

РОСЖЕЛДОР
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Ростовский государственный университет путей сообщения»
(ФГБОУ ВО РГУПС)

О.Г. Ведерникова, О.В. Игнатъева

ВИЗУАЛЬНОЕ ПРОГРАММИРОВАНИЕ
И ГРАФИЧЕСКИЕ ИНТЕРФЕЙСЫ

Учебно-методическое пособие
для лабораторных работ

Часть 1

Ростов-на-Дону
2019

УДК 004.41(07) + 06

Рецензент – кандидат технических наук, доцент В.В. Жуков

Ведерникова, О.Г.

Визуальное программирование и графические интерфейсы: учебно-методическое пособие для лабораторных работ. В 2 ч. Ч. 1 / О.Г. Ведерникова, О.В. Игнатъева; ФГБОУ ВО РГУПС. – Ростов н/Д, 2019. – 80 с.

Рассмотрены способы создания графических интерфейсов с использованием визуальных компонентов, обработки текстовой и числовой информации в табличной форме, создания многостраничных приложений. Приведены базовые приемы программирования в среде MS Visual Studio Express на языке C#, начиная с простейших приложений. Содержатся задания и методика выполнения лабораторных и практических работ по дисциплине, а также примеры выполнения лабораторных работ.

Предназначено для студентов и магистрантов направлений «Информатика и вычислительная техника» и «Информационные системы и технологии» для углубленного изучения визуального программирования на аудиторных занятиях и самостоятельного изучения материала по дисциплинам «Визуальное программирование и графические интерфейсы» и «Информатика и программирование», а также для всех изучающих смежные дисциплины и спецкурсы.

Одобрено к изданию кафедрой «Вычислительная техника и автоматизированные системы управления».

Учебное издание

Ведерникова Ольга Геннадьевна
Игнатъева Олеся Владимировна

ВИЗУАЛЬНОЕ ПРОГРАММИРОВАНИЕ И ГРАФИЧЕСКИЕ ИНТЕРФЕЙСЫ

Часть 1

Печатается в авторской редакции
Технический редактор Т.И. Исаева

Подписано в печать 17.09.19. Формат 60×84/16.
Бумага офсетная. Печать офсетная. Усл. печ. л. 4,65.
Тираж экз. Изд. № 5066. Заказ .

Редакционно-издательский центр ФГБОУ ВО РГУПС.

Адрес университета: 344038, г. Ростов н/Д, пл. Ростовского Стрелкового Полка
Народного Ополчения, д. 2.

© Ведерникова О.Г., Игнатъева О.В., 2019
© ФГБОУ ВО РГУПС, 2019

ОГЛАВЛЕНИЕ

ЛАБОРАТОРНАЯ РАБОТА № 1. ЗНАКОМСТВО СО СРЕДОЙ ВИЗУАЛЬНОЙ РАЗРАБОТКИ ПРИЛОЖЕНИЙ	4
Варианты заданий для лабораторной работы № 1	19
ЛАБОРАТОРНАЯ РАБОТА № 2. ФУНКЦИИ ПРЕОБРАЗОВАНИЯ ТЕКСТА В ЧИСЛО	24
Варианты заданий для лабораторной работы № 2	29
ЛАБОРАТОРНАЯ РАБОТА № 3. ИСПОЛЬЗОВАНИЕ ВИЗУАЛЬНЫХ КОМПОНЕНТОВ И НАСТРОЙКА ИХ СВОЙСТВ	32
Варианты заданий для лабораторной работы № 3	37
ЛАБОРАТОРНАЯ РАБОТА № 4. ОБРАБОТКА СОБЫТИЙ ДЛЯ ФОРМАТИРОВАНИЯ ТЕКСТА В МНОГОСТРОЧНОМ ПОЛЕ ВВОДА	42
Варианты заданий для лабораторной работы № 4	46
ЛАБОРАТОРНАЯ РАБОТА № 5. ОБРАБОТКА СТРУКТУРНОЙ ИНФОРМАЦИИ В ТАБЛИЦЕ DATAGRIDVIEW	51
Варианты заданий для лабораторной работы № 5	64
ЛАБОРАТОРНАЯ РАБОТА № 6. СОЗДАНИЕ МНОГОСТРАНИЧНЫХ ПРИЛОЖЕНИЙ ИЛИ РАБОТА С НЕСКОЛЬКИМИ ФОРМАМИ	70
Варианты заданий для лабораторной работы № 6	77
Библиографический список	80

Лабораторная работа № 1

ЗНАКОМСТВО СО СРЕДОЙ ВИЗУАЛЬНОЙ РАЗРАБОТКИ ПРИЛОЖЕНИЙ

Для установки среды Microsoft Visual Studio Express 2012 для Windows Desktop можно бесплатно скачать ее с официального сайта по ссылке: <http://www.microsoft.com/ru-ru/download/details.aspx?id=34673>.

Рассмотрим процесс разработки программы в Microsoft Visual C# – создадим *приложение* (программы, предназначенные для решения прикладных задач, принято называть *приложениями*) (примерно как на рис. 1.1).

