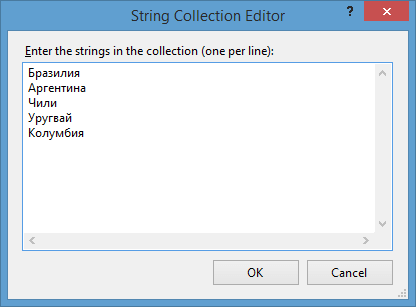
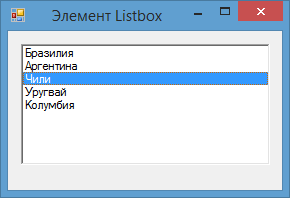
КОМПОНЕНТ *LISTBOX.*

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:



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:



Программное управление элементами в ListBox

Добавление элементов

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:

|  |  |
| --- | --- |
| 1  2 | string[] countries = { "Бразилия", "Аргентина", "Чили", "Уругвай", "Колумбия" };  listBox1.Items.AddRange(countries); |

Вставка элементов

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

|  |  |
| --- | --- |
| 1 | listBox1.Items.Insert(1, "Парагвай"); |

В данном случае вставляем элемент на вторую позицию в списке, так как отсчет позиций начинается с нуля.

Удаление элементов

Для удаления элемента по его тексту используется метод Remove:

|  |  |
| --- | --- |
| 1 | listBox1.Items.Remove("Чили"); |

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

|  |  |
| --- | --- |
| 1 | listBox1.Items.RemoveAt(1); |

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

|  |  |
| --- | --- |
| 1 | listBox1.Items.Clear(); |

Доступ к элементам списка

Используя индекс элемента, можно сам элемент в списке. Например, получим первый элемент списка:

|  |  |
| --- | --- |
| 1 | string firstElement = listBox1.Items[0]; |

Метод Count позволяет определить количество элементов в списке:

|  |  |
| --- | --- |
| 1 | int number = listBox1.Items.Count(); |

Выделение элементов списка

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

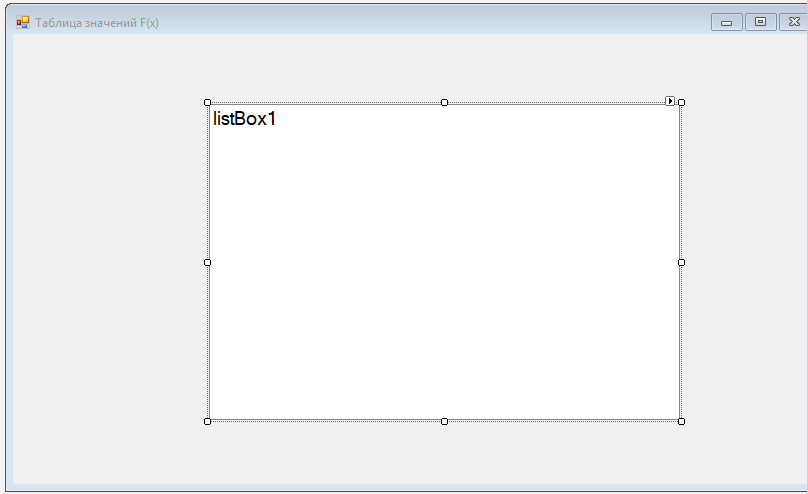
SelectedItem: возвращает или устанавливает текст выделенного элемента

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.

