МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ» (ДГТУ)

Колледж экономики, управления и права

Методические указания по организации практических занятий и самостоятельной работы студентов

по ПМ.02. Участие в разработке информационных систем

МДК.02.01 Информационные технологии и платформы разработки информационных систем

Visual Studio 2015

Специальность

09.02.04 Информационные системы (по отраслям)

Ростов-на-Дону 2018

УДК 004 (075.32)

Методические указания по организации практических занятий и самостоятельной работы студентов по ПМ.02. Участие в разработке информационных систем МДК.02.01 Информационные технологии и платформы разработки информационных систем

В данных методических указаниях представлены задания и пошаговые инструкции по разработке приложений в среде MS Visual Studio 2015 на языке C#, а также даны задания для самостоятельной работы студентов и вопросы для самоконтроля.

Определяют этапы выполнения работы на практическом занятии, содержат рекомендации по выполнению индивидуальных заданий и образцы решения задач, а также список рекомендуемой литературы.

Разработаны с учетом ФГОС среднего профессионального образования специальности 09.02.04 Информационные системы (по отраслям), предназначены для студентов и преподавателей колледжа.

Автор-составитель: С.В.Шинакова

Одобрены решением учебно-методического совета колледжа и рекомендованы к практическому применению в образовательном процессе.

Протокол № 1 от «31» августа 2018 г

Председатель учебно-методического совета колледжа
_____ С.В.Шинакова
личная подпись

Ответственный за выпуск к.п.н., заместитель директора колледжа И.И.Джужук

заказ 1087 от 30.10.18

СОДЕРЖАНИЕ

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 1 4
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 2 11
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 3 17
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 4 20
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 5 24
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 6 32
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 7 35
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 8 41
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 9 48
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 10 54
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 11 55
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 12 – 13 61
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 14 71
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 15 75
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 16 80
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 17 82
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 18-19 86
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ№ 20 94
ПРАКТИЧЕСКАЯ РАБОТА № 21 105
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 22-23 108
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 24 112

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 1
ЯЗЫК ПРОГРАММИРОВАНИЯ И СРЕДА РАЗРАБОТКИ

Цель занятия: формирование навыка работы со средой программирования MS Visual Studio 2015 .

Этапы выполнения работы:

1. Все студенты на занятии выполняют задание, указанное ниже.
2. Дома студенты готовят ответы на вопросы для самоконтроля и выполняют задания для самостоятельной работы.
3. На следующем занятии каждый студент отвечает на вопросы преподавателя.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

MS Visual Studio – это высокоэффективная графическая среда для визуальной разработки программ.

Интерфейс MS Visual Studio 2015

1 - Главное окно MS Visual Studio 2015 (оно имеет заголовок «MS Visual Studio»).

2 - Окно Формы (заголовок «Form1»).

Окно Формы представляет собой своеобразную макетницу для сборки на ней окна разрабатываемого приложения. Сама Форма и все то, что находится в её поле, с точки зрения MS Visual Studio, представляют собой *объекты*. Каждому объекту MS Visual Studio присваивает своё уникальное имя, которое, при необходимости, можно изменить.

3 - Окно Кода Программы (заголовок «Form1.cs»).

В меню Вид находятся вкладки **Обозреватель решений, Командный обозреватель, Панель элементов, Свойства** и др., которые при необходимости можно включать и выключать.

С помощью вкладки **Свойства** можно просматривать и изменять характеристики (свойства) выделенного объекта. Сразу после запуска на этой вкладке находятся свойства такого объекта, как сама Форма Form1: заголовок (Text), имя объекта (Name), его геометрические размеры (Size), цвет фона (BackColor) и др.

Свойство формы	Описание
Name	Имя формы
Text	Текст заголовка
Location	Положение компонента на поверхности формы
Size.Width	Ширина формы
Size.Height	Высота формы
StartPosition	Положение формы в момент первого появления на форме
Location.X	Расстояние от верхней границы формы до верхней границы экрана
Location.Y	Расстояние от левой границы формы до левой границы экрана
FormBorderStyle	Вид границы формы
ControlBox	Если значение равно false, то кнопки управления окном и контекстное меню не отображаются
MaximazeBox	Признак доступности кнопки Развернуть
MinimizeBox	Признак доступности кнопки Свернуть
StartPosition	Расположение окна приложения при запуске
Icon	Значок в заголовке окна
Font	Шрифт, используемый по умолчанию компонентами, находящимися на поверхности формы
ForeColor	Цвет, используемый по умолчанию компонентами, находящимися на поверхности формы, для отображения текста
BackColor	Цвет фона
BackGroundImage	Фоновое изображение
Opacity	Степень прозрачности формы

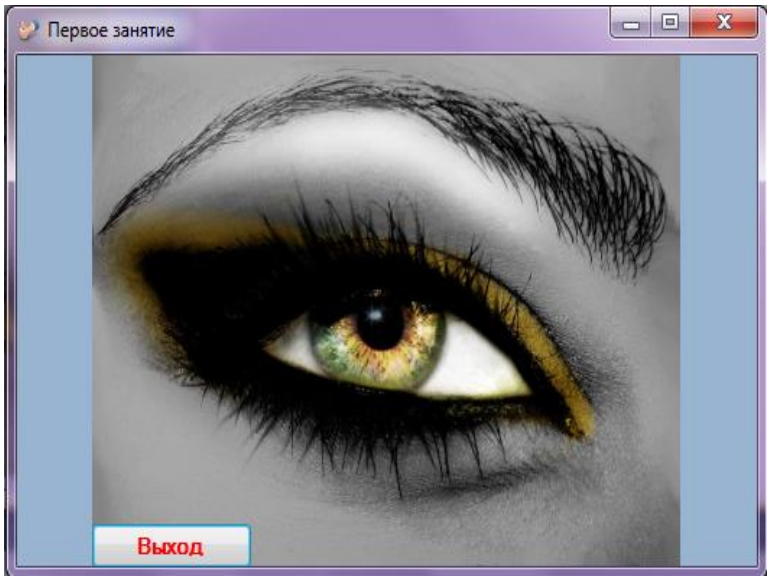
ХОД РАБОТЫ

Задание 1. Создать форму, соответствующую следующим требованиям: переименовать форму; фон задать изображением, заполнив всю форму; изменить системный значок; изменить вид границы формы; расположить форму по центру экрана при запуске приложения; расположить кнопку на форме; изменить размер и цвет шрифта на кнопке.

Запрограммировать кнопку Выход:

```
private void button1_Click(object sender, EventArgs e)
        {
            this.Close();
        }
```

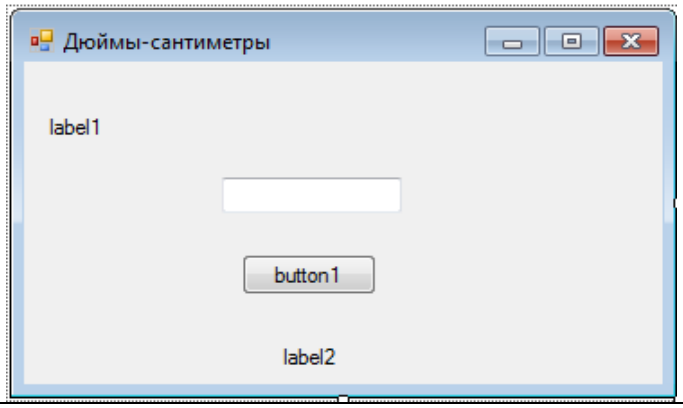
Пример интерфейса:



Задание 2. Создать простейшее приложение в MS Visual Studio - программу пересчета.

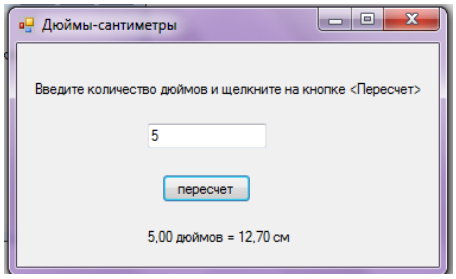
Алгоритм создания приложения:

1. Создайте папку для файлов проекта «Дюймы-сантиметры».	а) MS Visual Studio/ Создать проект Приложение Windows Forms; б) В окне «Создать проект» задаем имя Сумма, выбираем место сохранения проекта; в) Нажать кнопку Ok.

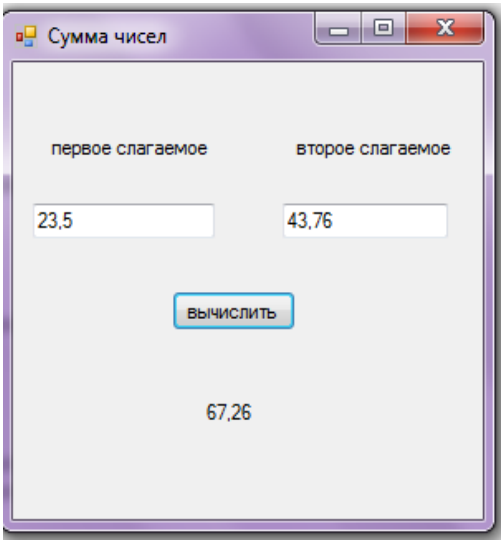
2. Студенты форму размещают (textBox1, как		самостоятельно переименовывают (Form1 ➔ «Дюймы-сантиметры»), на форме компоненты label1, label2, button1) показано на рисунке:
3.		

4. Удалить текст на компоненте label2	label2 → Свойства label2→ Text, удаляем текст.
5. Переименовать label1	Свойства label1→ Text, пишем Введите количество дюймов и щелкните на кнопке <Пересчет>
6. Переименовать button1	button1 → Свойства Button1→ Text, пишем Пересчет
7. Запрограммировать кнопку «Пересчет»	`private void button1_Click (object sender, EventArgs e)` `{` `  double duym;` `  double sm;` `  duym=Convert.ToDouble(textBox1.Text);` `  sm = duym * 2.54;` `  label2.Text = duym.ToString("N")+" дюймов = "+sm.ToString("N")+" см";` `}`

Приложение имеет вид:



Задание 3. Создать приложение, позволяющее находить сумму двух чисел. Интерфейс приложения:



Алгоритм создания приложения:

1. Создайте папку для файлов проекта «Сумма чисел».	
2. Студенты самостоятельно переименовывают форму (Form1 ➔ «Сумма чисел»), размещают на форме компоненты (label1, label2, label3, textBox1, textBox2, button1)	
3. Расположенные объекты выровнять по горизонтали.	
4. Переименовать label1, label2, button1	
5. Label3 очистить	
6. Запрограммировать кнопку «Вычислить»	`private void button1_Click(object sender, EventArgs e)` `{` `double a;` `double b;` `double c;` `a = Convert.ToDouble(textBox1.Text);` `b = Convert.ToDouble(textBox2.Text);` `c = a + b;` `label3.Text = c.ToString("N");` `}`

Задание 4. Создать проект с использованием библиотеки базовых классов. Использование таймера.

Алгоритм создания приложения:

1. Создайте форму, поместите на форму элементы label, textbox, timer, button. Создадим проект, выключающий компьютер через введенное число минут.	
2. Запрограммировать кнопку «Выключить»	В переменной ShutTime будет храниться время выключения компьютера `Добавьте using System.Diagnostics;` `DateTime ShutTime;` `private void button1_Click(object sender, EventArgs e)` `{`

	ShutTime = DateTime.Now.AddMinutes(Convert.ToInt32(textBox1.Text)); label1.Text = "До выключения осталось " + textBox1.Text + " мин."; timer1.Start(); }
3. Для элемента Timer создайте событие Tick.	private void timer1_Tick(object sender, EventArgs e) { if (DateTime.Now < ShutTime) { TimeSpan ts = ShutTime - DateTime.Now; label1.Text = "До выключения осталось " + ts.Minutes + " мин. " + ts.Seconds + " сек."; } else { Close(); Process.Start("Shutdown.exe", " -s"); } }

Использование блокировки

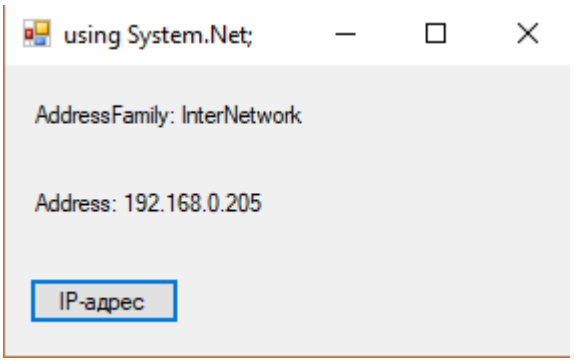
```
using System.Threading;

Thread.Sleep(TimeSpan.FromSeconds(5));
```

Получение параметров сети

Добавьте `using System.Net;`

Код кнопки:

```
IPHostEntry heserver = Dns.GetHostEntry(Dns.GetHostName());
foreach (IPAddress curAdd in heserver.AddressList)
{
 label1.Text="AddressFamily: " + curAdd.AddressFamily.ToString();
 label2.Text = "Address: " + curAdd.ToString();
}
```

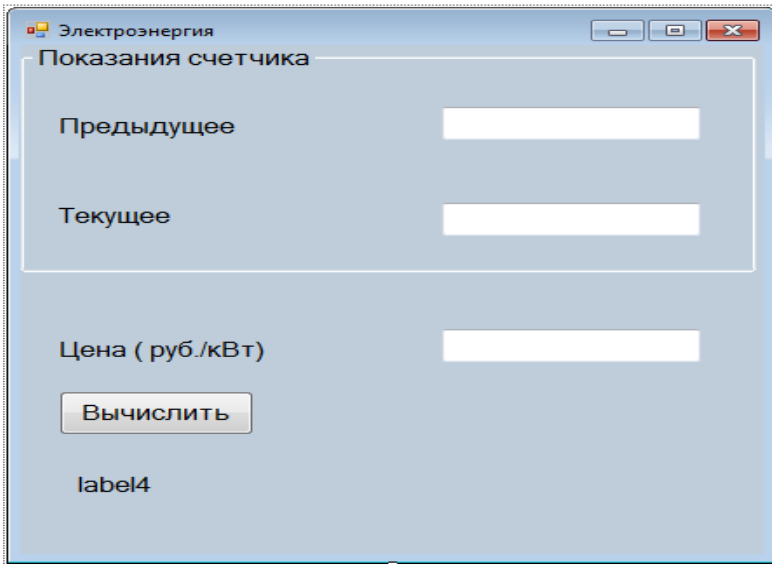


Получение информации о папке с файлом

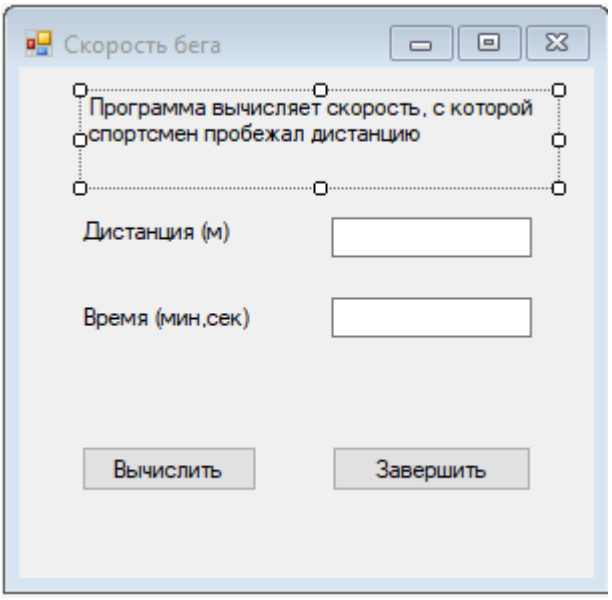
```
label1.Text= Environment.CurrentDirectory;
```

САМОСТОЯТЕЛЬНАЯ РАБОТА

1. Создать проект Конвертер, пересчитывающий цену из долларов по данному курсу в рубли.
2. Изменить значок приложения в задании 1, создав, например, значок с изображением букву «К» в белом круге. Расположить окно работающего приложения по центру экрана.
3. Разработать проект, содержащий форму, на которую помещена кнопка, элементы управления Label и MonthCalendar. При нажатии на кнопку в элементе управления Label1 выдается текущая дата.
4. Разработать проект, содержащий форму, на которую помещена кнопка и элемент управления Label. При нажатии на кнопку содержимое элемента управления Label изменяется на название вашей группы.
5. Разработать проект, содержащий форму, на которую помещена кнопка и три элемента управления label. При нажатии на кнопку в элементы управления label выдаются имя машины, имя пользователя, количество процессоров.
6. Разработать проект, содержащий форму, на которую помещена кнопка и три элемента управления label. При нажатии на кнопку в элементы управления label выдаются текущий каталог, версия операционной системы, путь к каталогу операционной системы.
7. Создать проект «Электроэнергия».



8. Создать проект, используя следующий интерфейс.



Примечание: чтобы предложение - Программа вычисляет скорость, с которой спортсмен пробежал дистанцию - на форме было написано в 2 строчки, нужно свойству AutoSize компонента label1 задать значение false.

9. Разработать проект, содержащий форму, на которую помещена кнопка, элементы управления timer и textBox. При нажатии на кнопку должен вызваться диспетчер задач через интервал, указанный в textbox.
10. Разработать проект, содержащий форму, на которую помещена кнопка и элемент управления timer. При нажатии на кнопку через 5 секунд должен вызваться Блокнот.
11. Разработать проект, содержащий форму, на которую помещена кнопка и элемент управления timer. При нажатии на кнопку через 1 минуту должен вызваться графический редактор Paint.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Зачем нужен компонент «label» и где он находится?
2. Зачем нужен компонент «textBox»?
3. Зачем нужен компонент «button» и какими свойствами он обладает?
4. Как увеличить размер шрифта объекта label?
5. Как на объект «label» вывести результат?
6. Как в объект «textBox» вывести результат?
7. Как убрать кнопки изменения размером окна?
8. Назначение using System.Net.
9. Назначение using Threading.
10. Назначение using Diagnostics.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 2
ЯЗЫК ПРОГРАММИРОВАНИЯ И СРЕДА РАЗРАБОТКИ

Цель занятия: формирование навыков работы по созданию консольных приложений и DLL - проекта в среде MS Visual Studio.

Этапы выполнения работы:

1. Создание DLL – проекта.
2. Самостоятельная работа по созданию консольного приложения:
   1) студенты, записанные в журнале под четными номерами выполняют задания 2, 4 и 6. Остальные – 1, 3 и 5. Время выполнения – 40 минут.
   2) далее проходит тестирование по теоретическому материалу лекций 1 и 2.

ХОД РАБОТЫ

Задание 1. Создать dll-проект.

Алгоритм создания приложения:

<table>
<tr><td>1. Создать проект</td><td>1. Запустим Visual Studio 2015, со стартовой страницы перейдем к созданию проекта и в качестве типа проекта укажем тип «Библиотека классов».

2. В открывшемся окне создания DLL, в поле Имя задается имя строящейся DLL – MathMat.

3. Указывается путь к папке, где будет храниться Решение, содержащее проект.

4. В поле Решение выбран элемент «Создать новое решение», создающий новое Решение. Альтернативой является элемент списка, указывающий, что проект может быть добавлен к существующему Решению. Здесь выбрано имя Занятие 2, указывающее на то, что все проекты данного занятия вложены в одно Решение.

Задав требуемые установки и щелкнув по кнопке «ОК», получим автоматически построенную заготовку проекта DLL, открытую в среде разработки проектов Visual Studio 2015.</td></tr>
<tr><td colspan="2">Примечание.

В окне проектов Обозреватель решений показано Решение с именем «Занятие 2», содержащее проект DLL с именем «MathMat». В папке «Properties» проект содержит файл с описанием сборки – ее имя и другие характеристики. В папке «References» проект содержит ссылки на основные пространства имен библиотеки FCL, которые могут понадобиться в процессе работы DLL.

Поскольку всякая DLL содержит один или несколько классов, то для одного класса, которому по умолчанию дано имя «Class1», заготовка построена. Класс этот, показанный в окне кода, пока что пуст – не содержит никаких элементов.</td></tr>
<tr><td>2. Изменим имя «Class1» на имя «MyMath»</td><td>В контекстном меню выбрать команду Переименовать.</td></tr>
<tr><td colspan="2">Примечание.

Имя класса и имя файла, хранящего класс, должны совпадать. Переименование имени файла делается непосредственно в окне проектов Обозреватель решений.</td></tr>
</table>

3. Создать для класса заголовочный комментарий	Для добавления комментария достаточно в строке, предшествующей заголовку класса набрать три подряд идущих слеша (три косых черты). В результате перед заголовком класса появится заголовочный комментарий – тег «*summary*», в который и следует добавить краткое, но содержательное описание сути класса. Комментарий: «Аналог класса Math библиотеки FCL».

Примечание.
Теги «*summary*», которыми следует сопровождать классы, открытые (public) методы и поля класса играют три важные роли.
1. Они облегчают разработку и сопровождение проекта, делая его само документируемым.
2. Клиенты класса при создании объектов класса получают интеллектуальную подсказку, поясняющую суть того, что можно делать с объектами.
3. Специальный инструментарий позволяет построить документацию по проекту, включающую информацию из тегов «*summary*».

4. Зададим заголовочный комментарий к методу	В тело класса добавим следующий код: /// <summary> /// Sin(x) /// </summary> /// <param name="x"> угол в радианах - аргумент функции Sin </param> /// <returns> Возвращает значение функции Sin для заданного угла </returns> public static double Sin(double x) { }

Примечание. Для вычисления применим формулу:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots = \sum_{i=0}^{\infty}(-1)^i \frac{x^{2i+1}}{(2i+1)!} \qquad (1)$$

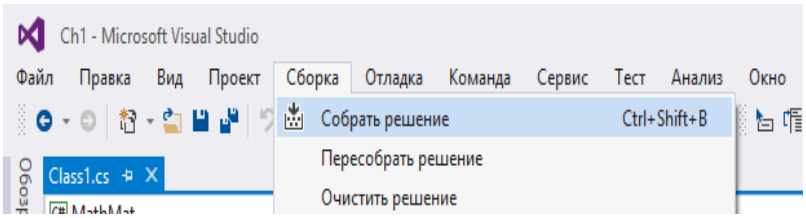
5. Полный код проекта DLL:	

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace MathMat
{    /// <summary>
     /// Аналог класса Math библиотеки FCL
     /// </summary>

    public class MyMath
    {
        //Константы класса
        const double TWOPI = 2 * Math.PI;
        const double EPS = 1E-9;

        //Статические методы класса
```

```
        /// <summary>
        /// Sin(x)
        /// </summary>
        /// <param name="x">угол в радианах - 
аргумент функции Sin</param>
        /// <returns>Возвращает значение функции Sin 
для заданного угла</returns>
        public static double Sin(double x)
        {
            //Оптимизация - приведение к интервалу
            x = x % TWOPI;

            //Инициализация
            double a = x;
            double res = 0;
            int k = 0;

            //Основные вычисления
            while (Math.Abs(a) > EPS)
            {
                res += a;
                a *= -x * x / ((2 * k + 2) * (2 * k + 
3));
                k++;
            }
            return res;
        }
    }
}
```

6. Построить Решение, содержащее проект

В меню среды выберем пункт Сборка / Собрать решение



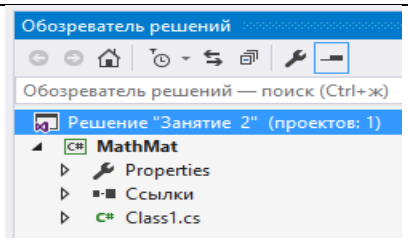
В результате успешной компиляции будет построен файл dll.

Так как построенная сборка не содержит выполняемого файла, то непосредственно запустить наш проект на выполнение не удастся. Построим консольный проект, к которому присоединим DLL.

Задание 2. Создать консольное приложение и присоединить, созданную DLL.

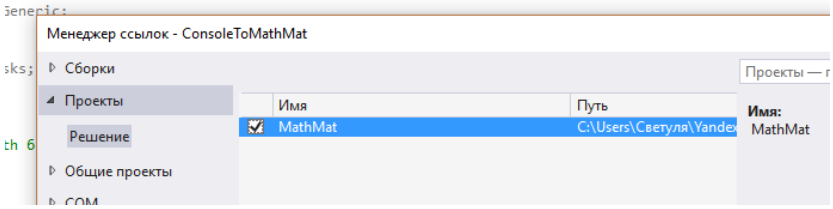
Алгоритм создания приложения:

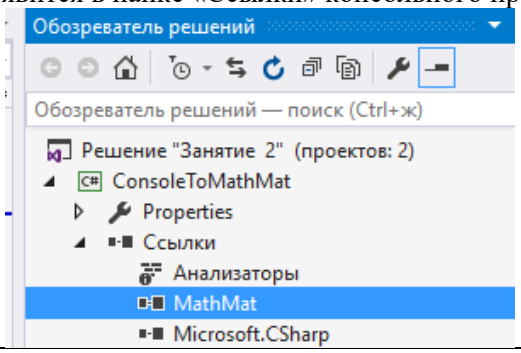
1. Добавить проект в решение Занятие 2	1. В обозревателе решений выделить строку

<table>
<tr><td></td><td> 

В контекстном меню выбрать Добавить / Создать проект / Консольное приложение.

2. В поле Имя задается ConsoleToMathMat.</td></tr>
<tr><td colspan="2">Примечание.

Автоматически создается класс с именем Program, содержащий единственный статический метод – процедуру Main</td></tr>
<tr><td>2. Связать два построенных проекта</td><td>В консольный проект добавим ссылку на проект с DLL MathMat.
1. В окне Обозреватель решений подвести указатель мыши к имени консольного проекта и из контекстного меню, выбрать пункт Добавить / Ссылка.
2. В открывшемся окне добавления ссылок выберем вкладку «Проекты».
 

3. Щелкнуть по появившемуся в окне имени MathMat. Ссылка на DLL появится в папке «Ссылки» консольного проекта.
 
</td></tr>
<tr><td colspan="2">Примечание.

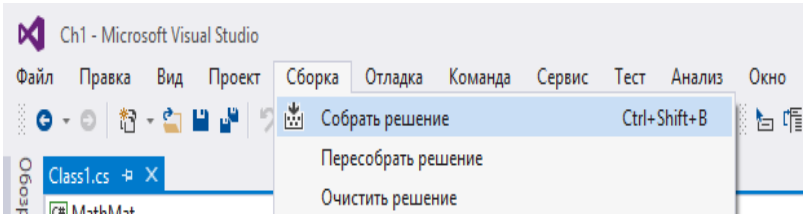
Так как проект MathMat включен в Решение, то он автоматически появится в открывшемся окне. Если ссылку нужно установить на проект, не включенный в Решение, то в окне добавления ссылок нужно задать путь к проекту. Теперь проекты связаны и из консольного проекта доступны сервисы, предоставляемые DLL.</td></tr>
<tr><td>4. Полный код проекта DLL:</td><td><pre>
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace MathMat
{   /// <summary>
     /// Аналог класса Math библиотеки FCL
     /// </summary>
</pre></td></tr>
</table>

```
    public class MyMath
    {
        //Константы класса
        const double TWOPI = 2 * Math.PI;
        const double EPS = 1E-9;

        //Статические методы класса

        /// <summary>
        /// Sin(x)
        /// </summary>
        /// <param name="x">угол в радианах - 
аргумент функции Sin</param>
        /// <returns>Возвращает значение функции Sin 
для заданного угла</returns>
        public static double Sin(double x)
        {
            //Оптимизация - приведение к интервалу
            x = x % TWOPI;

            //Инициализация
            double a = x;
            double res = 0;
            int k = 0;

            //Основные вычисления
            while (Math.Abs(a) > EPS)
            {
                res += a;
                a *= -x * x / ((2 * k + 2) * (2 * k + 
3));
                k++;
            }
            return res;
        }
    }
}
```

5. Построить Решение, содержащее проект

В меню среды выберем пункт Сборка / Собрать решение



В результате успешной компиляции будет построен файл dll.

Так как построенная сборка не содержит выполняемого файла, то непосредственно запустить наш проект на выполнение не удастся. Построим консольный проект, к которому присоединим нашу DLL.

Код проекта:

```
namespace ConsoleToMathMat1
{
```

```
    class Program
    {
        static void Main(string[] args)
        {
              //Входные данные
                double x = 0;

                const string INVITE = "Введите вещественное число x" +  "-
аргумент функции Sin(x)";
                const string CONTINUE =  "Продолжим? (Yes/No)";
                string answer = "yes";
                do
                {
                    //Организация ввода данных
                    Console.WriteLine(INVITE);
                    string temp = Console.ReadLine();
                    x = Convert.ToDouble(temp);

                    //Вычисления и вывод результата
                    double res = 0;
                    res = Math.Sin(x);
                    Console.WriteLine("Math.Sin(x) = " + res.ToString());

                    res = MathMat.MyMath.Sin(x);
                    Console.WriteLine("MathTools.MyMath.Sin(x) = " +
res.ToString());

                    //диалог с пользователем
                    Console.WriteLine(CONTINUE);

                    answer = Console.ReadLine();
                } while (answer == "yes");
            }

        }
}
```

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Построить консольное приложение «Выражение». Приложение вычисляет значение X и выводит его на консоль, где в вычисляемом выражении m, n, p, a, b, c, d, e – это имена переменных, значения которых задает пользователь.
2. Создать приложение, вычисляющее сумму и частное двух целых чисел. Оформить форматированный вывод.
3. Построить Windows-приложение «Круг». Дано: r – радиус круга. Вычислить: диаметр, длину окружности, площадь круга.
4. Построить Windows-приложение «Треугольник». Дано: стороны треугольника a, b, c. Вычислить периметр и площадь треугольника.
5. Постройте Windows-приложение, в котором тип источника – string, тип цели – один из подтипов арифметического типа, выбираемый из списка. Преобразование выполните с использованием метода Parse.

6. Постройте Windows-приложение, в котором тип источника – string, тип цели – один из подтипов арифметического типа, выбираемый из списка. Преобразование выполните с использованием метода класса Convert.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 3
ОПЕРАТОРЫ ЯЗЫКА C#

Цель занятия: формирование навыка работы со средой программирования MS Visual Studio при разработке приложений с применением условного оператора.

Этапы выполнения работы:

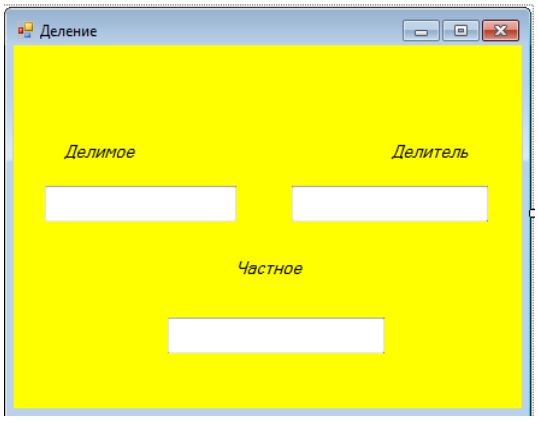
1. Все студенты на занятии выполняют задания, указанные ниже.
2. Дома студенты готовят ответы на вопросы для самоконтроля и выполняют задания для самостоятельной работы.
3. На следующем занятии каждый студент отвечает на вопросы преподавателя.

ХОД РАБОТЫ

Задание 1. Разработать приложение, позволяющее находить частное от деления двух целых чисел.

Алгоритм создания приложения:

1. Самостоятельно создать форму с необходимыми на ней компонентами изменить свойства объектов



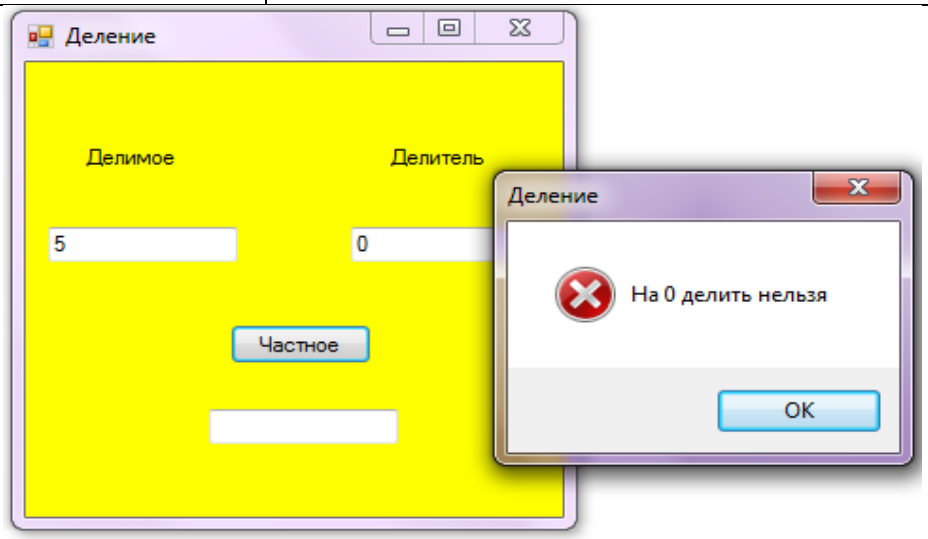
2. Создать код обработчика события, протестировать приложение.