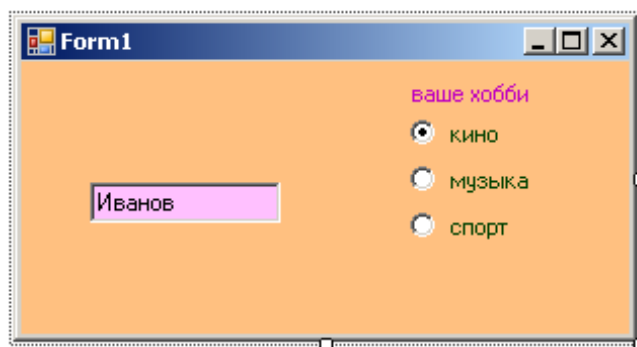


Рис. 1.1. Окно программы

НАЧАЛО РАБОТЫ НАД ПРОЕКТОМ

Чтобы начать работу над новым проектом, надо:

1. В меню **File** выбрать команду **New Project**.
2. В открывшемся окне **New Project** выбрать тип приложения – **Windows Forms Application – Visual C#**.
3. В поле **Name** ввести имя проекта – **profit** (любое латинскими буквами) и нажать кнопку **OK** (рис. 1.2).

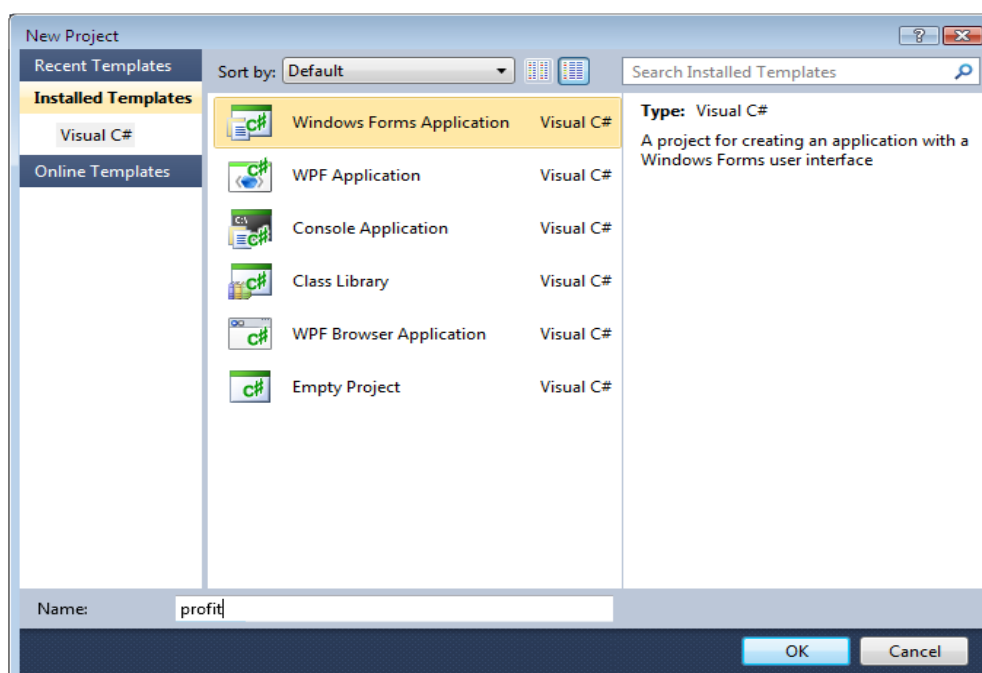


Рис. 1.2. Начало работы над новой программой

В результате описанных действий в папке временных проектов (по умолчанию это `C:\Users\User\Visual AppData\Local\Temporary Projects`) будет создана папка `profit`, а в ней – проект `profit`.

ФОРМА

Работа над приложением начинается с создания стартовой *формы* – главного окна программы. Сначала нужно установить требуемые значения свойств формы, затем – поместить на форму необходимые *компоненты* (поля ввода информации, командные кнопки, поля отображения текста и др.) и выполнить.

В левой части главного окна Visual C# отображается окно **Toolbox** в котором (рис. 1.3) находятся *компоненты*. Компонент – это объект, реализующий некоторую функциональность. Например, в группе **Common Controls** находятся компоненты, реализующие пользовательский интерфейс (например: `Label` – область отображения текста; `TextBox` – поле редактирования текста; `Button` – командная кнопка, а в группе **Data** – компоненты, обеспечивающие доступ к базам данных).

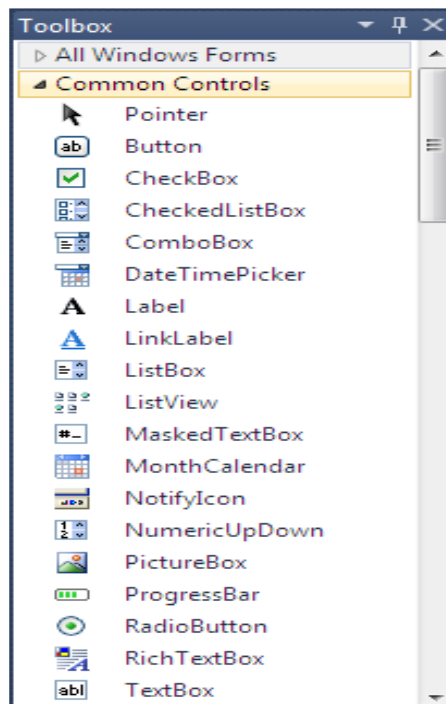


Рис. 1.3

В окне **Properties** (рис. 1.4) отображаются *свойства* выбранного в данный момент объекта-компонента или, если ни один из элементов (компонентов) формы не выбран, самой формы. Обратите внимание, в верхней части окна **Properties** отображаются имя выбранного объекта и его тип. Редактирование свойств объекта ведет к изменению его вида и поведения. (Например, результатом изменения значения свойства `Text` для объекта `Form` (формы) является изменение текста, находящегося в ее заголовке, а свойство `BackColor` к изменению цвета фона.)