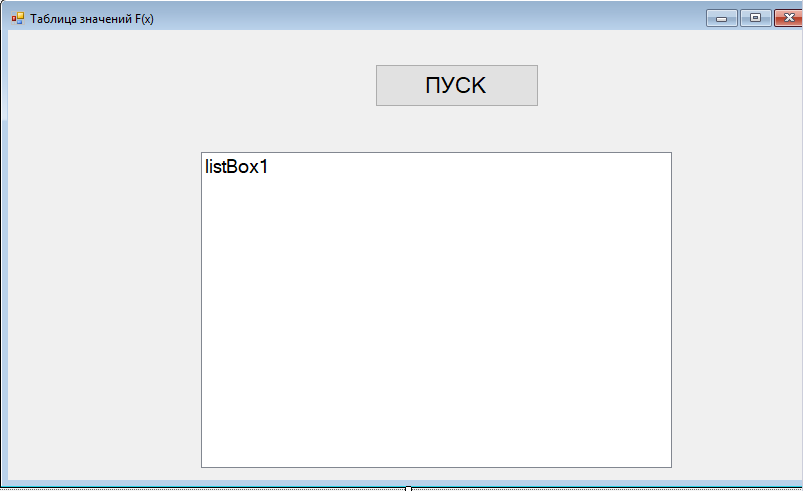
**Задание 1.** Выведите таблицу значений функции *у = sin(x),* если *х* меняется от 1 до 2 с шагом 0,2.

* Откройте *Visual Studio,* далее нажмите на Началь­ной странице пункт меню **Создать проект** и в разделе *Visual C#* выберите приложение *Windows Forms.*
* Задайте имя проекта и нажмите ***ОК.***
* Измените заголовок формы, в свойстве ***Text*** устано­вите значение **Таблица значений *F(x).***
* Разместите компонент ***ListBox.*** В свойстве ***Font*** уста­новите полужирное начертание и размер шрифта рав­ным 14 пт.



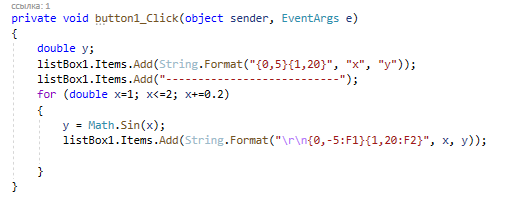
Интерфейс приложения

* Разместите на форме элемент ***Button.*** В свойстве ***Font*** установите полужирное начертание и размер шриф­та равным **14** пт. В свойстве ***Text*** установите значение **Пуск**



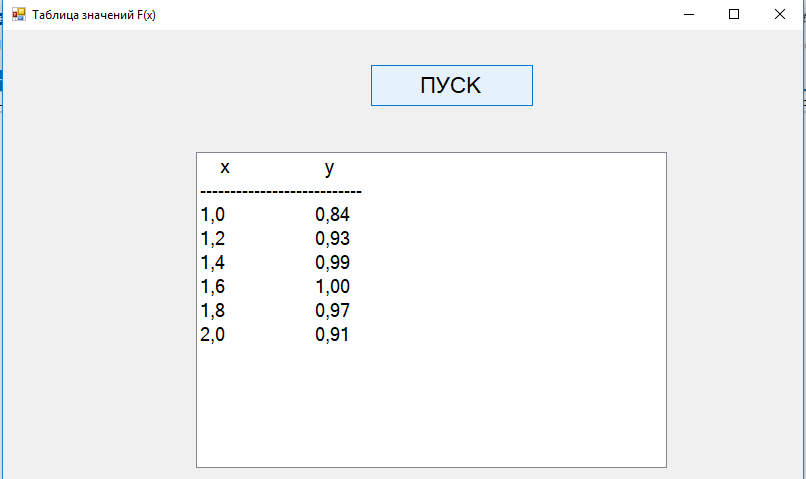
Интерфейс приложения

* Создайте событие для элемента ***Button 1,*** для этого достаточно дважды щелкнуть мышью по данному эле­менту.



Код проекта

* Запустите приложение на выполнение и протести­руйте программу.

