

## **ЛАБОРАТОРНАЯ РАБОТА №1**

### **ТЕМА: «ВВЕДЕНИЕ В РАЗРАБОТКУ ГРАФИЧЕСКИХ ПОЛЬЗОВАТЕЛЬСКИХ ИНТЕРФЕЙСОВ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИИ WINDOWS FORMS»**

#### **ЦЕЛЬ РАБОТЫ**

Научиться размещать и настраивать внешний вид элементов управления на форме и создавать обработчики событий.

#### **ХОД ВЫПОЛНЕНИЯ РАБОТЫ**

Для Windows-приложений, ориентированных на выполнение в общезыковой исполняющей среде (CLR), в качестве основы построения графического пользовательского интерфейса используются классы Windows Forms из библиотек .NET Framework, т.е. Windows Forms – это набор средств для создания Windows-приложений, выполняющихся в среде CLR.

Средства Windows Forms обеспечивают быструю разработку пользовательского интерфейса, поскольку он создается из стандартных элементов, а соответствующий программный код генерируется автоматически. Затем требуется лишь доработать сгенерированный код, чтобы добиться необходимой функциональности.

Применяя Windows Forms, можно построить удобный пользовательский интерфейс, в том числе главное окно приложения, выбирая подходящие элементы управления, с которыми взаимодействует пользователь.

Чтобы создать Windows-приложение на основе Windows Forms, следует выбрать команду главного меню «Файл» (File) → «Создать» (New) → «Проект» (Project), затем выбрать тип проекта «Рабочий стол» (Windows Desktop), а в качестве шаблона проекта указать Windows Forms Application (Рисунок 1).

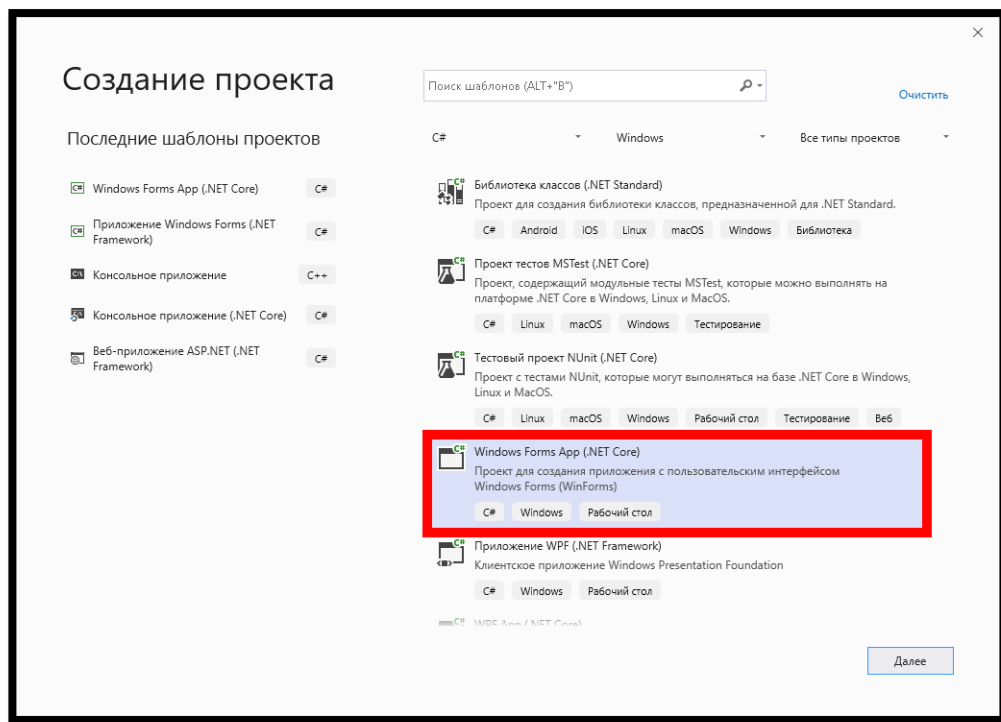


Рисунок 1

После этого следует ввести имя проекта, например, PR1, и указать папку для хранения проекта.

После этого Visual Studio откроет наш проект с созданными по умолчанию файлами:

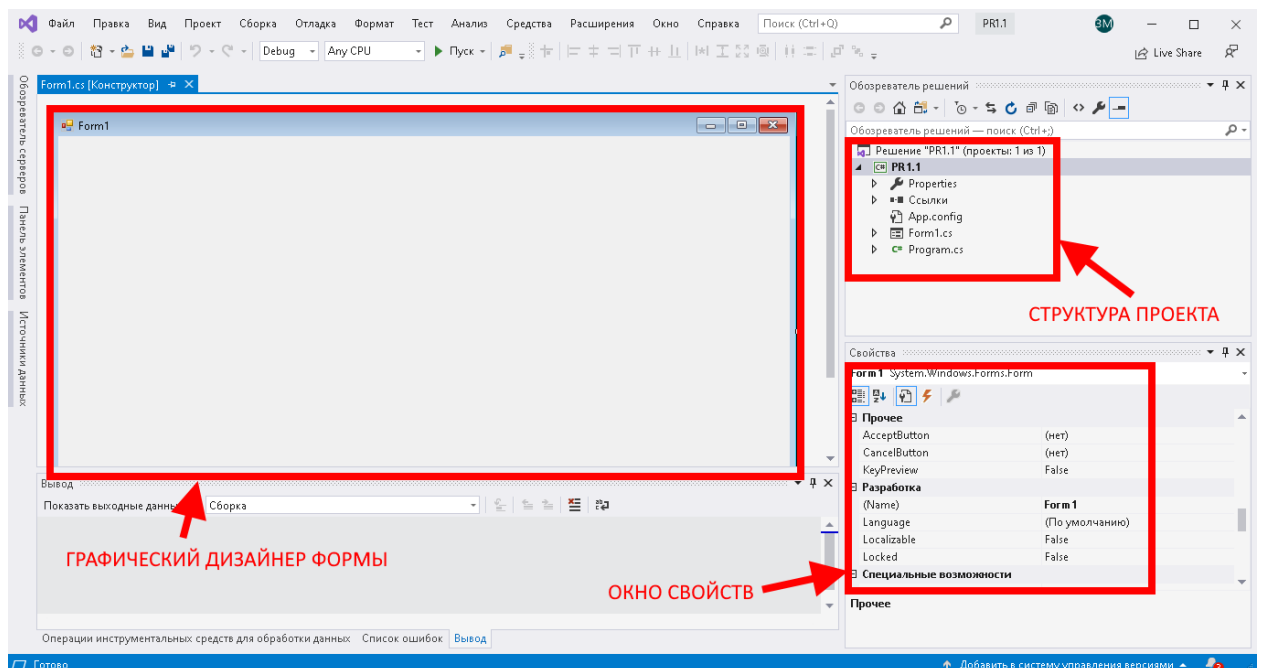


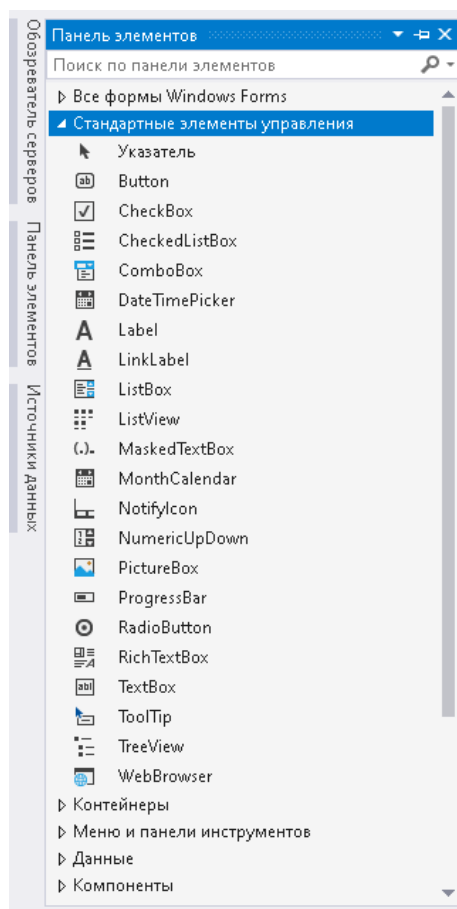
Рисунок 2

Большую часть пространства Visual Studio занимает графический дизайнер, который содержит форму будущего приложения. Пока она пуста и имеет только заголовок Form1. Справа находится окно файлов решения/проекта - Solution Explorer (Обозреватель решений). Там и находятся все связанные с нашим приложением файлы, в том числе файлы формы Form1.cs.

Внизу справа находится окно свойств - Properties. Так как в данный момент выбрана форма как элемент управления, то в этом поле отображаются свойства, связанные с формой.