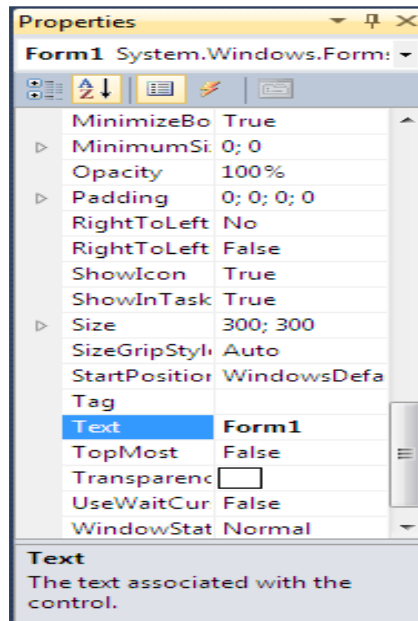


Рис. 1.4

Свойства в окне **Properties** могут быть объединены в группы по функциональному признаку (**Categorized**) или упорядочены по алфавиту (**Alphabetical**). Чтобы изменить способ отображения, надо сделать щелчок на соответствующей кнопке, находящейся в верхней части окна **Properties** (рис. 1.5).

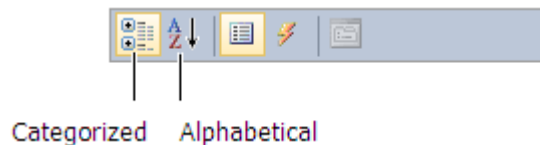


Рис. 1.5

Чтобы в заголовке окна формы отображалось название программы, следует щелкнуть левой кнопкой мыши в поле значение свойства **Text** (в поле появится курсор), ввести в поле редактирования текст **Доход** (или ваша фамилия) и нажать клавишу <Enter> (рис. 1.6).

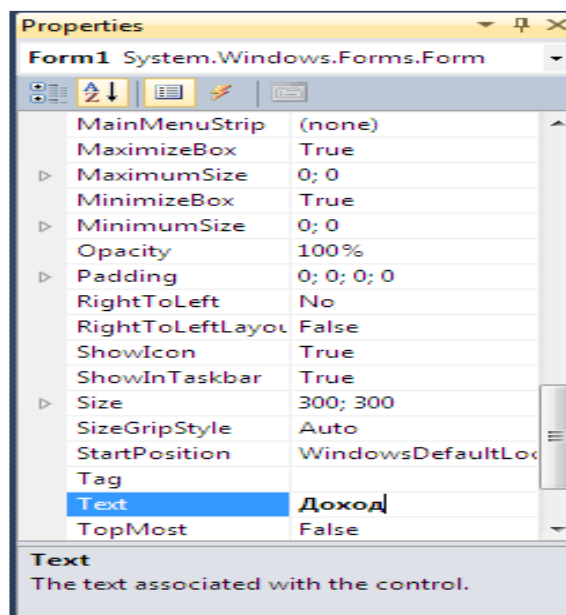


Рис. 1.6

При выборе некоторых свойств, например `FormBorderStyle`, справа от текущего значения свойства появляется значок раскрывающегося списка. Очевидно, что значение таких свойств можно задать путем выбора из списка (рис. 1.7).

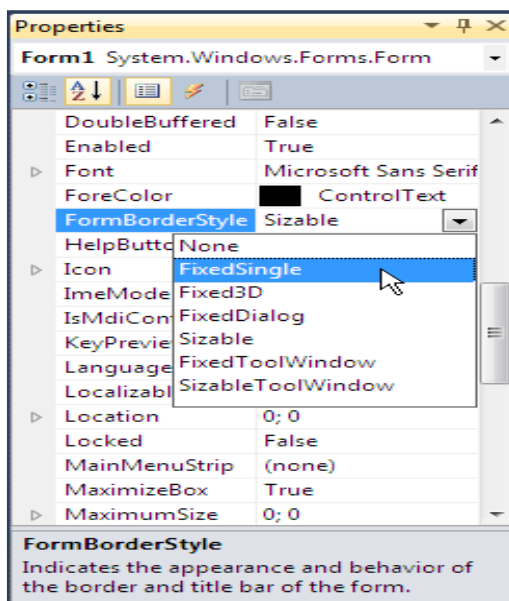


Рис. 1.7

Свойство `FormBorderStyle`, определяет Тип формы (границы). Форма может представлять собой обычное окно (`Sizable`), окно фиксированного размера (`FixedSingle`, `Fixed3D`), диалог (`FixedDialog`) или окно без кнопок **Свернуть** и **Развернуть** (`SizeableToolWindow`, `FixedToolWindow`). Если свойству присвоить значение `None`, у окна не будет заголовка и границы

Некоторые свойства являются сложными. Они представляют собой совокупность других (уточняющих) свойств. Например, свойство `Size`, определяющее размер формы, представляет собой совокупность свойств `Width` и `Height`. Перед именами сложных свойств стоит значок , в результате щелчка на котором раскрывается список уточняющих свойств (рис. 1.8). Значение уточняющего свойства можно задать (изменить) обычным образом – ввести нужное значение в поле редактирования.

Размер формы можно изменить и с помощью мыши, точно так же, как и любого окна, т. е. путем перемещения границы. По окончании перемещения границы значения свойств **Width** и **Height** будут соответствовать установленному размеру формы.

В результате выбора некоторых свойств, например `Font`, в поле значения свойства отображается кнопка, на которой изображены три точки. Это значит, что задать значение свойства можно в дополнительном диалоговом окне, которое появится в результате щелчка на этой кнопке. Например, значение свойства `Font` можно задать путем ввода значений уточняющих свойств (`Name`, `Size`, `Style` и др.), а можно воспользоваться стандартным диалоговым окном **Шрифт**, которое появится в результате щелчка на кнопке с тремя точками (рис. 1.9).