```
private void button1_Click (object sender,
EventArgs e)
   {
      double a;
      double b;
      double c;
      a = Convert.ToDouble(textBox1.Text);
      b = Convert.ToDouble(textBox2.Text);
      c = a / b;
      textBox3.Text = c.ToString("N");
    }
```

3. Создать модальное окно, применяя MessageBox.Show

```
private void button1_Click(object sender, EventArgs
e)
  {
     double a;
     double b;
     double c;
     a = Convert.ToDouble(textBox1.Text);
     b = Convert.ToDouble(textBox2.Text);
     if (b = = 0)
     {
```

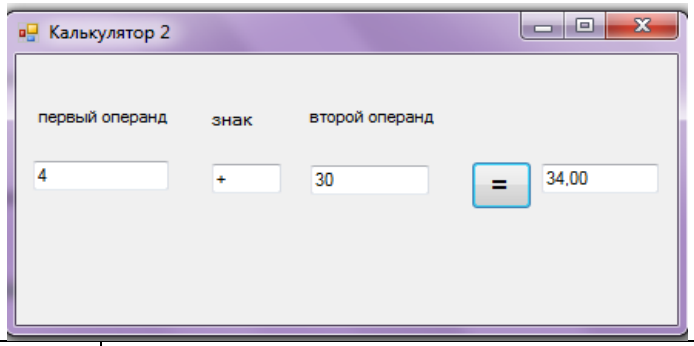
```
    MessageBox.Show("На 0 делить нельзя",
"Деление", MessageBoxButtons.OK,
MessageBoxIcon.Error);
      }
     else
       {
          c = a / b;
          textBox3.Text = c.ToString("N");
        }
    }
```



4. Протестировать приложение.

Задание 2. Создать приложение с учетом следующего интерфейса

Алгоритм создания приложения:



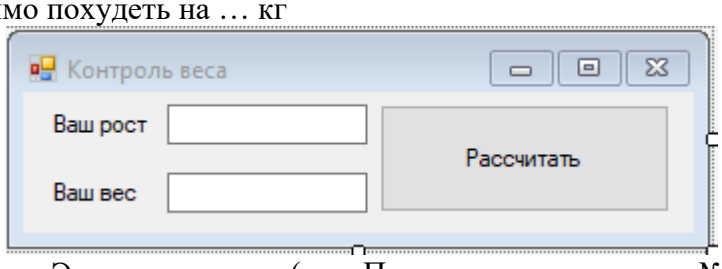
1. Применить оператор switch

```
double  a, b, c;
char operation;
   a = Convert.ToInt32 (textBox1.Text);
   b = Convert.ToInt32 (textBox3.Text);
   operation = Convert.ToChar (textBox2.Text);

   switch (operation)
   {
      case '+':
      c = a + b;
      textBox4.Text = c.ToString("N");
      break;
```

	`` case '-': c = a - b; textBox4.Text = c.ToString("N"); break; case '*': c = a * b; textBox4.Text = c.ToString("N"); break; case '/': if (b = = 0) MessageBox.Show ("На 0 делить нельзя", "Калькулятор"); else { c = a / b; textBox4.Text = c.ToString("f"); } break; default: MessageBox.Show ("Вы ввели не тот символ", "Калькулятор 2"); break; } ```
2. Протестировать приложение.	
3. Изменить тип данных переменных a, b на int. Протестировать приложение, сравнить результаты при делении, например 2/4	

САМОСТОЯТЕЛЬНАЯ РАБОТА

1. Создать приложение, позволяющее решать квадратное уравнение, заданное коэффициентами. Создать обработчик события, который будет подготавливать окно к новому расчету (кнопка "Новое"). Осуществить выход из приложения по кнопке "Выход".
2. Составить приложение для вычисления стоимости телефонного разговора в зависимости от дня недели. Цена одной минуты разговора 1.5 руб., в субботу и воскресенье скидка 50%.
3. Создать приложение для определения оптимального веса. Программа запрашивает Ваш вес и рост, вычисляет оптимальное для Вас значение веса (рост минус сто), сравнивает его с реальным и выводит одно из следующих сообщений:
   - Ваш вес оптимален
   - Вам надо поправиться на … кг
   - Вам необходимо похудеть на … кг



4. Создать проект Электроэнергия (см. Практическое занятие №1, задание №2), учитывая, что новое показание счетчика должно быть не меньше предыдущего.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 4
ОПЕРАТОРЫ ЯЗЫКА C#

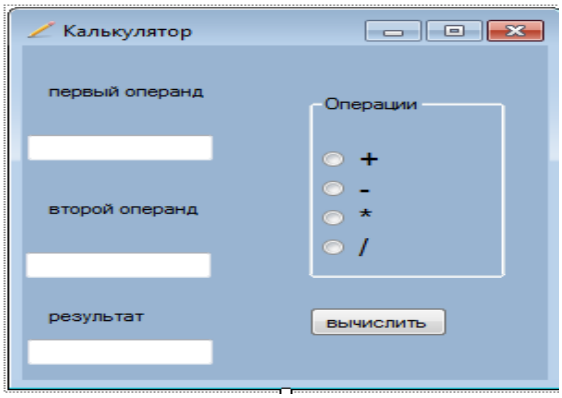
Цель занятия: формирование навыка работы со средой программирования MS Visual Studio при разработке приложений с применением оператора выбора, компонентов: radioButton, checkBox, groupBox. Изучение методов отладки в Visual Studio.

ХОД РАБОТЫ

Задание 1. Создать приложение, которое позволяет вводить два целых числа и выполнять над ними арифметические операции. Для выбора операции используйте переключатели, вывод сообщения об ошибке при вводе делителя, равного нулю, отображается в отдельном окне сообщения.

Алгоритм создания приложения:

1.Самостоятельно создать форму с необходимыми на ней компонентами изменить свойства объектов



2. Создать код обработчика события, протестировать приложение.

```
private void button1_Click(object sender, EventArgs e)
   {
        int a, b;
        double c;
        a = Convert.ToInt32(textBox1.Text);
        b = Convert.ToInt32(textBox2.Text);
            if (radioButton1.Checked)
            {
              c = a + b;
              textBox3.Text =c.ToString("N");
            }
            if (radioButton2.Checked)
            {
             c = a - b;
             textBox3.Text = c.ToString("N");
            }

            if (radioButton3.Checked)
            {
             c = a * b;
             textBox3.Text = c.ToString("N");
            }
            if (radioButton4.Checked)
            {
```

	`` if (b == 0) MessageBox.Show ("На 0 делить нельзя","Калькулятор"); else { c = a / b; textBox3.Text = c.ToString("N"); } } } ```
3. Протестировать приложение	

ТЕОРЕТИЧЕСКИЙ МАТЕРИАЛ

Отладчик Visual Studio является мощным инструментом наблюдения за поведением программы во время выполнения и поиска логических ошибок. Отладчик работает со всеми языками программирования Visual Studio и библиотеками. С помощью отладчика можно прерывать или приостанавливать выполнение программы с целью проверки кода, вычислять и редактировать значения переменных программы, отслеживать состояние регистров процессора, просматривать инструкции, созданные из исходного кода, а также просматривать область памяти, используемую приложением. В режиме «Изменить и продолжить» можно внести изменения в код при отладке и затем продолжить выполнение.

Отладчик Visual Studio предоставляет меню «Отладка» для доступа к средствам отладчика. Окна и диалоговые окна отладчика отображают сведения о программе и позволяют вводить дополнительные сведения.

В отладчике Visual Studio предусмотрены эффективные команды для управления выполнением приложения. В следующих разделах описаны задачи, которые можно решать с помощью команд отладчика, предназначенных для управления выполнением:

- Начать (или продолжить) отладку;
- Прервать все;
- Остановить отладку;
- Перезапустить;
- Пошаговое выполнение приложения;
- Выполнение до заданного места;
- Задание точки останова.

Задание 2. Применение отладки проекта.

Порядок выполнения задания:

1. Начало отладки

1.1 В меню **Отладка** выберите **Запуск, Шаг с заходом или Шаг с обходом**.

1.2. В окне исходного кода щелкните правой кнопкой мыши строку исполняемого кода и выберите команду **Выполнять до текущей позиции.**

Если выбрана команда **Запуск**, приложение запустится и будет выполняться до точки останова. Выполнение можно прервать в любой момент, чтобы просмотреть или изменить значения переменных, либо для выполнения других операций, связанных с проверкой работы программы. При выборе команд **Шаг с заходом или Шаг с обходом** выполнение приложения после его запуска будет прервано на первой строке кода.

При выборе команды **Выполнять до текущей позиции** приложение запустится и будет выполняться либо до точки останова, либо до текущего положения курсора, если точка останова встречена не будет. Положение курсора определяется в окне исходного кода. В некоторых случаях прерывания не происходит. Это означает, что код, на котором стоит курсор, так и не был достигнут в ходе выполнения.

Решение может содержать несколько проектов. В этом случае запускаемый проект можно выбрать с помощью команд из меню **Отладка.** Выбранный проект можно запустить также из обозревателя решений.

Для запуска проекта без отладчика служит команда **Запуск без отладки** (она находится в меню **Отладка**).

2. Приостановка выполнения программы вручную

2.1 В меню **Отладка** выберите **Прервать все.**

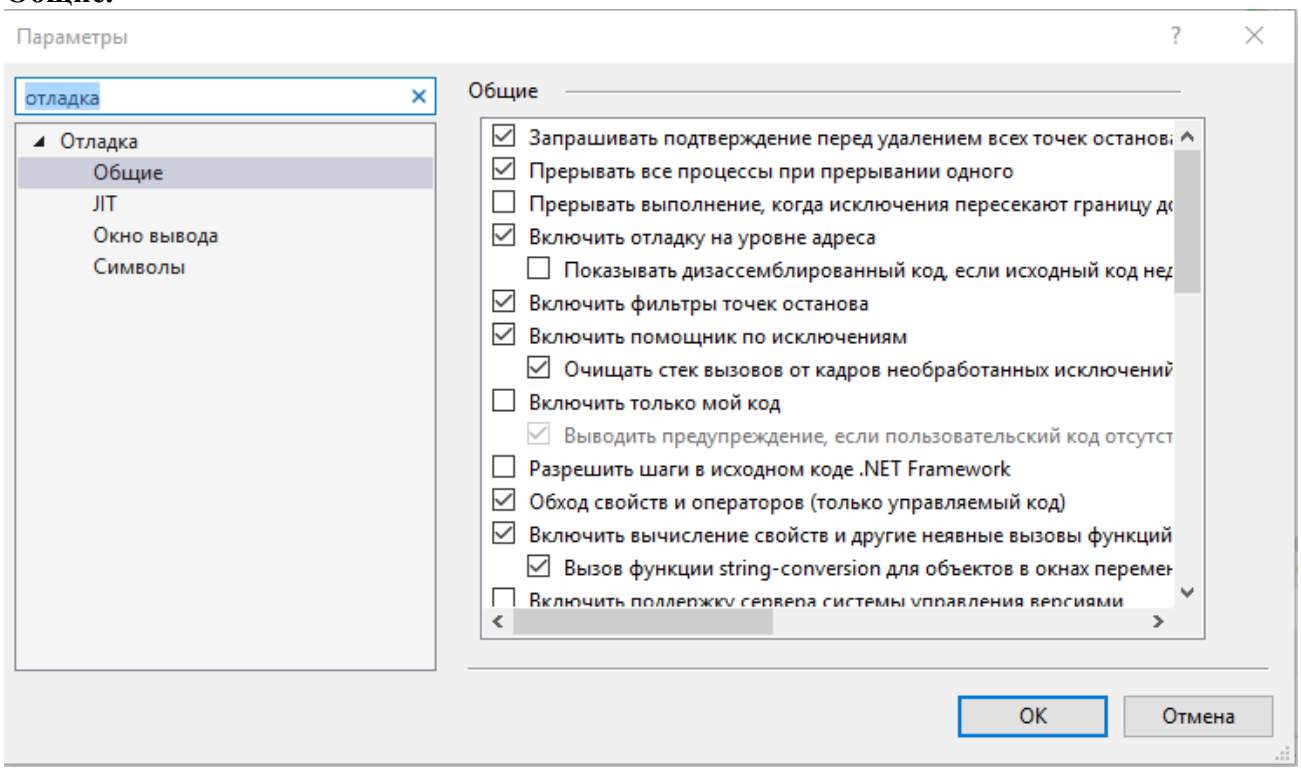
Отладчик останавливает выполнение всех программ, запущенных под его управлением. Эти программы не завершаются, и можно возобновить их выполнение в любой момент. Отладчик и приложение теперь находятся в режиме приостановки.

2.2 При отладке нескольких программ команда **Прервать все или точка останова** по умолчанию влияет на выполнение всех отлаживаемых программ. Для прерывания только текущей программы можно изменить этот режим, используемый по умолчанию.

3. Изменение режима приостановки при отладке нескольких программ

3.1 В меню **Сервис** выберите пункт **Параметры.**

3.2 В диалоговом окне **Параметры** откройте папку **Отладка** и выберите категорию **Общие.**



3.3 Установите или снимите флажок **Прерывать все процессы при прерывании одного.** Нажмите кнопку **ОК.**

4. Пошаговое выполнение

Одной из наиболее распространенных процедур отладки является пошаговое выполнение. При пошаговом выполнении код выполняется по одной строке за раз.

В меню **Отладка** предусмотрены команды для пошаговой отладки кода.

Шаг с заходом

Шаг с обходом

Команды **Шаг с заходом и Шаг с обходом** отличаются только в одном — в способе обработки вызовов функций.

Обе команды указывают отладчику на то, что необходимо выполнить следующую строку кода. Если строка содержит вызов функции, команда **Шаг с заходом** выполняет только сам вызов, а затем останавливает выполнение в первой строке кода внутри функции.

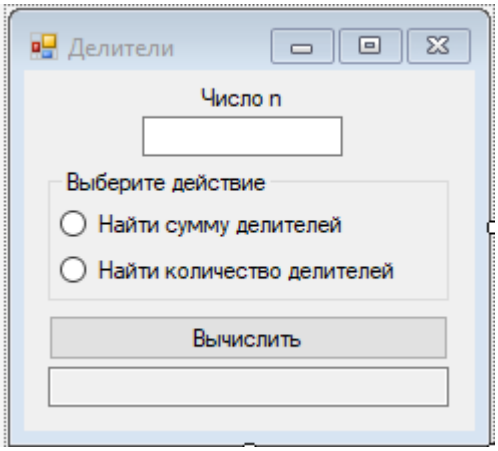
Команда **Шаг с обходом** выполняет целую функцию, а затем останавливает выполнение в первой строке, расположенной вне функции.

Команду **Шаг с заходом** следует использовать, если требуется заглянуть внутрь вызова функции.

Команду **Шаг с обходом** следует использовать, если требуется избежать попадания внутрь функции.

САМОСТОЯТЕЛЬНАЯ РАБОТА

1. Создать приложение для расчета характеристик вектора: длина вектора, произведение векторов. Выбор характеристики осуществить с помощью компонента: 1) radioButton; 2) checkBox.
2. Создать оконное приложение, позволяющее менять цвет формы. Выбор цвета осуществить с помощью: 1) radioButton; 2) введя цвет в поле textBox. Для задания цвета применить команду: `this.BackColor = Color.Red.`
3. Создать оконное приложение, позволяющее для натурального числа n, введенного в поле textBox, выполнить действие, которое можно выбрать с помощью компонента: 1) radioButton; 2) checkBox.



4. Создать оконное приложение, позволяющее для формы настроить выбор цвета, размер шрифта, цвет шрифта, активность компонента textBox. Выполнить действие с помощью компонента checkBox.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 5

ОПЕРАТОРЫ ЯЗЫКА C#

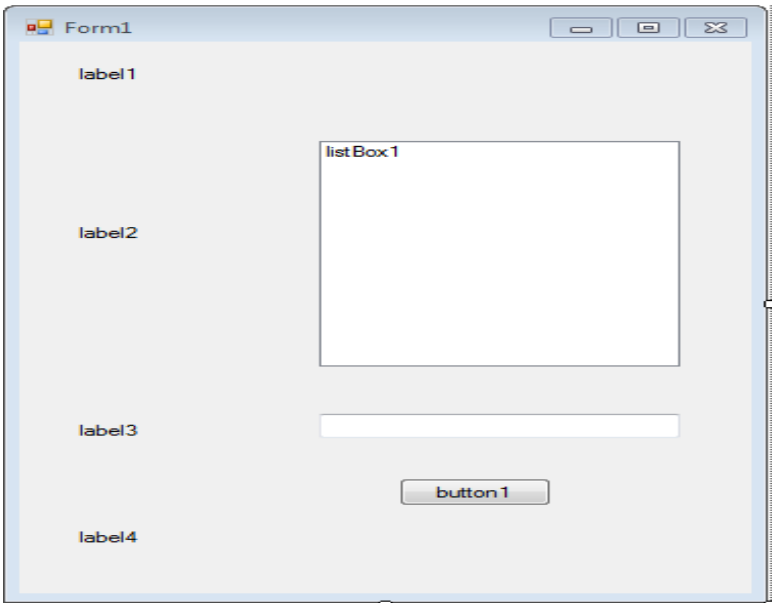
Цель занятия: формирование навыка работы со средой программирования MS Visual Studio 2015 при разработке приложений с применением компонентов listBox и comboBox, с применением оператора выбора.

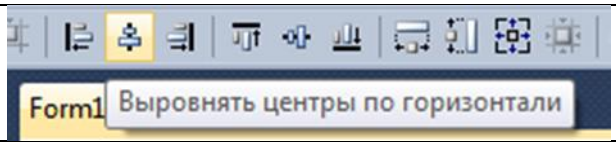
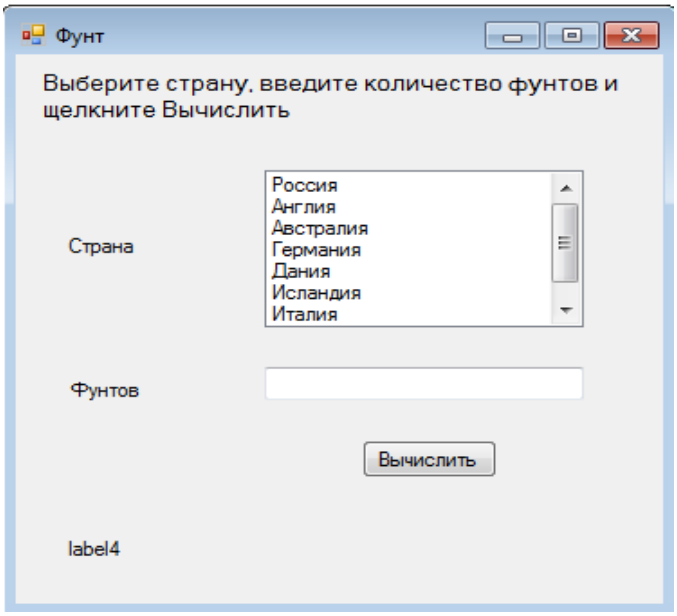
ХОД РАБОТЫ

Задание 1. Написать программу, которая пересчитывает вес из фунтов в килограммы с учетом того, что в разных странах фунт «весит» по-разному.

Россия	*0,4059*
Англия	*0,453592*
Австралия	*0,56001*
Германия	*0,5*
Дания	*0,5*
Исландия	*0,5*
Италия	*0,31762*
Нидерланды	*0,5*

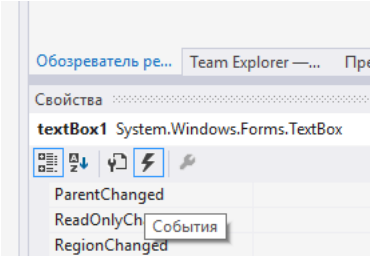
Алгоритм создания приложения:

1. Создайте папку для файлов проекта «Фунт».	А) Файл/Создать/Проект… Visual C# - Приложение Windows Forms Б) В окне «Создать проект» задаем имя «Фунт», выбираем место сохранения проекта В) Нажать кнопку Ok
2. Студенты самостоятельно переименовывают форму (Form1 ➔ «Фунт»), размещают на форме компоненты (textBox1, label1, label2, label3, label4, button1, listBox1) как показано на рисунке:  	
3. Расположенные объекты выровнять по горизонтали.	При нажатой клавише SHIFT выделяем компоненты и в строке меню выбираем команду выравнивания:

<table>
<tr><td></td><td> 
</td></tr>
<tr><td colspan="2">4. Добавить недостающие компоненты на форму, чтобы она приняла следующий вид:
 
</td></tr>
<tr><td>5. Заполнить listBox</td><td>Свойства → Items → щелкнуть на кнопке редактора, на которой изображены три точки. В появившемся диалоговом окне Коллекции следует набрать список, поместив каждый элемент списка на отдельной строке. Ввод очередного элемента списка должен заканчиваться нажатием клавиши Enter.</td></tr>
<tr><td>6. Запрограммировать кнопку «Вычислить». Для вывода применить форматирование</td><td><pre>
private void button1_Click(object sender,
EventArgs e)
{
   double funt, kg, k = 0;
   switch (listBox1.SelectedIndex)
     {
       case 0: k = 0.4059; break;
       case 1: k = 0.453592; break;
       case 2: k = 0.056001; break;
       case 3: k = 0.5; break;
       case 4: case 5: case 7: k = 0.5;break;
       case 6: k = 0.31762; break;
      }
      funt = Convert.ToDouble(textBox1.Text);
      kg = k * funt;
  label4.Text = textBox1.Text + " фунт(а/ов) -
это " + String.Format("{0:0.00}",
Convert.ToString(kg)) + " кг";
}
</pre></td></tr>
<tr><td>7. Откомпилируйте и запустите приложение на исполнение</td><td>а) Тестируем, указывая операнд.
б) Не задаем значение операнда.
<u>Замечание:</u> В случае б) выводится окно сообщения о некорректности значения операнда</td></tr>
<tr><td>8. Отредактировать текст модуля таким образом, чтобы перед выполнением вычислений</td><td>Помещаем перед оператором присваивания
<code>funt = Convert.ToDouble(textBox1.Text);</code>
Строку</td></tr>
</table>

<table>
<tr><td>выполнялась проверка, заданы ли значения операндов.
Если значения не заданы, то следует вывести сообщение об этом в отдельном окне.</td><td>

```
if (textBox1.Text != "")
{
…код программы…
}
else  MessageBox.Show("Не заданы значения");
```

</td></tr>
<tr><td colspan="2">9. Сохранить, откомпилировать и запустить приложение на исполнение.
Проверить работу для случая, когда:
а) не заданы значения;
б) введены, не цифры, а другие символы.</td></tr>
<tr><td>10. Создать обработчик события, запрещающий ввод любых символов, крое цифр от 0 до 9 и знаков + и -.</td><td>В окне свойств компонента textBox1 нажимаем значок События
 

и выбираем KeyPress, делаем справа двойной щелчок левой кнопкой мыши.
После этого окно редактора немедленно получит фокус и в разделе interface появится запись процедуры обработчика события:

```
private void textBox1_KeyPress(object sender,
KeyPressEventArgs e)
        {

        }
```

В тело процедуры вставляем следующий оператор:

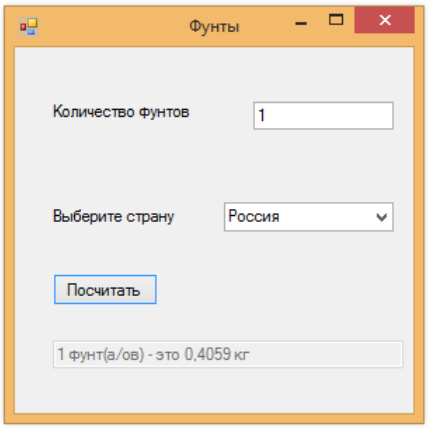
```
if (!char.IsControl(e.KeyChar) &&
(!char.IsDigit(e.KeyChar)) && (e.KeyChar != '-'))
e.Handled = true;
```

</td></tr>
<tr><td>11. Модифицировать обработчик события так, чтобы знаки + и – воспринимались только вначале числа</td><td>

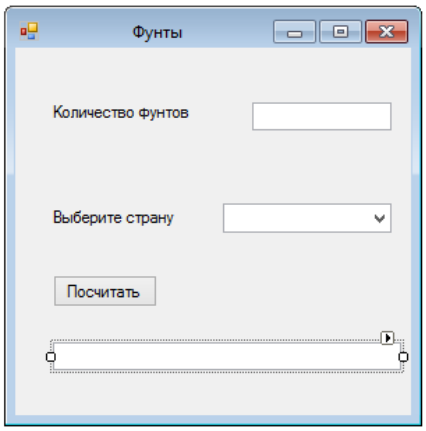
```
Заменяем предыдущий оператор новым:
if (e.KeyChar == '-' && (sender as
TextBox).Text.Length > 0)
e.Handled = true;
```

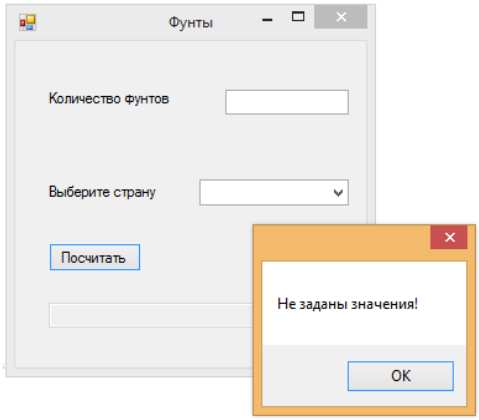
</td></tr>
<tr><td>12. Сохранить, откомпилировать и запустить приложение на исполнение.</td><td>Вводим в качестве операндов не цифры, а другие символы.</td></tr>
</table>

Задание 2. Написать программу, которая пересчитывает вес из фунтов в килограммы, применяя компонент comboBox. Интерфейс приложения представлен ниже.



Алгоритм создания приложения:

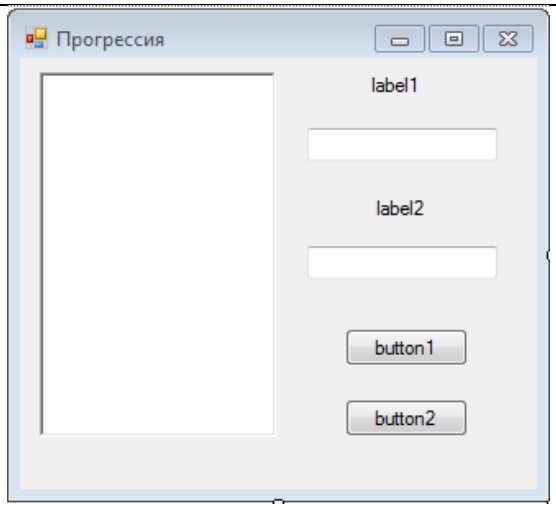
1. Создать папку для файлов проекта «Фунт2».	
2. Переименовать форму (Form1 → «Фунты»), разместить на форме компоненты (textBox1, textBox2, label1, label2, button1, comboBox1)	
3. Форма должна иметь следующий вид: 	
4. Заполнить comboBox1	comboBox1 → Items → щелкнуть на кнопке редактора, на которой изображены три точки правее Коллекции. В появившемся диалоговом окне следует набрать список, поместив каждый элемент списка на отдельной строке. Ввод очередного элемента списка должен заканчиваться нажатием клавиши Enter.
5. Настроить возможность в поле textBox2 только отображать данные	textBox2 → Enabled → False
6. Запрограммировать кнопку «Вычислить»	private void button1_Click(object sender, EventArgs e) { double funt, kg, k=0; funt= Convert.ToDouble(textBox1.Text); switch (comboBox1.SelectedIndex) { case 0: k = 0.4059; break; case 1: k = 0.453592; break; case 2: k = 0.56001; break;

<table>
<tr><td></td><td>case 3: k = 0.5; break;
case 4: k = 0.5; break;
case 5: k = 0.5; break;
case 6: k = 0.31762; break;
case 7: k = 0.5; break;
}
kg = k * funt;
textBox2.Text = textBox1.Text + " фунт(а/ов) - это " + Convert.ToString(kg) + " кг";
}</td></tr>
<tr><td>7. Откомпилируйте и запустите приложение на исполнение</td><td>а) Тестируем, указывая, операнд.
б) Не задаем значение операнда.
<u>Замечание:</u> В случае б) запускается отладка</td></tr>
<tr><td>8. Отредактировать текст модуля таким образом, чтобы перед выполнением вычислений выполнялась проверка, заданы ли значения операндов.</td><td>Если значения не заданы, то следует вывести сообщение об этом в отдельном окне.
 
</td></tr>
<tr><td colspan="2">9. Сохранить, откомпилировать и запустить приложение на исполнение.
Проверить работу для случая, когда:
а) не заданы значения;
б) введены, не цифры, а другие символы.</td></tr>
<tr><td colspan="2">10. Создать обработчик события, запрещающего ввод любых символов</td></tr>
</table>

Задание 3. Создать оконное приложение, вычисляющее первые 20 элементов арифметической прогрессии, для которой заданы первый элемент и разность.

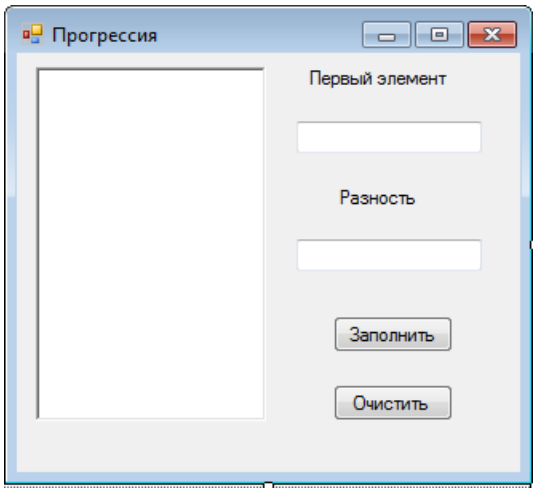
Алгоритм создания приложения:

<table>
<tr><td>1. Создайте папку для файлов проекта «Прогрессия».</td><td>А) Файл/Создать/Проект…
Visual C# - Приложение Windows Forms
Б) В окне «Создать проект» задаем имя «Прогрессия», выбираем место сохранения проекта
В) Нажать кнопку Ok</td></tr>
<tr><td colspan="2">2. Студенты самостоятельно переименовывают форму (Form1 → «Прогрессия»), размещают на форме компоненты (listBox1, textBox1, textBox2, label1, label2, button1, button2) как показано на рисунке:</td></tr>
</table>



3. Расположенные объекты выровнять по горизонтали.

4. Добавить недостающие компоненты на форму, чтобы она приняла следующий вид:



5. Запрограммировать кнопку «Заполнить»

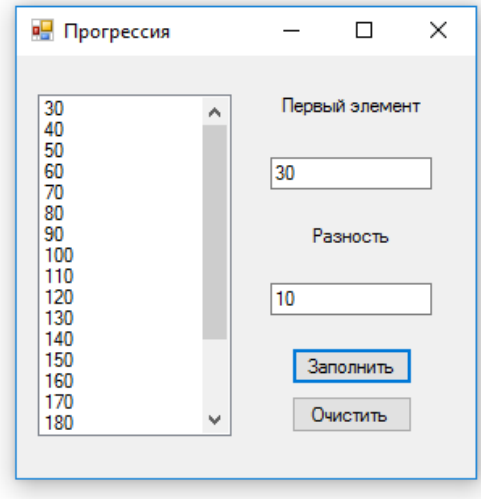
```
private void button1_Click(object sender,
EventArgs e)
{
   int a, d, i, an;
   a = Convert.ToInt32(textBox1.Text);
   d = Convert.ToInt32(textBox2.Text);
   an = a;
   listBox1.Items.Add(a);
   for (i = 1; i <= 20; i++)
            {
                an = an + d;
                listBox1.Items.Add(an);
            }
}
```

6. Запрограммировать кнопку «Очистить»

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

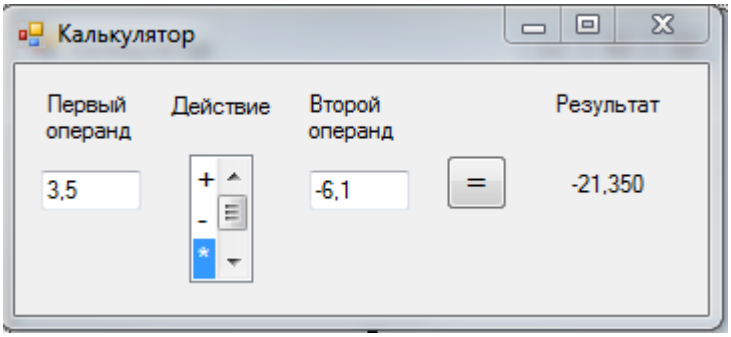
<table>
<tr><td>7. Откомпилируйте и запустите приложение на исполнение</td><td>а) Тестируем, указывая, два операнда.
б) Не задаем значение операндов.
<u>Замечание:</u> В случае б) выводится окно сообщения о некорректности значения операнда</td></tr>
<tr><td>8. Отредактировать текст модуля таким образом, чтобы перед выполнением вычислений выполнялась проверка, заданы ли значения операндов.
Если значения не заданы, то следует вывести сообщение об этом в отдельном окне.</td><td>Помещаем перед оператором присваивания
A= StrToInt (Edit1.Text);
строку
<code>if (textBox1.Text != "" && textBox2.Text != "")</code>
остальной код выделяем фигурными скобками
{…}
А перед последним оператором END прописываем следующее:
<code>else
MessageBox.Show("Не заданы значения!");</code></td></tr>
<tr><td colspan="2">9. Сохранить, откомпилировать и запустить приложение на исполнение.
Проверить работу для случая, когда:
а) не заданы значения;
б) введены, не цифры, а другие символы.</td></tr>
<tr><td>10. Создать обработчик события, запрещающего ввод любых символов, кроме цифр от 0 до 9 и знаков + и -.</td><td>См. задачу №1</td></tr>
<tr><td>11. Модифицировать обработчик события так, чтобы знаки + и – воспринимались только вначале числа</td><td>См. задачу №1</td></tr>
<tr><td>12. Сохранить, откомпилировать и запустить приложение на исполнение.</td><td>Вводим в качестве операндов не цифры, а другие символы.</td></tr>
</table>

Приложение имеет вид:



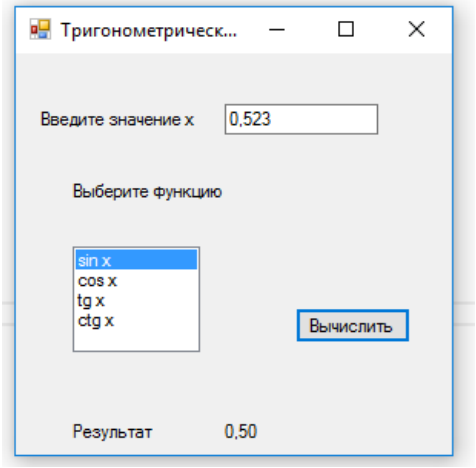
САМОСТОЯТЕЛЬНАЯ РАБОТА

1. Написать программу, выполняющую работу арифметического калькулятора с четырьмя арифметическими действиями над действительными числами. Окно работающего приложения:

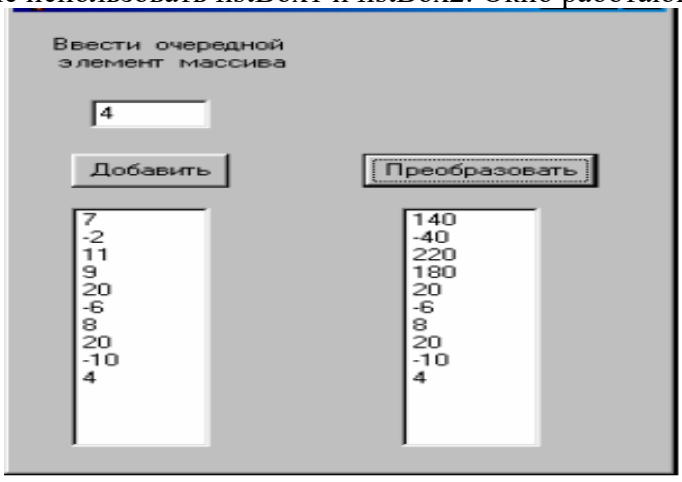


2. Написать программу для вычисления тригонометрической функции (sin x, cos x, tg x, ctg x) для данного действительного числа x. решить задачу двумя способами: с применением listBox и comboBox.

Окно работающего приложения:



3. Даны *a* элементов. Все элементы этой последовательности, предшествующие максимальному элементу, умножить на этот максимальный элемент. Для отображения элементов в форме использовать listBox1 и listBox2. Окно работающего приложения:



4. Заполнить список listBox числами, введенными с клавиатуры в поле textbox. Получить новый список, удвоив нечетные элементы первого списка, а четные оставить без изменения. Результат отобразить в поле listBox2.
5. Заполнить список a числами, введенными с клавиатуры. Получить новый список b, каждый элемент которого равен сумме цифр соответствующего элемента списка a. Полученный список b отобразить в поле listBox2.

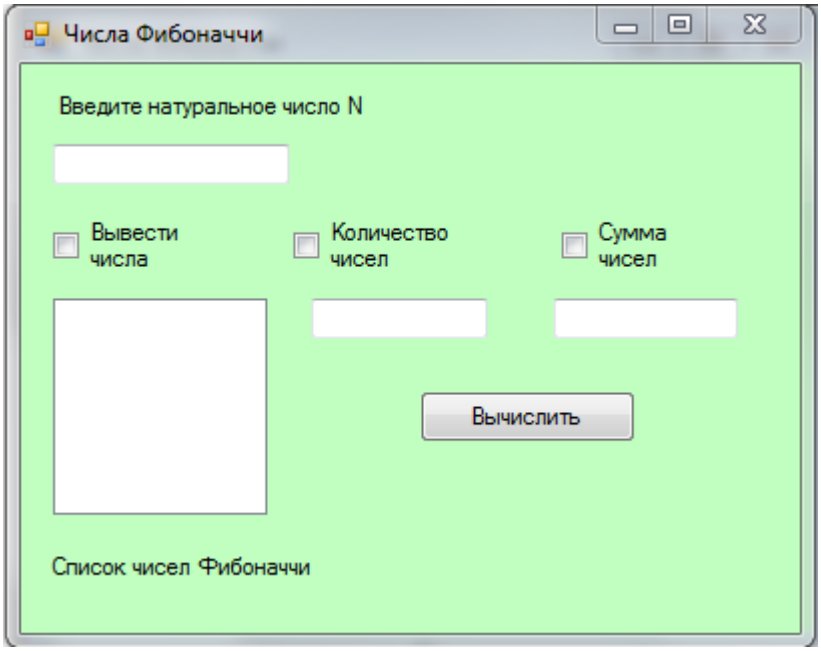
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 6

ОПЕРАТОРЫ ЯЗЫКА C#

Цель занятия: формирование навыка работы со средой программирования Visual Studio при разработке приложений с применением компонентов checkbox, listBox, с применением оператора цикла.

ХОД РАБОТЫ

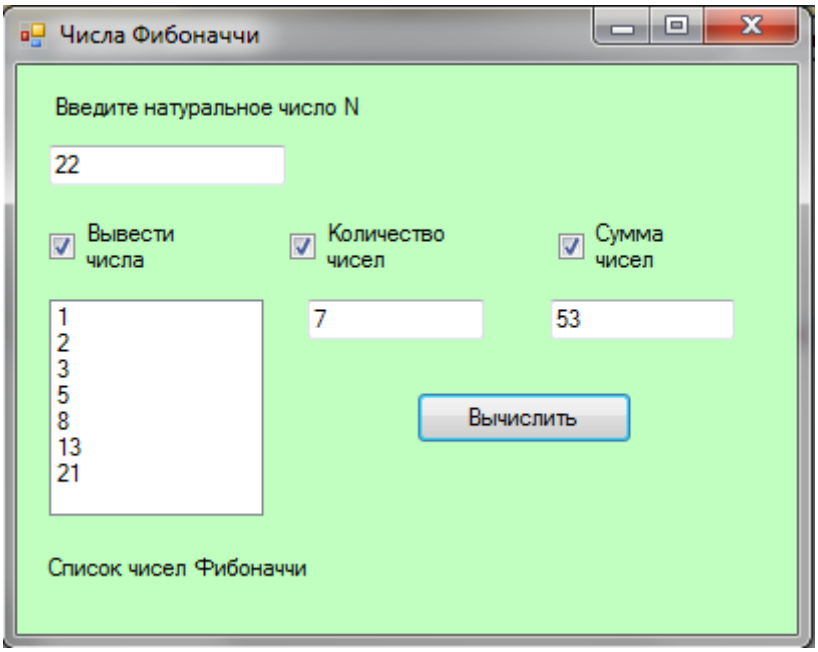
Задание 1. Разработать приложение, позволяющее по заданному натуральному числу N находить числа Фибоначчи, не превышающие данного N, а также определяло бы количество найденных чисел и их сумму (по требованию). Список чисел вывести в компоненте listBox.



Алгоритм создания приложения:

1. Самостоятельно создать форму с необходимыми на ней компонентами, изменить свойства объектов в соответствии с рисунком. Организовать запрет ввода в поле для натурального числа N любого текста, кроме целых чисел.	
2. Если пользователь введет единицу, то в окне должно появиться сообщение «Чисел Фибоначчи до 1 не существует».	
3. Обращение к компоненту CheckBox организовать можно так:	if (checkBox1.Checked) { for (i = 0; i <= k-1; i++) listBox1.Items.Add(mas[i]); } или так: if (checkBox1.CheckState == CheckState.Checked) { for (i = 0; i <= k-1; i++) listBox1.Items.Add(mas[i]); }
4. Откомпилируйте программу (Отладка/Начать отладку). Проверьте работу приложения.	

Приложение имеет следующий вид:



Программный код:

```
private void textBox1_KeyPress(object sender, KeyPressEventArgs e)
  {
    if (!char.IsDigit(e.KeyChar) == true && e.KeyChar != 8)
            {
                e.Handled = true;
            }
  }
int n;
private void button1_Click(object sender, EventArgs e)
{
     listBox1.Items.Clear();
     n = int.Parse(textBox1.Text);
     int k = 2;
     int sum = 3;
     int i = 2;
     if (textBox1.Text != "1")
       {
          int[] mas = new int[n];
          mas[0] = 1; mas[1] = 2;
          do
            {
              mas[i] = mas[i - 2] + mas[i - 1];
              if (mas[i] < n)
                {
                   sum += mas[i]; k++; i++;
                }
               else break;
            }
           while (mas[i-1] <= n);

            if (checkBox1.Checked)
              {
                 for (i = 0; i <= k-1; i++) listBox1.Items.Add(mas[i]);
              }
            if (checkBox2.Checked)
              {
```

```
                textBox2.Text = Convert.ToString(k);
              }
            if (checkBox3.Checked)
              {
                textBox3.Text = Convert.ToString(sum);
              }
        }
        else
            {
               MessageBox.Show("Чисел Фибоначчи до 1 не существует");
               textBox1.Clear();
               textBox1.Focus();
            }
}
```

САМОСТОЯТЕЛЬНАЯ РАБОТА

1. Даны 2 вектора с координатами (x1,y1) и (x2,y2). Создать приложение для расчета длины вектора, суммы векторов, скалярного произведения векторов. Выбор характеристик осуществить с помощью компонента listBox.
2. Даны 2 вектора с координатами (x1,y1) и (x2,y2). Создать приложение для расчета длины вектора, суммы векторов, скалярного произведения векторов. Выбор характеристик осуществить с помощью компонента checkBox.
3. Найти все трёхзначные числа, каждое из которых удовлетворяет условию: сумма кубов равняется самому числу. Найденные числа отобразить в поле listBox1, подсчитать и вывести их количество.
4. Дан одномерный целочисленный массив из 20 элементов. Ввести его элементы с клавиатуры и удалить последний максимальный элемент массива.
5. Дан одномерный целочисленный массив a из 20 элементов. Ввести его элементы с клавиатуры, а затем переставить элементы массива a так, чтобы они расположились в следующем порядке a1, a11, a2, a12, . . . ,a10, a20.

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Каково назначение визуального компонента checkBox? Каким способом можно запрограммировать состояние компонента checkBox?
2. Как задать список элементов в объекте listBox?
3. Как запретить ввод символов в textbox?
4. Как запретить ввод недопустимых символов в textbox?

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 7

ОПЕРАТОРЫ ЯЗЫКА C#

Цель занятия: формирование навыка работы со средой программирования Visual Studio при разработке приложений с применением компонента richTextBox; обработка нескольких форм; применение условного оператора.

ХОД РАБОТЫ

Задание 1. Разработать приложение «Денежный калькулятор».

Программа должна выполнять функции специализированного калькулятора по пересчету денежных сумм из рублей в доллары США и обратно. Предусмотреть возможность ввода нового курса доллара в соответствии с информацией ЦБ РФ. Встроить в проект справочную информацию. Рекомендуемый интерфейс программы изображён на рисунке 1.

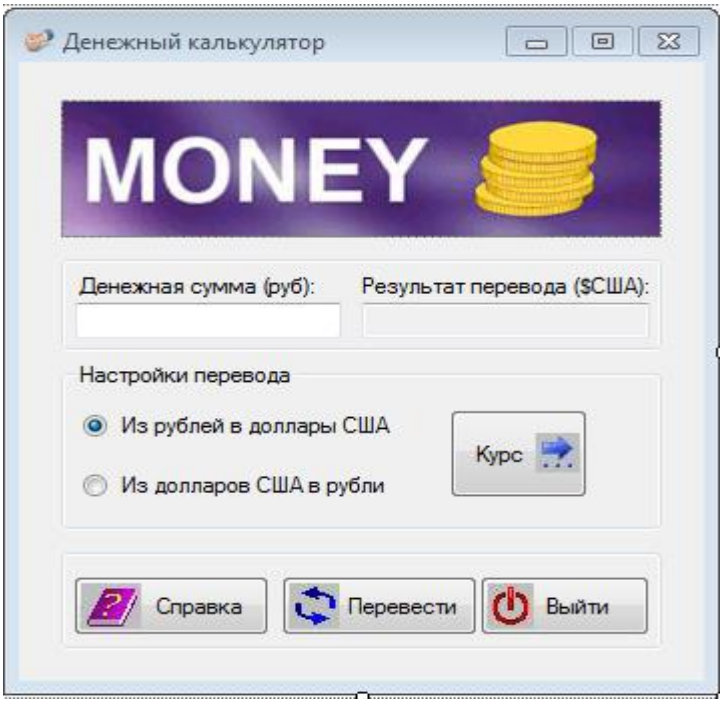


Рисунок 1 – Интерфейс приложения

Разработка интерфейса

1. Создайте проект с именем money. Измените содержимое Строки заголовка в окне Формы с «Form1» на «Денежный калькулятор». Стадии разработки интерфейса (по шагам) наглядно показаны на рисунке 2.	
Примечание: при необходимости сделайте ручную подгонку размеров остальных объектов Формы, ориентируясь при этом на рисунок 1.	
2. Заблокируйте действие кнопки «Развернуть», находящейся в строке заголовка формы «Денежный калькулятор».	свойство MaximizeBox принимает значение False
3. Вставьте рисунок.	Добавьте компонент PictureBox. Растяните внутри формы прямоугольную область, отводимую под рисунок. В свойстве Image задайте путь и имя файла с рисунком.

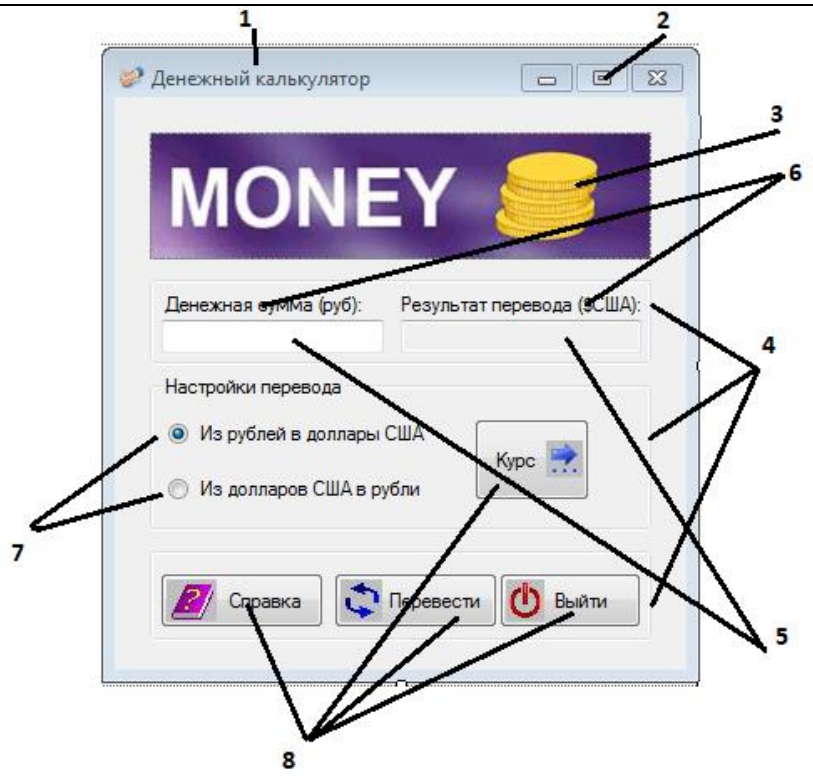


Рисунок 2 - Стадии разработки интерфейса

4. С помощью компонента GroupBox изобразите три декоративные рамки (см. рисунок 2, шаг 8°), каждая из которых будет охватывать свою группу объектов.	Для верхней и нижней очистите свойство Text, а для средней задайте этому свойству значение «Настройка перевода».
5. В соответствии с рисунком 2, используя компонент textBox, разместите на Форме «Денежный калькулятор» два текстовых поля: textBox1 и textBox2.	Чтобы заблокировать возможность ввода с клавиатуры информации в текстовое поле textBox2 (используемое для вывода результата расчета), придайте его свойству ReadOnly значение True.
6. В соответствии с рисунком 2, разместите на Форме «Денежный калькулятор» надписи «Денежная сумма (руб):» и «Результат перевода ($США):» с применением label1 и label2, соответственно.	
7. В соответствии с рисунком 2, используя компонент radioButton, разместите на Форме «Денежный калькулятор» два переключателя режимов: radioButton1 и RadioButton2.	Для первого из них задайте текст: «Из рублей в доллары США». Для второго – «Из долларов США в рубли». Первый переключатель переведите в положение «Включен». Для этого задайте свойству Checked значение True.
8. а) разместите на Форме кнопки управления. Создайте четыре кнопки, как показано на рисунке 2. б) добавьте на этих кнопках рисунки. в) расположите рисунки кнопок слева от текста надписи	а) button1 - «Выйти», button2 – «Перевести», button3 – «Справка», button4 – «Курс». б) в свойстве Image для каждой из кнопок задайте имя её картинки, находящейся по адресу, указанному преподавателем. Для button1 – exit.bmp, для button2 – replace.bmp, для button3 – help.bmp, а для button4 – rate.bmp. в) в свойстве ImageAlign выберите положение рисунка – MiddleLeft, а для Button4 – MiddleRight.

Интерфейс первой Формы завершен. Обратите внимание на то, что на кнопке <Курс> изображена стрелка с многоточием. Это означает, что при щелчке мышью по этой кнопке должно открыться одноимённое диалоговое окно.

Опишем его интерфейс во второй Форме. Пусть он будет выглядеть так, как на рисунке 3.

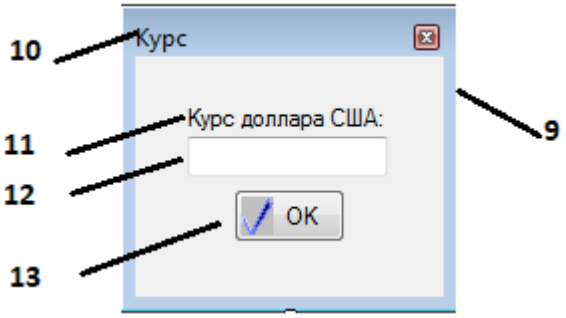


Рисунок 3 – Интерфейс окна Курс

9. Добавить вторую форму	Для создания второй Формы (Form2) задайте команду – Проект/ Добавить форму Windows/Форма/ Windows Forms. Сделайте это окно диалоговым. Для этого свойству FormBorderStyle присвойте значение FixedToolWindow.

10. Создайте с помощью свойства Text заголовок у Формы 2 – «Курс». Имя формы (Name) оставьте Form2.

11. В соответствии с рисунком 3, используя компонент label, разместите на Форме «Курс» надпись «Курс доллара США: »

12. В соответствии с рисунком 3, используя компонент textBox, разместите на Форме «Курс» текстовое поле textBox1. Оно понадобится нам для ввода курса доллара.

13. В соответствии с рисунком 3, используя компонент Button, разместите на Форме «Курс» кнопку button1. Привяжите к этой кнопке рисунок.

Интерфейс второй Формы завершен. Нам потребуется создать ещё и третью Форму для формирования справочного окна «О программе».

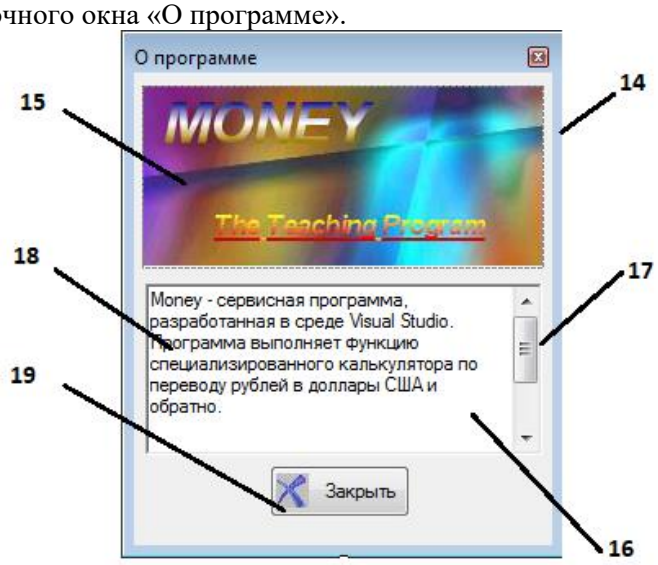


Рисунок 4 – Справочное окно

14. Создайте третью Форму (Form3) такой же, как и Form2, т.е. в виде диалогового окна.

15. С помощью компонента pictureBox разместите на Форме картинку.

16. Добавьте на Форму компонент richTextBox с текстом справки.

17. Дополните объект richTextBox1 вертикальной полосой прокрутки.	Для этого в окне свойств у scrollBars установите значение Vertical.
18. Заполните richTextBox1 текстовой информацией.	Сделайте двойной щелчок по параметру свойства Text или одиночный щелчок по знаку «∨» и в открывшемся окне можно ввести текст вручную с клавиатуры.

19. Создайте кнопку для закрытия окна – button1 с рисунком находящимся в папке, указанной преподавателем.

Программирование событий

1. Сделайте активной Форму 1 (Form1) и перейдите к коду:
2. Опишите событие, связанное с щелчком по кнопке <Выйти>, находящейся на Form1.

```
private void button1_Click(object sender, EventArgs e)
    {
      Application.Exit();
    }
```

3. Опишите события, связанные со щелчком мыши по переключателям radioButton1 или radioButton2:

```
   byte x = 0;
    private void radioButton1_CheckedChanged(object sender, EventArgs e)
    {
      x = 0;
      label1.Text = "Денежная сумма (руб)";
      label2.Text = "Результат перевода ($США)";
    }

    private void radioButton2_CheckedChanged(object sender, EventArgs e)
    {
      x = 1;
      label1.Text = "Денежная сумма ($США)";
      label2.Text = "Результат перевода (руб)";
    }
```

4. Опишите событие, связанное со щелчком мыши по кнопке <Справка>. Эта кнопка, находящаяся на Form1, имеет имя Button2.

```
  Form3 f3 = new Form3();
    private void button4_Click(object sender, EventArgs e)
    {
      f3.Show();
    }
```

5. Опишите событие, связанное со щелчком по самой главной кнопке – <Перевести> (Button3). Именно с помощью этой кнопки выполняется решение поставленной задачи:

```
      public double k;
      public void button3_Click(object sender, EventArgs e)
    {
      double b = 0;
         if (f2.k != 0)
        {
          k = f2.k;
        }
        double a = Convert.ToDouble(textBox1.Text);
```

```
// Если перевод из рублей в доллары
if (x = = 0)
{
   b = a / k;
}

// Если перевод из долларов в рубли
if (x = = 1)
{
      b = a * k;
}
string s = String.Format("{0:0.00}", b);
textBox2.Text = s;
}
```

6. Опишите событие, связанное со щелчком по кнопке <Закрыть> (Button1) на Форме3:

```
public void button1_Click(object sender, EventArgs e)
    {
      this.Close();
    }
```

7. Опишите событие, связанное со щелчком по кнопке <Курс> (Button2) на Форме 1:

```
Form2 f2 = new Form2();
 public void button2_Click(object sender, EventArgs e)
 {
    f2.Show();
 }
```

8. Опишите событие, связанное со щелчком по кнопке <OK> (Button1) на Форме 2:

```
public double k;
 private void button1_Click(object sender, EventArgs e)
  {
    k = Convert.ToDouble(textBox1.Text);
    this.Visible = false;
  }
```

Сохраните проект и запустите на исполнение.

САМОСТОЯТЕЛЬНАЯ РАБОТА

1. В приложении Калькулятор добавить кнопку «Очистить».
2. Создать приложение, позволяющее вычислять площадь геометрической фигуры: квадрат, прямоугольник, прямоугольный треугольник. Выбор фигуры осуществить с помощью компонента radioButton. После выбора фигуры открывается вторая форма, на которой находятся поля для ввода данных. Количество полей должно соответствовать числу вводимых значений.

ВОПРОСЫ ДЛЯ КОНТРОЛЯ

3. Как заблокировать кнопку «Развернуть» в строке заголовка приложения?
4. Как изобразить прямоугольную кювету в окне Формы?
5. Как вставить рисунок в окно Формы?

6. Можно ли доработать рисунок, предназначенный для размещения на окне Формы, с помощью какого-нибудь графического редактора?
7. Можно ли украсить окно разрабатываемого приложения какой-нибудь фотографией?
8. Как добиться того, чтобы вставляемый в окно Формы рисунок совпал по своим геометрическим размерам с полем, которое вы для него подготовили?
9. Как изобразить декоративную рамку в окне формы?
10. Как изобразить в окне Формы поле для ввода текстовой информации?
11. Как изобразить надпись в окне Формы?
12. Может ли надпись состоять из нескольких строк?
13. Как изобразить в окне Формы переключатели режимов?
14. Как изобразить в окне Формы кнопку с картинкой и надписью?
15. Возможно ли такое, чтобы приложение создавалось из нескольких Форм?
16. Чем принципиально отличается диалоговое окно от окна приложения?
17. С помощью какого компонента можно разместить на поле Формы многострочный текст?
18. Как вставить в окно с многострочным текстом полосу прокрутки этого текста?
19. Каково назначение визуального компонента groupBox? Как задать надпись в этом компоненте?
20. Чем отличаются компоненты radioButton и groupBox? Как задать список элементов в объекте radioButton?
21. Для чего и каким образом выполняется фиксация положения компонентов на форме?

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 8
МАССИВЫ

Цель занятия: формирование навыка работы со средой программирования Visual Studio при разработке приложений по обработке линейных массивов с применением компонентов textbox, richTextBox, listBox.

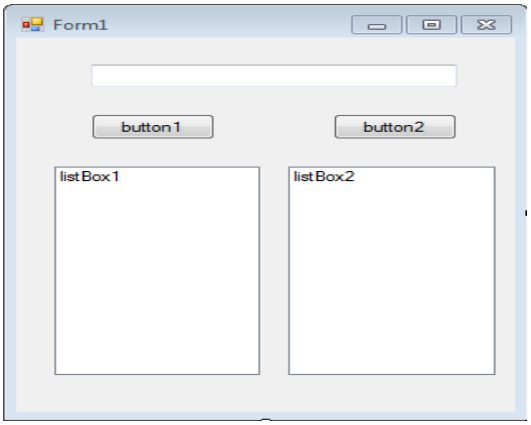
ХОД РАБОТЫ

Задание 1. В целочисленном массиве из 10 элементов (все элементы различны) найти максимальный и минимальный элементы и поменять их местами. (Для отображения массивов в форме использовать компонент listBox).

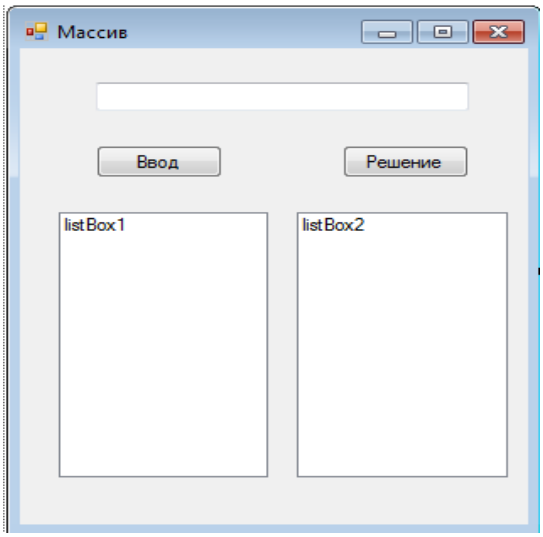
Алгоритм создания приложения:

1. Создайте папку для файлов проекта «Массив».	А) Файл/Создать/Проект… Visual C# - Приложение Windows Forms Б) В окне «Создать проект» задаем имя «Массив», выбираем место сохранения проекта В) Нажать кнопку Ok

2. Студенты самостоятельно переименовывают форму (Form1 ➔ «Массив»), размещают на форме компоненты (textBox1, button1, button2, listBox1, listBox2) как показано на рисунке:



3. Форма должна принять следующий вид:



4. Для ввода целых чисел используется поле редактирования textBox1. Ввод каждого числа

<table>
<tr><td colspan="2">завершается щелчком мыши по кнопке Ввод. Введенный массив отображается в поле listBox1. После щелчка по кнопке «Решение» преобразованный массив отображается в поле listBox2.</td></tr>
<tr><td>5. Вставим описание массива и используемых переменных.</td><td>После
public partial class Form1 : Form
{
пишем:
int[] a = new int[10];
int min, max, imin, imax;</td></tr>
<tr><td>6. Зададим начальное значение индекса массива.</td><td>int i = 0;</td></tr>
<tr><td>11. Запрограммировать кнопку «Ввод»</td><td>Двойной щелчок по кнопке «Ввод»: создаем обработчик нажатия клавиши. В открывшемся редакторе кода пишем:
private void button1_Click(object sender, EventArgs e)
{
listBox1.Items.Add(textBox1.Text);
a[i] = int.Parse(textBox1.Text);
i++;
textBox1.Clear();
textBox1.Focus();
}</td></tr>
<tr><td colspan="2"><u>Примечание:</u> Функция Add формирует список listBox1, заполняя его числами, вводимыми в поле textBox1. Одновременно этими же числами заполняется массив a. Процедура Focus устанавливает фокус ввода в поле textBox1.</td></tr>
<tr><td>12. Запрограммировать кнопку «Решение»</td><td>private void button2_Click(object sender, EventArgs e)
{
max = a[0]; imax = 0;
min = a[0]; imin = 0;
for (int k = 1; k < 10; k++)
{
if (max < a[k])
{
max = a[k];
imax = k;
}
if (min > a[k])
{
min = a[k];
imin = k;
}
}
a[imax] = min;
a[imin] = max;
for (int k = 0; k < 10; k++)
{ listBox2.Items.Add(Convert.ToString(a[k]));
}
}</td></tr>
<tr><td colspan="2">8. Сохранить, откомпилировать и запустить приложение на исполнение.
Проверить работу для случая, когда:
а) не заданы значения;
б) введены, не цифры, а другие символы.
в) создать обработчик события, запрещающего ввод любых символов, крое цифр от 0 до 9 и знаков + и -.</td></tr>
</table>

Задание 2. Создайте приложение, предлагающее пользователю задать размер линейного массива, заполнить этот массив случайными целыми числами, вывести список

элементов массива, а затем по выбору пользователя определить минимальный и максимальный элементы, сумму всех элементов и количество положительных элементов. Образец оформления приложения представлен на рисунке 1.

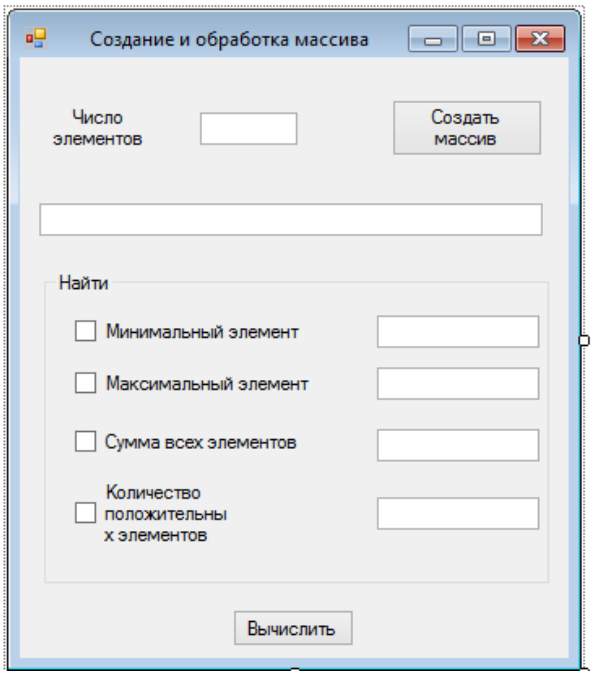


Рисунок 1 - Форма к заданию 2

Алгоритм создания приложения:

1. Самостоятельно создайте верхнюю часть формы с необходимыми на ней компонентами, измените свойства объектов в соответствии с рисунком. Вторая часть формы «Найти» организована с помощью панели groupBox[1]. В панели groupBox разместите компоненты checkBox1, checkBox2, checkBox3, checkBox4. Напротив этих компонентов разместите компоненты textBox3 - textBox6. Выровняйте компоненты на форме и зафиксируйте их положение (ПКМ[2] → Блокировка элементов управления – Lock Controls). **Замечание:** если компоненты checkBox1, checkBox2, checkBox3, checkBox4 окажутся под панелью groupBox1, то надо выделить эту панель и в контекстном меню выбрать «На задний план – Send to Back».	
2. В классе формы запишем:	`int [] ar;` `int N;` где N – это размер массива; M – динамический массив целых чисел.
3. Создайте обработчик события, запрещающего ввод любых символов, кроме цифр от 0	

[1] Панель groupBox – это контейнер с рамкой и надписью, объединяющий группу связанных органов управления, таких как переключатели radioButton, флажки checkBox и др. Эта панель используется для выделения на форме группы функционально объединенных компонентов.

[2] ПКМ – правая кнопка мыши

до 9. Учесть возможность стирания содержимого textBox1 с помощью кнопки BS.

4. В процедуре обработчика события щелчка мышью на кнопке button1 описываем создание массива целых чисел.

```
private void button1_Click(object sender, EventArgs
e)
  {
    N = int.Parse(textBox1.Text);
    textBox2.Text = "";
    ar = new int[N];
    Random random = new Random();
    for (int i = 0; i <= N - 1; i++)
      {
       ar[i] = random.Next(-10, 10);
       textBox2.Text += ar[i] + " ";
      }
        return;
}
```

5. Сохраните проект, откомпилируйте и запустите программу на выполнение.

6. Обработку массива описываем в процедуре обработчика события щелчка мышью на кнопке Button2. В обработчике события добавим локальные переменные:

int max, min, kol, sum;

где max – максимальный элемент массива;

min –минимальный элемент массива;

sum – сумма всех элементов массива;

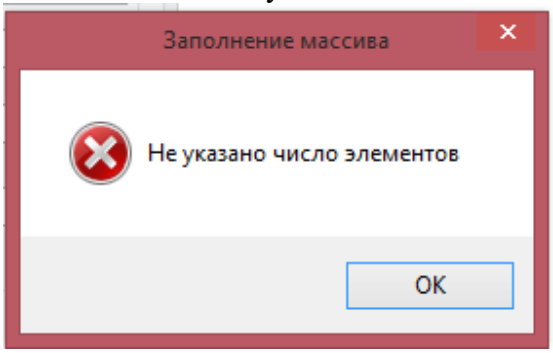
kol – количество положительных элементов массива.

Текст обработки массива может быть записан следующим образом:

```
private void button2_Click(object sender, EventArgs e)
 {
        int max, min, kol, sum;
          min = ar[0];
          max = ar[0];
          sum = 0;
          kol = 0;
          for (int i = 0; i <= N - 1; i++)
       {
        if (checkBox1.Checked)
          {
           if (min > ar[i])
             min = ar[i];
          }
        if (checkBox2.Checked)
          {
           if (max < ar[i])
              max = ar[i];
          }
        if (checkBox3.Checked)
           sum += ar[i];