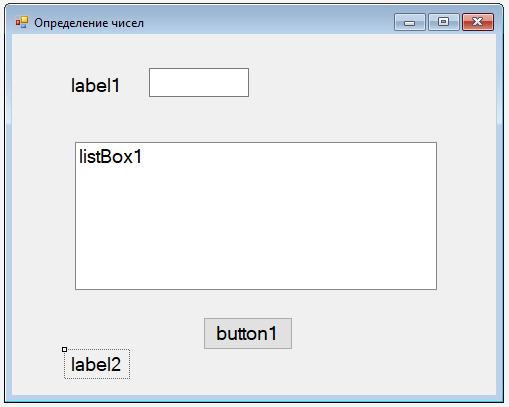


Результат тестирования приложения

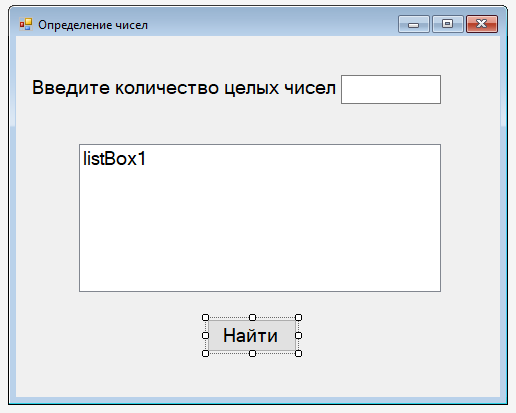
* Экспериментальным путем выясните значение каж­дого параметра функции ***Format*** (удалите один символ, запустите и сравните результаты, верните символ, затем удалите другой и т.д.)

**Задание 2.** Известно *N* чисел (числа задаются слу­чайным образом в промежутке [-50; 50]). Составьте про­грамму, которая определяет количество отрицательных, количество положительных и количество нулей среди введенных чисел. Значение *N* вводится с клавиатуры.

* Откройте ***Visual Studio,*** далее нажмите на Началь­ной странице пункт меню **Создать проект** и в разделе ***Visual C#*** выберите приложение ***Windows Forms.***
* Задайте имя проекта и нажмите ***ОК.***
* Измените заголовок формы, в свойстве ***Text*** устано­вите значение **Определение чисел.**
* Разместите компоненты ***ListBox, Button, TextBox, Label***. В свойстве ***Font*** установи­те размер шрифта **14** пт.
* В свойствах ***Text*** задайте значения,.
* Создайте событие для элемента ***Button 1,*** для это­го достаточно дважды щелкнуть мышью по данному элементу



Размещение элементов на форме проекта



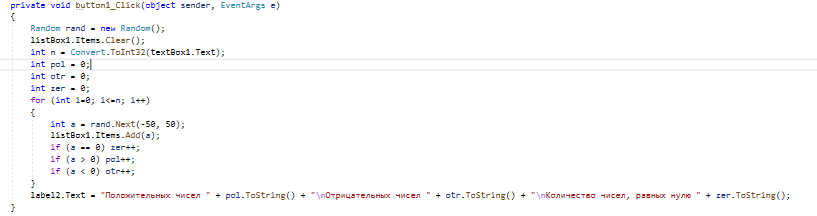
Интерфейс проекта

*Алгоритм данного события*

1. Создается новый экземпляр класса ***Random*** (генера­тора случайных чисел).
2. Очищается ***ListBox1.***
3. Объявление переменных: ***п —*** целый тип и перво­начальное значение берет из компонента ***TextBox1,*** пред­варительно преобразовав тип; ***pol —*** количество положи­тельных чисел; ***otr*** — количество отрицательных чисел; ***zer —*** количество нулей.
4. В цикле от 1 до ***п*** выполняется:

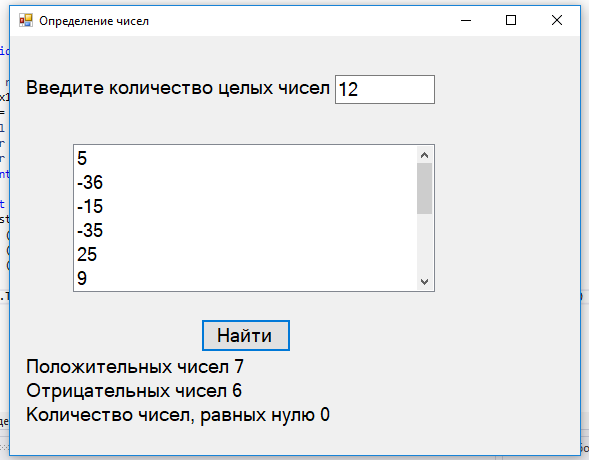
* присваивание случайного числа целочисленной переменной ***а***;
* добавление элемента в компонент ***ListBox1*,** если а равна нулю, тогда переменная ***zer*** увеличи­вается на единицу;
* если а положительная, тогда переменная ***pol*** увели­чивается на единицу;
* если а отрицательная*, тогда переменная* ***otr*** *увели­чивается на единицу;*

1. Вывод результата в элементе ***Label 2 \п*** означает вы­вод с новой строки.