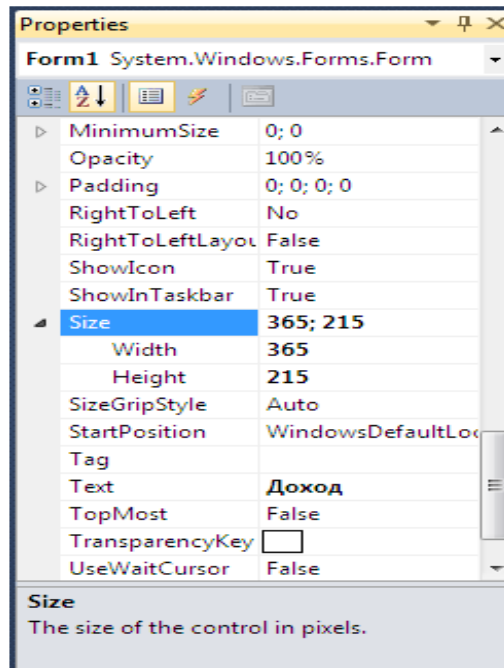


Рис. 1.8. Изменение значения уточняющего свойства

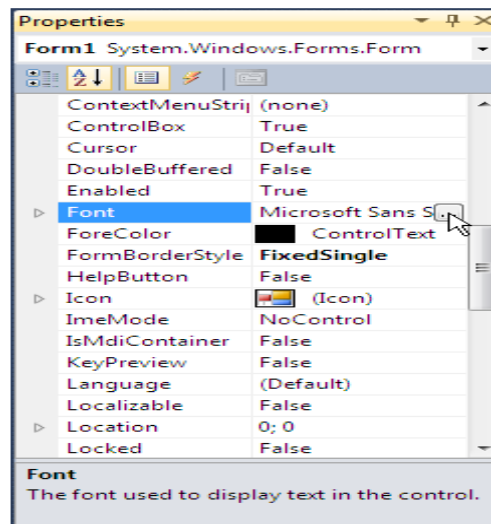


Рис. 1.9

Для задания цвета форме выберем свойство **BackColor** и перейдем на вкладку **Custom**, далее выберем нужный цвет.

Для программного задания цвета: `textBox1.BackColor= Color.Red;`

После того как будут установлены значения свойств формы, она должна выглядеть так, как показано на рис. 1.10. Теперь на форму надо добавить *компоненты*.

Чтобы на форму добавить компонент **TextBox**, надо:

1) в палитре компонентов (окно **Toolbox**) раскрыть вкладку **Common Controls**;

2) сделать щелчок на значке компонента **TextBox** (рис. 1.11).

3) установить указатель мыши в ту точку формы, в которой должен быть левый верхний угол компонента, и сделать щелчок левой кнопкой мыши.

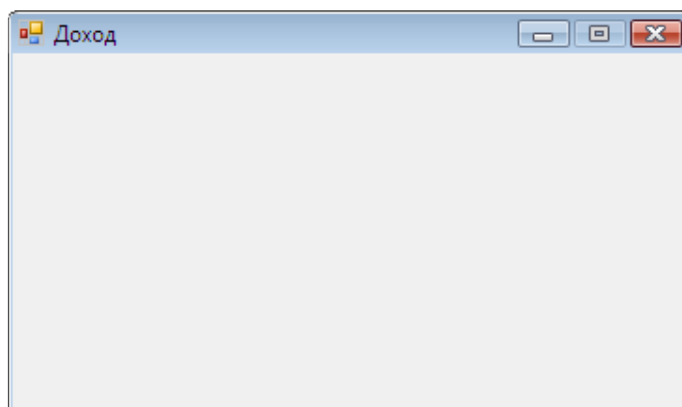


Рис. 1.10. Форма после изменения значений ее свойств

В результате на форме появляется поле ввода/редактирования – компонент TextBox (см. рис. 1.11).

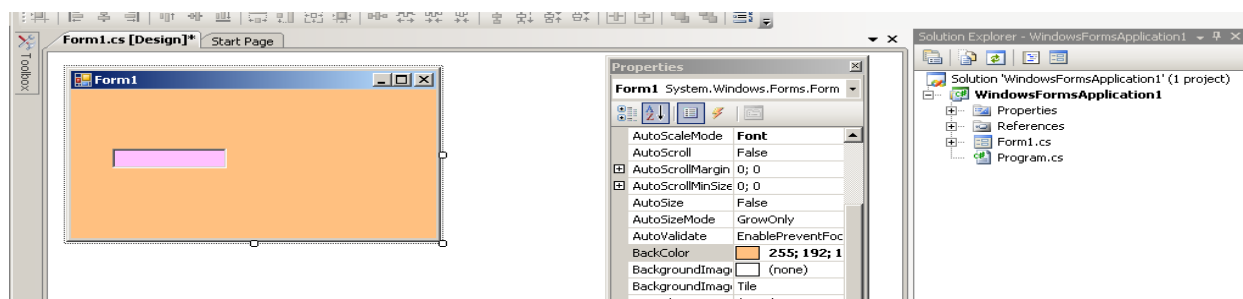


Рисунок 1.11

Основные свойства компонента **TextBox** приведены в табл.