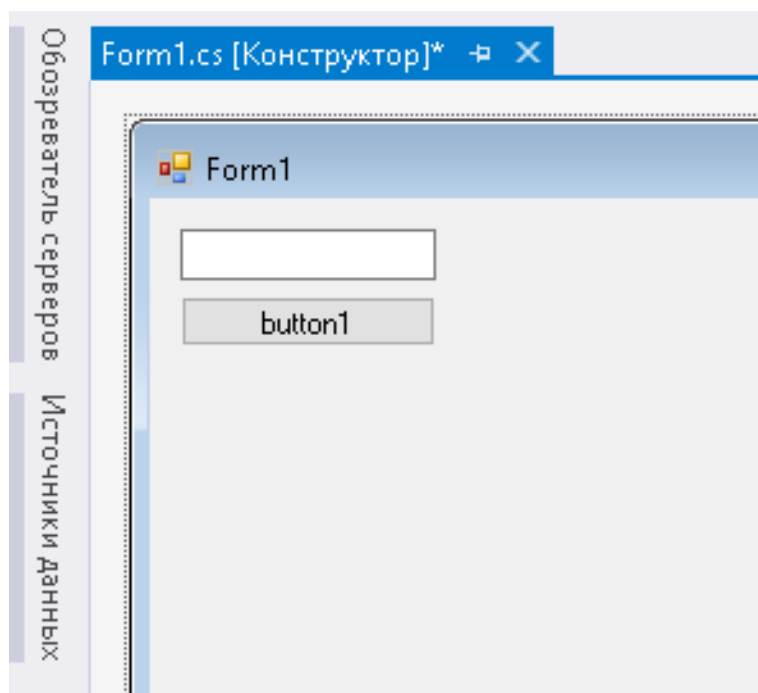
### **Размещения элементов управления на форме**

Для размещения различных элементов управления на форме используется Панель инструментов (Toolbox). Панель элементов содержит элементы управления, сгруппированные по типу. Каждую группу элементов можно свернуть, если она в настоящий момент не нужна. (Рисунок 3)



*Рисунок 3*

Чтобы добавить элементы управления на форму, необходимо щелкнуть на добавляемый элемент в панели инструментов, а затем щелкнуть в нужном месте формы. После этого элемент появится на форме (Рисунок 4).



*Рисунок 4 – Форма с добавленными элементами управления (TextBox и Button)*

Элемент можно перемещать по форме, схватившись за него левой кнопкой мыши. Если элемент управления позволяет изменять размеры, то на соответствующих его сторонах появятся белые кружки, ухватившись за которые и можно изменить размер. При размещении элемента управления на форме, его можно выделить щелчком мыши и получить доступ к его свойствам в окне свойств.

### **Размещение строки ввода**

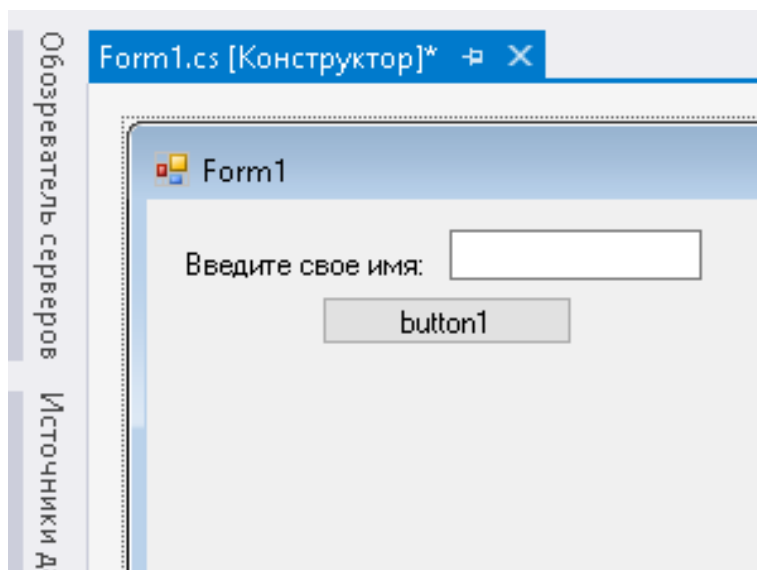
Если необходимо ввести из формы в программу или вывести в форму информацию, которая вмещается в **одну** строку, используют окно однострочного редактора текста, представляемого элементом управления TextBox.

Создадим поле для ввода имени пользователя. Выберем на панели инструментов пиктограмму с названием «TextBox», щелкните мышью в том месте формы, где хотите ее разметить. После размещения элемента можно в

тексте программы использовать переменную `textBox1`, которая соответствует добавленному элементу управления. В этой переменной, в свойстве `Text` будет содержаться строка символов (типа `string`) и отображаться в соответствующем окне `TextBox`. С помощью окна свойств можно установить шрифт и размер символов, отражаемых в строке `TextBox` (свойство `Font`).

### **Размещение надписей**

На форме могут размещаться пояснительные надписи. Для размещения таких надписей на форму используется элемент `Label`. Выберите на панели инструментов пиктограмму с названием `Label`, щелкните на ней мышью. После этого в нужном месте формы щелкните мышью, появится надпись `label1`. Щелкнув на ней мышью, можно отрегулировать размер и, изменив свойство `Text` в окне свойств, введите строку, например, «Введите свое имя:», а также выберите размер символов (свойство `Font`). Добавленный элемент представлен в форме на рисунке 5.



*Рисунок 5 – Форма с добавленным элементом `Label`*

В тексте программы можно обращаться к новой переменной типа `Label`. В ней хранится пояснительная строка, которую можно изменять в процессе работы программы.

### **Написание программы обработки события**

С каждым элементом управления на форме и с самой формой могут происходить события во время работы программы. Например, с кнопкой

может произойти событие – нажатие кнопки, а с окном, которое проектируется с помощью формы, может произойти ряд событий: создание окна, изменение размера окна, щелчок мыши на окне и т.п. Эти события могут обрабатываться программно. Для обработки таких событий используются специальные методы – обработчики событий. Создать такие методы можно двумя способами.

Первый способ – создать обработчик для события по умолчанию. Например, при нажатии кнопки на форме таким образом создается обработчик события нажатия.

### **Написание программы обработки события нажатия кнопки**

Поместите на форму кнопку, которая описывается элементом управления Button. С помощью окна свойств измените заголовок (Text) на слово «Привет» или другое по вашему желанию. Отрегулируйте положение и размер кнопки.

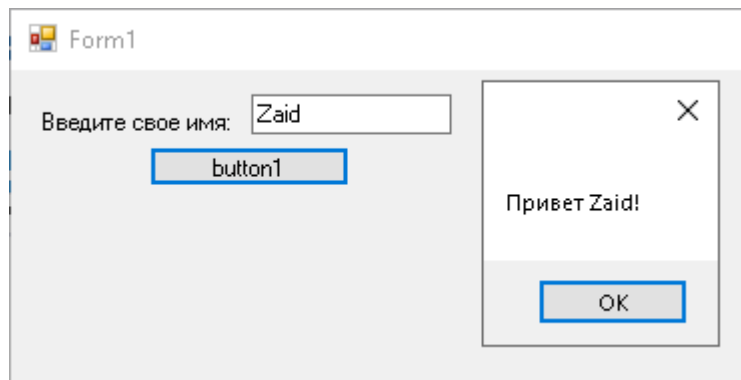
После этого два раза щелкните мышью на кнопке, появится текст программы:

```
private void button1_Click(object sender, EventArgs e)
{
}
}
```

Этот метод и будет являться обработчиком события нажатия кнопки. Вы можете добавлять свой код между скобками {...}. Например, наберите:


```
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show("Привет " + textBox1.Text + "!");
}
```

При нажатии на кнопку выведется окно с сообщением «Привет», которое соединено с текстом, записанным пользователем в textBox1 (Рисунок 6).



*Рисунок 6 – Окно с сообщением, появившееся при нажатии кнопки button1*

### **Написание программы обработки события загрузки формы**

Второй способ создания обработчика события заключается в выборе соответствующего события для выделенного элемента на форме. При этом используется окно свойств и его закладка . Рассмотрим этот способ. Выделите форму щелчком по ней, чтобы вокруг нее появилась рамка из точек. В окне свойств найдите событие Load. Щелкните по данной строчке дважды мышкой. Появится метод:

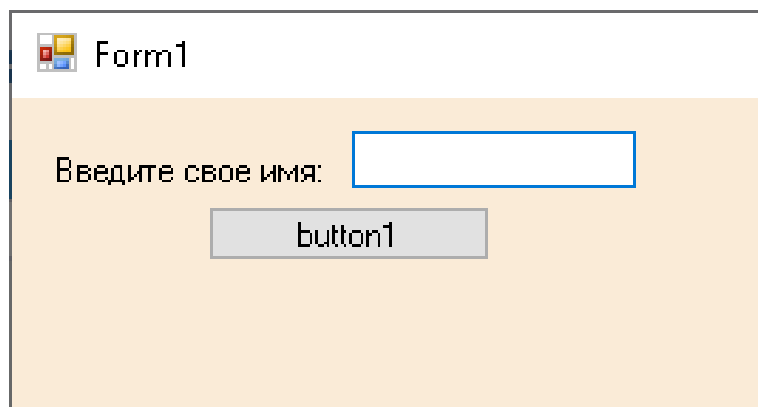
```
private void Form1_Load(object sender, EventArgs e)
{

}
```

Между скобками {...} вставим текст программы:

```
private void Form1_Load(object sender, EventArgs e)
{
    BackColor = Color.AntiqueWhite;
}
```

Свойство `BackColor` позволяет изменить цвет фона. Результат выполнения данного метода представлен на рисунке 7.