Код проекта

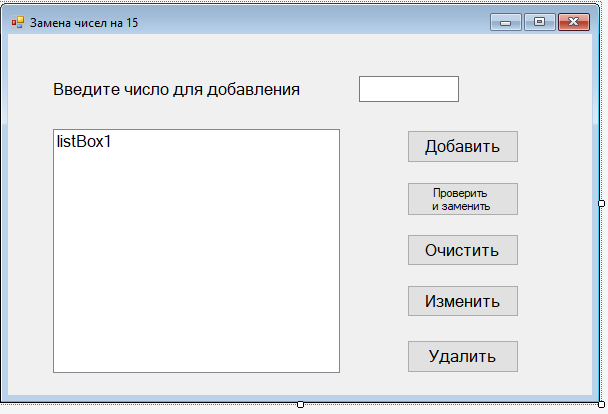
* Запустите приложение на выполнение и протести­руйте программу.



Результат тестирования приложения

**Задание 3.** Введите с клавиатуры 10 чисел. Если среди них есть числа, большие 15, замените их на 15.

* Откройте *Visual Studio,* далее нажмите на Началь­ной странице пункт меню **Создать проект** и в разделе *Visual C#* выберите приложение *Windows Forms.*
* Задайте имя проекта и нажмите ***ОК.***
* Измените заголовок формы, в свойстве ***Text*** устано­вите значение **Определение чисел.**
* Разместите компоненты ***ListBox, Button, TextBox, Label,*** как показано на рис. В свойстве ***Font*** установи­те размер шрифта **12** пт.
* В свойствах ***Text*** задайте значения, соответствую­щие рис.



**Рис.** Интерфейс проекта

* Создайте событие для элемента ***Button 1*** (кнопка **Добавить),** для этого достаточно дважды щелкнуть мы­шью по данному элементу.

*Алгоритм данного события*

1. Объявляется целочисленная переменная ***п,*** началь­ное значение которой равно количеству строк элемента ***ListBox1.***
2. Если ***п*** (количество элементов списка) меньше или равно 9 (т. к. номера строк в списке ***ListBox*** начинают нумероваться с нуля) и если текстовый блок пустой, добавить строку в список ***ListBoxl*** из элемента ***TextBox1,*** преобразовав значение к целому типу, иначе вывести со­общение о невозможности ввода чисел в список

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

{

//1. Объявляется целочисленная переменная п,

//начальное значение которой равно количеству строк элемента ListBox1

int n = listBox1.Items.Count;

//2. Если п (количество элементов списка) меньше или равно 9

//(т. к. номера строк в списке ListBox начинают нумероваться с нуля)

//и если текстовый блок пустой, добавить строку в список ListBoxl из элемента TextBox1,

//преобразовав значение к целому типу, иначе вывести

//сообщение о невозможности ввода чисел в список.

if (n <= 9 && textBox1.Text != "")

{

listBox1.Items.Add(Convert.ToInt32(textBox1.Text));