Свойство	Описание
Name	Имя компонента. Используется для доступа к компоненту и его свойствам (лучше не менять его)
Text	Текст, который находится в поле редактирования
Location	Положение компонента на поверхности формы
Size	Размер компонента
Font	Шрифт, используемый для отображения текста в поле компонента
ForeColor	Цвет текста, находящегося в поле компонента
BackColor	Цвет фона поля компонента
Enabled	= False поле недоступно для ввода, объект серого цвета недоступный,
ReadOnly	= True только для чтения, невозможность редактирования текста
Visible	Задаёт видимость компонента. Если = True , то компонент виден, False – нет
WordWrap	Разрешает разбивку и перенос непомещающихся строк по словам
TextAlign	Выравнивание текста по центру, по левому краю, по правому краю

Каждому добавленному компоненту среда разработки присваивает имя, которое состоит из названия компонента и его порядкового номера. Например, первый добавленный на форму компонент `TextBox` получает имя `textBox1`, второй – `textBox2`. Программист путем изменения значения свойства `Name` может поменять имя компонента. Однако в простых программах имена компонентов, как правило, не меняют.

Чтобы изменить положение компонента, необходимо установить курсор мыши на его изображение, нажать левую кнопку мыши и, удерживая ее нажатой, переместить компонент в нужную точку формы (рис. 1.12).

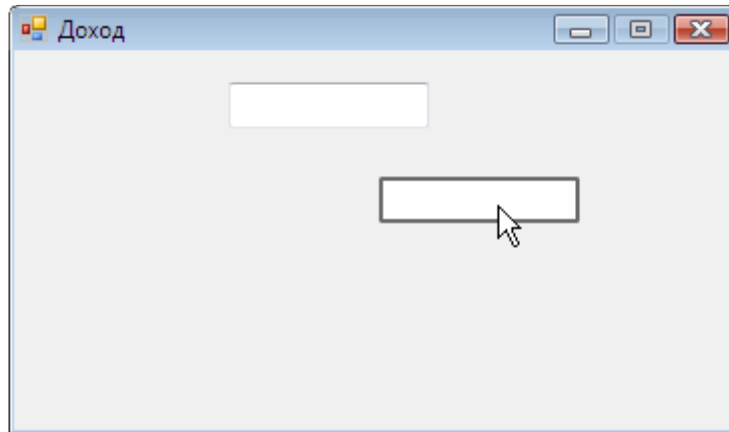


Рис. 1.12

Изменить положение компонента программно можно используя следующий код:

`textBox.Left+=10;` – перемещение от левой границы формы
`textBox.Top+=12;` перемещение от верхней границы формы

Для того чтобы изменить размер компонента, необходимо сделать щелчок на его изображении (в результате чего компонент будет выделен), установить указатель мыши на один из маркеров, помечающих границу компонента, нажать левую кнопку мыши и, удерживая ее нажатой, изменить положение границы компонента (рис. 1.13).

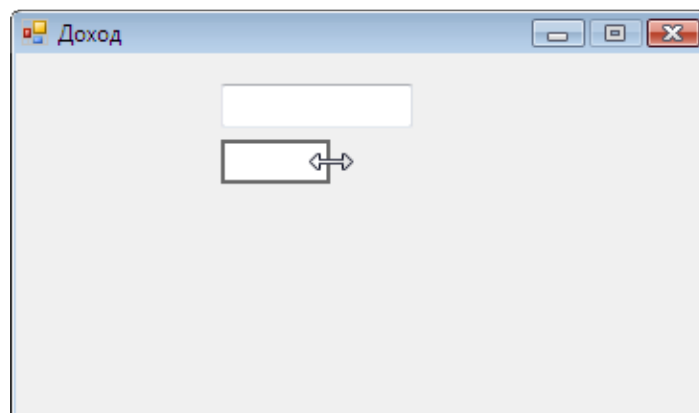


Рис. 1.13

Для изменения цвета компонента `TextBox` нужно выделить его, выбрать свойство `BackColor` и перейти на вкладку `Custom` там выбрать нужный цвет.

Для задания надписи в поле ввода `TextBox` выбрать свойство `Text` и ввести своё имя (рис. 1.14).

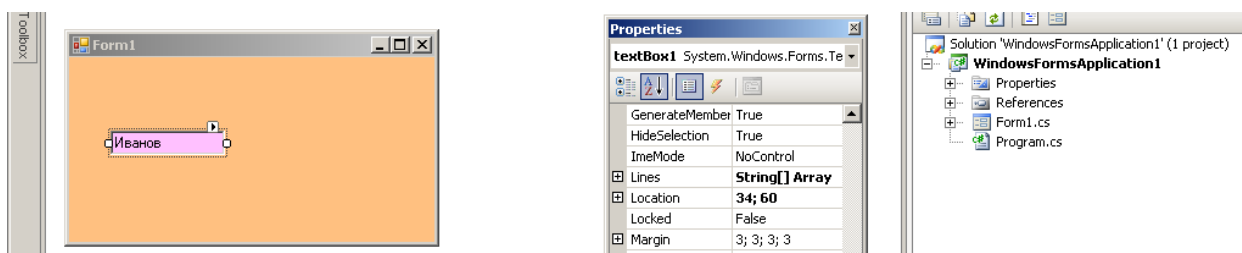


Рис. 1.14

Разместим на форме ещё компонент `Label` который обеспечивает отображение текста на поверхности формы (подсказок, результата расчета). Добавляется компонент `Label` на форму точно так же, как и поле редактирования (компонент `TextBox`). В свойство `Text` введем «Ваше хобби:»