*Рисунок 7 – Форма с измененным фоном*

Каждый элемент управления имеет свой набор обработчиков событий, однако некоторые из них присущи большинству элементов управления. Наиболее часто применяемые события описаны ниже:

1) Load – событие, возникающее при загрузке формы. В обработчике данного события следует задавать действия, которые должны происходить в момент создания формы, например, установка начальных значений.

2) KeyPress – событие, возникающее при нажатии кнопки на клавиатуре. Параметр `e.KeyChar` имеет тип `char` и содержит код нажатой клавиши (например, клавиша Enter клавиатуры имеет код #13, клавиша Esc - #27 и т.д.). Обычно это событие используется в том случае, когда необходима реакция на нажатие одной из клавиш.

3) KeyDown – событие, возникающее при нажатии кнопки на клавиатуре. Обработчик этого события получает информацию о нажатой клавише и состоянии клавиш Shift, Alt и Ctrl, а также о нажатой кнопке мыши. Информация о клавише передается параметром `e.KeyCode`, который представляет собой перечисление `Keys` с кодами всех клавиш, а информацию о клавишах-модификаторах Shift и др. можно узнать из параметра `e.Modifiers`.

4) KeyUp – является парным событием для KeyDown и возникает при отпускании ранее нажатой клавиши.

5) Click – возникает при нажатии кнопкой мыши в области элемента управления.



6) DoubleClick – возникает при двойном нажатии кнопки мыши в области элемента управления.

**Важное примечание** - если какой-то обработчик был добавлен по ошибке или больше не нужен, то для его удаления нельзя просто удалить программный код обработчика. Сначала нужно удалить строку с именем обработчика в окне свойств в закладке с событиями. В противном случае программа может перестать компилироваться и даже отображать форму в дизайнера VS.

### **Динамическое изменение свойств**

Свойства элементов на окне могут быть изменены динамически во время выполнения программы. Например, можно изменять текст надписи или цвет формы. Изменение свойств происходит внутри обработчика события (например, обработчика события нажатия на кнопку). Для этого используют оператор присвоения вида:

`<имя элемента>.<свойство> = <значение>;`

Например,

`label1.Text = "Hello";`

`<имя элемента>` определяется на этапе проектирования формы, при размещении элемента управления на форме. Например, при размещении на форме ряда элементов TextBox, эти элемента получают имена `textBox1`, `textBox2`, `textBox3` и т.д., которые могут быть заменены в окне свойств в свойстве (Name) для текущего элемента. Список свойств для конкретного элемента можно посмотреть в окне свойств.

Если требуется изменить свойства формы, то никакое имя элемента перед точкой вставлять не нужно, как и саму точку. Например, чтобы задать цвет формы, нужно просто написать:

`BackColor = Color.Green;`

### **Ввод и вывод данных в программу**

Рассмотрим один из способов ввода данных через элементы, размещенные на форме. Для ввода данных чаще всего используют элемент

управления TextBox, через обращение к его свойству Text. Свойство Text хранит в себе строку введенных символов. Поэтому данные можно считать следующим образом:

```
private void button1_Click(object sender, EventArgs e)
{
    string s = textBox1.Text;
}
```

Однако со строкой символов трудно производить арифметические операции, поэтому лучше всего при вводе числовых данных перевести строку в целое или вещественное число. Для этого у типов int и double существуют методы Parse для преобразования строк в числа. С этими числами можно производить различные арифметические действия. Таким образом, предыдущий пример можно переделать следующим образом:

Перед выводом числовые данные следует преобразовать назад в строку. Для этого у каждой переменной существует метод ToString(), который возвращает в результате строку с символьным представлением значения. Вывод данных можно осуществлять в элементы TextBox или Label, используя свойство Text. Например,

```
private void button1_Click(object sender, EventArgs e)
{
    string s = textBox1.Text;
    int a = int.Parse(s);
    int b = a * a;
    label1.Text = b.ToString();
}
```

### **Кнопки-переключатели**

При создании программ в Visual Studio для организации разветвлений часто используются элементы управления в виде кнопок-переключателей. Кнопки-переключатели представлены в панели инструментов как RadioButton. Состояние такой кнопки (включено-выключено) визуально отражается на форме, а в программе можно узнать его помощью свойства Checked: если кнопка включена, это свойство будет содержать True, в противном случае

False. Если пользователь выбирает один из вариантов переключателя в группе, все остальные автоматически отключаются.

Группируются радиокнопки с помощью какого-либо контейнера – часто это бывает элемент `GroupBox`. Радиокнопки, размещенные в различных контейнерах, образуют независимые группы.

Контейнер с кнопками переключателями представлен на рисунке 8.

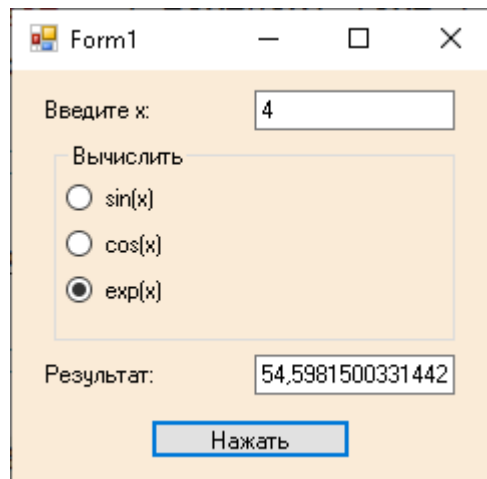


Рисунок 8. Группа кнопок-переключателей

Взаимодействие с кнопками-переключателями описывает следующий код:

```
if (radioButton1.Checked)
{
    textBox1.Text =
Convert.ToString(Math.Sin(Convert.ToDouble(textBox2.Text)));
}
else if (radioButton2.Checked)
{
    textBox1.Text =
Convert.ToString(Math.Cos(Convert.ToDouble(textBox2.Text)));
}
else if (radioButton3.Checked)
{
```

```

        textBox1.Text =
Convert.ToString(Math.Exp(Convert.ToDouble(textBox2.Text)));
    }

```

### Флажок для множественного выбора

Элемент CheckBox или флажок предназначен для установки одного из двух значений: отмечен или не отмечен. Чтобы отметить флажок, надо установить у его свойства Checked значение true.

Кроме свойства Checked у элемента CheckBox имеется свойство CheckState, которое позволяет задать для флажка одно из трех состояний - Checked (отмечен), Indeterminate (флажок не определен - отмечен, но находится в неактивном состоянии) и Unchecked (не отмечен).

Также следует отметить свойство AutoCheck - если оно имеет значение false, то мы не можем изменять состояние флажка. По умолчанию оно имеет значение true. Форма с элементами-флажками представлена на рисунке 9.

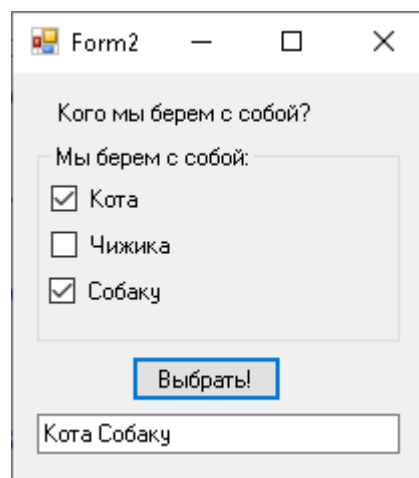


Рисунок 9. Форма с CheckBox

Взаимодействие с CheckBox позволяет описать следующий код:

```

private void button1_Click(object sender, EventArgs e)
//кнопка "Выбрать"
{
    textBox1.Text = "";
    switch(checkBox1.Checked)
    {
        case true:
        {
            textBox1.Text += checkBox1.Text + " ";