}

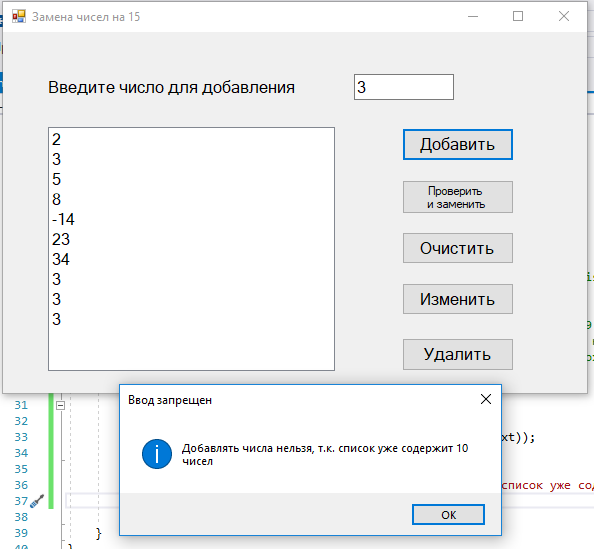
else

MessageBox.Show("Добавлять числа нельзя, т.к. список уже содержит 10 чисел", "Ввод запрещен",MessageBoxButtons.OK, MessageBoxIcon.Asterisk);

textBox1.Text = "";

}

Код нажатия на кнопку «Добавить»



Протестируйте работу кнопки «Добавить»

Создайте событие для элемента ***Button 2*** (кнопка **Про­верить и изменить),** для этого достаточно дважды щел­кнуть мышью по данному элементу.

*Алгоритм данного события*

1. Объявляется целочисленная переменная ***п,*** началь­ное значение которой равно количеству строк элемента ***ListBox1.***
2. Если ***п*** (количество элементов списка) меньше 9 (т.к. номера строк в списке ***ListBox*** начинают нумеро­ваться с нуля), тогда вывести сообщение о недостаточном количестве строк в списке, иначе в цикле от ***i = 0*** до ***п*** проверять, если значение строки с номером ***i*** больше 15, тогда изменить данную строку на значение, равное 15.

private void button2\_Click(object sender, EventArgs e)

{

int n = listBox1.Items.Count;

if (n < 9)

{

MessageBox.Show("Проверить список нельзя, т.к. список содержит менее 10 чисел", "Введите недостающие числа", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);

}

else

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

if (Convert.ToInt32(listBox1.Items[i]) > 15)

{

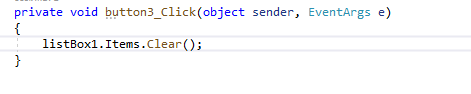
listBox1.Items[i] = 15;

}

}

Код нажатия на кнопку «Проверить и изменить»

* Создайте событие для элемента ***Button 3*** (кнопка **Очистить),** для этого достаточно дважды щелкнуть мы­шью по данному элементу .



Код нажатия на кнопку «Очистить»

* Создайте событие для элемента ***Button 4*** (кнопка **Изменить),** для этого достаточно дважды щелкнуть мы­шью по данному элементу.

Данное событие содержит один условный оператор. Если элемент ***TextBox1*** пустой, тогда выведите сообще­ние о необходимости ввести значение, на которое следу­ет поменять строку списка, иначе выделенному номеру строки списка присвоить значение из элемента ***TextBox1,*** преобразовав к типу ***Int.***

private void button4\_Click(object sender, EventArgs e)

{

if (textBox1.Text == "")

{

MessageBox.Show("Введите число в текстовый блок", "Введите заменяемое число", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);

}

else

{

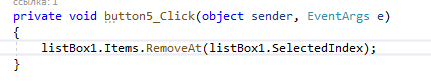
listBox1.Items[listBox1.SelectedIndex] = Convert.ToInt32(textBox1.Text);

}

}

Код нажатия на кнопку «Изменить»

* Создайте событие для элемента ***Button 4*** (кнопка **Удалить),** для этого достаточно дважды щелкнуть мы­шью по данному, элементу. ***Remove At*** — удаляет элемент по указанному индексу.



Код нажатия на кнопку «Удалить»

**Протестируйте работу приложения!**

**Самостоятельное задание № 1**

**1.** Белоснежка на Рождество решила купить гномам новые рубашки и измерила рост каждого. В ма­газине продаются рубашки трех размеров: I — 25x35 см, II — 36x45 см и III — 46x55 см. Выведите таблицу, в ко­торой будут указаны реальный рост каждого гнома и соответствующий ему размер рубашки.

**2.** Клиент внес в банк *K* рублей. Каждый ме­сяц эта сумма увеличивается на *Y* процентов. Составьте таблицу ежемесячных выплат клиенту.