Основные свойства компонента **Label**

Свойство	Описание
Name	Имя компонента. Используется в программе для доступа к свойствам компонента
Text	Отображаемый текст
Location	Положение компонента на поверхности формы
AutoSize	Признак автоматического изменения размера компонента. Если значение свойства равно <code>True</code> , то при изменении значения свойства <code>Text</code> (или <code>Font</code>) автоматически изменяется размер компонента
Size	Размер компонента (области отображения текста). Определяет (если значение свойства <code>AutoSize</code> равно <code>False</code>) размер компонента (области отображения текста)
Font	Шрифт, используемый для отображения текста
ForeColor	Цвет текста, отображаемого в поле компонента
BackColor	Цвет заливки области вывода текста
TextAlign	Способ выравнивания (расположения) текста в поле компонента. Всего существует девять способов расположения текста. На практике наиболее часто используют выравнивание по левой верхней границе (<code>TopLeft</code>), посередине (<code>TopCenter</code>) и по центру (<code>MiddleCenter</code>)
Visible	Задаёт видимость компонента. Если = True , то компонент виден, False – нет
Enabled	= False поле недоступно, объект серого цвета недоступный,
AutoSize	= <code>False</code> –Отменяет автоподбор размера метки под размер текста и позволяет переносить текст по словам при изменении размеров метки в ручную.

Далее Разместим аналогично три компонента **RadioButton** введем текст «Кино» «Музыка» «Спорт». Для кнопки **RadioButton1** зададим свойство **checked=true**, для того чтобы она было выбрана(выделена) (рис. 1.15).

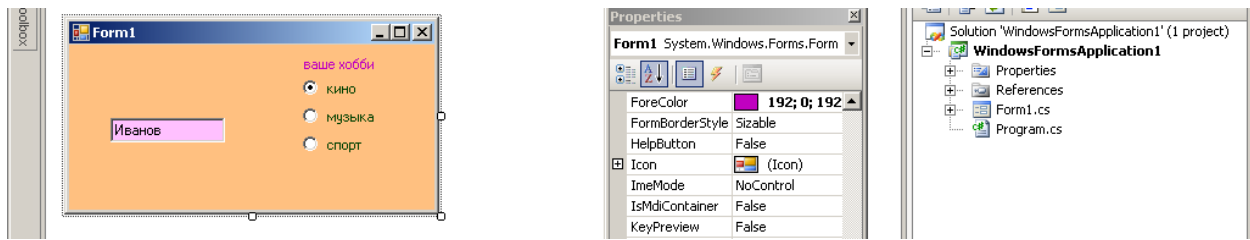
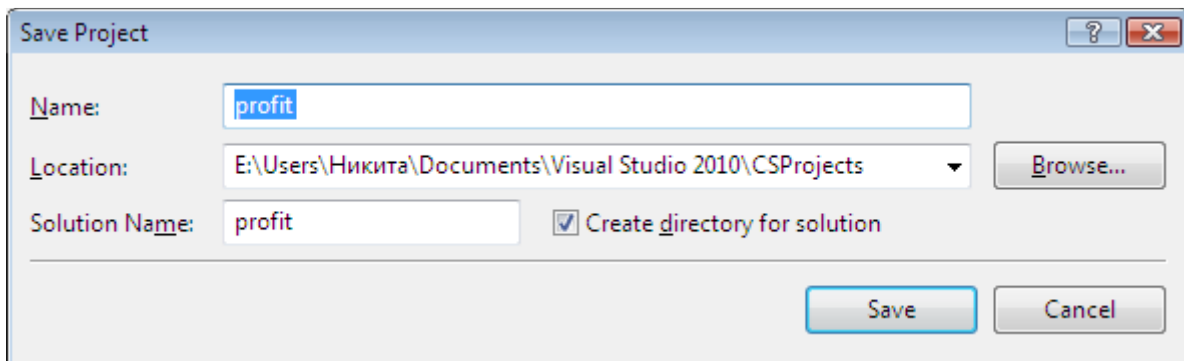


Рис. 1.15

СОХРАНЕНИЕ ПРОЕКТА

Как было сказано ранее, в момент создания проекта среда разработки в папке временных проектов (по умолчанию `C:\Users\User\AppData\Local\Temporary Projects`, где *User* – имя пользователя в системе) создает каталог проекта. Чтобы программист мог работать с программой в дальнейшем, проект надо сохранить явно. Для этого в меню **File** надо выбрать команду **Save All** и в появившемся окне **Save Project** нажать кнопку **Save**.



Следует обратить внимание, что в момент сохранения проекта в папке, имя которой указано в поле **Location**, для сохраняемого проекта будет создана новая папка (если установлен флажок **Create directory for solution**).

КОМПИЛЯЦИЯ

Процесс преобразования исходной программы в выполняемую называется *компиляцией* или построением (build). Укрупненно процесс построения программы можно представить как последовательность двух этапов: компиляция и компоновка. На этапе компиляции выполняется перевод исходной программы (модулей) в некоторое внутреннее представление. На этапе компоновки – объединение модулей в единую программу.

Процесс построения программы активизируется в результате выбора в меню **Debug** команды **Build solution**, а также в результате запуска программы из среды разработки (меню **Debug**, команда **Start Debugging**), если с момента последней компиляции в программу были внесены изменения.

Результат компиляции отражается в окне **Error List**. Если в программе нет ошибок, то по завершении процесса компиляции окно **Error List** выглядит так, как показано на рис. 1.16.

