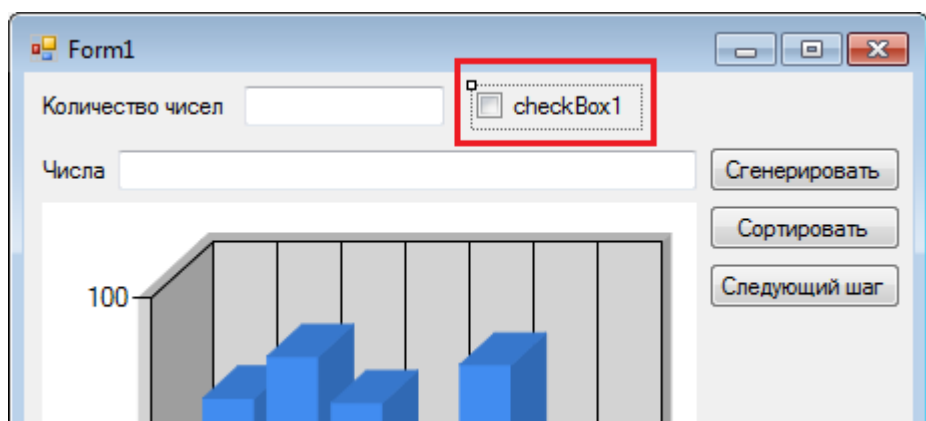
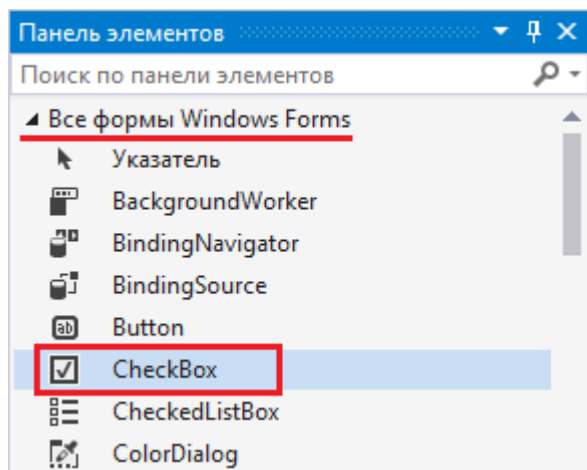
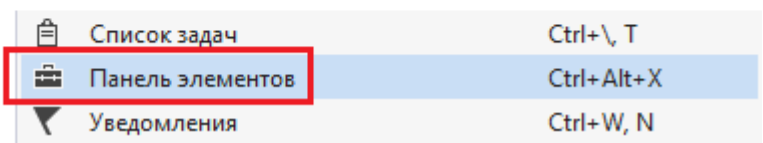
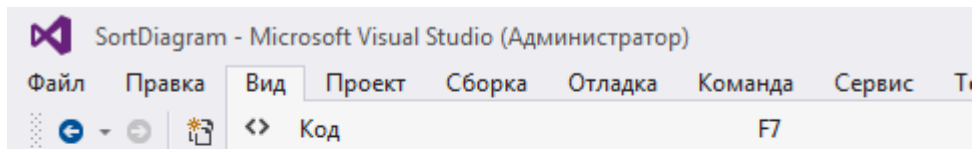




Улучшение программы

В данном случае сортировка массива происходит только по возрастанию. Изменим код программы так, чтобы при нажатии на кнопку **Сортировать** можно было сортировать массив ещё и по убыванию.