```

```

        break;
    }
    case false:
    {
        break;
    }
}
switch(checkBox2.Checked)
{
    case true:
    {
        textBox1.Text += checkBox2.Text + " ";
        break;
    }
    case false:
    {
        break;
    }
}
switch(checkBox3.Checked)
{
    case true:
    {
        textBox1.Text += checkBox3.Text + " ";
        break;
    }
    case false:
    {
        break;
    }
}
}

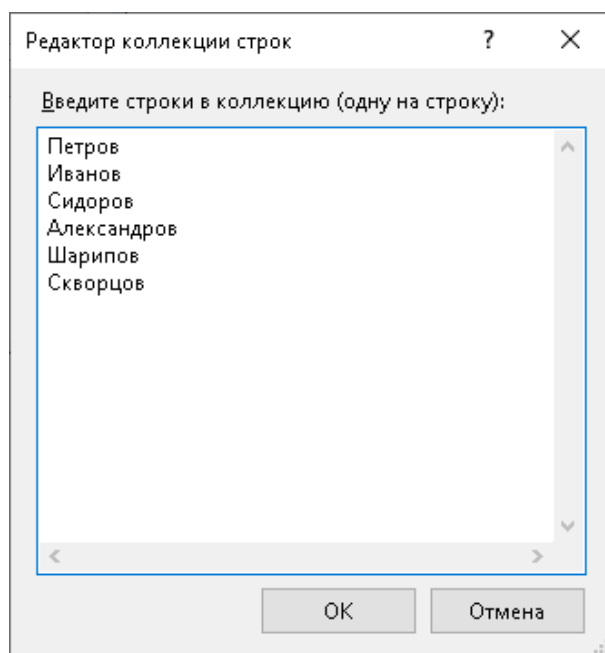
```

### Элемент **ListBox**

Элемент **ListBox** представляет собой простой список. Ключевым свойством этого элемента является свойство **Items**, которое как раз и хранит набор всех элементов списка.

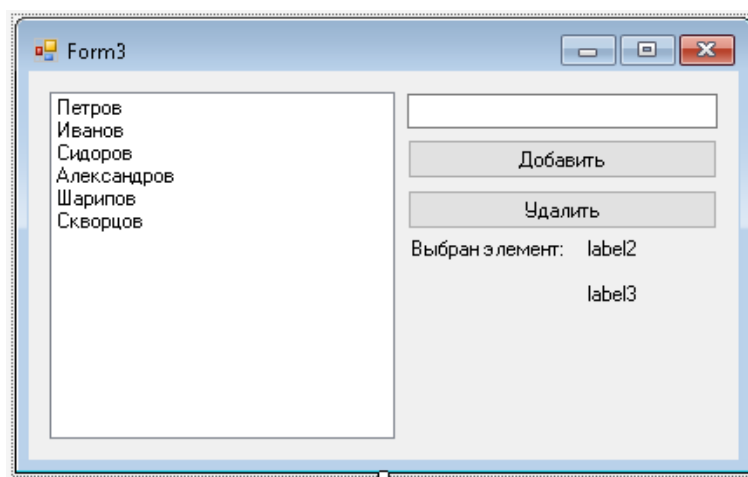
Элементы в список могут добавляться как во время разработки, так и программным способом. В Visual Studio в окне **Properties** (Свойства) для элемента **ListBox** мы можем найти свойство **Items**. После двойного щелчка на

свойство нам отобразится окно для добавления элементов в список (Рисунок 10).



*Рисунок 10 – Редактор коллекции строк в ListBox*

В пустое поле мы вводим по одному элементу списка - по одному на каждой строке. После этого все добавленные нами элементы окажутся в списке, и мы сможем ими управлять. Создадим форму (Рисунок 11) для управления элементами формы. В качестве функционала добавим возможность добавления элемента в конец списка, удаление и отображение выделенного элемента.



*Рисунок 11 – Макет формы для управления содержимым списка*

Итак, все элементы списка входят в свойство Items, которое представляет собой коллекцию. Для добавления нового элемента в эту

коллекцию, а значит и в список, надо использовать метод `Add`, например:  
`listBox1.Items.Add("Новый элемент");`. При использовании этого метода каждый добавляемый элемент добавляется в конец списка.

Можно добавить сразу несколько элементов, например, массив. Для этого используется метод `AddRange`:

```
string[] students = {"Кондратьев", "Дормидонтова", "Валеева"};  
listBox1.Items.AddRange(students);
```

В отличие от простого добавления вставка производится по определенному индексу списка с помощью метода `Insert`:

```
listBox1.Items.Insert(listBox1.Items.Count, textBox1.Text);
```

Свойство `listBox1.Items.Count` определяет текущее количество элементов в списке. В вышеописанной строчке кода вставка элемента будет производиться постоянно в конец списка.

Для удаления элемента по его тексту используется метод `Remove`:

```
listBox1.Items.Remove("Петров");
```

Чтобы удалить элемент по его индексу в списке, используется метод `RemoveAt`:

```
listBox1.Items.RemoveAt(1);
```

В качестве параметра передается индекс удаляемого элемента.

Кроме того, можно очистить сразу весь список, применив метод `Clear`:

```
listBox1.Items.Clear();
```

Используя индекс элемента, можно получить доступ к самому элементу в списке. Например, получим первый элемент списка:

```
string firstElement = listBox1.Items[0];
```

При выделении элементов списка мы можем ими управлять как через индекс, так и через сам выделенный элемент. Получить выделенные элементы можно с помощью следующих свойств элемента `ListBox`:

- `SelectedIndex`: возвращает или устанавливает номер выделенного элемента списка. Если выделенные элементы отсутствуют, тогда свойство имеет значение `-1`

- **SelectedIndices**: возвращает или устанавливает коллекцию выделенных элементов в виде набора их индексов
- **SelectedItem**: возвращает или устанавливает текст выделенного элемента
- **SelectedItems**: возвращает или устанавливает выделенные элементы в виде коллекции

По умолчанию список поддерживает выделение одного элемента. Чтобы добавить возможность выделения нескольких элементов, надо установить у его свойства **SelectionMode** значение **MultiSimple**.

Чтобы выделить элемент программно, надо применить метод **SetSelected(int index, bool value)**, где **index** - номер выделенного элемента. Если второй параметр - **value** имеет значение **true**, то элемент по указанному индексу выделяется, если **false**, то выделение наоборот скрывается:

```
listBox1.SetSelected(2, true); // будет выделен третий
                               элемент
```

Чтобы снять выделение со всех выделенных элементов, используется метод **ClearSelected**.

Из всех событий элемента **ListBox** надо отметить в первую очередь событие **SelectedIndexChanged**, которое возникает при изменении выделенного элемента. Рассмотрим реализацию обработчика события на примере нашего задания.

```
private void listBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    label2.Text = Convert.ToString(listBox1.SelectedItem);
    label3.Text = Convert.ToString(listBox1.SelectedIndex);
}
```

Обработчик события изменяет текст в надписях. В **label2** выводится текст выбранного пользователем элемента, а в **label3** – его индекс.

При нажатии на кнопку «Добавить» будет происходить добавление нового элемента в конец списка. Конец элемента будет определять индекс, равный **listBox1.Items.Count** – текущему количеству элементов в списке.



Текстовое наполнение нового элемента будет извлекаться из данных, введенных в `textBox1`.

```
private void button1_Click(object sender, EventArgs e)
{
    listBox1.Items.Insert(listBox1.Items.Count,
textBox1.Text);
}
```

Кнопка «Удалить» будет производить удаление выбранного в списке элемента. `listBox1.SelectedIndex` будет возвращать индекс выбранного пользователем элемента.

```
private void button2_Click(object sender, EventArgs e)
{
    listBox1.Items.RemoveAt(listBox1.SelectedIndex);
}
```

### Пример написания программы

Выполним следующее задание, разработав для него интерфейс с помощью Windows Forms.

**Задание** – Поле шахматной доски определяется парой натуральных чисел, каждое из которых не превосходит 8: первое число — номер вертикали (при счете слева направо), второе—номер горизонтали (при счете снизу-вверх). Даны натуральные числа *a*, *b*, *c*, *d*, *e*, *f*, каждое из которых не превосходит 8. Определить, может ли белая ладья, расположенная на поле (*a*, *b*), одним ходом пойти на поле (*e*, *f*), не попав при этом под удар черной ладьи, находящейся на поле (*c*, *d*).

**Решение** – Упростим решение задачи. Поля шахматного поля будем представлять в виде флажков `CheckBox` и будем создавать их динамически, то есть явно через программный код. Финальный макет разрабатываемого приложения показан на рисунке 12.



Рисунок 12

Промежуточный макет разрабатываемого приложения, который необходимо собрать вручную, представлен на рисунке 13.



Рисунок 13

Label будет хранить информацию для взаимодействия с пользователем: «Выберите позицию первой ладьи», «Выберите позицию первой ладьи» и «Выберите новую позицию первой ладьи». Контейнер GroupBox будет хранить флажки checkBox, каждый из которых будет представлять ячейку в шахматном поле. Кнопка «Начать заново» перезапустит ход игры, вернув в исходное положение все checkBox.