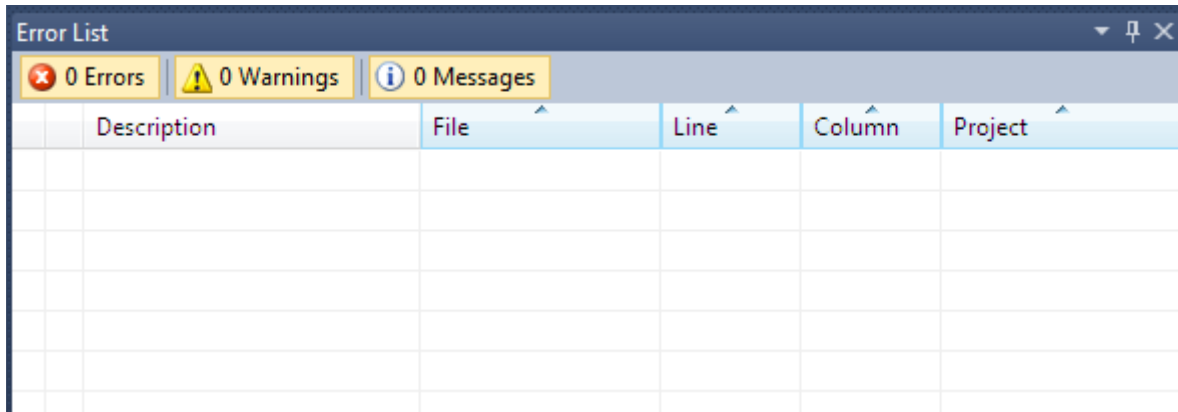


Рис. 1.16

Если в процессе построения в программе обнаруживаются ошибки, то в окне **Error List** (см. рис. 1.16) выводится их список.

ЗАПУСК ПРОГРАММЫ

Пробный запуск программы можно выполнить из Visual Studio, не завершая работу со средой разработки. Для этого в меню **Debug** надо выбрать команду **Start Debugging**. Можно также сделать щелчок на находящейся в панели инструментов **Debug** кнопке **Start Debugging** (рис.1.17) или нажать клавишу <F5>.

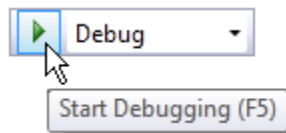


Рис. 1.17. Для запуска программы сделайте щелчок на кнопке **Start Debugging**

Результат этого алгоритма приведен на рис.1.18.

ВНЕСЕНИЕ ИЗМЕНЕНИЙ (редактирование)

Чтобы внести изменения в программу (после ее закрытия), нужно открыть соответствующий проект. Для этого надо в меню **File** выбрать команду **Open Project**, открыть папку проекта и сделать щелчок на значке файла проекта, т. е. на файле с расширением ***.sh** (рис. 1.19). Далее двойной клик по **Form1.cs**

Нужный проект для загрузки можно выбрать также из списка проектов, над которыми в последнее время работал программист. Этот список становится доступным в результате выбора в меню **File** команды **Recent Projects and Solutions**.