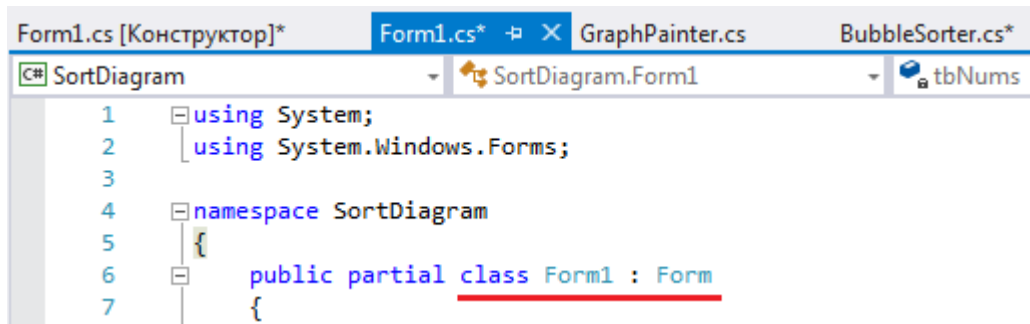
19. На **Панели элементов** (меню **Вид – Панель элементов**) найти компонент **CheckBox** и добавить его на форму.



Правой кнопкой мыши по компоненту **CheckBox** – **Свойства** - в **Окне свойств** компонента **CheckBox** на вкладке **Свойства** изменить:

- свойство **(Name)** с **checkBox1** на **chbInvers**
- свойство **Text** с **checkBox1** на **Инверсия**.

20. Перейти в класс **Form1**.



Найти метод **btnSort_Click**.

В этом методе изменить код **_sorter.Sort()**; на **_sorter.Sort(chbInvers)**;

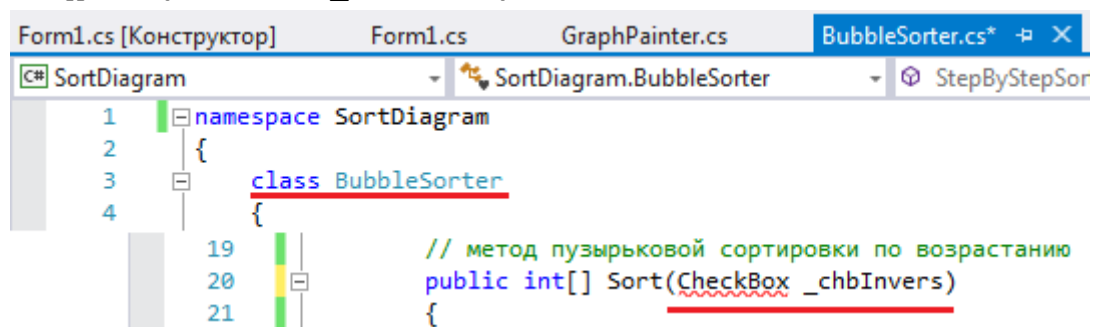
```
// метод сортировки данных
private void btnSort_Click(object sender, EventArgs e)
{
    // если данных нет - то выходим из метода
    if (tbNums.Text.Length == 0 || _sorter == null) return;
    // если данные есть - сортируем их
    _datacolumns = _sorter.Sort(chbInvers);
    // и выводим отсортированные данные на диаграмму
    _painter.PaintGraph(_datacolumns);
}
```

Таким образом первоначально метод **Sort** объекта **_sorter** не принимал никаких параметров. Теперь данному методу в качестве единственного параметра передаётся объект **chbInvers** (с типом **CheckBox**). Среда **Visual Studio** подчёркивает метод **Sort** красным цветом, показывая предупреждение, что ни одна из текущих реализаций (в нашем случае текущая реализация всего одна) метода **Sort** не принимает ни одного параметра.

Так как объект **_sorter** является экземпляром класса **BubbleSorter**, и, соответственно, метод **Sort** реализован в классе **BubbleSorter**, то необходимо изменить метод **Sort** в классе **BubbleSorter**.

21. Перейти в класс **BubbleSorter** к методу **Sort**.

В параметрах метода указать, что метод **Sort** принимает данные типа **CheckBox**, и значение этих данных присваивает локальной переменной **_chbInvers**, т.е. заменить строку **public int[] Sort()** на строку **public int[] Sort(CheckBox _chbInvers)**.