В разрабатываемой программе можно выделить три сущности, которые можно представить в виде классов:

1. Класс Rook (Ладья) – будет описывать объект «Ладья» со свойствами coord\_x (Позиция ладьи по горизонтали) и coord\_y (Позиция ладьи по вертикали). Конструктор класса будет инициализировать новые координаты ладьи в шахматном поле при ее создании. Создание будет происходить при выборе соответствующего флажка на игровом поле.

1	<code>public class Rook</code>
2	<code>{</code>
3	<code>    public int coord_x { get; set; }</code>
4	<code>    public int coord_y { get; set; }</code>
5	<code>    public Rook(int x, int y)</code>
6	<code>    {</code>
7	<code>        this.coord_x = x;</code>
8	<code>        this.coord_y = y;</code>
9	<code>    }</code>
10	<code>}</code>

2. Класс NewCheckBox – данный класс будет наследоваться от стандартного класса CheckBox и будет определять для него дополнительные свойства – это координаты (по горизонтали и вертикали) checkBox в шахматном поле. Координаты будут инициализироваться при создании элементов шахматного поля.

1	<code>public class NewCheckBox : CheckBox</code>
2	<code>{</code>
3	<code>    public int Coord_x { get; set; }</code>
4	<code>    public int Coord_y { get; set; }</code>
5	<code>    public NewCheckBox(int x, int y)</code>
6	<code>    {</code>
7	<code>        this.Coord_x = x;</code>

8	<code>this.Coord_y = y;</code>
9	<code>}</code>
10	<code>}</code>

3. Класс **ChessField** (Шахматное поле) – данный класс представляет описание объекта «Шахматное поле» и будет хранить следующие свойства: это массив флажков **checkBox** размерностью 8\*8 для дальнейшего их добавления в форму и переменная **state**, отвечающая за этап решения задачи (0 – выбор позиции первой ладьи, 1 – выбор позиции второй ладьи, 2 – выбор новой позиции первой ладьи).

Конструктор класса будет отвечать за динамическое создание элементов **checkBox** в форме.

1	<code>public class ChessField</code>
2	<code>{</code>
3	<code>    public NewCheckBox[,] checkBoxes;</code>
4	<code>    public int state;</code>
5	<code>    public ChessField(int x, int y)</code>
6	<code>    {</code>
7	<code>        state = 0;</code>
8	<code>        checkBoxes = new NewCheckBox[x, y];</code>
9	<code>        for (int i = 0; i &lt; 8; i++)</code>
10	<code>        {</code>
11	<code>            for (int j = 0; j &lt; 8; j++)</code>
12	<code>            {</code>
13	<code>                checkBoxes[i, j] = new NewCheckBox(i,</code>
14	<code>                j)</code>
15	<code>                {</code>
16	<code>                    Checked = false,</code>
17	<code>                    Location = new Point(i * 30 + 20, j</code>
18	<code>                    * 30 + 20),</code>
19	<code>                    Size = new Size(20, 20),</code>
20	<code>                    Name = "cell" + Convert.ToString(i</code>
21	<code>                    + 1) + Convert.ToString(j + 1),</code>
22	<code>                    Visible = true</code>
23	<code>                };</code>
24	<code>            }</code>
25	<code>        }</code>
26	<code>    }</code>

Конструктор в качестве параметров принимает два целочисленных аргумента (строка 5), которые отвечают за количество элементов в каждом измерении матрицы `checkboxes` при ее создании. Свойство `state` инициализируется нулем (строка 7), сигнализирующем о начальном состоянии программы (расстановка на шахматное поле первой ладьи).

В двойном цикле происходит обработка двумерного массива `checkboxes`. Во вложенном цикле динамически создается элемент `checkboxes` как экземпляр класса `NewCheckBox` (строка 13). Далее происходит инициализация свойств экземпляра класса `NewCheckBox` (строка 15- 19). Свойство `Checked` (строка 15) задает неотмеченное состояние флажка. Свойство `Location` (строка 16), которое создается динамически через вызов конструктора класса `Point`, определяет новое расположение элемента относительно верхнего левого края формы. Как аргументы конструктора задаются координаты по горизонтали и координаты по вертикали. Свойство `Size` (строка 17) задает размерность элемента по ширине и высоте. Свойство `Name` (строка 18) изменяет значение названия элемента, по которому в дальнейшем можно к нему обращаться. Свойство `Visible` (строка 19) изменяет видимость и невидимость элемента (если `True`, то элемент видимый).

После создания необходимых для решения задачи классов можно приступить за создание функциональности генерируемых событий в форме. При загрузке формы необходимо проинициализировать текстовое наполнение требуемых элементов формы и добавить в контейнер `groupBox1` все элементы, сгенерированные в классе `ChessField`. Обработчик события будет выглядеть следующим образом:

1	<code>private void Chess_Load(object sender, EventArgs e)</code>
2	<code>{</code>
3	<code>label1.Text = "Выберите позицию первой ладьи!";</code>
4	<code>groupBox1.Text = "Шахматное поле";</code>
5	<code>chessField = new ChessField(8, 8);</code>

6	<code>foreach (NewCheckBox i in</code>
7	<code>chessField.checkBoxes)</code>
8	<code>{</code>
9	<code>groupBox1.Controls.Add(i);</code>
10	<code>i.CheckedChanged += new</code>
11	<code>System.EventHandler(this.checkBox_CheckedChanged);</code>
	<code>}</code>
	<code>}</code>

В строке 5 происходит создание экземпляра класса `ChessField`, создается «Шахматное поле» размерностью 8\*8. Далее, используя цикл `foreach`, организуем обработку элементов двойного массива `checkboxes`, который представляет собой свойство экземпляра класса `ChessField`, где каждый элемент массива добавляем в контейнер `groupBox1` (строка 8). В строке 9 каждому элементу массива привязывается для его события `CheckedChanged` обработчик, представляющий собой метод `checkBox_CheckedChanged`. Название метода передается как аргумент при вызове конструктора обработчика события `System.EventHandler`.

Самое интересное кроется в методе обработчике события нажатия на флажок. При изменении состояния флажка он генерирует событие `CheckedChanged`. Обработывая это событие, мы можем получать измененный флажок и производить определенные действия.

1	<code>private void checkBox_CheckedChanged(object sender, EventArgs e)</code>
2	<code>{</code>
3	<code>NewCheckBox checkBox = (NewCheckBox)sender; // приводим отправителя к элементу типа CheckBox</code>

Здесь мы параметр `sender`, представляющий собой элемент отправитель события (флажок), преобразуем в пользовательский тип `NewCheckBox` и сохраняем в локальную переменную `checkbox`. Проще говоря, в переменную `checkbox` сохраняется тот элемент-флажок, который выбрал пользователь.

После инициализации переменной идет проверка состояния `Checked` флажка (выбран/не выбран).

4	<code>if (checkBox.Checked == true)</code>
---	--------------------------------------------

5	{
---	---

Далее идет проверка значения свойства `chessField.state`. `state = 0` – это выбор позиции первой ладьи в шахматном поле, `state = 1` – это выбор позиции второй ладьи в шахматном поле, `state = 2` – это выбор новой позиции для первой ладьи.

6	<code>if (chessField.state == 0)</code>
7	<code>{</code>
8	<code>    checkBox.AutoCheck = false;</code>
9	<code>    checkBox.BackColor = Color.Aqua;</code>
10	<code>    rookFirst = new Rook(checkBox.Coord_x,</code> <code>    checkBox.Coord_y);</code>
11	<code>    chessField.state = 1;</code>
12	<code>    label1.Text = "Выберите позицию второй ладьи!";</code>
13	<code>}</code>
14	<code>else if (chessField.state == 1)</code>
15	<code>{</code>
16	<code>    checkBox.AutoCheck = false;</code>
17	<code>    checkBox.BackColor = Color.Blue;</code>
18	<code>    rookSecond = new Rook(checkBox.Coord_x,</code> <code>    checkBox.Coord_y);</code>
19	<code>    chessField.state = 2;</code>
20	<code>    label1.Text = "Выберите новую позицию первой</code> <code>    ладьи!";</code>
21	<code>}</code>
22	<code>else if (chessField.state == 2)</code>
23	<code>{</code>
24	<code>    if (!(rookFirst.coord_x == checkBox.Coord_x   </code> <code>    rookFirst.coord_y == checkBox.Coord_y))</code>
25	<code>    {</code>
26	<code>        checkBox.Checked = false;</code>
27	<code>        MessageBox.Show("Ладья так ходить не</code> <code>        может!");</code>
28	<code>    }</code>
29	<code>    else</code>
30	<code>    {</code>
31	<code>        rookFirst.coord_x = checkBox.Coord_x;</code>
32	<code>        rookFirst.coord_y = checkBox.Coord_y;</code>
33	<code>        checkBox.BackColor = Color.Red;</code>
34	<code>        if (rookFirst.coord_x == rookSecond.coord_x   </code> <code>        rookFirst.coord_y == rookSecond.coord_y)</code>
35	<code>        {</code>

36	<code>MessageBox.Show("Ладья под ударом!");</code>
37	<code>}</code>
38	<code>else</code>
39	<code>{</code>
40	<code>MessageBox.Show("Ладья в безопасности!");</code>
41	<code>}</code>
42	<code>checkBox.AutoCheck = false;</code>
43	<code>chessField.state = 0;</code>
44	<code>label1.Text = "";</code>
45	<code>}</code>
46	<code>}</code>

Свойство флажка `AutoCheck` равное `false` блокирует изменение состояния флажка. По умолчанию данное свойство имеет значение `true`.