        if (checkBox4.Checked)
          {
            if (ar[i]>0)
             kol ++;
          }
        }
       if (checkBox1.Checked)
   textBox3.Text = min.ToString();
      if (checkBox2.Checked)  textBox4.Text =
  max.ToString();
    if (checkBox3.Checked)  textBox5.Text =
sum.ToString();
    if (checkBox4.Checked)  textBox6.Text =
kol.ToString();
```

	}

7. Сохраните файлы проекта и программного модуля, откомпилируйте и запустите программу на выполнение. Проверить работу для случая, когда:
а) не заданы значения;
б) вывести сообщение в новом окне «Не указано число элементов»;



в) введены, не цифры, а другие символы;
г) создать обработчик события, разрешающего только ввод положительных чисел.

Задание 3. Создайте приложение, предлагающее пользователю задать размер линейного массива, заполняет этот массив случайными целыми числами, выводит список элементов массива, а затем по выбору пользователя выполняет сортировку массива (по возрастанию, по убыванию) и выводит новый массив. Образец оформления приложения представлен на рисунке 2.

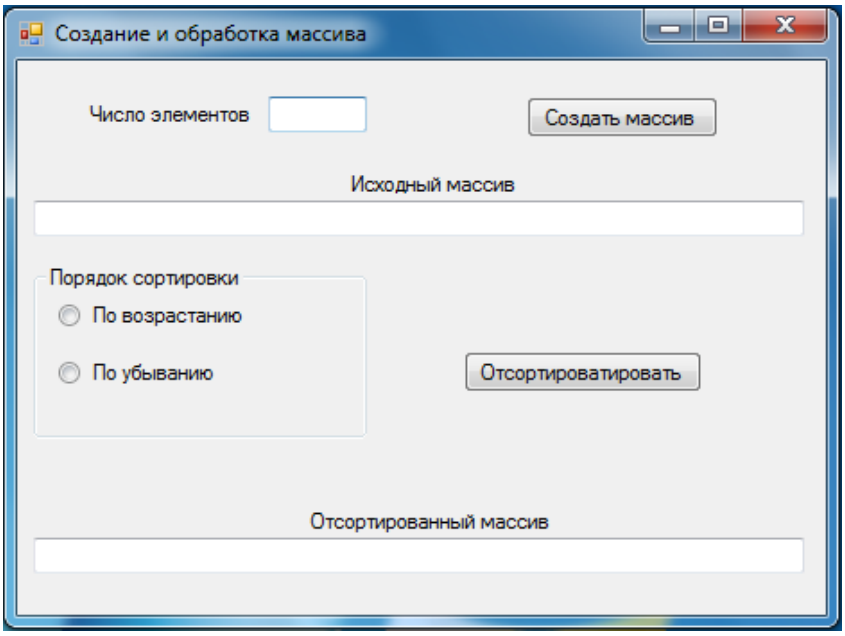


Рисунок 2 - Форма к заданию 3

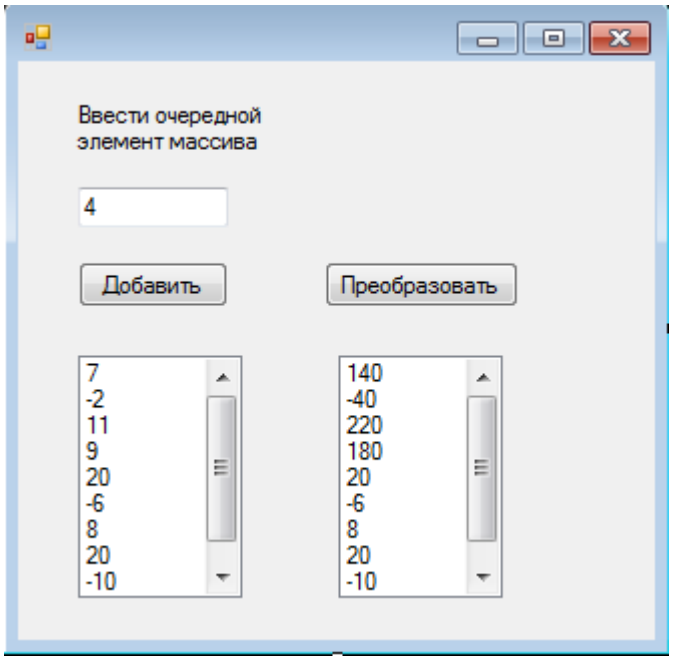
Алгоритм создания приложения:

1. Самостоятельно расположить на форме нужные компоненты в соответствии с рисунком. Изменить свойства формы и компонентов. Зафиксировать положение компонентов на форме.	
2. Сохранение изменений	Файл/Сохранить все
3. Построение решения	Построение/Построить решение
4. Запуск программы и завершение работы приложения	Начать отладку (F5), Alt+F4

5. Расположение окна приложения	Свойства/StartPosition/CenterScreen
6. Создание массива	См. п. 4, задание 1
7. Сохранение изменений	Файл/Сохранить все
8. Обработка события нажатия кнопки Button2 «Отсортировать»	`for (int i = 0; i <= N - 2; i++)` `{` `  for (int j = i+1; j <= N-1 ; j++)` `  {` `  if ((radioButton1.Checked && ar[i] > ar[j]) | (radioButton2.Checked && ar[i] < ar[j]))` `  {` `  int tmp=ar[i];` `  ar[i]=ar[j];` `  ar[j]=tmp;` `  }` `  }` `}`
9. Вывод отсортированного массива	`for (int i = 0; i <= N - 1; i++)` `  {` `  textBox3.Text += ar[i] + " ";` `  }`
10. Сохраните файлы проекта, откомпилируйте и запустите программу на выполнение.	

САМОСТОЯТЕЛЬНАЯ РАБОТА

1. Дан одномерный массив A из n элементов. Все элементы этого массива, предшествующие первому по порядку элементу со значением max(a1, ...,a10), умножить на этот максимальный элемент. Для отображения массивов в форме использовать listBox1 и listBox2. Окно работающего приложения:



2. Заполнить одномерный целочисленный массив A числами, введенными с клавиатуры в поле listBox1. Получить новый массив B, возведя в квадрат четные элементы массива

A, а нечетные оставить без изменения. Полученный массив B отобразить в поле listBox2.

3. В поле listBox1 ввести числа, произвести их анализ. Распределить отрицательные и положительные числа в поля listBox2и listBox3.
4. Изменить предыдущее задание так, чтобы числа можно было вводить в поле textBox1.
5. Дан массив A. Вывести сумму положительных элементов и произведение отрицательных. Выбор осуществить с помощью radioButton.
6. Изменить приложение в № 3 так, чтобы можно было применить компонент checkBox.

ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Опишите отличия статических и динамических массивов.
2. Каков порядок описания и применения динамических массивов?
3. Описание одномерных массивов на языке C#.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 9
МАССИВЫ

Цель занятия: формирование навыка работы со средой программирования Visual Studio при разработке приложений по обработке двумерных массивов с применением компонента dataGridView1.

ТЕОРЕТИЧЕСКИЙ МАТЕРИАЛ

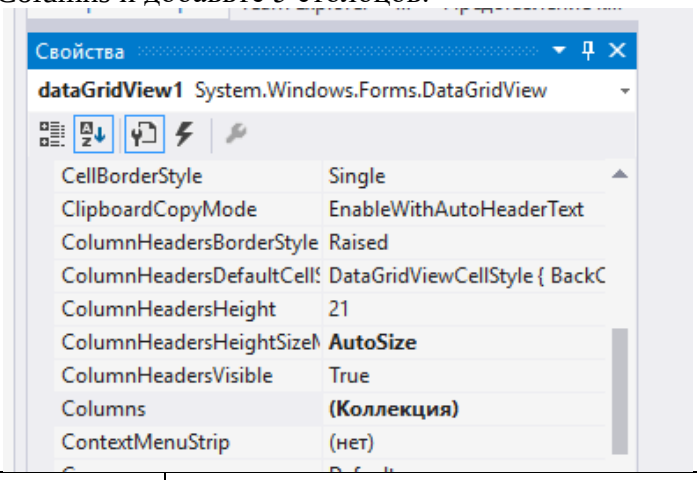
В таблице описаны некоторые свойства компонента dataGridView1.

Свойство	**Обозначение**
Количество колонок таблицы	Columns.Count
Количество строк таблицы	Rows.Count
Соответствующий таблице двумерный сроковый массив (индексы нумерованы от 0)	Rows[i].Cells[j].Value

ХОД РАБОТЫ

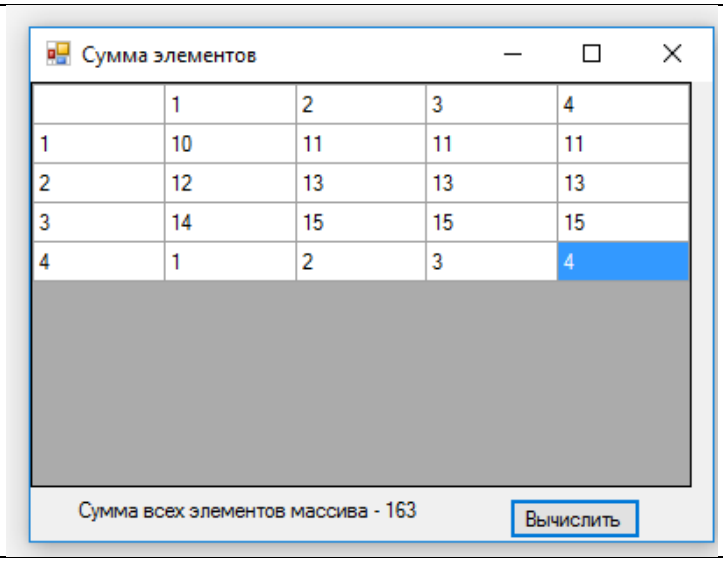
Задание 1. Создайте приложение с компонентом dataGridView1, обеспечивающее пользователю возможность ввода значений элементов двумерного массива в ячейки и выполняющее вычисление суммы элементов массива.

Алгоритм создания приложения:

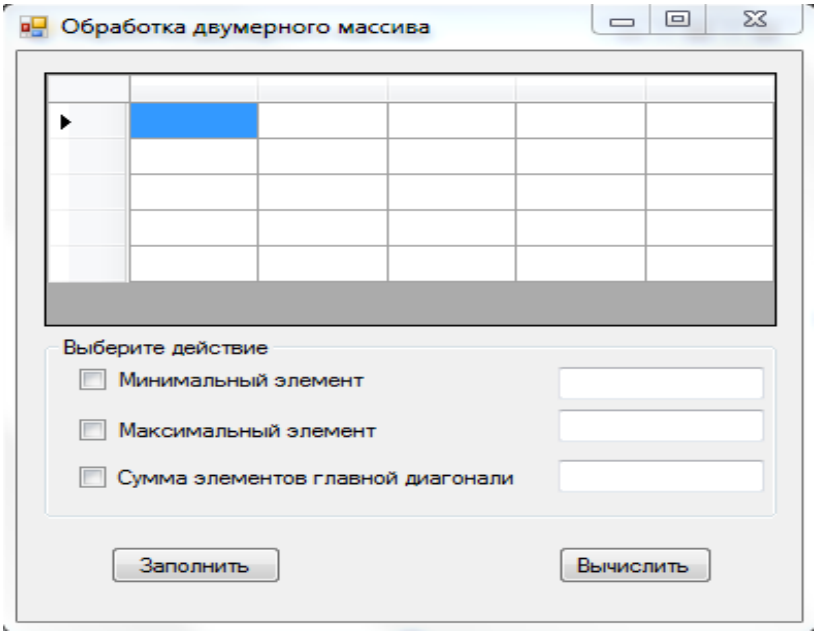
1. Самостоятельно расположить на форме нужные компоненты. Изменить свойства формы и компонентов.	
2. Сохраните файл проекта и программного модуля.	В окне «Создать проект» задаем имя «Сумма элементов», выбираем место сохранения проекта Нажать кнопку Ok
На странице свойств разверните список свойств dataGridView1 и задайте значения для его подсвойств: ColumnHeadersVisible, RowHeadersVisible, AllowUserToAddRow значение False, AutoSizeColoumnMode = Fill. Выберите свойство Columns и добавьте 5 столбцов. 	
4. В окне редактора кода в процедуру `Form1_Load` вставьте	Так как эта процедура сводится к выводу в окне приложения таблицы dataGridView1 подписи номеров

<table>
<tr><td>следующий код:</td><td>строк и столбцов, то текст этой процедуры можно записать следующим образом:
dataGridView1.Rows.Add(5);
int i = 0;
int j = 0;
int r = dataGridView1.Rows.Count;
int c = dataGridView1.Columns.Count;
for (j = 1; j <= r - 1; j++)
{
dataGridView1.Rows[i].Cells[j].Value = j;
}
j = 0;
for (i = 1; i <= c - 1; i++)
{
dataGridView1.Rows[i].Cells[j].Value = i;
}</td></tr>
<tr><td>5. Выберите объект buttonl и на странице События произведите двойной щелчок на пустом поле списка в событии OnClick, наступающем в момент получения элементом фокуса.</td><td>После этого в окне Редактора кода будет сгенерирована заготовка процедуры обработчика события procedure
private void button1_Click(object sender, EventArgs e)
в которую следует добавить следующие операторы:

int r = dataGridView1.Rows.Count;
int c = dataGridView1.Columns.Count;
int sum = 0;
int i,j;
int[,] mas = new int[c,r];
for (i = 1; i <= c - 1; i++)
{
for (j = 1; j <= r - 1; j++)
{
mas[i,j] =
Convert.ToInt32(dataGridView1.Rows[i].Cells[j].Value);
sum += mas[i, j];
}
}
label1.Text = String.Format("Сумма всех элементов массива - {0}",sum);</td></tr>
<tr><td colspan="2">5. Сохраните файлы модуля и проекта, откомпилируйте и запустите приложение.</td></tr>
</table>



Задание 2. Создайте приложение, которое выводит двумерный массив случайных целых чисел в объекте dataGridView1 и определяет минимальный и максимальный элементы, а также сумму элементов массива, расположенных на главной диагонали.



Алгоритм создания приложения:

1. Добавление строк на форму	В процедуре обработки формы: dataGridView1.Rows.Add(5); Глобальные переменные: int i, j; int[,] mas = new int[5, 5];
2. Создание кода – 1-го обработчика события	Random random = new Random(); for (i = 0; i <= 4; i++) { for (j = 0; j <= 4; j++) {

	dataGridView1.Rows[i].Cells[j].Value = random.Next(1, 100); } }
3. Создание кода – 2-го обработчика события	private void button2_Click(object sender, EventArgs e) { for (i = 0; i <= 4; i++) { for (j = 0; j <= 4; j++) { mas[i,j] = Convert.ToInt32(dataGridView1.Rows[i].Cells[j].Value); } } if (checkBox1.Checked) { int min = mas[1, 1]; for (i = 0; i <= 4; i++) { for (j = 0; j <= 4; j++) { if (mas[i, j] < min) min = mas[i, j]; } } textBox1.Text = min.ToString(); } if (checkBox2.Checked) { int max = mas[1, 1]; for (i = 0; i <= 4; i++) { for (j = 0; j <= 4; j++) { if (mas[i, j] > max) max = mas[i, j]; } } textBox2.Text = max.ToString(); } if (checkBox3.Checked) { int sum = mas[1, 1]; for (i = 0; i <= 4; i++) { for (j = 0; j <= 4; j++) { if (i == j) sum += mas[i, j]; } } textBox3.Text = sum.ToString(); } }
4. Сохраните файлы проекта и программного модуля, откомпилируйте и запустите программу на выполнение.	

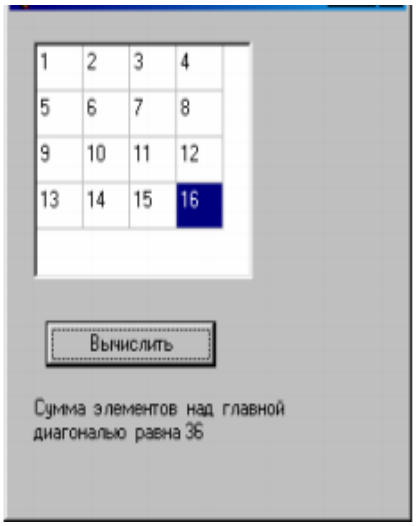
5. Щелкнув мышью на кнопке Заполнить, проверьте заполнение надписей номеров строк и столбцов, а также заполнение объекта dataGridView значениями элементов двумерного массива.

САМОСТОЯТЕЛЬНАЯ РАБОТА

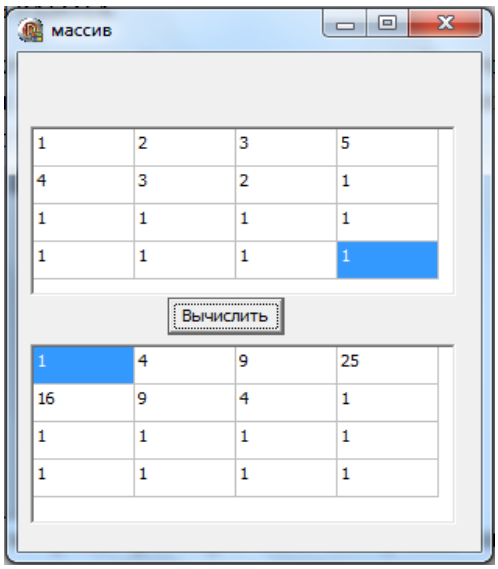
Студент выполняет первую задачу, номер которой совпадает с номером его фамилии в журнале группы.

Задача номер 21 обязательная для всех.

1. Дан двумерный целочисленный массив размера 4x4. Подсчитать сумму элементов, стоящих над главной диагональю. Для отображения массива использовать компонент dataGridView1.



2. Дана целочисленная матрица размера 4x4. Получить квадрат этой матрицы.



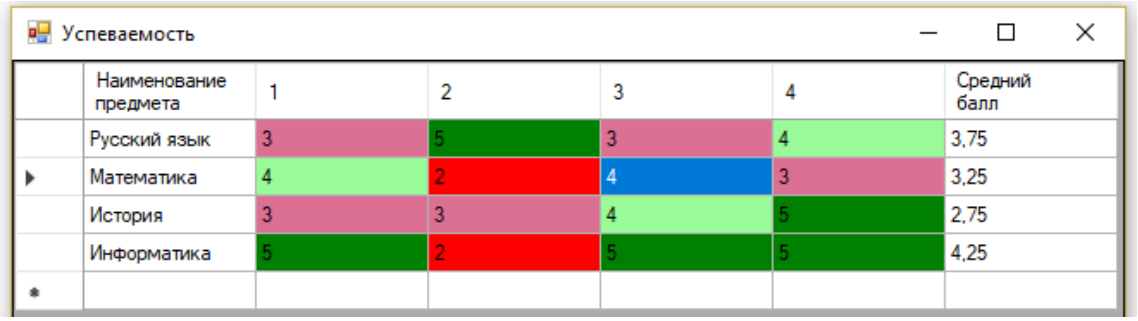
3. Создать приложение, вычисляющее произведение двух матриц.
4. Повернуть исходную матрицу на 90 градусов по часовой стрелке.

5. Обнулить первую строку матрицы (5X5), если во второй строке найдется хотя бы один отрицательный элемент.
6. Дана квадратная матрица 5x5 целых чисел. Подсчитать количество элементов попавших в отрезок [5,15].
7. Дана матрица целых чисел размером 4x4. Увеличить все чётные элементы на 16, а нечётные элементы увеличить втрое.
8. Дана матрица 5x5 целых чисел. Зеркально отразите ее относительно главной диагонали.
9. Дан двумерный массив целых чисел. Поменять местами элементы первой и третьей строки.
10. Для прямоугольной матрицы целочисленной 3x4 определить номер самого левого столбца, содержащего только положительные элементы. Если такого столбца нет, то вывести сообщение.
11. Дана матрица вещественных чисел: A[m,m]. Транспонировать матрицу относительно побочной диагонали.
12. Дана матрица A(5x6). Сформировать массив из сумм положительных элементов каждой строки. Если их в строке нет, то результат положить равный нулю.
13. Дана матрица целых чисел размером 3x4. Определите сумму каждого столбца этой матрицы.
14. Дана матрица A(3x4). Сформировать массив из произведений отрицательных элементов каждой строки. Если их в строке нет, то результат положить равный нулю.
15. Повернуть матрицу 5x5 на 90 градусов против часовой стрелки.
16. Дана матрица M(6,6) действительных чисел. Найти минимум среди сумм элементов диагоналей главной и побочной.
17. Дана квадратная матрица целых чисел A(3x3). Зеркально отразить относительно вертикальной оси симметрии.
18. Дана матрица A(N,N). Переписать элементы её главной диагонали в одномерный массив Y(N) и разделить их на максимальный элемент главной диагонали.
19. Дана матрица A(N,M). Поменять местами её наибольший и наименьший элементы.
20. Найти наибольший элемент побочной диагонали заданной матрицы A(N, N) и вывести на печать всю строку, в которой он находится.

Дополнительная задача

21. Создать приложение, осуществляющее вывод в окно двумерного массива – табеля успеваемости по нескольким школьным предметам с оценками за 1-4 четверти и за год, причем годовая оценка должна вычисляться как среднее арифметическое всех оценок по данному предмету. Используйте закраску ячеек таблицы разными цветами, которые могут определяться как на стадии разработки формы приложения, так и во время работы приложения, в зависимости от значения элемента массива.

Образец работающего приложения:



	Наименование предмета	1	2	3	4	Средний балл
	Русский язык	3	5	3	4	3,75
▶	Математика	4	2	4	3	3,25
	История	3	3	4	5	2,75
	Информатика	5	2	5	5	4,25
*						

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 10
РУБЕЖНЫЙ КОНТРОЛЬ

Цель занятия: формирование навыка работы со средой программирования Visual Studio при разработке приложений по изученным операторам и компонентам.

Вид занятия: контрольная работа.

Тип занятия: контроль и оценка знаний.

Разработать приложение с применением изученных операторов: условный, оператор выбора, циклы. Задания даются по вариантам. Темы для создания приложений:

1. Геометрическая задача по выводу площадей фигур. Выбирается одна из трех фигур. В зависимости от вида фигуры на форме появляются поля для ввода. Количество полей соответствует количеству вводимых данных.
2. Заказ в кафе: выбираются блюда (несколько), указывается их количество. Подтверждается наличие скидки и ее размер. Выводится сумма заказа.
3. Автозаправка: выбирается вид оплаты – наличный или безналичный; выбирается количество литров (для наличного расчета) или сумма (для безналичного расчета). Поле для ввода количества литров или суммы должно иметь правильный поясняющий текст. После завершения на форме или в окне должно появиться сообщение, сколько литров, и на какую сумму была проведена заправка.
4. Геометрическая задача по выводу объема тел. Выбирается одно из трех тел. В зависимости от вида тела на форме появляются поля для ввода. Количество полей соответствует количеству вводимых данных.
5. Рекламное агентство: выбирается вид рекламы, материал для рекламных баннеров или буклетов, указывается количество рекламных объектов, допускается наличие скидки, выбирается вид оплаты. Вывод стоимости.
6. Склад: выбираются товары из нескольких категорий, указывается их количество. Выводится производитель и стоимость товара на складе.
7. Фотоателье: виды фотографий – цветное или черно-белое; матовое или глянцевое; количество фотографий. Вывод стоимости.
8. Магазин: выбираются товары, указывается их количество. Учесть, что один товар может поставляться разными производителями. Выводится стоимость покупки.
9. Автосалон: выбираем марку авто, модель авто, цвет. В зависимости от цвета меняется цена. Учесть возможность установки дополнительного оборудования, например, сигнализации. Вывод стоимости.
10. Туристическое агентство: выбирается страна, количество путевок (взрослые / дети). В зависимости от возраста детей считается скидка. Учесть возможность дополнительной оплаты, например, за экскурсии. Вывод стоимости.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 11
СТРОКИ

Цель занятия: формирование навыка работы со средой программирования Visual Studio при разработке приложений, обрабатывающих строки.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Таблица 1 - Статические методы и свойства класса string

Метод	Описание
Empty	Возвращается пустая строка. Свойство со статусом read only.
Compare	Сравнение двух строк. Метод перегружен. Реализации метода позволяют сравнивать как строки, так и подстроки. При этом можно учитывать или не учитывать регистр, особенности национального форматирования дат, чисел и т.д.
CompareOrdinal	Сравнение двух строк. Метод перегружен. Реализации метода позволяют сравнивать как строки, так и подстроки. Сравниваются коды символов.
Concat	Конкатенация строк. Метод перегружен, допускает сцепление произвольного числа строк.
Copy	Создается копия строки.
Format	Выполняет форматирование в соответствии с заданными спецификациями формата. Ниже приведено более полное описание метода.
Intern, IsIntern	Отыскивается и возвращается ссылка на строку, если таковая уже хранится во внутреннем пуле данных. Если же строки нет, то первый из методов добавляет строку во внутренний пул, второй – возвращает null. Методы применяются обычно тогда, когда строка создается с использованием построителя строк – класса StringBuilder.
Join	Конкатенация массива строк в единую строку. При конкатенации между элементами массива вставляются разделители. Операция, заданная методом Join, является обратной к операции, заданной методом **Split.** Последний является динамическим методом и, используя разделители, осуществляет разделение строки на элементы.

Таблица 2 - Динамические методы и свойства класса string

Метод	Описание
Insert	Вставляет подстроку в заданную позицию.
Remove	Удаляет подстроку в заданной позиции.
Replace	Заменяет подстроку в заданной позиции на новую подстроку.
Substring	Выделяет подстроку в заданной позиции.
IndexOf, IndexOfAny, LastIndexOf, LastIndexOfAny	Определяются индексы первого и последнего вхождения заданной подстроки или любого символа из заданного набора
StartsWith, EndsWith	Возвращается true или false, в зависимости от того, начинается или заканчивается строка заданной подстрокой.
PadLeft, PadRight	Выполняет набивку нужным числом пробелов в начале и в конце строки.
Trim, TrimStart, TrimEnd	Обратные операции к методам Pad. Удаляются пробелы в начале и в конце строки, или только с одного ее конца.
ToCharArray	Преобразование строки в массив символов.

Таблица 3 Статические методы и свойства класса char

Метод	Описание
GetNumericValue	Возвращает численное значение символа, если он является цифрой, и (-1) в противном случае.
GetUnicodeCategory	Все символы разделены на категории. Метод возвращает Unicode категорию символа.
IsControl	Возвращает true, если символ является управляющим.
IsDigit	Возвращает true, если символ является десятичной цифрой.
IsLetter	Возвращает true, если символ является буквой.
IsLetterOrDigit	Возвращает true, если символ является буквой или цифрой.
IsLower	Возвращает true, если символ задан в нижнем регистре.
IsNumber	Возвращает true, если символ является числом (десятичной или шестнадцатеричной цифрой).
IsPunctuation	Возвращает true, если символ является знаком препинания.
IsSeparator	Возвращает true, если символ является разделителем.
IsSurrogate	Некоторые символы Unicode с кодом в интервале [0x1000, 0x10FFF] представляются двумя 16-битными «суррогатными» символами. Метод возвращает true, если символ является суррогатным.
IsUpper	Возвращает true, если символ задан в верхнем регистре.
IsWhiteSpace	Возвращает true, если символ является «белым пробелом». К белым пробелам, помимо пробела, относятся и другие символы, например, символ конца строки и символ перевода каретки.
Parse	Преобразует строку в символ. Естественно, строка должна состоять из одного символа, иначе возникнет ошибка.
ToLower	Приводит символ к нижнему регистру.
ToUpper	Приводит символ к верхнему регистру.
MaxValue, MinValue	Свойства, возвращающие символы с максимальным и минимальным кодом. Возвращаемые символы не имеют видимого образа.

ХОД РАБОТЫ

Задание 1. Создайте приложение, предлагающее пользователю ввести строку текста, а затем подсчитывающее количество символов в тексте. Количество символов должно сочетаться со словом «символ», «символа», «символов». Например, «21 символ», «15 символов», «33 символа». Образец оформления приложения представлен на рисунке 1.

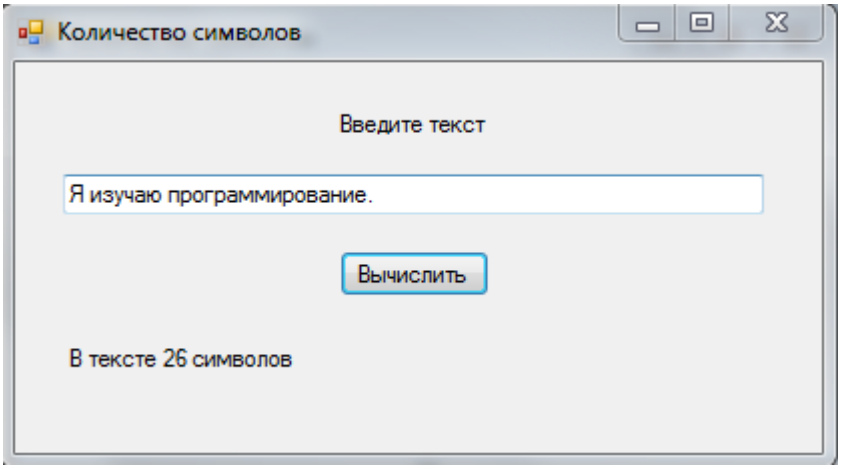


Рисунок 1 - Форма к заданию 1

Алгоритм создания приложения:

1. Самостоятельно создать форму с необходимыми на ней компонентами, изменить свойства объектов в соответствии с рисунком.	
2. Используя оператор switch, создаем код обработчика события кнопки Button1.	`private void button1_Click(object sender, EventArgs e)` `{` `string s = "";` `int dop = 0;` `int n = textBox1.Text.Length;` `if (n > 20) dop = n % 10;` `else dop = n;` `switch (dop)` `{` `case 1: s = " символ";` `break;` `case 2: case 3: case 4: s = " символа";` `break;` `default: s = " символов";` `break;` `}` `label2.Text = "В тексте " + n + s;` `}`
3. Сохранить проект, откомпилировать и запустить программу на выполнение. Протестировать программу при различном количестве символов.	

Задание 2. Создайте приложение, предлагающее пользователю ввести строку текста, а затем заменяющее символы в тексте и подсчитывающее количество заменённых символов. Вариант замены символов должен определяться по положению соответствующего флажка checkBox.

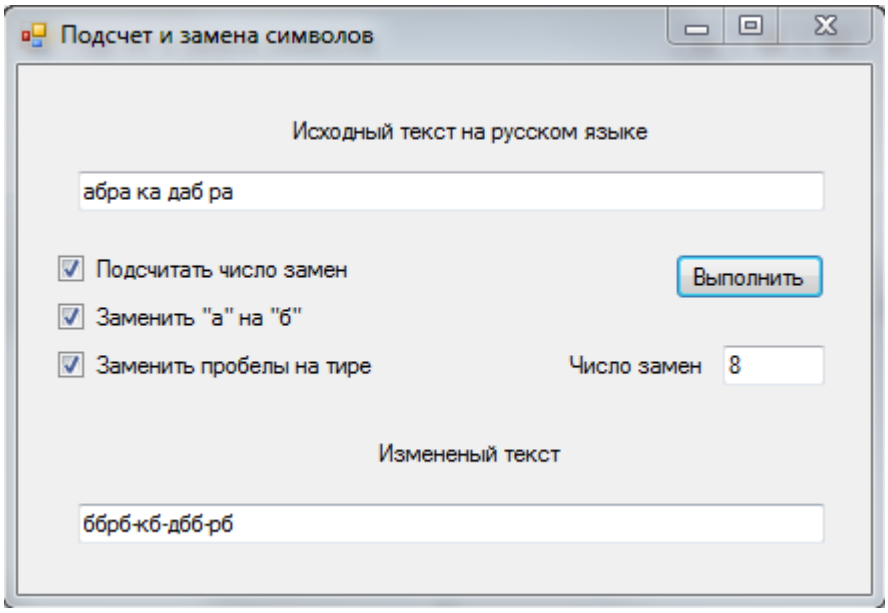


Рисунок 2 - Форма к заданию 2

Алгоритм создания приложения:

1. Создание кода – обработчика события	Для того, чтобы приложение выполнило вычисления при щелчке по кнопке Button1 («=») надо написать код обработки этого события. `private void button1_Click(object sender, EventArgs e)` `{` `int n = 0;` `string s = textBox1.Text;` `if (checkBox2.Checked)` `{` `n = s.Split('a').Length - 1;` `s = s.Replace("a", "6");` `}` `if (checkBox3.Checked)` `{` `n += s.Split(' ').Length - 1;` `s = s.Replace(" ", "-");` `}` `textBox3.Text = s;` `if (checkBox1.Checked) textBox2.Text = n.ToString();` `}`
2. Сохранение изменений	
3. Запуск приложения на исполнение	

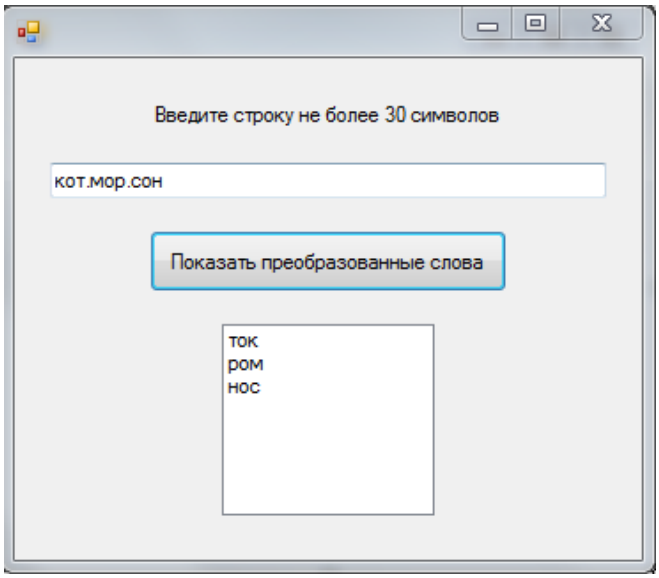
Задание 3. Создать приложение, которое позволяет вводить список строк в поле редактора richTextBox1, а затем при нажатии на кнопку переписывает все строки, содержащие более 5 символов, в поле редактора richTextBox2.

Алгоритм создания приложения:

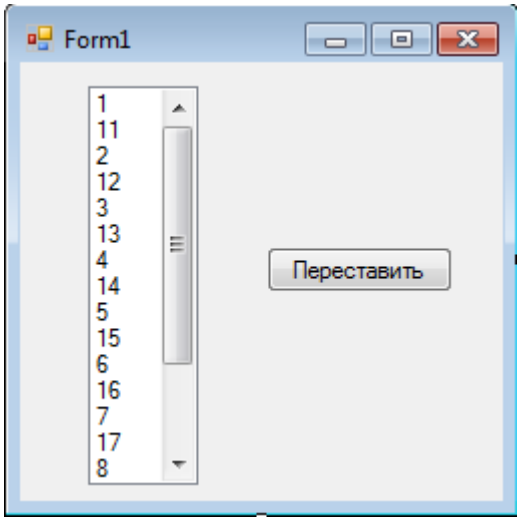
Процедура обработки события щелчка по кнопке будет иметь вид:	`private void button1_Click(object sender, EventArgs e)` `{` `string[] lines = new string[richTextBox1.Lines.Length];` `lines = richTextBox1.Lines;` `for (int i = 0; i <= richTextBox1.Lines.Length-1; i++ )` `if (lines[i].Length > 5)` `{ richTextBox2.AppendText(lines[i] + "\n"); }` `}`

САМОСТОЯТЕЛЬНАЯ РАБОТА

1. Создать приложение, которое определяет, является ли введенная строка перевертышем.
2. Создать приложение, которое позволяет вводить строку и выводит ее в обратном виде, т.е. при вводе «Привет» выведется «тевирП».
3. Создать приложение, которое позволяет вводить список строк в поле редактора richTextBox1, а затем переставить строки в обратном порядке.
4. Создать приложение, которое позволяет вводить список строк в поле редактора textBox1, а затем при нажатии на кнопку удаляет все строки, содержащие более 5 символов.
5. Дана строка, содержащая не более 30 символов, являющихся строчными русскими буквами и запятыми. Последний символ – точка. Назовем словом – последовательность букв между ближайшими двумя знаками препинания. Вывести в поле компонента listBox в столбец все слова, меняя местами в каждом слове первую и последнюю букву.



6. Дан одномерный целочисленный массив *A* из 20 элементов. Ввести его элементы с клавиатуры в поле richTextBox, а затем, используя методы класса Strings, переставить элементы массива а так, чтобы они расположились в следующем порядке a1, a11, a2, a12, . . . ,a10, a20. Окно работающего приложения:



ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Дать характеристику существующим типам строк: String, ShortString, AnsiString, WideString.
2. Дать характеристику процедурам и функциям обработки строк: Length, Copy, Remove, Insert, Split, ToString, Replace.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 12 – 13

РАБОТА С ТЕКСТОВЫМИ ФАЙЛАМИ

Цель занятия: формирование навыка работы со средой программирования MS Visual Studio 2015 при разработке приложений с применением компонентов monthCalendar, pictureBox, folderBrouserDialog с применением команд по обработке файлов.

ХОД РАБОТЫ

Задание 1. Создать консольное приложение, позволяющее отображать текст из текстового файла.

Алгоритм создания приложения:

1. Создайте форму	Файл/Создать/Проект… Visual C# - Консольное приложение
2. Сохраните файл проекта и программного модуля.	Задаем имя и выбираем место сохранения проекта, после чего нажимаем кнопку Ok
3. Пропишите using System.IO; // Пространство имен IO (Input / Output) содержит множество полезных классов для работы с файлами.	
4. В поле класса Program объявите переменную input	static StreamReader input;
// Чтобы не перегружать процедуру Main, выполняя непосредственно в ней все работы, связанные с файлом, разобьём нашу задачу на подзадачи. Для каждой задачи зададим свой метод, а в процедуре Main будем вызывать эти методы	
5. В Main добавьте операции, позволяющие открыть файл для чтения	bool found; found = OpenReadFile();
// Булевская переменная found будет принимать значение true, если файл удалось открыть, и false – в случае неудачи. Метод OpenReadFile обеспечивает открытие файла.	
6. В поле класса Program добавьте новую операцию для доступа к файлу	static bool OpenReadFile() { **// Задать путь к директории, хранящей проект и // файл для чтения** String path = @"C:\1.txt"; **// Проверка существования файла в указанной** **// директории;** **// Открытие текстового файла для чтения** if (File.Exists(path)) { input = new StreamReader(path); Console.WriteLine("Файл открыт для чтения"); return true; } else { Console.WriteLine("Ошибка в задании пути файла!");

	Console.ReadLine(); return false; } **}**
7. В поле класса Program добавьте новую операцию для отображения текста из файла **(первый способ)**	static void ReadingFile() { Int64 n = Convert.ToInt64(input.ReadLine()); for (int i = 0; i < n; i++) { string current = input.ReadLine(); Console.WriteLine(current); } }
8. В поле класса Program добавьте новую операцию для отображения текста из файла **(второй способ)**	static void NextReading() { Console.WriteLine("Повторное открытие и чтение файла"); OpenReadFile(); while (!input.EndOfStream) { string current = input.ReadLine(); Console.WriteLine(current); } input.Close(); }
// необходимо использовать оба способа	
9. В Main добавьте операции, отображающие текст из текстового файла	if (found) { ReadingFile(); Console.WriteLine(""); NextReading(); input.Close(); Console.ReadLine(); }
10. Сохраните файлы модуля и проекта, откомпилируйте и запустите приложение.	

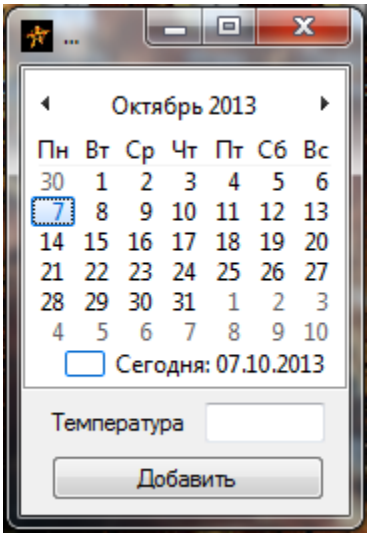
Задание 2. Создать консольное приложение, позволяющее изменять текст в текстовом файле.

Алгоритм создания приложения:

1. Создайте форму	Файл/Создать/Проект… Visual C# - Консольное приложение
2. Сохраните файл проекта и программного модуля.	Задаем имя и выбираем место сохранения проекта, после чего нажимаем кнопку Ok
3. Пропишите using.System IO; // Пространство имен IO (Input / Output) содержит множество полезных классов для работы с файлами.	
4. В поле класса Program объявите переменные **output**, **path** и **n**	static StreamWriter output; static string path; static int n;
// Чтобы не перегружать процедуру Main, выполняя непосредственно в ней все работы,	

<table>
<tr><td colspan="2">связанные с файлом, разобьём нашу задачу на подзадачи. Для каждой задачи зададим свой метод, а в процедуре Main будем вызывать эти методы</td></tr>
<tr><td>5. В Main добавьте операции, позволяющие открыть файл для записи</td><td>OpenWriteFile();</td></tr>
<tr><td>6. В поле класса Program добавьте новую операцию для доступа к файлу</td><td>static void OpenWriteFile()
{
// Задать путь к директории, хранящей проект и //
файл для чтения
path = @"C:\1.txt";
// Проверка существования файла в указанной
// директории;
// Открытие текстового файла для записи
if (File.Exists(path))
{
output = new StreamWriter(path);
}
else
{
output = new StreamWriter(path, true);
}
Console.WriteLine("Файл открыт для записи");
}</td></tr>
<tr><td>7. В поле класса Main добавьте операции для изменения текста в текстовом файле</td><td>WritingFile();
output.Close();
Console.ReadLine();</td></tr>
<tr><td>8. В поле класса Program добавьте новую операцию для изменения текста в текстовом файле по заданным программой параметрам</td><td>static void WritingFile()
{
n = 10;
output.WriteLine(n.ToString());
for (int i = 0; i < n; i++)
{
string current = (10 * i).ToString();
output.WriteLine(current);
}
Console.WriteLine("Файл output успешно записан");
}</td></tr>
<tr><td colspan="2">9. Сохраните файлы модуля и проекта, откомпилируйте и запустите приложение.</td></tr>
</table>

Задание 3. Создать приложение, позволяющее изменять текст в текстовом файле.

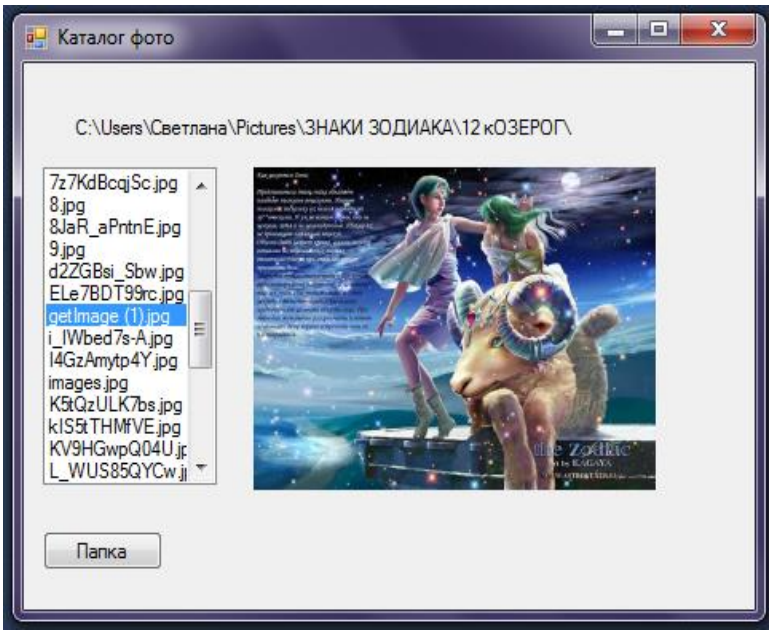


Алгоритм создания приложения:

1. Создайте форму	Файл/Создать/Проект… Visual C# - Приложение Windows Forms
2. Сохраните файл проекта и программного модуля.	Задаем имя и выбираем место сохранения проекта, после чего нажимаем кнопку Ok
3. Самостоятельно расположите на форме компоненты **monthCalendar**, **label**, **textbox** и **button**	
4. Пропишите using.System IO; // Пространство имен IO (Input / Output) содержит множество полезных классов для работы с файлами.	
5. Объявите новые переменные	StreamWriter sw; FileInfo fi; double tp; DateTime dt;
6. Создайте защищённый блок	try { } catch { }
7. В защищённом блоке задайте значение переменной**(1)** и введите действие для исключения**(2)**	**1.** tp = Convert.ToDouble(textBox1.Text); **2.** MessageBox.Show("Введенное значение имеет неверный формат.", "Погода", MessageBoxButtons.OK, MessageBoxIcon.Error); return;
8. Укажите путь к текстовому файлу	fi = new FileInfo(Application.StartupPath + "\\meteo.txt");
9. Укажите действия с текстовым файлом в переменную	if (fi.Exists) { sw = fi.AppendText(); } else { sw = fi.CreateText(); }
10. Присвойте переменной информацию о дате	dt = monthCalendar1.SelectionStart;

11. Создайте цикл, создающий в текстовом файле новую запись	do { sw.WriteLine(dt.ToShortDateString() + " " + tp.ToString("N")); dt = dt.AddDays(1); } while(DateTime.Compare(monthCalendar1.SelectionEnd, dt) >= 0);
12. Создайте операцию для завершения работы с текстовым файлом и очистки поля textbox	sw.Close(); textBox1.Text = string.Empty;
13. Сохраните файлы модуля и проекта, откомпилируйте и запустите приложение.	

Задание 4. Создать приложение, позволяющее просматривать графические изображения. Предварительно пользователь выбирает папку для просмотра. Интерфейс приложения представлен на рисунке:



Алгоритм создания приложения:

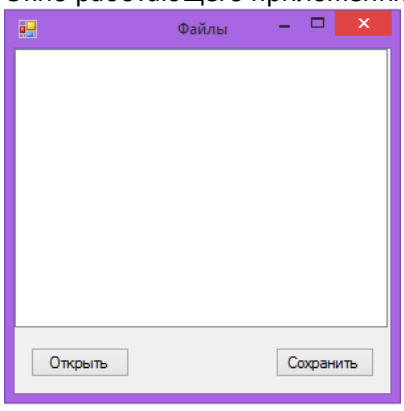
1. Расположить на форме следующие компоненты: listBox, folderBrouserDialog, pictureBox, label	
2. Для получения прямого доступа к типам DyrectoryInfo и FileInfo необходимо в директиву Using программы добавить System.IO	`using System.IO;`
3. Задать	`private int pbh,// высота`

исходную высоту и ширину элемента pictureBox1	pbw; // ширина элемента PictureBox1
4. Заполнение списка	private Boolean FillListBox (string aPath) { DirectoryInfo di = new DirectoryInfo(aPath); FileInfo[] fi = di.GetFiles("*.jpg"); listBox1.Items.Clear(); foreach(FileInfo fc in fi) { listBox1.Items.Add(fc.Name); } label1.Text=aPath; if (fi.Length==0) return false; else { listBox1.SelectedIndex=0; return true; } }
5. Загрузка формы	private void Form1_Load(object sender, EventArgs e) { pbh=pictureBox1.Height; pbw=pictureBox1.Width; listBox1.Sorted=true; FillListBox(Application.StartupPath+"\\"); }
6. Щелчок на кнопке «Папка»	private void button1_Click(object sender, EventArgs e) { FolderBrowserDialog fb=new FolderBrowserDialog(); fb.Description="Выберите папку"; fb.ShowNewFolderButton=false; if (fb.ShowDialog()==DialogResult.OK) if (!FillListBox(fb.SelectedPath+"\\")) pictureBox1.Image=null; }
7. Выбор элементов списка	private void listBox1_SelectedIndexChanged(object sender, EventArgs e) { double mh,mw; pictureBox1.Image=new Bitmap (label1.Text+listBox1.SelectedItem.ToString()); if ((pictureBox1.Image.Width>pbw)||(pictureBox1.Image.Height >pbh))

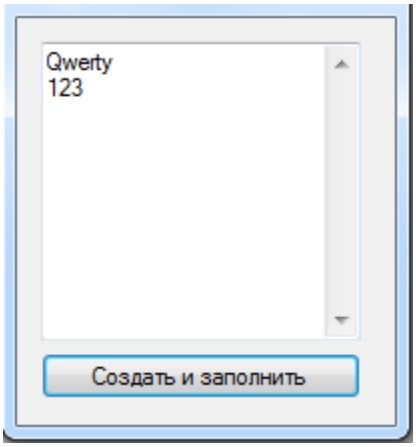
```
            {

pictureBox1.SizeMode=PictureBoxSizeMode.StretchImage;

mh=(double)pbh/(double)pictureBox1.Image.Height;

mw=(double)pbw/(double)pictureBox1.Image.Width;
                if (mh<mw)
                {

pictureBox1.Width=Convert.ToInt16(pictureBox1.Image.Width*mh
);
                    pictureBox1.Height=pbh;
                }
                else{
                    pictureBox1.Width=pbw;

pictureBox1.Height=Convert.ToInt16(pictureBox1.Image.Height*
mw);
                    }
            }
            else
                 if
(pictureBox1.SizeMode==PictureBoxSizeMode.StretchImage)