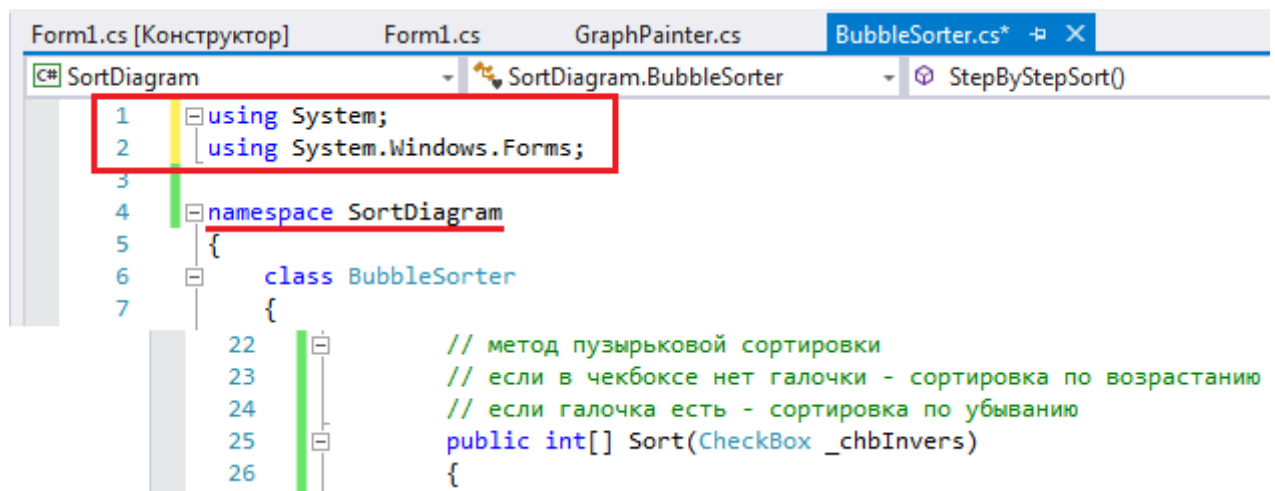


Среда **Visual Studio** подчёркивает тип **CheckBox** красным цветом, показывая предупреждение: **Не удалось найти тип или пространство имён «CheckBox» (возможно отсутствует директива using или ссылка на сборку).**

Для этого перед объявлением пространства имён **namespace SortDiagram** добавить две директивы **using**:

- **using System.Windows.Forms;** (эта директива нужна для использования в коде данных типа **CheckBox**).
- **using System;** (эта директива нужна для использования в коде методов обработки массивов **Array**, которые понадобятся ниже).

После этого тип **CheckBox** воспринимается корректно.



22. Изменим тело метода **Sort**.

```
// метод пузырьковой сортировки
// если в чекбоксе нет галочки - сортировка по возрастанию
// если галочка есть - сортировка по убыванию
public int[] Sort(CheckBox _chbInvers)
{
    if (_chbInvers.Checked == false)
    {
        Array.Sort(_numbers, 0, _numbers.Length);
    }
    else
    {
        Array.Sort(_numbers, 0, _numbers.Length);
        Array.Reverse(_numbers);
    }
    // возврат отсортированных данных
    return _numbers;
}
```

Т.е. если в компоненте **chbInvers** (с типом **CheckBox**) галочка не стоит, тогда свойство **Checked** данного компонента находится в состоянии **false** (ложь). В этом случае происходит сортировка массива по возрастанию.

В противном случае **else** (если галочка стоит) происходит сортировка массива по убыванию (сначала массив сортируется по возрастанию командой **Array.Sort**, затем командой **Array.Reverse** происходит замена порядка элементов в массиве на обратный).

Таким образом, в качестве параметра в метод **Sort** поступает весь объект типа **CheckBox** и далее в коде используется свойство этого объекта.

Сам метод **Array.Sort** принимает в качестве параметров собственно массив **_numbers** (который надо отсортировать), начальный индекс диапазона сортировки (в данном случае массив сортируется с самого начала, т.е. с элемента с нулевым индексом), и число элементов в диапазоне сортировки **_numbers.Length** (в данном случае все элементы массива **_numbers**), а возвращает отсортированный массив **_numbers**.

23. Запустить программу на выполнение.