Свойство `BackColor` позволяет изменить цветовую заливку элемента управления. Объект `Color` представляет собой цвет фона элемента управления.

При выборе позиции для первой и второй ладьи происходит динамическое создание двух экземпляров класса `Rook` с передачей в конструктор класса в качестве аргументов координаты по горизонтали и вертикали элементов управления, символизирующих ячейку в шахматном поле.

При достижении `state` значения 2 идет проверка поставленных в условии задачи ограничений нового расположения первой ладьи.

Последний метод, который необходимо добавить – это обработчик события нажатия кнопки «Нажать заново». При ее нажатии мы изменяем стартовое сообщение, обнуляем значение поля `state`, и с помощью цикла обходим все хранящиеся в массиве элементы-флажки и изменяем их свойства: цветовую заливку `BackColor` устанавливаем на значение `SystemColors.Control` (цвет по умолчанию), устанавливаем `AutoCheck` на значение `true`, чтобы разрешить изменение состояния всех флажков и устанавливаем `Checked` на значение `false`, чтобы изменить состояние всех флажков в контейнере на «Не отмеченный».



1	<code>private void button1_Click(object sender, EventArgs e)</code>
2	<code>{</code>
3	<code>label1.Text = "Выберите позицию первой ладьи!";</code>
4	<code>chessField.state = 0;</code>
5	<code>foreach (NewCheckBox i in</code> <code>chessField.checkBoxes)</code>
6	<code>{</code>
7	<code>i.BackColor = SystemColors.Control;</code>
8	<code>i.AutoCheck = true;</code>
9	<code>i.Checked = false;</code>
10	<code>}</code>
11	<code>}</code>

Результат работы приложения представлен на рисунке 14.

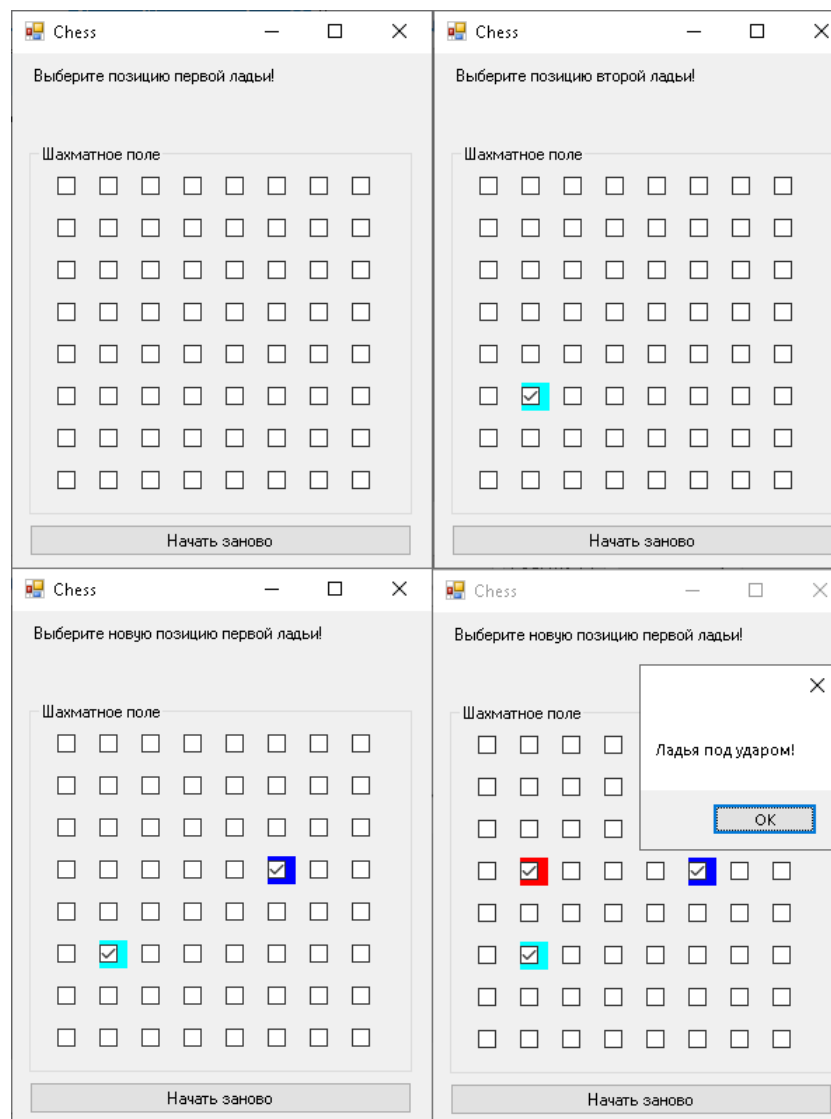


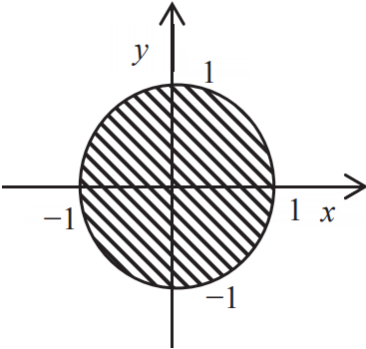
Рисунок 14

### ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ

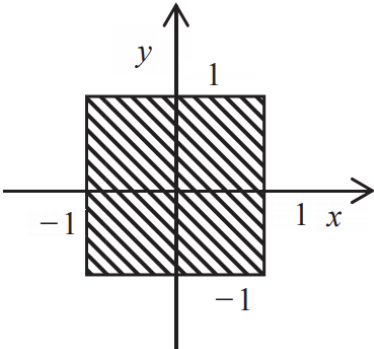
Выбрать пять заданий из приложения 1 и решить их, разработав графический интерфейс с использованием технологии Windows Forms.

**Вариант заданий необходимо согласовать у преподавателя перед непосредственной разработкой!**

## ПРИЛОЖЕНИЕ 1

№	Индивидуальное задание
1	Ввести целое число N. Выяснить, кратно ли оно трем.
2	По заданным x, y составить программу вычисления значения z: $z = \frac{\min(x, y) + 0,5}{1 + \max^2(x, y)}$
3	По заданным x, y составить программу вычисления значения z: $z = \begin{cases} \max(x, y), & \text{при } x < 0 \\ \min(x, y), & \text{при } x \geq 0 \end{cases}$
4	Даны действительные числа x, y. определить, принадлежит ли точка с координатами (x, y) заштрихованной части плоскости. 
5	Даны действительные числа x, y, z. Получить: $L = 2 * \max(x, z) - 3 * \min(x, y, z)$
6	По заданным вещественным числам a, b, c вычислить: $P = \frac{\max(a, b, c) - \min(a, b, c)}{2}$
7	Ввести два целых числа. Выяснить, являются ли они оба четными или нечетными, либо одно четное, а другое нечетное.
8	Даны действительные числа a, b, c. проверить, выполняются ли неравенства $a < b < c$ .
9	Даны действительные числа a, b, c. удвоить эти числа, если $a \geq b \geq c$ , и заменить их абсолютными значениями (значение по модулю), если это не так.
10	Даны два действительных числа. заменить первое число

№	Индивидуальное задание
	нулем, если оно меньше второго или равно ему, и оставить числа без изменения в противном случае.
11	Даны действительные положительные числа $x, y, z$ . Выяснить, существует ли треугольник с длинами $x, y, z$ .
12	Дано действительное число $a$ . Вычислить $f(a)$ , если $f(x) = \begin{cases} x^2 + 4x + 5, & \text{при } x \leq 2 \\ \frac{1}{x^2 + 4x + 5}, & \text{в противном случае} \end{cases}$
13	Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 10% предоставляется, если сумма покупки больше 1000 руб.
14	Написать программу определения стоимости разговора по телефону с учетом скидки 20%, предоставляемой по воскресеньям. У пользователя должен запрашиваться день недели, в который производится разговор.
15	Даны три действительные числа. Если все числа положительны, найти среднее арифметическое, иначе произведение.
16	Даны три действительные числа. Возвести в квадрат те из них, значения которых положительны, и в четвертую степень — отрицательные.
17	Даны три действительных числа. выбрать из них те, которые принадлежат полуинтервалу $(3, 7]$ .
18	По заданным $x, y$ составить программу вычисления значения $z$ : $z = \begin{cases} \max(x^2, y^2), & \text{при } y < 0 \\ \min(x, y), & \text{при } y \geq 0 \end{cases}$
19	Даны действительные числа $x, y$ . определить, принадлежит ли точка с координатами $(x, y)$ заштрихованной части плоскости.