pictureBox1.SizeMode=PictureBoxSizeMode.AutoSize;
}
```

САМОСТОЯТЕЛЬНАЯ РАБОТА

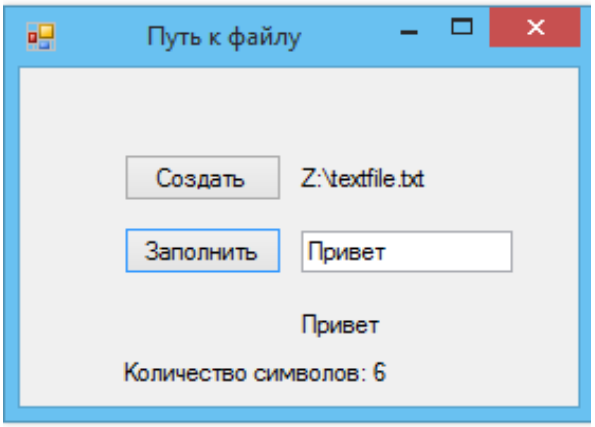
1. Создать текстовый редактор, позволяющий с помощью диалоговых окон открывать и сохранять текстовые файлы. Окно работающего приложения:



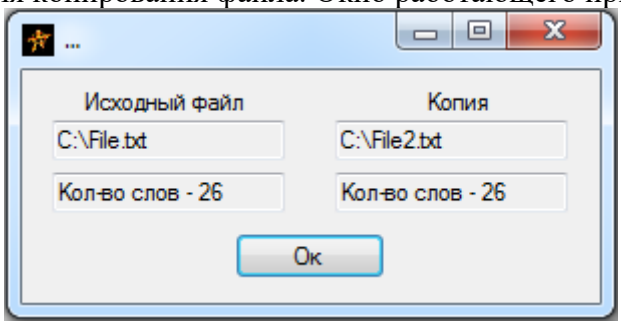
2. Создать текстовый файл и заполнить его информацией, введённой в textBox. Окно работающего приложения:



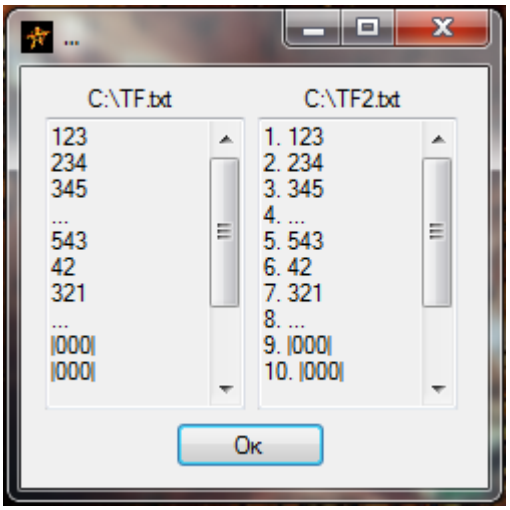
3. Создать файл, состоящий из символов, введённых в поле ввода textBox. Вывести путь к файлу. Вывести содержимое созданного файла в поле метки label. Посчитать количество символов во введенном тексте и вывести в поле метки label. Окно работающего приложения:



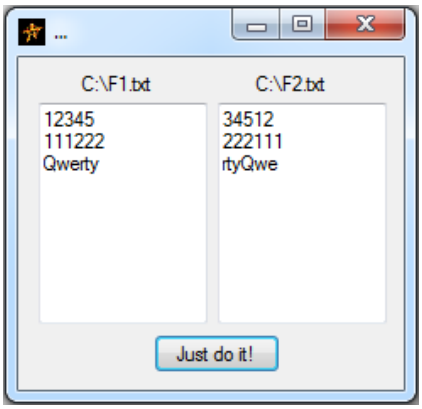
4. Составить программу для копирования файла. Окно работающего приложения:



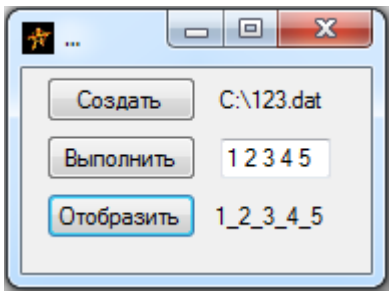
5. В начало каждой строки файла вставить её номер и записать преобразованные строки в новый файл. Окно работающего приложения:



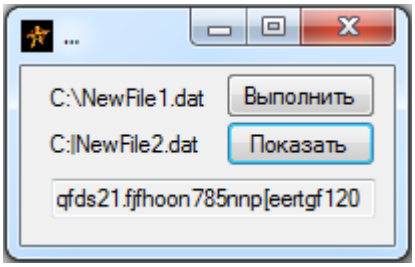
6. Дан текстовый файл, содержащий строки. Переписать содержимое файла в другой файл, сдвигая циклически каждую строку вправо на 3 символа. Например, результатом сдвига строки abcdewrt будет строка wrtabcde. Для выбора имён исходного и создаваемого файла использовать диалоговые окна. Окно работающего приложения:



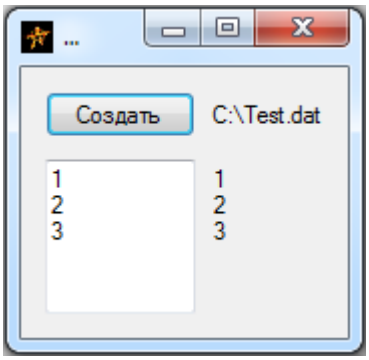
7. Создать типизированный файл, состоящий из символов, введённых в окно ввода textBox. Заменить в файле каждый пробел символом '_ '. (Не используя вспомогательный файл.) Окно работающего приложения:



8. Переписать содержимое данного символьного файла в новый символьный файл, заменяя каждую встречающуюся строчную английскую букву (кроме буквы z) на следующую по алфавиту, а все остальные элементы оставляя без изменения. Окно работающего приложения:



9. Создать типизированный файл, состоящий из целых чисел, введенных в столбик в окно редактора Memo. Вывести содержимое созданного файла в поле метки Label1. Окно работающего приложения:



ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 14

ПРОГРАММИРОВАНИЕ ГРАФИКИ

Цель занятия: формирование навыков разработки приложений по обработке графики, а также планирование и реализация модульной структуры программного продукта; создание многооконного проекта в среде Microsoft Visual Studio 2015.

ТЕОРЕТИЧЕСКИЙ МАТЕРИАЛ

Графические примитивы GDI+

Класс	Описание
Bitmap	Представляет растровые изображения
Font	Определяет атрибуты шрифта: размер, гарнитуру и т. д
HatchBrush	Определяет заливку решетчатым узором
LinearGradientBrush	Определяет заливку с линейным градиентом
Pen	Определяет внешний вид линий и кривых
SolidBrush	Определяет заливку сплошным цветом
TextureBrush	Определяет заливку с использованием растрового изображения

Некоторые из этих классов определены в пространстве имен System.Drawing, другие — в пространстве имен System.Drawing.Drawing2D.

Перья, представленные экземплярами класса Pen, управляют внешним видом линий и кривых.

Объект Font задает внешний вид текста.

Кисти, представленные объектами классов HatchBrusb, LinearGradientBrush, SolidBrush и TextureBrusb определяют виды заливки.

Объект Pen, передаваемый DrawRectangle, обводит каждый прямоугольник черной линией толщиной в 1 единицу. По умолчанию — это один 1 пиксель.

Кисти, передаваемые FillRectangle, управляют заливкой внутренней части прямоугольников. Заметьте: рамки и заливки рисуются по отдельности.

Класс Graphics содержит разнообразные открытые методы Draw и Fill для рисования линий, кривых, прямоугольников, эллипсов и других фигур.

В следующей таблице перечислены методы класса Graphics, для рисования линий, кривых и фигур.

Метод	Описание
DrawArc	Рисует дугу
DrawBezier	Рисует кривую Безье
DrawCurve	Рисует фундаментальный сплайн
DrawEllipse	Рисует круг или эллипс
DrawIcon	Рисует значок
DrawImage	Рисует картинку
DrawLine	Рисует линию
DrawPie	Рисует сектор
DrawPolygon	Рисует многоугольник
DrawString	Рисует строку текста
FillEllipse	Рисует круг или эллипс с заливкой
FillPie	Рисует сектор с заливкой
FillPolygon	Рисует многоугольник с заливкой
FillRectangle	Рисует прямоугольник с заливкой

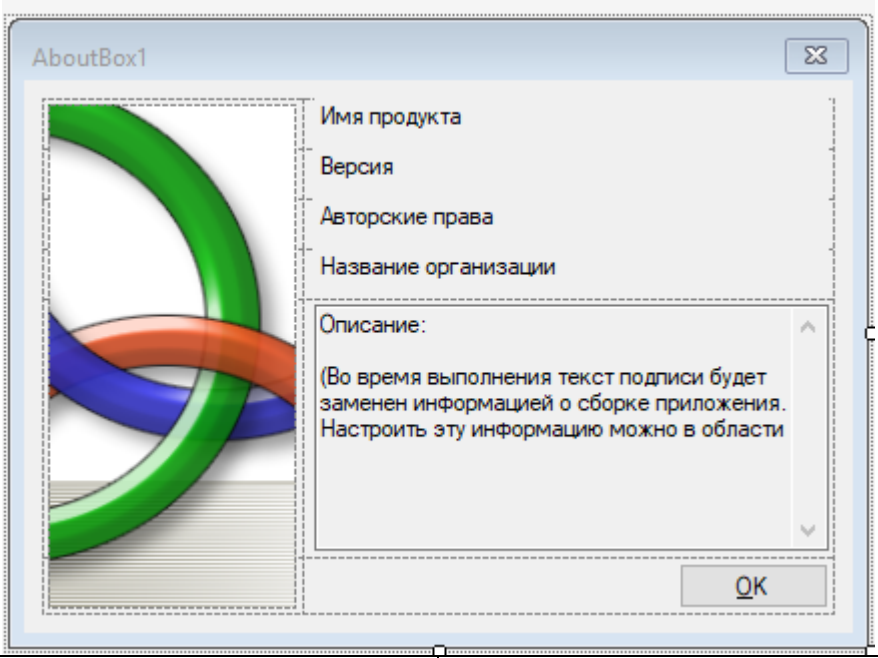
При вызове DrawRectangle и FillRectangle вы определяете координаты, указывающие положение левого верхнего угла прямоугольника, и расстояния, задающие его ширину и высоту.

По умолчанию расстояния измеряются в пикселах. Координаты задают расположение в двумерной системе, начало которой располагается в левом верхнем углу формы, а оси X и Y направлены вправо и вниз соответственно.

ХОД РАБОТЫ

Задание 1. Создать окно, выводящее информацию о программном продукте.

Алгоритм создания приложения:

1. В меню «**Файл**» последовательно выберите пункты «**Создать**» и «**Проект**».
2. В области «**Категории шаблонов**» откройте Visual C#, а затем щелкните »Windows».
3. В области **Шаблоны** щелкните приложение **Windows Forms**. Введите имя проекта в поле Имя. Нажмите кнопку ОК. В Обозревателе решений появится новый проект.
4. Добавьте два элемента button на форму1. Button1 переименовать в «Прямоугольники», button2 - в «Многоугольник»
5. В меню «**Проект**» выберите «**Добавить форму Windows**». Добавьте две формы.
6.В меню «**Проект**» выберите «**Добавить компонент**». Выберите элемент форма Окно «О программе». 

7. Запрограммировать button1 на открытие окна «О программе»	AboutBox1 box = new AboutBox1(); box.ShowDialog();
8. Запрограммировать button2 и button3 на открытие второй и третьей форм	Form2 form2 = new Form2(); form2.Show(); Form3 form3 = new Form3(); Form3.Show();
9. Протестируйте программу.	

Задание 2. Создать метод, рисующий прямоугольники трех стилей: 1) без заливки, 2) закрашенный красным цветом и 3) с градиентной заливкой, переходящей из красного цвета в синий.

Алгоритм создания приложения:

1. Создать графическую среду, задать цвет пера	`private void button1_Click(object sender, EventArgs e)` `{` `  Graphics im = CreateGraphics();` `  Pen pen = new Pen(Color.Black);`
2. Нарисовать пустой прямоугольник	`im.DrawRectangle(pen, 10, 10, 390, 90);`
3. Нарисовать прямоугольник со сплошной заливкой	`SolidBrush solid = new SolidBrush(Color.Red);` `im.FillRectangle(solid, 10, 110, 390, 90);`
4. Нарисовать прямоугольник с градиентной заливкой	1. Добавьте `using System.Drawing.Drawing2D;` `Rectangle rect = new Rectangle(10, 210, 390, 90);` `LinearGradientBrush gradient = new LinearGradientBrush(rect, Color.Red, Color.Blue, LinearGradientMode.Horizontal);` `im.FillRectangle(gradient, rect);` `im.DrawRectangle(pen, rect);`

Задание 3. Создать метод, который рисует прямоугольник шириной 300 и высотой 100 единиц в левом верхнем углу

Алгоритм создания приложения:

1. Создать графическую среду	`protected override void OnPaint (PaintEventArgs e)` `{` `  int x = 0, y = 101;` `  Graphics im = CreateGraphics();` `  SolidBrush brush = new SolidBrush(Color.Red);` `  im.FillRectangle(brush, 0, 0, 300, 100);`
2. Вставить рисунок	`Image newImage = Image.FromFile("sky.jpg");` `  im.DrawImage(newImage, x, y);`
3. Создать точки для полигона и указать цвета заливки	`SolidBrush blueBrush = new SolidBrush(Color.Blue);` `Point point1 = new Point(60, 50);` `Point point2 = new Point(110, 25);` `Point point3 = new Point(160, 25);` `Point point4 = new Point(210, 50);` `Point point5 = new Point(160, 75);` `Point point6 = new Point(110, 75);` `Point point7 = new Point(60, 50);` `Point[] curvePoints = { point1, point2, point3, point4, point5, point6, point7 };`
4. Вывести полигон на форму	`im.FillPolygon(blueBrush, curvePoints);` `}`

Задание 4. Создать приложение, выводящее на экран, гистограмму.

САМОСТОЯТЕЛЬНАЯ РАБОТА

Создать проект, содержащий форму. Расположить на форме 2 кнопки, при нажатии на которых отображаются Герб и Флаг государства. Флаг строить из прямоугольных элементов соответствующего цвета. Флаг и Герб отображаются на разных формах.

Номер задания определяется по номеру студента в списке группы.

1. Англия	2. Армения
3. Белоруссия	4. Болгария
5. Германия	6. Греция
7. Грузия	8. Дания
9. Индия	10. Испания
11. Италия	12. Казахстан
13. Китай	14. Латвия
15. Литва	16. Нидерланды
17. Норвегия	18. Польша
19. Россия	20. Румыния
21. Таджикистан	22. Туркменистан
23. Узбекистан	24. Украина
25. Финляндия	26. Франция
27. Чехия	28. Швейцария
29. Швеция	30. Япония

Примечание: для вставки изображения применить команду:

```
Image newImage = Image.FromFile(Application.StartupPath + "\\gerb.png");
```

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 15

ПРОГРАММИРОВАНИЕ ГРАФИКИ

Цель занятия: формирование навыков создания проектов с использованием диалоговых окон, элементов управления: openFileDialog, fontDialog, colorDialog, pageSetupDialog; применение методов ООП; разработка приложений по обработке графики.

ХОД РАБОТЫ

Задание 1. Создать проект с функциями:
- выбора графического файла для вывода на форму;
- вывода текста, введенного в textBox с использованием выбранного шрифта и цвета;
- вывода текста, введенного в textBox с использованием текстуры, выбранной в openFileDialog;
- отображения линии;
- падающего прямоугольника.

Алгоритм создания приложения:

1. Запустите Visual Studio.	
2. В меню «**Файл**» последовательно выберите пункты «**Создать**» и «**Проект**».	
3. В области «**Категории шаблонов**» откройте «Visual C#», а затем щелкните «Windows».	
4. В области «**Шаблоны**» щелкните приложение **Windows Forms**.	
5. Введите имя проекта в поле Имя. Нажмите кнопку ОК. В Обозревателе решений появится новый проект.	
6. С левой стороны найти вкладку «**Панель элементов**». Добавьте элемент Button, textBox, fontDialog, colorDialog, openFileDialog на форму.	
7. Вставьте следующий код для программирования кнопки «Открыть»:	{

```
        Graphics im = CreateGraphics();
        OpenFileDialog openFileDialog1 = new OpenFileDialog();

        if (openFileDialog1.ShowDialog() == DialogResult.OK)
        {
           try
           {
              Image newImage = Image.FromFile(openFileDialog1.FileName);
              im.DrawImage(newImage, 10, 10);
            }
           catch (Exception ex)
           {
              MessageBox.Show("Ошибка "+ex);
           }
           Font font1 = new Font("Arial",16);
           FontDialog fontDialog1 = new FontDialog();
           if (fontDialog1.ShowDialog() != DialogResult.Cancel)
           {
              font1 = fontDialog1.Font;
           }
           Color color1 =  Color.FromArgb(0,0,0);
           ColorDialog colorDialog1 = new ColorDialog();
           if (colorDialog1.ShowDialog() == DialogResult.OK)
```

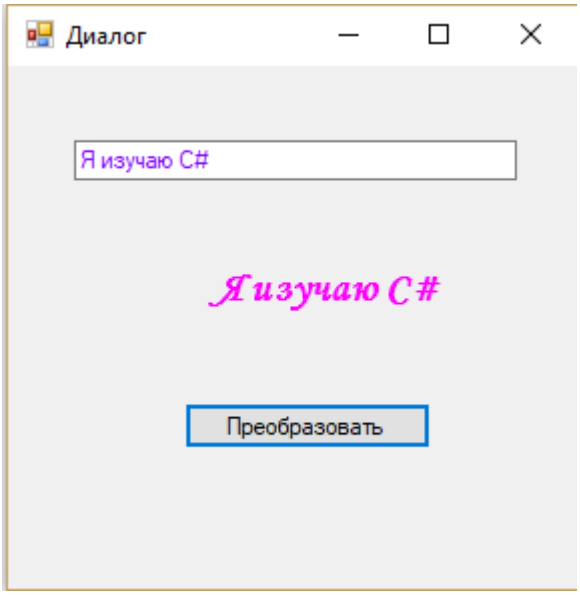
```
                color1 = colorDialog1.Color;
            SolidBrush newbrush = new SolidBrush(color1);
            im.DrawString(textBox1.Text, font1, newbrush, 10, 10);
            if (openFileDialog1.ShowDialog() == DialogResult.OK)
            {
                Image newImage =
Image.FromFile(openFileDialog1.FileName);
                im.DrawImage(newImage, 10, 10);
                TextureBrush texture = new TextureBrush (newImage);
                im.Clear (Color.Aqua);
                im.DrawString (textBox1.Text, font1, texture, 20, 20);
            }
        }
    }
```

Задание 2. Создать проект, в котором выбирается цвет шрифта, тип шрифта, способ начертания для текста, вводимого в текстовое поле.

На форму поместите pageSetupDialog, fontDialog, colorDialog. Для обработки событий вставьте код:

```
ColorDialog MyDialog = new ColorDialog();

// Не позволяет пользователю выбрать собственный цвет.
            MyDialog.AllowFullOpen = false;

// Позволяет пользователю получать справку. (Значение по
умолчанию - false.)
            MyDialog.ShowHelp = true;

// Устанавливает исходный цвет для текущего цвета текста.
            MyDialog.Color = textBox1.ForeColor;

// Обновите цвет текстового поля, если пользователь нажимает
кнопку OK.
            if (MyDialog.ShowDialog() == DialogResult.OK)
                textBox1.ForeColor = MyDialog.Color;

Graphics im = this.CreateGraphics();
Pen bpen = new Pen(Color.Black);
SolidBrush drawBrush = new SolidBrush(Color.Red);
Font drawFont = new Font("Arial", 16);
ColorDialog MyDialog1 = new ColorDialog();
if (MyDialog1.ShowDialog() == DialogResult.OK)
drawBrush.Color = MyDialog1.Color;
FontDialog MyDialog2 = new FontDialog();
if (MyDialog2.ShowDialog() == DialogResult.OK)
drawFont = MyDialog2.Font;
im.DrawString(textBox1.Text, drawFont, drawBrush, 100, 100);
```

Ниже представлен результат работы проекта.



Задание 3. Создать приложение, выводящее на экран, летящий по небу самолет.
Задание 4. Создать приложение, выводящее на экран, бегущую строку.

САМОСТОЯТЕЛЬНАЯ РАБОТА

Часть 1

Задачи первой части выполняет каждый студент. Номер подзадачи студент выбирает самостоятельно.

1. Создать проект, содержащий форму. При нажатии на кнопку «Показать» выдать график заданной функции. На графике начертить оси X и Y . Предусмотреть выбор цвета линии. Подписать график, выбирая шрифт текста.
   № 1 y= 3x - 6
   № 2 y= 3-2x
   № 3 y=5x+12

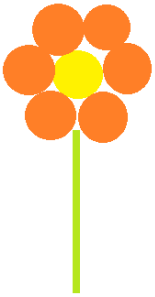
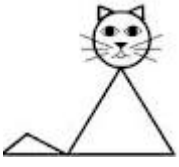
2. Создать проект, содержащий форму. При нажатии на кнопку «Показать» нарисовать рисунок. При нажатии на кнопку «Изменить» поменять рисунок:
   № 1 улыбающийся смайл – грустный смайл
   № 2 женский силуэт - мужской силуэт

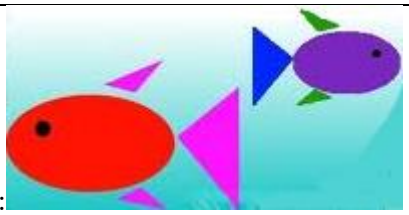
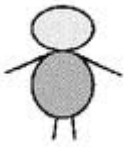
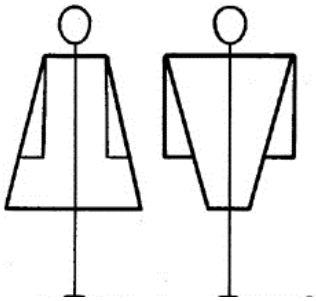
3. Создать проект, содержащий форму. При нажатии на кнопку «Показать» нарисовать рисунок:
   № 1 движущийся силуэт
   № 2 газированная вода
   № 3 моросящий дождь

4. Разработать проект, содержащий форму, на которую помещена кнопка. При нажатии на кнопку нарисовать строку текста выбранного цвета и шрифта, через 10 секунд цвет должен поменяться, смену повторить циклически 5 раз.
5. Разработка проекта, содержащего форму, на которую помещена кнопка. При нажатии на кнопку нарисовать мыльные пузыри, возникающие в разных местах формы.
6. Разработать проект, содержащий форму, на которую помещена кнопка. При нажатии на кнопку нарисовать прямоугольник, залитый градиентной заливкой от синего к красному, через 5 секунд цвет должен поменяться на градиентную заливку от желтого к зеленому.

Часть 2

В части номер 2 студент выбирает одну задачу. Номер задачи соответствует номеру студента в списке группы.

Номер задачи	**Задание**
1	Нарисовать:  Изменить: сделать из дерева лес, поменять заливку крон с задержкой на градиентную (желтая-зеленая, желтая-красная), далее на желтую.
2	Нарисовать:  Изменить: нарисовать дым из трубы.
3	Нарисовать:  Изменить: зажечь мерцающие огни на елке
4	 Нарисовать: Изменить: отправьте корабль в плавание
5	 Нарисовать: Изменить: сорвите лепестки с ромашки (задержка по времени)
6	 Нарисовать: Изменить: кошка машет хвостом

7	Нарисовать:  Изменить: рыбки плывут в разные стороны
8	Нарисовать:  Изменить: человечек подпрыгивает
9	Нарисовать:  Изменить: покрасить лепестки задержкой в разные цвета
10	Нарисовать:  Изменить: сделайте подмигивающий смайл
11	Нарисовать:  Изменить: разведите фигуры в разные стороны
12	 Нарисовать: Изменить: Сделать кошку полосатой (использовать HatchBrush)
13	Нарисовать и изменить с задержкой 
14	Нарисовать и изменить с задержкой 

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 16

РАБОТА С COM-ОБЪЕКТАМИ

Цель занятия: изучение механизма подключения COM-объектов в пользовательский проект; создание приложения Excel, рабочей книги, рабочего листа; работа с элементами коллекции.

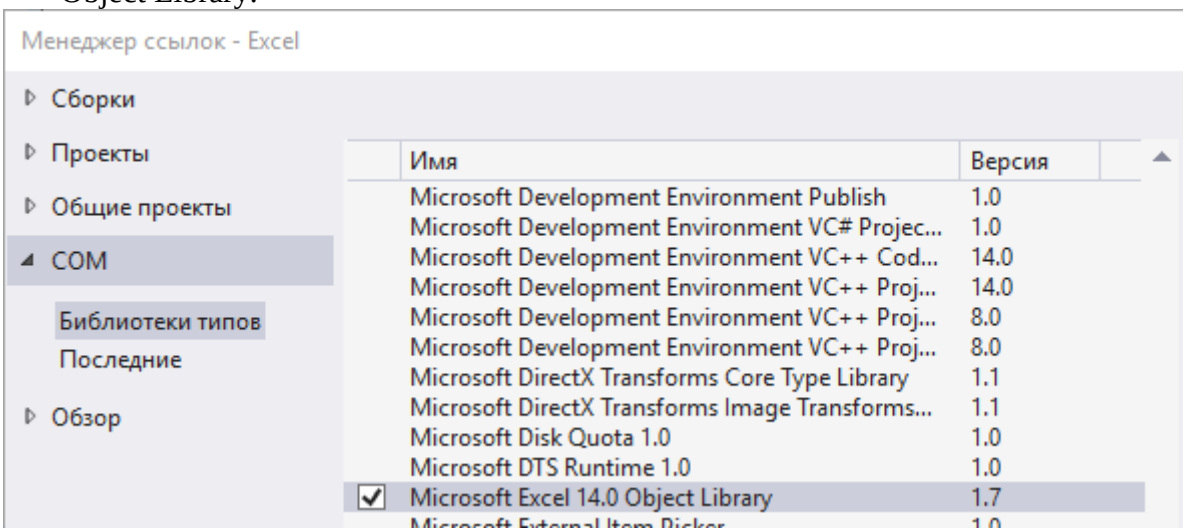
ХОД РАБОТЫ

Задание 1. Разработать приложение, создающее рабочую книгу Excel и заполняющее лист данными.

Алгоритм создания приложения:

1. Создайте Windows Forms приложение с именем COM_Excel.

1. В меню «Проект» на вкладке «Добавить ссылку» присоедините Microsoft Excel 14.0 Object Library.



2. На форме создайте кнопку «Excel». В файле кода Form1.cs добавьте:

```
using Excel = Microsoft.Office.Interop.Excel;
```

4. Добавьте следующий код для кнопки

```
//Создание приложения Excel
Excel.Application Excel_ = new Excel.Application();
          Excel_.Visible = true;
//Книга
Excel.Workbook WorkBook_;
            Excel.Worksheet Sheet_;
            WorkBook_ = Excel_.Workbooks.Add();
//Лист
            Sheet_ = (Excel.Worksheet)WorkBook_.Sheets[1];
//Присвоение имени листа
            Sheet_.Name = "Первый";
//Цикл для заполнения ячеек листа
            for (int i = 2; i < 20; i++)
                for (int j = 1; j < 20;j++)
                    Sheet_.Cells[i, j].Value = i * j;
//Присвоение значения отдельной ячейке листа
```

```
            Sheet_.Cells[20, 1].Value = "Значение";
//Определение цвета окантовки ячейки
           Sheet_.Cells.Borders.Color =
      Color.BlueViolet;//Определение типа линии окантовки
      ячейки
            Sheet_.Cells.Borders.LineStyle =Excel.
XlLineStyle.xlDash;
//определение ширины  ячейки
            Sheet_.Cells.ColumnWidth = 8;
//определение формулы в ячейке
            Sheet_.Cells[20,2].Formula = "=sin(A2)";
//объединение ячеек
           Sheet_.Range["A1", "s1"].Merge();
            Sheet_.Cells[1, 1].Value = "Таблица
значений";
 // выравнивание: 1- по значению; 2 - влево; 3-
посередине; 4- вправо;
            Sheet_.Cells.HorizontalAlignment = 3;
```

САМОСТОЯТЕЛЬНАЯ РАБОТА

Создайте форму для ввода требуемой информации. Спроектируйте пользовательский интерфейс. Из введенной информации сформируйте файл в Excel, содержащий таблицу. Перенесите названия форм ввода в заголовки таблицы. В заголовке таблицы укажите название таблицы. Обозначьте границы ячеек.

При нажатии на кнопку должна вводиться информация в следующую строку таблицы. Номер строки объявите как public.

Задания для индивидуальной работы:

1. Создайте форму для ввода анкетных данных студента.
2. Создайте форму для ввода анкетных данных преподавателя.
3. Создайте форму для ввода данных о группе колледжа.
4. Создайте форму для ввода анкетных данных абитуриента.
5. Создайте форму для ввода плана работы колледжа.
6. Создайте форму для ввода посещаемости студента в сессию.
7. Создайте форму для ввода дисциплин специальности.
8. Создайте форму для ввода успеваемости студента в сессию.
9. Создайте форму для ввода расписания занятий группы.
10. Создайте форму для ввода данных об аудиториях колледжа.
11. Создайте форму для ввода данных в телефонный справочник.
12. Создайте форму для ввода данных о технической оснащенности колледжа.
13. Создайте форму для ввода данных об оснащенности колледжа компьютерами.
14. Создайте форму для ввода данных о сотрудниках колледжа.
15. Создайте форму для ввода данных о методическом обеспечении специальности.
16. Создайте форму для ввода данных о кураторах групп.
17. Создайте форму для ввода данных о поставщиках товаров.
18. Создайте форму для ввода данных о мероприятиях колледжа.
19. Создайте форму для ввода данных о сотрудниках колледжа.
20. Создайте форму для ввода данных о дисциплинах специальности колледжа.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 17

ПРОГРАММИРОВАНИЕ БАЗ ДАННЫХ

Цель занятия: изучение возможности подключения к базе данных, созданной в MS ACCESS с помощью механизма ADO.NET: создание базы данных с требуемыми параметрами; создание обработчика для подключения к базе данных; выдача полученных данных в элементы управления; изучение возможности метода ExecuteNonQuery.

ТЕОРЕТИЧЕСКИЙ МАТЕРИАЛ

Чтение из базы данных

Для соединения с базой данных MS Access (файл .accdb) в C# следует использовать класс OleDbConnection со следующими параметрами соединения:

provider= Microsoft.Ace.OLEDB.12.0[3];data source=databaseFile

Здесь databaseFile — абсолютный путь к файлу базы данных Access. Провайдер соединения должен иметь значение Microsoft.Ace.OLEDB.12.0.

Метод ExecuteNonQuery

Метод ExecuteNonQuery выполняет инструкцию SQL применительно к свойству Connection. Метод ExecuteNonQuery можно применять для выполнения операций с каталогами, например, для запроса структуры базы данных, или для создания объектов базы данных, таких как таблицы, или для изменения данных в базе данных без использования DataSet путем выполнения операторов UPDATE, INSERT или DELETE.

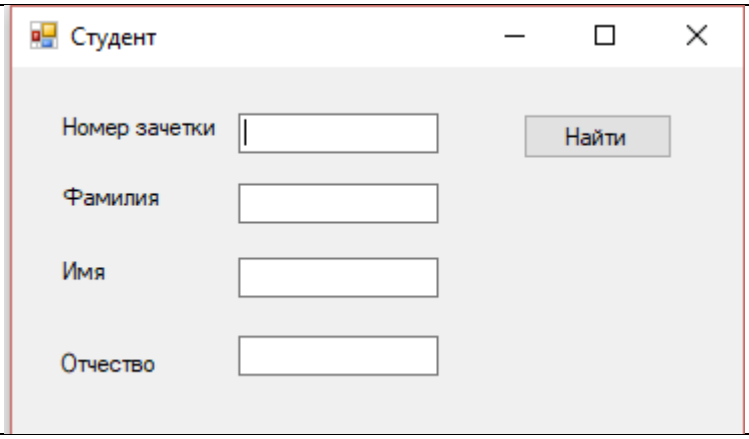
ХОД РАБОТЫ

Задание 1. Создать базу данных Колледж.accdb. в таблице Студент создать поля: `НомерЗачетки, Фамилия, Имя, Отчество.` Создать проект, выводящий информацию о студенте по номеру его зачетки.

Алгоритм создания приложения:

1. Для начала следует создать базу данных и заполнить базу содержимым.
2. Создайте проект с именем “Приложение Windows”.
3. Перетащите по 4 элемента управления textbox и label, 1 элемент управления button и 1 oleDbConnection из панели элементов в свою форму.

[3] Число зависит от версии программного продукта



4.Для элемента управления button создайте событие button1_Click.
Не забудьте добавить using System.Data.OleDb;

5. Вставьте следующий код в обработчик события.