№	Индивидуальное задание
	
20	<p>По четырехзначному номеру трамвайного билета определить, является ли он счастливым (билет считается счастливым, если сумма первых двух цифр номера совпадает с суммой двух его последних цифр).</p>
21	<p>Разработать простой тест: на экране по очереди появляются вопросы (вопросы выбираются студентом), с вариантами ответов. В конце работы программа выдает количество заработанных баллов по результатам ответа. Минимальное количество вопросов – пять.</p>
22	<p>Ввести 2 массива строк – по 12 элементов в каждом. Сформировать новый массив, на четных местах которого будут элементы с нечетными индексами из первого массива, а на нечетных, с четными индексами из второго.</p>
23	<p>Ввести целочисленный массив, состоящий из 15 элементов. Вычислить сумму цифр всех чисел, входящих в данный массив.</p>
24	<p>Дана действительная квадратная матрица <math>[a_{ij}]_{i,j=\overline{1,n}}</math>. Размерность матрицы задается пользователем. Получить две квадратные матрицы <math>[b_{ij}]_{i,j=\overline{1,n}}</math> и <math>[c_{ij}]_{i,j=\overline{1,n}}</math>, для элементов которых выполняются условия:</p> $b_{ij} = \begin{cases} a_{ij}, & \text{при } j < i \\ a_{ji}, & \text{при } j \geq i \end{cases}$ $c_{ij} = \begin{cases} a_{ij}, & \text{при } j < i \\ -a_{ij}, & \text{при } j \geq i \end{cases}$

№	Индивидуальное задание
25	Дана действительная матрица размерности $m \times n$ , в которой не все элементы равны нулю. Получить новую матрицу путем деления всех элементов данной матрицы на ее наибольший по модулю элемент.
26	Дана действительная матрица размерности $m \times n$ . Найти сумму наибольшего значений элементов ее строк.
27	В действительной матрице размерности $m \times n$ поменять местами строку, содержащую элемент с наибольшим значением, со строкой, содержащей элемент с наименьшим значением. Предполагается, что эти элементы единственны.
28	Дана действительная матрица размера $m \times n$ , все элементы в которой различны. В каждой строке выбирается элемент с наибольшим значением, затем среди чисел выбирается наименьшее. Указать индексы элемента с найденным значением.
29	Дана действительная квадратная матрица порядка 12. Заменить нулями все элементы, расположенные на главной диагонали, и единицами, расположенные выше главной диагонали.
30	Дана действительная квадратная матрица порядка 11. Получить целочисленную квадратную матрицу того же порядка, в которой элемент равен единице, если соответствующий ему элемент исходной матрицы больше элемента, расположенного в его строке на главной диагонали, и равен нулю в противном случае.
31	Определить, является ли введенная действительная квадратная матрица симметричной относительно своей главной диагонали.
32	Известна зарплата каждого из 12 работников фирмы за каждый месяц первого квартала.

№	Индивидуальное задание																										
	<table><tr><th rowspan="2">Работники</th><th colspan="3">Месяц</th></tr><tr><th>1</th><th>2</th><th>3</th></tr><tr><td>1</td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td></tr><tr><td>...</td><td></td><td></td><td></td></tr><tr><td>12</td><td></td><td></td><td></td></tr></table> <p>Организовать ввод информации по этой таблице и определить:</p> <p>а) общую сумму, выплаченную за квартал всем работникам;</p> <p>б) зарплату, полученную за квартал каждым работником;</p> <p>в) общую зарплату всех работников за каждый месяц.</p>				Работники	Месяц			1	2	3	1				2				...				12			
Работники	Месяц																										
	1	2	3																								
1																											
2																											
...																											
12																											
33	Даны натуральные числа $i, j$ , действительная матрица размера $18 \times 24$ ( $1 \leq i < j \leq 24$ ). Поменять в матрице местами $i$ -й и $j$ -й столбцы.																										
34	Дана действительная квадратная матрица порядка 9. Вычислить сумму тех из ее элементов, расположенных на главной диагонали и выше неё, которые превосходят по величине все элементы, расположенные ниже главной диагонали. Если на главной диагонали и выше неё нет элементов с указанным свойством, то вывести сообщение об этом.																										
35	Будем называть соседями элемента с индексами $i, j$ некоторой матрицы такие элементы этой матрицы, соответствующие индексы которых отличаются от $i$ и $j$ не более чем на единицу. Для данной целочисленной матрицы $[a_{ij}]_{i=\overline{1,n}, j=\overline{1,m}}$ найти матрицу из нулей и единиц $[b_{ij}]_{i=\overline{1,n}, j=\overline{1,m}}$ , элемент которой $b_{ij}$ равен единице, когда все соседи $a_{ij}$ меньше самого $a_{ij}$ .																										
36	Таблица футбольного чемпионата задана квадратной матрицей порядка $n$ , в которой все элементы, принадлежащие главной диагонали, равны нулю, а каждый элемент, не принадлежащий главной диагонали, равен 2, 1 или 0 (числу очков, набранных в игре: 2 – выигрыш, 1 – ничья, 0 – проигрыш). Найти число команд, имеющих																										

№	Индивидуальное задание
	больше побед, чем поражений и определить номера команд, прошедших чемпионат без поражений.
37	<p>Найти решение системы линейных алгебраических уравнений</p> $\begin{cases} a_{11}x_1 = b_1 \\ a_{21}x_1 + a_{22}x_2 = b_2 \\ \dots \\ a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n = b_n \end{cases}$ <p>Если задана действительная матрица <math>[a]_{n \times n}</math> и массив свободных членов <math>[b]_n</math>. Для решения СЛАУ можно использовать метод Крамера.</p>
38	В квадратной матрице $n \times n$ обменять значения элементов в каждой строке, расположенные на главной и побочной диагоналях.
39	Сформировать матрицу из чисел от 0 до 999, вывести ее на экран. Посчитать количество двузначных чисел в ней.
40	Записать элементов прямоугольной матрицы в одномерный массив в порядке следования столбцов.
41	<p>Вводятся пять вещественных чисел. Записать в первый столбец матрицы целую часть чисел, во второй - дробную часть, приведенную к пятизначному целому, в третий столбец - знак числа: 0 для положительных чисел и 1 - для отрицательных.</p> <p>Например, если вводится число 5.452456, то в первой ячейке строки присваивается 5, второй присваивается 45245, а третьей - число 0.</p>
42	Написать программу, вычисляющую объем и площадь поверхности цилиндра по известному радиусу основания и высоте.
43	Написать программу, определяющую максимальный элемент одномерного массива. При вводе/выводе элементов использовать индексы.
44	Написать программу, выполняющую транспонирование неквадратной матрицы. Матрицы должны храниться в памяти в виде двумерного динамического массива, размерности исходной матрицы вводятся



№	Индивидуальное задание
	пользователем с клавиатуры. Предусмотреть генерацию значений матрицы как случайных целых чисел в диапазоне от -10 до 10.
45	Написать программу для определения максимального из трех положительных чисел, введенных пользователем, а также их произведения.
46	Написать программу, определяющую модуль разности между количеством отрицательных и положительных элементов одномерного массива. При вводе/выводе элементов использовать индексы.
47	Написать программу, выполняющую расчет произведения двух неквадратных матриц. Матрицы должны храниться в памяти в виде двумерного динамического массива, размерности матриц вводятся пользователем с клавиатуры. Предусмотреть генерацию значений матриц как случайных вещественных чисел в диапазоне от -0.5 до 0.5.
48	Написать программу, вычисляющую сопротивление электрической цепи, состоящей из двух последовательно соединенных резисторов, а также падения напряжения на каждом из них (сила тока известна).
49	Написать программу, определяющую минимальный элемент в одномерном массиве и увеличивающую его в два раза.
50	Написать программу, выполняющую поворот неквадратной матрицы на 90° по часовой стрелке. Матрицы должны храниться в памяти в виде двумерного динамического массива, размерности исходной матрицы вводятся пользователем с клавиатуры. Предусмотреть генерацию значений матрицы как случайных целых чисел в диапазоне от -20 до 20.
51	Написать программу, вычисляющую значения $a^b$ и $b^a$ .
52	Написать программу, заменяющую положительные элементы массива на их квадраты. При вводе/выводе элементов использовать индексы.