```
string sql = String.Concat("SELECT * FROM Студент
WHERE НомерЗачетки=", textBox1.Text);
string connectionString;
connectionString = "Provider=Microsoft.Ace.OLEDB.12.0;"
+              @"Data Source= Колледж.accdb";

OleDbConnection connection = new
OleDbConnection(connectionString);
connection.Open();

OleDbCommand command = new OleDbCommand(sql,
connection);
OleDbDataReader dataReader = command.ExecuteReader();
        // Очистка от старых записей
        //Вывод найденной записи

 textBox2.Text = "";
 textBox3.Text = "";
 textBox4.Text = "";
 while (dataReader.Read())
  {
    textBox2.Text = textBox2.Text + dataReader["Фамилия"];
    textBox3.Text = textBox3.Text + dataReader["Имя"];
    textBox4.Text = textBox4.Text + dataReader["Отчество"];
  }
        // Очистка
 dataReader.Close();
 connection.Close();
```

6. Для работы с listBox, comboBox

```
comboBox.Items.Add(dataReader[“Фамилия”]);
listBox.Items.Add(dataReader[“Фамилия”]);
```

Обязательно: в `НомерЗачетки` – числовое значение!

Для сравнения с текстовым значением сравниваемый параметр должен быть в апострофах:

```
SELECT Фамилия, Имя, Отчество FROM student WHERE НомерЗачетки = номер
SELECT Фамилия, Имя, Отчество FROM student WHERE Фамилия = ‘Иванов’.
```

Задание 2. Создайте кнопки в задаче №1, позволяющие вносить данные в базу данных, а также удалять данные и изменять их.

Алгоритм создания приложения:

1. Настроить свойства компонентов	1. button2 / Text / Добавить данные 2. button3/ Text / Удалить 3. button4/ Text / Изменить
2. Вставить следующий код в обработчик события кнопки «Добавить данные» для добавления строк в таблицу БД	`OleDbConnection con = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=Колледж.accdb");` `con.Open();` `string queryString = "insert into Студент (НомерЗачетки, Фамилия, Имя, Отчество) values('" + textBox1.Text + "','" + textBox2.Text + "','" + textBox3.Text + "','" + textBox4.Text +"')";` `OleDbCommand command = new OleDbCommand(queryString, con);` `MessageBox.Show(queryString);` `command.ExecuteNonQuery();` `con.Close();`
3. Для удаления строк в таблице БД вставьте следующий код	`OleDbConnection con = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=Колледж.accdb");` `con.Open();` `// 1 способ – для одного номера зачетки` `// string queryString = "delete FROM Студент WHERE НомерЗачетки = 14";` `// 2 способ – для любого указанного номера зачетки` `string queryString = String.Concat("delete FROM Студент WHERE НомерЗачетки = ", textBox1.Text);` `OleDbCommand command = new OleDbCommand(queryString, con);` `MessageBox.Show(queryString);` `command.ExecuteNonQuery();` `con.Close();`
4. Для изменения строк в таблице БД вставьте следующий код в обработчик события	`OleDbConnection con = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=Колледж.accdb");` `con.Open();` `string queryString = String.Concat("UPDATE Студент SET Фамилия = '" + textBox2.Text + "' WHERE НомерЗачетки = ", textBox1.Text);` `OleDbCommand command = new OleDbCommand(queryString, con);` `MessageBox.Show(queryString);` `command.ExecuteNonQuery();` `con.Close();`

САМОСТОЯТЕЛЬНАЯ РАБОТА

1. Разработайте форму для программного продукта, поместите элементы управления для работы с базой данных. Вывести из БД информацию на форму.
2. Создайте функции модификации, вставки и удаления записей.

Задания для практической работы указаны в практической работе №16

Результат выполнения задания:

Программный продукт с подключенной базой данных, с возможностью вставки, модификации и удаления записей в таблицах.

Что нужно знать и уметь делать после выполнения практического занятия:

1. Уметь подключать базу данных, используя элементы управления.
2. Уметь осуществлять вывод данных в текстовые поля.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 18-19

РАЗРАБОТКА ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ С ПОМОЩЬЮ VISUAL STUDIO

Цель занятия: изучение элементов управления maskedTextBox, toolTip, toolStrip, bindingNavigator, menuStrip, contextMenuStrip, statusStrip для построения пользовательского интерфейса.

ТЕОРЕТИЧЕСКИЙ МАТЕРИАЛ

Класс MaskedTextBox является усовершенствованной версией элемента управления TextBox, которая поддерживает декларативный синтаксис, на основе которого принимаются или отклоняются данные, вводимые пользователем. С помощью свойства Mask можно задать следующие типы входных данных, не создавая в приложении специальный код для проверки.

Символ маски	Описание
0	Цифра, ввод обязателен. Пользователь вправе ввести любую цифру из диапазона 0-9
9	Цифра или пробел. Ввод необязателен.
#	Цифра, пробел, плюс или минус. Ввод необязателен.
L	Буква, ввод обязателен.
?	Буква, ввод необязателен.
&	Символ (в т.ч. Unicode), ввод обязателен. Любой не управляющий символ. При значении true свойства AsciiOnly данный символ маски совершенно аналогичен символу “L”
C	Символ (в т.ч. Unicode), ввод не обязателен. Любой не управляющий символ. При значении true свойства AsciiOnly данный символ маски совершенно аналогичен символу “?”
A	Буквенно-цифровой символ (Unicode не принимается), ввод обязателен. При значении true свойства AsciiOnly данный символ маски позволяет ввод лишь ASCII букв a-z и A-Z.
A	Буквенно-цифровой символ (Unicode не принимается), ввод опциональный. При свойстве AsciiOnly приравненным к true данный символ маски позволяет ввод лишь ASCII букв a-z и A-Z.
. (точка)	Место, где будет отображен десятичный разделитель. Какой символ будет использоваться, определяется свойством Culture control-a.
, (запятая)	Место, где будет отображен разделитель тысяч. Какой символ будет использоваться, определяется свойством Culture control-a.
: (двоеточие)	Место, где будет отображен разделитель времени. Какой символ будет использоваться, определяется свойством Culture control-a.
/ (прямой слеш)	Место, где будет отображен разделитель даты. Какой символ будет использоваться, определяется свойством Culture control-a.
$	Место, где будет отображен символ валюты. Какой символ будет использоваться, определяется свойством Culture control-a.
<	Весь ввод пользователя после этого символа маски автоматически приводится к нижнему регистру.
>	Весь ввод пользователя после этого символа маски автоматически приводится к верхнему регистру.
|	С этого места отменить действие любого из двух предыдущих символов маски. Принимать ввод пользователя в том регистре, в каком он его осуществляет.

richTextBox Данный элемент управления дает возможность пользователю вводить и обрабатывать большие объемы информации (более 64 кБт). RichTextBox включает все возможности текстового редактора Microsoft Word.

listBox — простейший вариант пролистываемого списка. Он позволяет выбирать один или несколько хранящихся в списке элементов. Кроме того, ListBox имеет возможность отображать данные в нескольких колонках. Это позволяет представлять данные в большем объеме.

toolTip – всплывающая подсказка широко используется в Windows-приложениях, позволяет добавить подсказку практически любому компоненту, наследующему класс Control.

Элемент управления **BindingNavigator** является специализированным элементом управления **ToolStrip**, который предназначен для навигации и управления привязанными к данным элементами управления формы.

Для создания профессиональных интерфейсных элементов Windows-приложений, соответствующих последним требованиям эргономики и схожих с интерфейсными элементами приложений компании Microsoft, предлагается использовать набор интерфейсных компонентов:

• MenuStrip — компонент для создания основных меню приложений;
• ContextMenuStrip — компонент для создания контекстных меню приложений;
• StatusStrip — компонент для создания статусных панелей;
• ToolStrip — компонент для создания панелей инструментов.

Компонент ToolStrip

Данный компонент служит в качестве базового класса для компонентов, реализующих меню и статусные панели, а также для создания панелей инструментов. Создаваемые с помощью компонента ToolStrip панели похожи на инструментальные панели в таких продуктах, как Microsoft Office, и обладают широкими возможностями настройки, включая поддержку визуальных шаблонов, а также поддерживают динамическое изменение расположения элементов.

Компонент ToolStrip может служить контейнером для следующих компонентов:

• ToolStripButton — реализует кнопку на инструментальной панели;
• ToolStripComboBox — реализует список на инструментальной панели;
• ToolStripSplitButton — реализует кнопку-разделитель на инструментальной панели;
• ToolStripLabel — реализует элемент инструментальной панели, который может содержать текст, графическое изображение или гиперссылку;
• ToolStripSeparator — реализует разделитель на инструментальной панели;
• ToolStripDropDownButton — реализует кнопку с раскрытием вложенных элементов на инструментальной панели;
• ToolStripTextBox — реализует строку ввода текста на инструментальной панели.

Помимо этого можно использовать компонент ToolStripControlHost для размещения в инструментальной панели любых других компонентов Windows Forms.

Компонент ToolStrip обеспечивает базовую функциональность инструментальных панелей, включая отрисовку элементов, обработку событий от мыши и клавиатуры, поддержку операций drag-and-drop и т.п. Для расширения функциональности панелей инструментов можно воспользоваться классом ToolStripManager, чтобы объединить элементы панели в специальные горизонтальные или вертикальные области. Для получения более полного контроля над стилями и отрисовкой панелей можно использовать класс ToolStripRenderer.

Компонент MenuStrip

Компонент MenuStrip представляет собой контейнер для создания структуры меню, каждый элемент которого реализуется с помощью компонента ToolStripMenuItem. Каждый такой элемент может либо представлять конкретную команду, либо содержать дочерние элементы.

Компонент MenuStrip может содержать компоненты ToolStripMenuItem, ToolStripComboBox, ToolStripSeparator и ToolStripTextBox.

Компонент ContextMenuStrip

Данный компонент используется для реализации контекстных меню, которые отображаются, когда пользователь щелкает правой кнопкой мыши по интерфейсному элементу или определенной области формы. Для связи интерфейсных элементов или формы с контекстным меню используется свойство ContextMenuStrip соответствующего элемента или формы.

Контекстное меню может состоять из следующих элементов: ToolStripMenuItem, ToolStripComboBox, ToolStripSeparator и ToolStripTextBox.

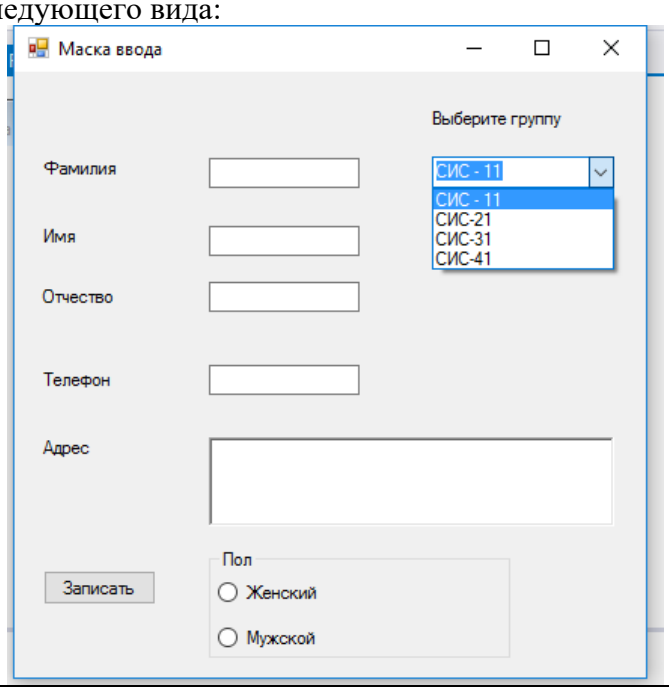
Компонент StatusStrip

Этот компонент используется для создания статусных панелей и обычно состоит из нескольких элементов StatusStripPanel, добавляемых с помощью метода AddRange, каждый из которых может содержать либо текст, либо графическое изображение, либо и то и другое. Статусная панель также может содержать компоненты ToolStripLabel и ToolStripProgressBar.

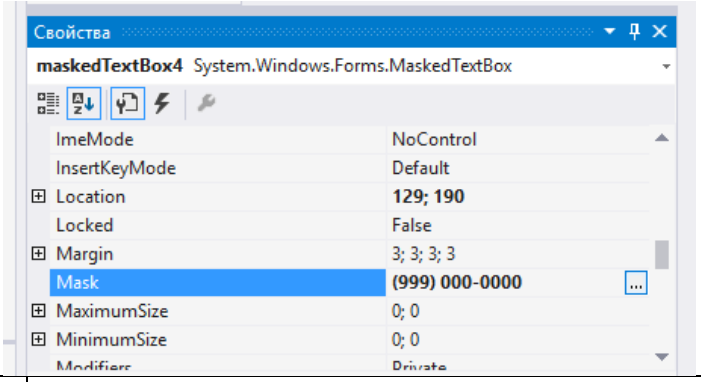
ХОД РАБОТЫ

Задание 1. Создать проект, с предложенным интерфейсом, позволяющий вводить информацию о студенте группы.

Алгоритм создания приложения:

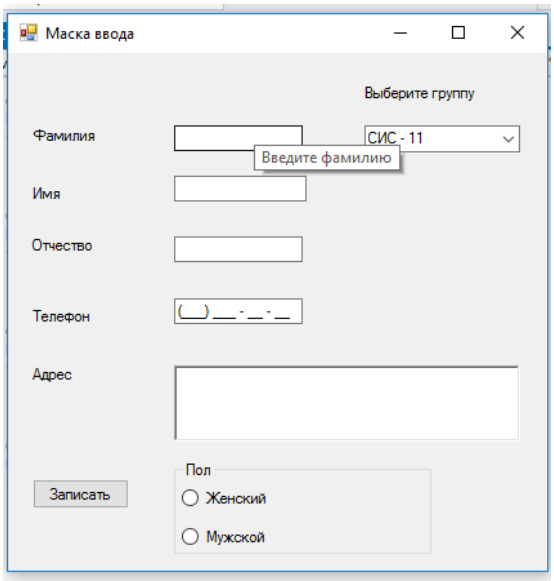
1. Создайте проект «Приложение Windows Forms».
2. Перетащите элементы управления: 3 button, maskedTextBox, richTextBox, comboBox, toolTip из панели элементов в свою форму
3. Создайте форму следующего вида: 

Для создания маски телефона используйте компонент maskedTextBox / свойство Mask и выберите нужный шаблон.



4.Вставьте следующий код в обработчик события формы	comboBox1.Items.Add(“СИС-11”); combBox1.Items.Add(“СИС-21 “); combBox1.Items.Add(“СИС-31”); combBox1.Items.Add(“СИС-41”); combBox1.SelectedIndex = 0;
5.Запрограммируйте radioButton	if (radioButton1.Checked) { MessageBox.Show (“Женский”); } else if (radioButton2.Checked) { MessageBox.Show (“Мужской”); }
6. Используйте ToolTip	ToolTip tip = new ToolTip(); tip.SetToolTip (textBox1, “Введите фамилию”); tip.SetToolTip (textBox2, “Введите имя”); tip.SetToolTip (textBox3, “Введите отчество”); tip.SetToolTip (maskedTextBox1, “Введите номер телефона”); tip.SetToolTip (richTextBox1, “Введите адрес”); tip.SetToolTip (comboBox1, “Выберите номер группы”); tip.SetToolTip (groupBox1, “Введите пол”);

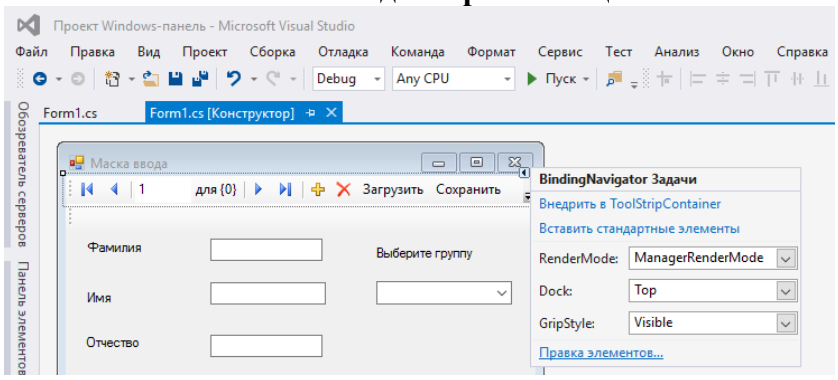
7. Результат:



8. Запретите ввод недопустимых символов в поля: фамилия, имя, отчество.

Задание 2. Добавление кнопок загрузки, сохранения и отмены в элемент управления BindingNavigator в формах Windows Forms.

Алгоритм создания приложения:

<table>
<tr><td colspan="2">1. Создайте проект «Приложение Windows».</td></tr>
<tr><td colspan="2">Примечание: так как это элемент управления ToolStrip, компонент BindingNavigator может быть легко изменен для включения дополнительных или альтернативных команд для пользователя.
В следующей процедуре элемент управления textBox привязан к данным, а добавленный в форму элемент управления ToolStrip дополнен кнопками загрузки, сохранения и отмены.</td></tr>
<tr><td colspan="2">2. Добавьте элемент управления textBox на форму.</td></tr>
<tr><td>3.Привяжите его к классу BindingSource, который привязан к источнику данных</td><td>Свойства BindingSourse1 / DataSourse / выбираете источник данных</td></tr>
<tr><td colspan="2">4.Когда таблица данных и адаптер таблиц будут сгенерированы, перетащите элемент управления bindingNavigator на форму</td></tr>
<tr><td>5 Присвойте свойству BindingSource элемента управления BindingNavigator значение BindingSource</td><td>bindingNavigator /bindingSource =bindingSource1</td></tr>
<tr><td>6 Добавьте кнопки</td><td>bindingNavigator / Щелкните (▶) и выберите пункт Изменение элементов. Появится окно Редактор коллекции элементов.
 

В этом окне выполните следующие действия:
1) Добавьте ToolStripSeparator и три элемента ToolStripButton, выбрав соответствующий тип элемента ToolStripItem и нажав кнопку Добавить.
2) Присвойте свойству кнопок Name соответственно значения LoadButton, SaveButton и CancelButton.
3) Присвойте свойству кнопок Text значения Загрузить, Сохранить и Отмена.
4) Присвойте свойству DisplayStyle каждой кнопки значение Text. Также можно присвоить этому свойству значение Image или ImageAndText и задать отображаемое изображение в свойстве Image.
5) Нажмите кнопку ОК, чтобы закрыть диалоговое окно. Кнопки будут добавлены в класс ToolStrip.</td></tr>
</table>

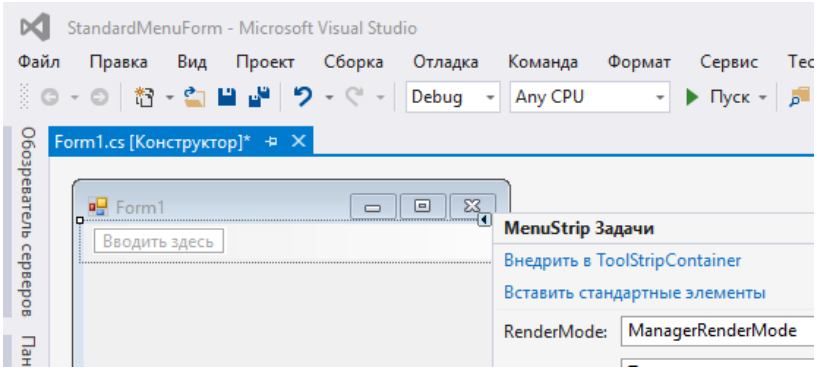
Задание 3. Создать проект, содержащий меню и строку состояния.

Алгоритм создания приложения:

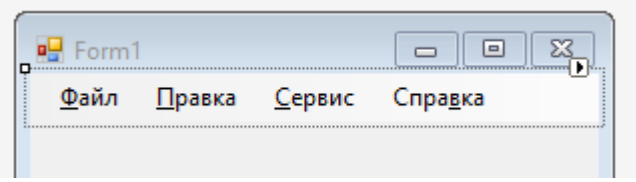
1. Создать проект и подготовить форму. Создайте проект "Приложение Windows" с названием StandardMenuForm.

2. Перетащите элемент управления MenuStrip из панели элементов в свою форму.

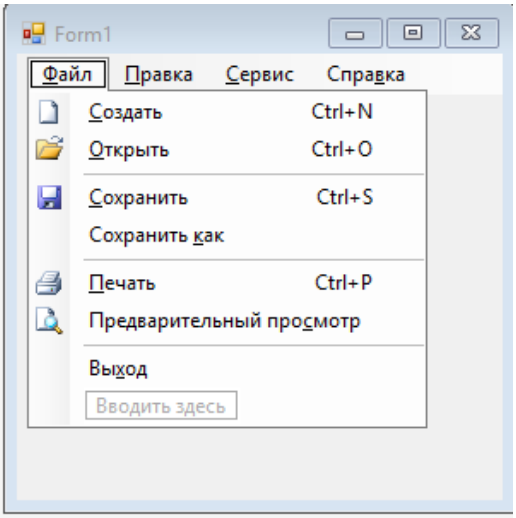
3. Щелкните для элемента управления MenuStrip «Свойства» и выберите «Вставить стандартные элементы».



4. Элемент управления MenuStrip заполняется стандартными элементами меню.

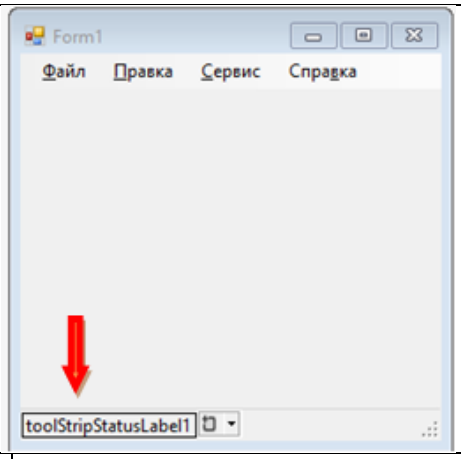
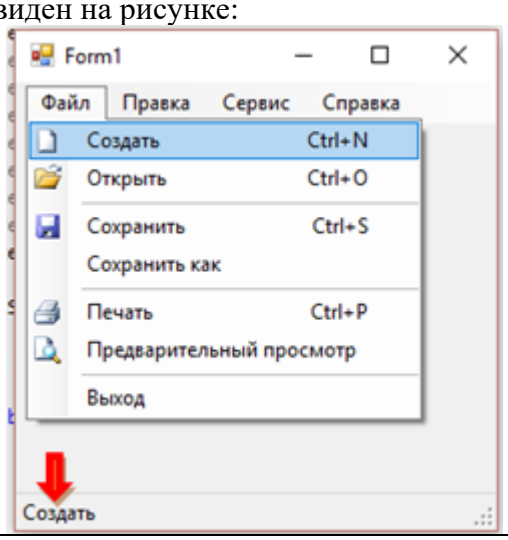
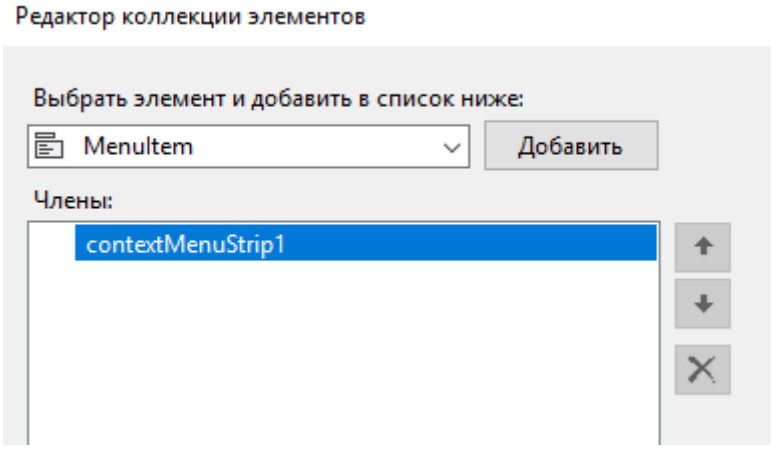


5. Щелкните меню Файл для просмотра элементов меню по умолчанию и соответствующих им значков.

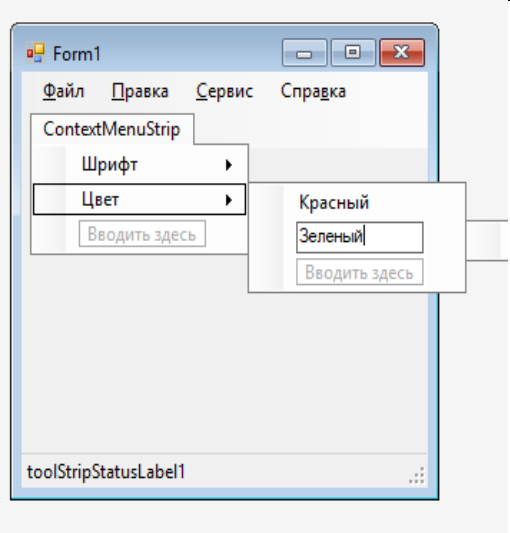


6. Перетащите элемент управления StatusStrip из панели элементов в свою форму. Элемент управления StatusStrip автоматически располагается в нижней части формы.

7. Нажмите кнопку раскрывающегося списка для элемента управления StatusStrip и выберите StatusLabel для добавления элемента управления ToolStripStatusLabel в элемент управления StatusStrip.

<table>
<tr><td></td><td> 
</td></tr>
<tr><td>8. Обработайте событие MouseEnter, чтобы оно реагировало на выделение элемента меню пользователем</td><td>1 Выберите пункт «Создать» в меню «Файл», созданный при создании стандартного раздела меню.
2 В окне «Свойства» щелкните «События».
3 Дважды щелкните событие MouseEnter. Конструктор Windows Forms генерирует обработчик событий для события MouseEnter</td></tr>
<tr><td>9. Вставьте следующий код в обработчик события</td><td><code>toolStripStatusLabel1.Text = «Создать»;</code>
Результат виден на рисунке:
 
</td></tr>
<tr><td>10. Перетащите элемент управления ContextMenuStrip из панели элементов в свою форму и создайте пункты меню</td><td>1. Настраиваем свойство contextMenuStrip1 / Items / Коллекция
2. В окне добавляем пункты меню
 

3. Изменить свойство Text для каждого созданного пункта меню – «Шрифт», «Цвет»
4. Создаем подпункты меню</td></tr>
</table>

	
11 Привязать контекстное меню к форме	Выделить форму / свойство ContextMenuStrip = contextMenuStrip1
12 Обработайте событие, чтобы оно реагировало на выделение элемента меню пользователем	`this.BackColor = Color.Red;`

САМОСТОЯТЕЛЬНАЯ РАБОТА

Задание: разработайте начальную форму для разрабатываемого программного продукта, содержащую пункты меню:

- Ввод первоначальных данных;
- Корректировка данных;
- Выходные документы;
- Справка;
- О программе;
- Выход.

Сформируйте обработчики событий для каждого пункта меню**.** В пункте меню **Ввод первоначальных данных и Корректировка данных** сформируйте формы для ввода и корректировки информации. В пункте меню **Выходные документы** осуществите выбор формы выходного документа. Заполните данные в форме **О программе.** В пункте меню **Справка** осуществите вывод окна MessageBox с краткой информацией о разрабатываемом программном продукте.

При разработке начальной формы применить компоненты: maskedTextBox, richTextBox, comboBox, toolTip, toolStrip и др.

Задания указаны в практической работе №16

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ№ 20

ПОСТРОЕНИЕ КЛАССОВ С ПОМОЩЬЮ VISUAL STUDIO

Цель занятия: изучение принципов построения классов.
Содержание занятия:

1.Добавление классов.
2. Работа с элементами класса.

ТЕОРЕТИЧЕСКИЙ МАТЕРИАЛ

Тип, который определен как *класс*, является *ссылочным типом*. Когда во время выполнения вы объявляете переменную ссылочного типа, такая переменная будет содержать значение *NULL*, пока вы явным образом не создадите экземпляр класса с помощью оператора *new* или не назначите его объекту совместимого типа, созданному в другом месте, как показано в следующем примере:

```
// Объявление объекта типа MyClass
MyClass mc = new MyClass();

// Объявление другого объекта того же типа, присваивая ему значение первого объекта
MyClass mc2 = mc;
```

При создании объекта выделяется достаточный объем памяти для этого объекта в управляемой куче, и переменная хранит только ссылку на расположение данного объекта. Хранение типов в управляемой куче требует дополнительных действий, как при выделении памяти, так и при удалении, которое выполняется функцией автоматического управления памятью в среде CLR, известной как сборка мусора. Сборка мусора является хорошо оптимизированным процессом и в большинстве случаев не создает помех для производительности.

Классы объявляются с помощью ключевого слова class, за которым следует уникальный идентификатор, как показано в следующем примере:

```
public class Customer
{
   Поля, свойства, методы и события
}
```

Ключевому слову `class` предшествует уровень доступа. Так как в этом случае используется открытый класс (public), любой пользователь может создавать его экземпляры. За именем класса следует ключевое слово `class`. Имя класса должно быть допустимым именем идентификатора C#. Оставшаяся часть определения — это тело класса, в котором задаются данные и поведение. Поля, свойства, методы и события в классе собирательно называются *членами класса*.

Классы полностью поддерживают *наследование*, фундаментальный механизм объектно-ориентированного программирования. Создаваемый класс может наследовать от любого другого интерфейса или класса, а другие классы могут наследовать от этого класса и переопределять его виртуальные методы.

При наследовании создается производный класс, то есть класс объявляется с помощью базового класса, от которого он наследует данные и поведение. Базовый класс задается добавлением после имени производного класса двоеточия и имени базового класса, как показано далее:

```
public class Manager : Employee
```

```
        {
          // Поля, свойства, методы и события, наследуемые от  Employee
          // Новые поля, свойства, методы и события нового класса Manager
        }
```

Когда класс объявляет базовый класс, он наследует все члены базового класса, за исключением конструкторов. В отличие от C++, класс в C# может только напрямую наследовать от одного базового класса. Тем не менее, поскольку базовый класс может сам наследовать от другого класса, класс может косвенно наследовать от нескольких базовых классов. Кроме того, класс может напрямую реализовать несколько интерфейсов.

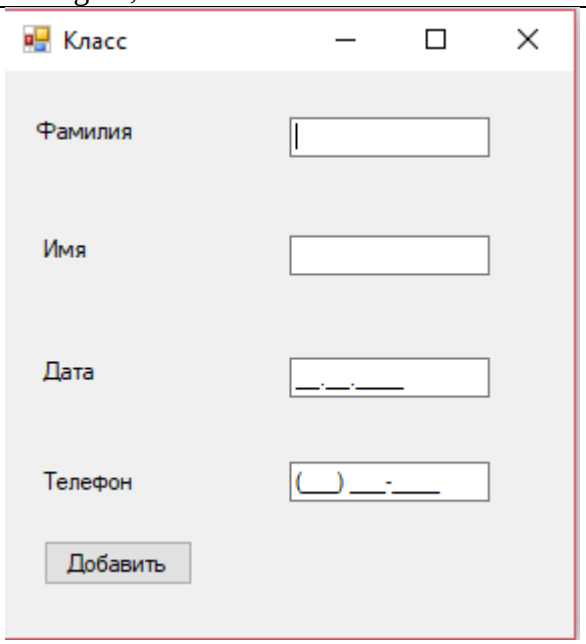
Задание 1. Создать консольное приложение, содержащее класс.

ХОД РАБОТЫ:

```
namespace класс_Персона
{
    public class Person
    {
        // Конструктор, который не принимает аргументов:
        public Person()
        {
            Name = "имя";
        }

        // Конструктор, с одним аргументом:
        public Person(string name)
        {
            Name = name;
        }

       // Авто-реализованное свойство readonly
        public string Name { get; }

        // Метод, который переопределяет реализацию базового класса (System.Object).
        public override string ToString()
        {
            return Name;
        }
    }
  class TestPerson
    {
        static void Main()
        {
            // Вызов конструктора, который не имеет параметров
            var person1 = new Person();
            Console.WriteLine(person1.Name);

            // Вызов конструктора с одним параметром
            var person2 = new Person("Иванов Иван");
            Console.WriteLine(person2.Name);