№	Индивидуальное задание
53	Написать программу, выполняющую расчет суммы двух неквадратных матриц. Матрицы должны храниться в памяти в виде двумерного динамического массива, размерности матриц вводятся пользователем с клавиатуры. Предусмотреть генерацию значений матриц как случайных вещественных чисел в диапазоне от -1.0 до 1.0.
54	Написать программу, вычисляющую доход по вкладу и сумму к выдаче. Исходными данными для функции являются: сумма вклада, процентная ставка (годовых) и срок вклада (количество дней).
55	Написать программу, заменяющую все элементы одномерного массива, кроме максимального, на их отрицательные значения. При вводе/выводе элементов использовать индексы.
56	Написать программу, выполняющую поворот неквадратной матрицы на $90^\circ$ против часовой стрелки. Матрицы должны храниться в памяти в виде двумерного динамического массива, размерности исходной матрицы вводятся пользователем с клавиатуры. Предусмотреть генерацию значений матрицы как случайных целых чисел в диапазоне от 0 до 100.
57	Написать программу, вычисляющую периметр и площадь треугольника по известным сторонам.
58	Написать программу, заменяющую в одномерном массиве максимальные элементы на минимальные, а минимальные на максимальные. При вводе/выводе элементов использовать индексы.
59	Написать программу, выполняющую отражение неквадратной матрицы относительно центральной горизонтальной оси. Матрицы должны храниться в памяти в виде двумерного динамического массива, размерности матриц вводятся пользователем с клавиатуры. Предусмотреть генерацию значений матриц как случайных вещественных чисел в диапазоне от -1.0 до 1.0.

№	Индивидуальное задание
60	Написать программу решающую квадратное уравнение.
61	Написать программу, определяющую разность между суммой модулей отрицательных элементов и суммой положительных элементов одномерного массива. При вводе/выводе элементов использовать индексы.
62	Написать программу, выполняющую поворот неквадратной матрицы на $90^\circ$ по часовой стрелке. Матрицы должны храниться в памяти в виде двумерного динамического массива, размерности исходной матрицы вводятся пользователем с клавиатуры. Предусмотреть генерацию значений матрицы как случайных вещественных чисел в диапазоне от 0.0 до 10.0.
63	Написать программу, вычисляющую определитель и произведение диагональных элементов матрицы второго порядка.
64	Написать программу, определяющую максимальный(-ые) элемент одномерного массива, и выводящую его номер. При вводе/выводе элементов использовать индексы.
65	Написать программу, выполняющую транспонирование неквадратной матрицы. Матрицы должны храниться в памяти в виде двумерного динамического массива, размерности матриц вводятся пользователем с клавиатуры. Предусмотреть генерацию значений матриц как случайных вещественных чисел в диапазоне от -2.0 до 2.0.
66	Создать класс EngMer для работы с английскими мерами длины: фунтами и дюймами, при этом учтем, что 1 фунт = 12 дюймов. Длина объекта будет задаваться парой чисел (фунты и дюймы), нужно реализовать: сложение и вычитание длин, умножение и деление длин, сравнение длин.
67	Создать класс vector3D, задаваемый тройкой координат. Обязательно должны быть реализованы: сложение и вычитание векторов, скалярное

№	Индивидуальное задание
	произведение векторов, умножение на скаляр, сравнение векторов, вычисление длины вектора, сравнение длины векторов.
68	Создать класс EngMoney для работы с устаревшей денежной системой Великобритании. В ней использовались фунты, шиллинги и пенсы. При этом: 1 фунт = 20 шиллингов, 1 шиллинг = 12 пенсов. Денежные суммы будут задаваться в фунтах, шиллингах и пенсах и результат выдаваться также в этих величинах. Должны быть реализованы: сложение и вычитание, умножение и деление, сравнение сумм.
69	Создать класс Money для работы с денежными суммами. Число должно быть представлено двумя полями: типа long int для рублей и типа int – для копеек. Дробная часть (копейки) при выводе на экран должна быть отделена от целой части запятой. Реализовать сложение, вычитание, деление сумм, деление суммы на дробное число, умножение на дробное число и операцию сравнения.
70	Создать класс Triangle для представления треугольника. Поля данных должны включать углы и стороны. Требуется реализовать операции: получения и изменения полей данных, вычисления площади, вычисления периметра, вычисления высот, а также определения вида прямо-угольника (равносторонний, равнобедренный или прямоугольный).
71	Создать класс Angle для работы с углами на плоскости, задаваемыми величинами в градусах и минутах. Обязательно должны быть реализованы: перевод в радианы, приведение к диапазону 0–360, увеличение и уменьшение угла на заданную величину, получение синуса, сравнение углов.
72	Создать класс Point для работы с точками на плоскости. координаты точки – декартовы. Обязательно должны быть реализованы:

№	Индивидуальное задание
	перемещение точки по оси X, перемещение по оси Y, определение расстояния до начала координат, расстояние между двумя точками.
73	Рациональная (несократимая) дробь представляется парой целых чисел (a,b), где a—числитель, b—знаменатель. Создать класс Rational для работы с рациональными дробями. Обязательно должны быть реализованы операции: сложения, вычитания, умножения, деления и сравнения дробей, причем результат должен приводиться в виде несократимой дроби (т.е. результат нужно сокращать).
74	Создать класс Date для работы с датами в формате «год.месяц.день». Дата представляется структурой с тремя полями типа unsigned int: для года, месяца и дня. Класс должен включать операции: вычисление даты через заданное количество дней от указанной, вычитание заданного количества дней из даты, определение високосного года, присвоение и получение отдельных частей (год, месяц, день), сравнение дат (равно, до, после), вычисление количества дней между датами.
75	Создать класс Time для работы со временем в формате «час:минута:секунда». Обязательными операциями являются: вычисление разницы между двумя моментами времени в секундах, сложение времени и заданного количества секунд, вычитание из времени заданного количества секунд, сравнение моментов времени, перевод в секунды, перевод в минуты (с округлением до целой минуты)
76	В морской навигации координаты точки измеряются в градусах и минутах широты и долготы. Один градус равен 60 минутам (ранее минуту делили на 60 секунд, но сейчас минуту делят на обычные десятичные доли). Долгота измеряется от 0 до 180° восточнее (E) или западнее (W) Гринвича. Широта принимает значения от 0 до

№	Индивидуальное задание
	<p>90°севернее (N) или южнее (S) экватора. Создать класс Koord, включающий следующие три поля: типа int для числа градусов, типа float для числа минут и типа char для указания направления (N, S, W или E). Объект этого класса может содержать значение как широты, так и долготы. Предусмотреть ввод координат точки, указания полушария для указанной точки, вычисления расстояния между двумя точками. Учитывать, что по широте 1°равен 111 км, а по долготе 1°равен 111 км * cos(широты).</p>
77	<p>Реализовать класс Account, представляющий собой банковский счет. В классе должны быть реализованы 4 поля: фамилия владельца, номер счета, процент начисления и сумма в рублях. Необходимо выполнять следующие операции: сменить владельца счета, снять некоторую сумму со счета, положить деньги на счет, начислить проценты, перевести сумму в доллары, перевести сумму в евро, получить сумму прописью (преобразовать в числительное).</p>
78	<p>Номиналы российских рублей могут принимать значения 1, 2, 5, 10, 50, 100, 500, 1000, 5000. Копейки представить как 0,01 (1 копейка), 0,05 (5 копеек), 0,1 (10 копеек), 0,5 (50 копеек). Создать класс Money для работы с денежными суммами. Сумма должна быть представлена полями-номиналами, значениями которых должны быть количества купюр данного достоинства. Реализовать сложение сумм, вычитание сумм, деление сумм, деление суммы на дробное число, умножение на дробное число и операцию сравнения. Дробная часть (копейки) при выводе на экран должна быть отделена от целой части запятой.</p>
79	<p>Создать класс Goods (товар). В классе должны быть представлены поля: наименование товара, дата оформления, цена товара, количество единиц товара, номер накладной, по которой товар поступил на склад. Реализовать методы изменения цены товара, изменения количества</p>

№	Индивидуальное задание
	товара (увеличения и уменьшения), вычисления стоимости товара. Должен быть метод для отображения стоимости товара в виде строки.
80	<p>Создать класс Payment(зарплата). В классе должны быть представлены поля: фамилия-имя-отчество, оклад, год поступления на работу, процент надбавки, подоходный налог, количество отработанных дней в месяце, количество рабочих дней в месяце, начисленная и удержанная сумма. Реализовать методы: вычисления начисленной суммы, вычисления удержанной суммы, вычисления суммы, выдаваемой на руки, вычисления стажа. Стаж вычисляется как полное количество лет, прошедших от года поступления на работу, до текущего года. Начисления представляют собой сумму, начисленную за отработанные дни, и надбавки. Удержания представляют собой отчисления в пенсионный фонд (1% от начисленной суммы) и подоходный налог (13%).</p>