            // Получите строковое представление экземпляра person2.
            Console.WriteLine(person2);
            Console.WriteLine("Нажмите клавишу для выхода.");
            Console.ReadKey();
        }
    }
 }
```

Задание 2. Добавить в Windows приложение класс.

Алгоритм создания приложения:

1. В классе определить объект класса	В меню Проект выберите вкладку «Добавить класс». public string FirstName; public string LastName; public string date; public string tel;
2. В основном модуле создайте форму	 Добавьте два компонента maskedTextBox, установив маски даты и номера телефона
3. Добавьте код для button1	Class1 sampleClass1 = new Class1(); sampleClass1.FirstName = textBox1.Text; sampleClass1.LastName = textBox2.Text; sampleClass1.date = maskedTextBox1.Text; sampleClass1.tel = maskedTextBox2.Text;
4. Создайте обработчик события, записывающий введенную информацию в текстовый файл.	

САМОСТОЯТЕЛЬНАЯ РАБОТА

Каждый студент выполняет одно задание, выбрав номер варианта в соответствии со своим номером в списке группы.

Вариант 1

Создайте программу с классом Student, который включает в себя следующие данные – элементы о студентах университета:

- ФИО;
- год поступления;
- курс;
- номер группы;
- размер стипендии;
- оценки по N предметам.

Номер для каждого создаваемого студента должен запрашиваться, а массив оценок за последнюю сессию создается в динамической памяти операцией new.

В состав класса входят следующие функции-члены класса:

- конструктор с параметрами;
- деструктор;
- функции просмотра текущего объекта;
- функция - друг, которая вычисляет средний балл и выводит ФИО студента, у которого он максимальный;
- виртуальная функция просмотра текущего объекта.

Производный класс Student_1 содержит следующие элементы:

- общественная работа;
- процент надбавки к стипендии;
- переопределенную функцию вывода данных об общественниках.

Вариант 2

Создайте программу с классом Firm, который включает в себя следующие данные – элементы о сотрудниках фирмы:

- ФИО сотрудника;
- табельный номер;
- количество отработанных часов за месяц;
- почасовой тариф.

Номер для каждого создаваемого студента должен запрашиваться, а массив оценок за последнюю сессию создается в динамической памяти операцией new.

В состав класса входят следующие функции-члены класса:

- конструктор с параметрами;
- деструктор;
- функции просмотра текущего состояния объекта;
- функция - друг, которая позволяет вывести размер заработанной платы каждого сотрудника фирмы, за вычетом подоходного налога, который составляет 13% от суммы заработка. Необходимо учесть, что рабочее время свыше 144 часов считается сверхурочным и оплачивается в двойном размере;
- виртуальная функция просмотра текущего объекта.

Производный класс Firm _1 содержит следующие элементы:

- количество командировочных дней в течение месяца;
- процент надбавки к зарплате за каждый день командировки;
- переопределенную функцию просмотра состояния объектов – сотрудников, которые в текущем месяце побывали в командировке.

Вариант 3

Создайте программу с классом Abitur, который включает в себя следующие данные – элементы об абитуриентах, сдавших вступительные экзамены в университет:

- ФИО;
- адрес;
- оценки по предметам.

Номер для каждого абитуриента запрашиваться, а массив оценок создается в динамической памяти.

В состав класса входят следующие функции-члены класса:

- конструктор с параметрами;
- деструктор;
- функции просмотра текущего объекта;

- функция – друг, позволяющая определить количество абитуриентов, проживающих в Москве;

- виртуальная функция просмотра текущего объекта.

Производный класс Abitur _1 содержит следующие данные о льготах:

- тип медали (золотая или серебряная);

- призовое место на Всероссийской олимпиаде;

- переопределенную функцию просмотра состояния объектов-абитуриентов, имеющих льготы при поступлении.

Вариант 4

Создайте программу с классом Student, который включает в себя следующие данные – элементы о студентах, желающих получить места в общежитии. Общежитие в первую очередь предоставляется тем студентам, у кого доход на члена семьи меньше двух минимальных зарплат. Класс включает в себя следующие данные-элементы:

- ФИО студента;

- номер группы (буква и четы;

- средний балл;

- доход на одного члена семьи.

В состав класса входят следующие функции-члены класса:

- конструктор с параметрами;

- деструктор;

- функции просмотра текущего объекта;

- функция - друг, которая должна вывести ФИО студентов со средним баллом выше указанного с клавиатуры;

- виртуальная функция просмотра текущего объекта.

Производный класс Student_1 содержит следующие элементы:

- вид общественной нагрузки;

- количество членов семьи;

- переопределенную функцию просмотра состояния объектов – студентов-внеочередников.

Вариант 5

Создайте программу с классом Bas, который включает в себя следующие данные – элементы об автобусных рейсах:

- номер рейса;

- тип автобуса;

- пункт назначения;

- цена билета;

- время отправления;

- время прибытия на конечный пункт.

В состав класса входят следующие функции-члены класса:

- конструктор с параметром (номер рейса);

- деструктор;

- функции просмотра текущего объекта;

- функция установки текущего состояния объектов для остальных элементов;

- функция - друг должна вывести информацию о рейсах, позволяющих добраться до указанного с клавиатуры пункта;

- виртуальная функция просмотра текущего объекта.

Производный класс Express содержит следующие данные об автобусах-экспрессах:

- дни недели работы;

- процентная надбавка на цену билета;
- выигрыш во времени;
- переопределенную функцию вывода данных об общественниках.

Вариант 6

Создайте программу с классом Team, который включает в себя следующие данные – элементы об участниках спортивных соревнований:

- ФИО игрока;
- игровой номер;
- возраст;
- рост;
- вес.

Номер для каждого создаваемого студента должен запрашиваться, а массив оценок за последнюю сессию создается в динамической памяти операцией new.

В состав класса входят следующие функции-члены класса:

- конструктор с параметрами;
- деструктор;
- функции просмотра текущего объекта;
- функция - друг, которая выведет информацию о самом легком спортсмене в команде;
- виртуальная функция просмотра текущего объекта.

Производный класс Inform содержит следующие данные:

- разряд спортсмена;
- категория;
- призер (Европы, мира и т.д.).
- переопределенную функцию просмотра состояния объекта.

Вариант 7

Создайте программу с классом Phone, который включает в себя следующие данные – элементы о разговорах на международной АТС:

- дату разговора;
- код и название города;
- продолжительность разговора;
- тариф;
- номер телефона в этом городе;
- номер телефона абонента.

В состав класса входят следующие функции-члены класса:

- конструктор по умолчанию;
- деструктор;
- функция установки текущего состояния объекта;
- функции просмотра текущего объекта;
- функция, которая позволяет вывести по номеру телефона название города;
- функция - друг, которая подсчитает суммарное время разговора с указанным городом;
- виртуальная функция просмотра текущего объекта.

Производный класс Phone _1 содержит следующие элементы:

- вид льгот;
- процент для льготного тарифа;
- надбавка к тарифу за срочность вызова;
- переопределенную функцию просмотра состояния объекта.

Вариант 8

Создайте программу с классом Goods, который включает в себя следующие данные – элементы о товарах, имеющихся на складе:

- страна-изготовитель;
- фирма-изготовитель;
- наименование товара;
- количество единиц товара.

В состав класса входят следующие функции-члены класса:

- конструктор с параметром для инициализации страны-изготовителя;
- деструктор;
- функция инициализации текущего состояния объектов остальных элементов;
- функции просмотра текущего объекта;
- функция подсчета общего количества товара указанной фирмы;
- функция - друг, которая позволяет вывести товары и их данные для указанной страны;
- виртуальная функция просмотра текущего объекта.

Производный класс Order содержит следующие элементы:

- страна-заказчик;
- дата заказа;
- количество заказанного товара;
- переопределенную функцию вывода данных об общественниках.

Вариант 9

Создайте программу с классом Student, который включает в себя следующие данные – элементы о студентах университета:

- ФИО;
- год поступления;
- курс;
- номер группы;
- размер стипендии;
- оценки по N предметам.

Номер для каждого создаваемого студента должен запрашиваться, а массив оценок за последнюю сессию создается в динамической памяти операцией new.

В состав класса входят следующие функции-члены класса:

- конструктор с параметрами;
- деструктор;
- функции просмотра текущего объекта;
- функция - друг , которая вычисляет средний балл и выводит ФИО студента, у которого он максимальный;
- виртуальная функция просмотра текущего объекта.

Производный класс Student_1 содержит следующие элементы:

- общественная работа;
- процент надбавки к стипендии;
- переопределенную функцию вывода данных об общественниках.

Вариант 10

Создайте программу с классом Firm, который включает в себя следующие данные – элементы о сотрудниках фирмы:

- ФИО сотрудника;
- табельный номер;
- количество отработанных часов за месяц;
- почасовой тариф.

Номер для каждого создаваемого студента должен запрашиваться, а массив оценок за последнюю сессию создается в динамической памяти операцией new.

В состав класса входят следующие функции-члены класса:

- конструктор с параметрами;
- деструктор;
- функции просмотра текущего состояния объекта;
- функция - друг, которая позволяет вывести размер заработанной платы каждого сотрудника фирмы, за вычетом подоходного налога, который составляет 13% от суммы заработка. Необходимо учесть, что рабочее время свыше 144 часов считается сверхурочным и оплачивается в двойном размере;
- виртуальная функция просмотра текущего объекта.

Производный класс Firm _1 содержит следующие элементы:

- количество командировочных дней в течение месяца;
- процент надбавки к зарплате за каждый день командировки;
- переопределенную функцию просмотра состояния объектов – сотрудников, которые в текущем месяце побывали в командировке.

Вариант 11

Создайте программу с классом Abitur, который включает в себя следующие данные – элементы об абитуриентах, сдавших вступительные экзамены в университет:

- ФИО;
- адрес;
- оценки по предметам.

Номер для каждого абитуриента запрашиваться, а массив оценок создается в динамической памяти.

В состав класса входят следующие функции-члены класса:

- конструктор с параметрами;
- деструктор;
- функции просмотра текущего объекта;
- функция – друг, позволяющая определить количество абитуриентов, проживающих в Москве;
- виртуальная функция просмотра текущего объекта.

Производный класс Abitur _1 содержит следующие данные о льготах:

- тип медали (золотая или серебряная);
- призовое место на Всероссийской олимпиаде;
- переопределенную функцию просмотра состояния объектов-абитуриентов, имеющих льготы при поступлении.

Вариант 12

Создайте программу с классом Student, который включает в себя следующие данные – элементы о студентах, желающих получить места в общежитии. Общежитие в первую очередь предоставляется тем студентам, у кого доход на члена семьи меньше двух минимальных зарплат. Класс включает в себя следующие данные-элементы:

- ФИО студента;
- номер группы (буква и четы;
- средний балл;

- доход на одного члена семьи.

В состав класса входят следующие функции-члены класса:

- конструктор с параметрами;
- деструктор;
- функции просмотра текущего объекта;
- функция - друг, которая должна вывести ФИО студентов со средним баллом выше указанного с клавиатуры;
- виртуальная функция просмотра текущего объекта.

Производный класс Student_1 содержит следующие элементы:

- вид общественной нагрузки;
- количество членов семьи;
- переопределенную функцию просмотра состояния объектов – студентов-внеочередников.

Вариант 13

Создайте программу с классом Bas, который включает в себя следующие данные – элементы об автобусных рейсах:

- номер рейса;
- тип автобуса;
- пункт назначения;
- цена билета;
- время отправления;
- время прибытия на конечный пункт.

В состав класса входят следующие функции-члены класса:

- конструктор с параметром (номер рейса);
- деструктор;
- функции просмотра текущего объекта;
- функция установки текущего состояния объектов для остальных элементов;
- функция - друг должна вывести информацию о рейсах, позволяющих добраться до указанного с клавиатуры пункта;
- виртуальная функция просмотра текущего объекта.

Производный класс Express содержит следующие данные об автобусах-экспрессах:

- дни недели работы;
- процентная надбавка на цену билета;
- выигрыш во времени;
- переопределенную функцию вывода данных об общественниках.

Вариант 14

Создайте программу с классом Team, который включает в себя следующие данные – элементы об участниках спортивных соревнований:

- ФИО игрока;
- игровой номер;
- возраст;
- рост;
- вес.

Номер для каждого создаваемого студента должен запрашиваться, а массив оценок за последнюю сессию создается в динамической памяти операцией new.

В состав класса входят следующие функции-члены класса:

- конструктор с параметрами;
- деструктор;

- функции просмотра текущего объекта;
- функция - друг, которая выведет информацию о самом легком спортсмене в команде;
- виртуальная функция просмотра текущего объекта.

Производный класс Inform содержит следующие данные:
- разряд спортсмена;
- категория;
- призер (Европы, мира и т.д.).
- переопределенную функцию просмотра состояния объекта.

Вариант 15

Создайте программу с классом Phone, который включает в себя следующие данные – элементы о разговорах на международной АТС:
- дату разговора;
- код и название города;
- продолжительность разговора;
- тариф;
- номер телефона в этом городе;
- номер телефона абонента.

В состав класса входят следующие функции-члены класса:
- конструктор по умолчанию;
- деструктор;
- функция установки текущего состояния объекта;
- функции просмотра текущего объекта;
- функция, которая позволяет вывести по номеру телефона название города;
- функция - друг, которая подсчитает суммарное время разговора с указанным городом;
- виртуальная функция просмотра текущего объекта.

Производный класс Phone _1 содержит следующие элементы:
- вид льгот;
- процент для льготного тарифа;
- надбавка к тарифу за срочность вызова;
- переопределенную функцию просмотра состояния объекта.

Вариант 16

Создайте программу с классом Goods, который включает в себя следующие данные – элементы о товарах, имеющихся на складе:
- страна-изготовитель;
- фирма-изготовитель;
- наименование товара;
- количество единиц товара.

В состав класса входят следующие функции-члены класса:
- конструктор с параметром для инициализации страны-изготовителя;
- деструктор;
- функция инициализации текущего состояния объектов остальных элементов;
- функции просмотра текущего объекта;
- функция подсчета общего количества товара указанной фирмы;
- функция - друг, которая позволяет вывести товары и их данные для указанной страны;
- виртуальная функция просмотра текущего объекта.

Производный класс Order содержит следующие элементы:
- страна-заказчик;
- дата заказа;
- количество заказанного товара;
- переопределенную функцию вывода данных об общественниках.

ПРАКТИЧЕСКАЯ РАБОТА № 21

ПОСТРОЕНИЕ СПРАВОЧНОЙ СИСТЕМЫ С ПОМОЩЬЮ VISUAL STUDIO

Цель занятия: изучение принципов построения справочной системы в программном продукте; создание .chm файла; подключение файла.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ И ХОД ВЫПОЛНЕНИЯ ЗАДАНИЯ

1. **Инструментарий Microsoft HTML Help** - новое поколение авторизованных справочных систем позволяет построить справочное руководство, страницы которого обладают всеми возможностями Web-страниц.

Разделы справочного руководства создаются как отдельные HTML - файлы и могут быть созданы в любом Редакторе, позволяющем работать с таким форматом, например, в Word или FrontPage. Подобный Редактор входит и в состав специального инструментального средства, используемого для создания справочных руководств, - HTML Help Workshop (HHW). Благодаря новому подходу расширяются стандартные возможности описания раздела справки. Разделы справки теперь могут содержать все элементы, присущие Web-страницам - гиперссылки, script-код, DHTML, элементы ActiveX, графические элементы, аудио и видео.

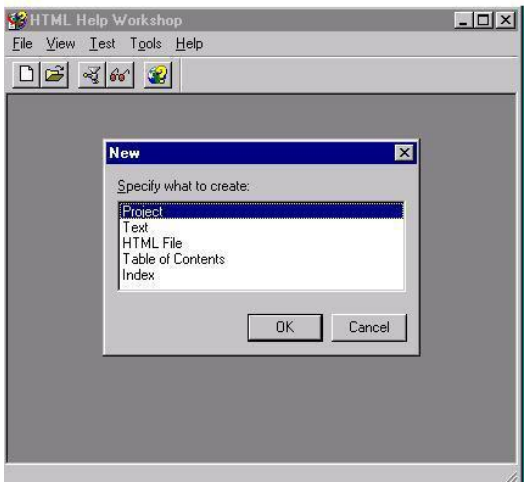


Рисунок 1- Начальный шаг

2.Построение файла проекта

Можно начинать создавать проект на пустом месте, когда никаких компонент еще нет. Для создания файла проекта, также как и для любых файлов, создаваемых в среде HHW, в меню File нужно выбрать пункт New, а затем в появившемся окне выбрать нужный тип создаваемого файла. Соответствующий Мастер ведет нас на всех этапах создания проекта. Вот как выглядит диалоговое окно на первом шаге работы Мастера:

Рисунок 2 - Начальный шаг создания проекта "Справка о справке"

3. Построение файла HTML

В меню File нужно выбрать пункт New и далее выбрать HTML File.На последующих шагах работы Мастера указывается, какие компоненты уже созданы и где они хранятся.В окне "HTML Title" введите "Form Help" и нажмите кнопку "ОК". Между тегами <BODY> введите фразу "Это основной раздел справки" (без кавычек). В меню "File" выберите пункт "Save File As". При этом откроется окно "Save File As". Сохраните файл под именем "Start.htm". Обратите внимание на папку, в которую выполняется сохранение. Закройте файл Start.htm.

После того, как Мастер завершил создание проекта, следует выполнить компиляцию проекта. Взгляните, как выглядит HHW после завершения компиляции проекта.

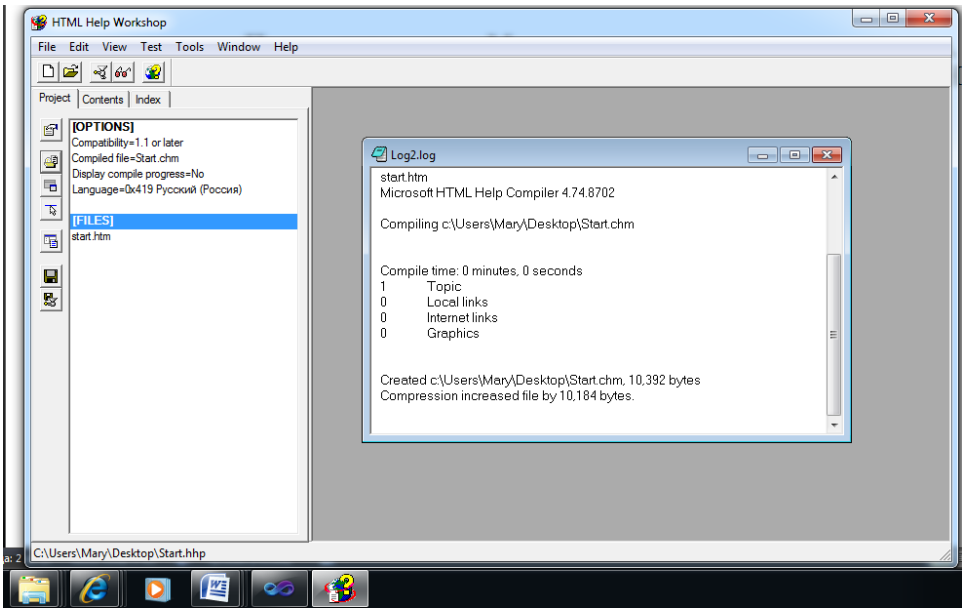


Рисунок 3 - Результат компиляции проекта "Справка о справке"

В левом окне отображаются данные о проекте, в правом - о результатах компиляции. В нашем случае никаких сообщений об ошибках компиляции не было, но сами понимаете, такие сообщения могут появляться. Указано время компиляции и другая полезная статистика, в частности, число разделов, число локальных гиперссылок и ссылок Интернет, объем файла до и после сжатия.

Теперь первый вариант задуманного нами справочного руководства создан, - он хранится в откомпилированном файле с уточнением ".chm". Его можно просмотреть его в специальном обозревателе HTML Help Viewer, который использует компоненты Internet Explorer, хотя и не совпадает с ним полностью.

4. Подключение .chm файла

Создайте проект Windows Form. Создайте кнопку button1, при нажатии на кнопку запишите код:

```
Help.ShowHelp(this, "start.chm");
```

САМОСТОЯТЕЛЬНАЯ РАБОТА

Постройте проект и запустите файл справки. Задания указаны в практической работе №16.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 22-23

РАЗРАБОТКА WINDOWS-ПРИЛОЖЕНИЙ

Цель занятия: изучение механизма связывания источника данных с элементом управления dataGridView; формирование компетенций при построении приложений по обработке информационных массивов с применением элементов управления: DataSet, DataGridView, BindingSourse, BindingNavigator.

Содержание занятия:

1.Создание пользовательского интерфейса.

2. Создание модулей для обработки кнопок ввода, редактирования и поиска данных.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

DataAdapter является средством связи между Dataset и Connection. С Dataset он связан посредством команд:

Fill - загрузить данные,

Update - записать изменения.

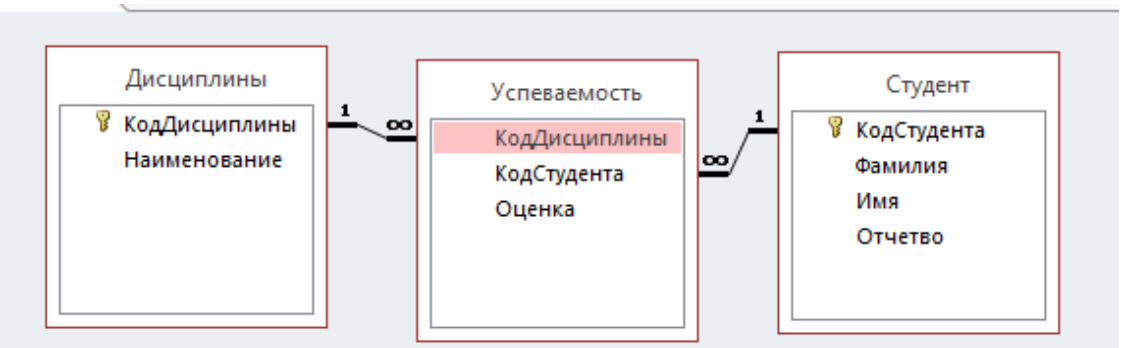
В момент записи создаётся список ошибок (Error Collection). DataAdapter работает с Connection с помощью реляционного метода. Команды взаимодействия с БД (Command) можно создавать как вручную, так и автоматически.

ХОД РАБОТЫ

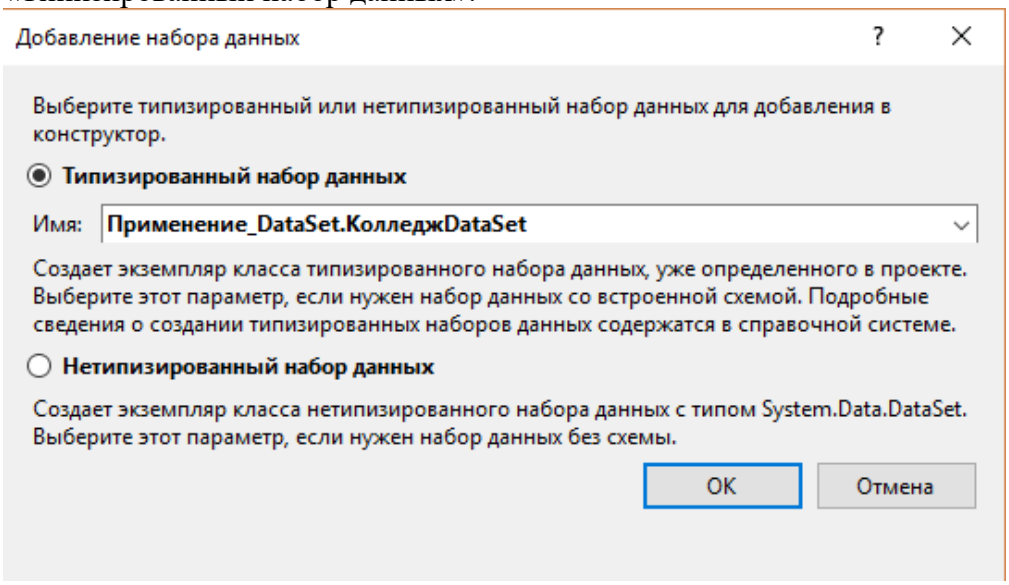
Задание 1. Создать приложение, позволяющее извлекать данные с помощью объекта DataSet.

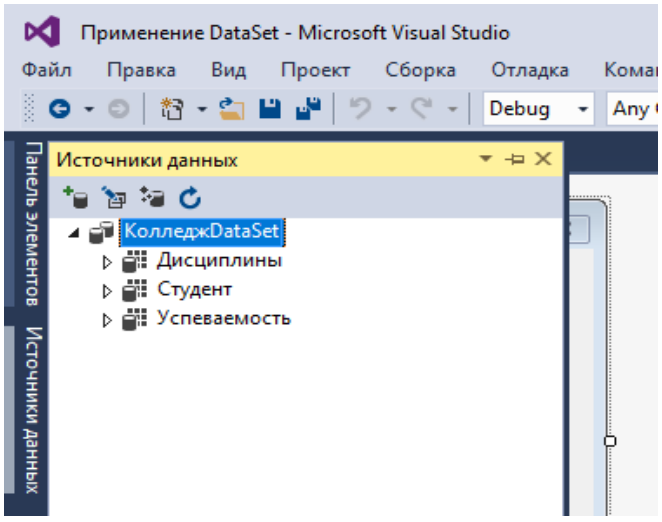
Примечание: в качестве альтернативы для заполнения элемента управления вручную, можно задать свойства для привязки DataGridView в данных источника и автоматически заполняет ее данными.

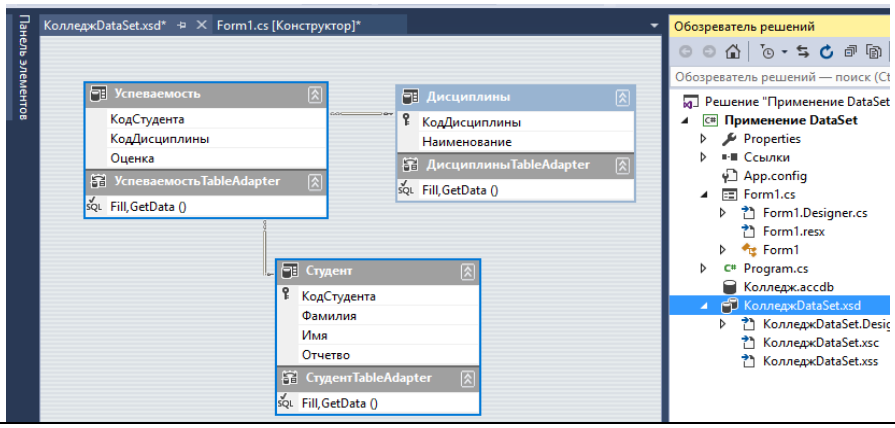
Алгоритм создания приложения:

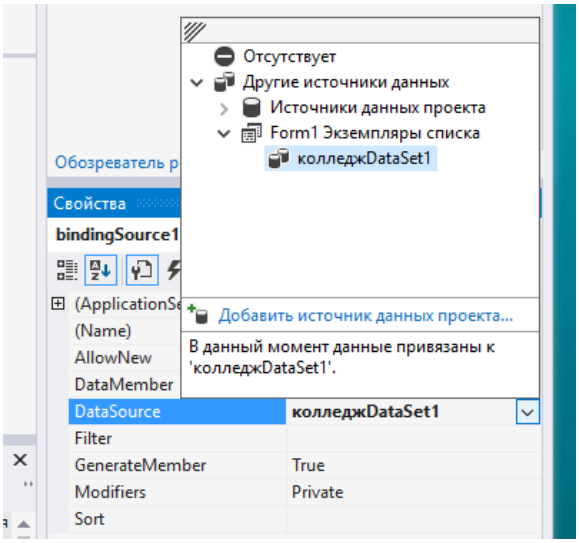
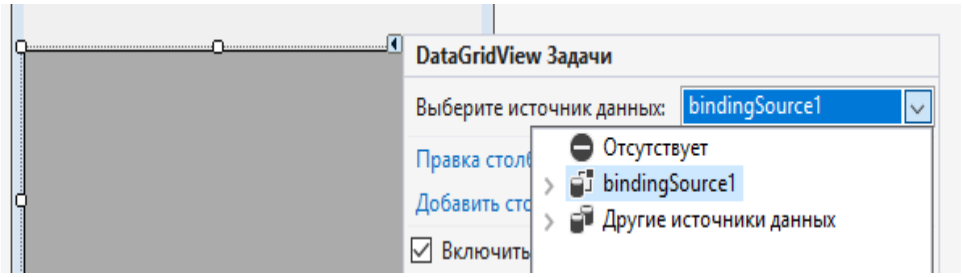
<table>
<tr><td colspan="2">В качестве источника данных выбрать базу данных Колледж.accdb, созданную в MS Access. Для обработки данных в приложении использовать компоненты DataGridView, BindingSourse, BindingNavigator, DataSet.</td></tr>
<tr><td colspan="2">Таблицы базы данных
 
</td></tr>
<tr><td>1. Выбрать источник данных</td><td>Проект / Добавить новый источник данных / указать путь к базе данных, выбрать таблицы и требуемые элементы таблицы:

в окне Мастер настройки источника данных выбрать База данных /</td></tr>
</table>

<table>
<tr><td></td><td>Набор данных / Создать подключение / Файл базы данных Microsoft Access / Обзор / Выбрать базу / Проверить подключение / Ok / Далее/ Да/ Далее/ Поставить галочку Таблицы / Готово</td></tr>
<tr><td>2. Добавить к своему проекту компонент DataSet</td><td>При добавлении Dataset, в мастере необходимо выбрать пункт «Типизированный набор данных».
 

Если все шаги были проделаны верно, то появится КолледжDataSet в источниках данных
 

Дважды щелкните на файле КолледжDataSet.xsd, появится схема данных.
 
</td></tr>
<tr><td>3. На форму из вкладки</td><td>Добавьте: BindingSource, DataGridView, BindingNavigator (два последних являются визуальными компонентами).</td></tr>
</table>

«Данные» панели элементов добавить компоненты	
4. Настроить свойства компонента bindingSource1	1) в элементе bindingSource1 выбрать в свойстве DataSource именно тот Dataset, который расположен на одной форме с этим компонентом  2) в поле DataMember нужно выбрать имя таблицы, связанной с этим компонентом - Студент. Если всё сделано, верно, то в списке невидимых компонентов должен появиться TableAdapter - студентTableAdapter. 3) в редакторе кода найдите строку, которая загружает данные в адаптер таблиц. Этот код был сгенерирован при установке привязки данных. Код должен иметь следующий вид: `this.студентTableAdapter.Fill(this.колледжDataSet1.Студент);` Он находится в событии формы `Form1_Load`.
5. Настроить свойства компонента dataGridView1	В dataGridView1 нужно свойству DataSource установить значение bindingSource1.  Появятся заголовки столбцов таблицы.
6. Настроить свойства компонента bindingNavigator 1	Установить свойству BindingSource соответствующий компонент формы - bindingSourse1

Задание 2. Запрограммировать кнопки навигационной панели.

Примечание: все операции над записями в программе происходят через BindingSource.

Алгоритм создания приложения:

1. Создайте обработчик события для кнопки ToolStripButton «Загрузить»	Синтаксис команды: TableAdapterName.Fill (DataSetName.TableName); Применительно к нашему проекту команда имеет следующий вид: `this.студентTableAdapter.Fill(this.колледжDataSet1.Студент);`
2. Создайте обработчик события для кнопки «Сохранить»	В некоторых случаях компонент BindingNavigator уже содержит кнопку **Сохранить**, но код, сгенерированный конструктором Windows Forms, при этом отсутствует. В этом случае приведенный код можно поместить в Click обработчик событий кнопки, вместо создания полностью новой кнопки ToolStrip. Кнопка по умолчанию отключена, свойству Enabled кнопки необходимо присвоить значение **true**, чтобы кнопка работала правильно. Синтаксис команды: TableAdapterName.Update(DataSetName.TableName); Добавим код: `this.студентTableAdapter.Update(this.колледжDataSet1);`
3. Создайте обработчик события для кнопки ToolStripButton «Отмена»	BindingSourceName.CancelEdit(); Метод CancelEdit распространяется на строку данных. Сохраните все изменения, сделанные при просмотре этой отдельной записи, до перехода к следующей записи.
4. Создайте обработчик события для кнопки ToolStripButton «Найти»	Фильтрация задается по следующему шаблону: bindingSource1.Filter = "Условие фильтра". Добавим код: `bindingSource1.Filter = "Фамилия like '" + textBox1.Text + "%'";`

Элемент TableAdapter предназначен для заполнения dataSet формы с помощью команды Fill, а также для сохранения этого набора данных в БД. Кроме того, с помощью этого компонента можно создавать команды для добавления записей - `TablAdapter.Insert()` - и их удаления – `TablAdapter.Delete()`.

1. **Сортировка:**
   bindingSource1.Sort ="Поле таблицы" asc/desc;
   asc/desc - по возрастанию или убыванию
2. **Перемещение по записям:**
   bindingSource1.MoveNext(); - следующий элемент
   bindingSource1.MovePrevious(); - предыдущий элемент
3. **Удалить запись в БД:**
   tableAdapter.Update(this.DataSet);
   `студентTableAdapter.Delete(1,"Хромов","","");`

Перед изменением данных необходимо выполнить команду:
bindingSource1.EndEdit(); - применить незавершенные изменения к базовому источнику данных.

САМОСТОЯТЕЛЬНАЯ РАБОТА

Каждый студент создает приложение с применением своей индивидуальной базы данных, закрепленной в дисциплине «Основы проектирования баз данных».

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 24

Цель занятия: контроль и оценка знаний.
Содержание занятия:

1.Создание проекта.
2. Демонстрация проекта.

Создайте программный продукт, содержащий основное меню с вкладками (Основная, Справка, О программе, Выход), строку статуса, всплывающие подсказки, выполняющий необходимые действия. Тема выдается на занятии по вариантам.

СПИСОК ЛИТЕРАТУРЫ

1. Биллиг В.А. Основы объектного программирования на C# (C# 3.0, Visual Studio 2008)/ В.А. Биллиг. – М.: Интернет-Университет Информационных Технологий: БИНОМ. Лаборатория знаний, 2013. – 582 с.: ил.
2. Культин Н.Б. C# в задачах и примерах. – СПб.: БХВ-Петербург, 2007. – 240 с.: ил.+CD-ROM.
3. Осипов Н.А., Разработка Windows приложений на C# - СПб: НИУ ИТМО, 2012. – 74 с.

Интернет-источники:

4. Заполнение наборов данных с помощью адаптера таблицы/ URL: https://docs.microsoft.com/ru-ru/visualstudio/data-tools/fill-datasets-by-using-tableadapters?view=vs-2017