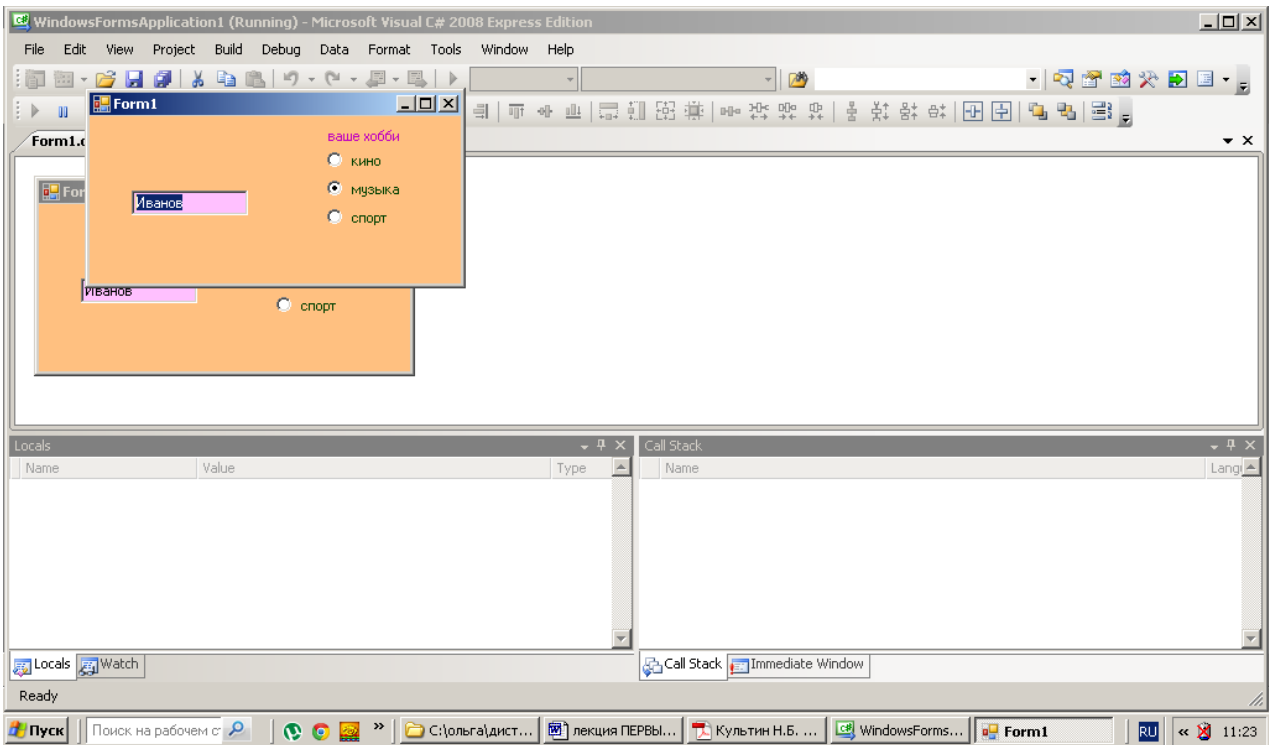


Рис. 1.18

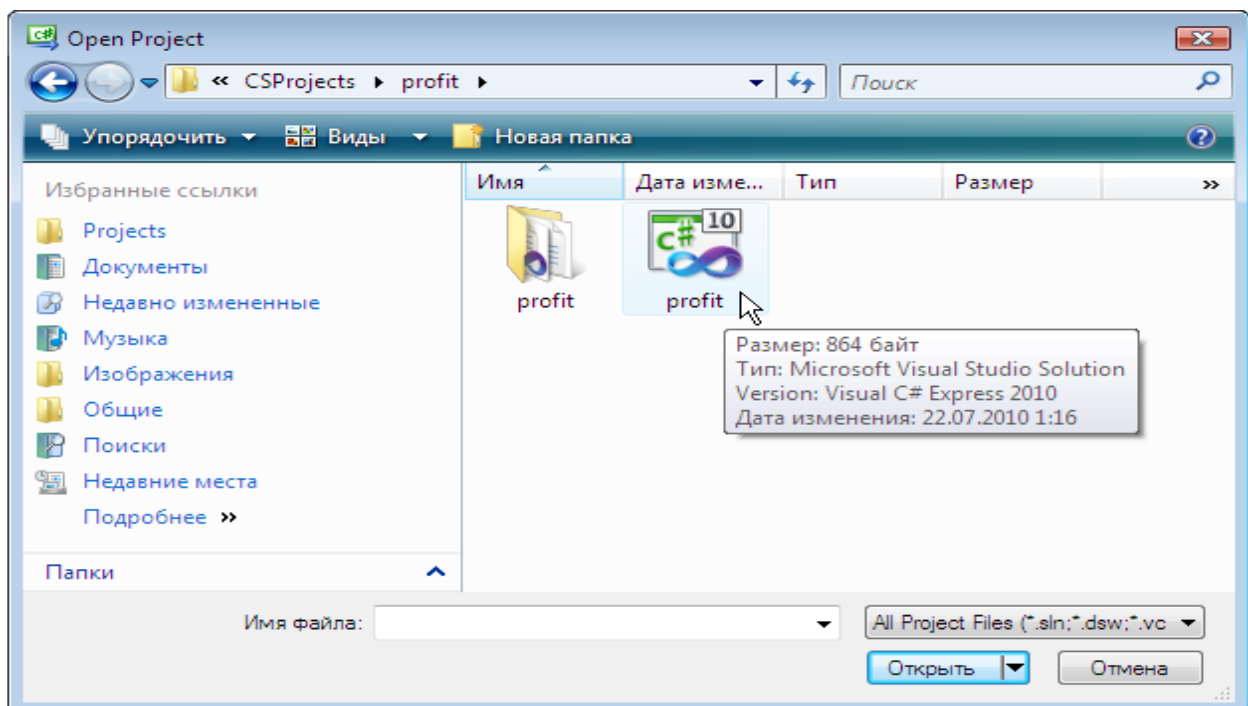
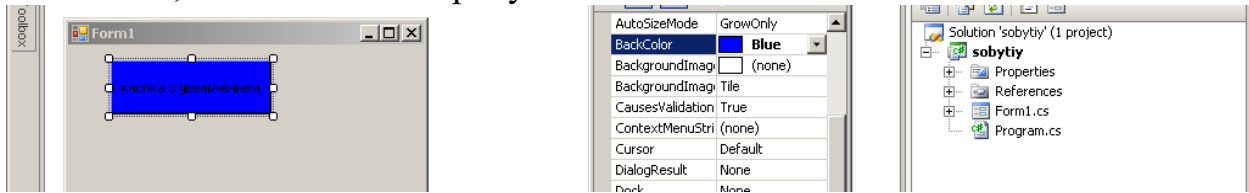


Рис. 1.19. Загрузка проекта для редактирования

ОБРАБОТКА СОБЫТИЙ

Создадим новый проект (в меню **File** команда **New Project**. В окне **New Project** тип приложения – **Windows Forms Application – Visual C#**. В поле **Name** ввести имя проекта латиницей) и разместим на форме кнопку Button.

Установим значения свойства Text = «Кнопка с увеличением», свойства BackColor = Blue, как показано на рисунке ниже.



Приведем основные свойства компонента **Button**.

Свойство	Описание
Name	Имя компонента. Используется для доступа к компоненту и его свойствам
Text	Текст на кнопке
BackColor	Цвет фона
Size	Размер кнопки
Location	Положение кнопки на поверхности формы. Уточняющее свойство X определяет расстояние от левой границы кнопки до левой границы формы, уточняющее свойство Y – от верхней границы кнопки до верхней границы клиентской области формы (нижней границы заголовка)
Enabled	Признак доступности кнопки. Кнопка доступна, если значение свойства равно True, и недоступна, если значение свойства равно False (в этом случае нажать кнопку нельзя, событие Click в результате щелчка на ней не возникает)
Visible	Позволяет скрыть кнопку (False) или сделать ее видимой (True)
TextAlign	Положение текста на кнопке. Текст может располагаться в центре кнопки (MiddleCenter), быть прижат к левой (MiddleLeft) или правой (MiddleRight) границе. Можно задать и другие способы размещения надписи (TopLeft, TopCenter, TopRight, BottomLeft, BottomCenter, BottomRight)
FlatStyle	Стиль. Кнопка может быть стандартной (Standard), плоской (Flat) или «всплывающей» (PopUp)

Сделав в окне **Properties** щелчок на кнопке **Events** (рис. 1.20), можно увидеть *события* (рис. 1.21), которые способен воспринимать выбранный объект (компонент или форма). Событие – это то, что происходит во время работы программы. Например, командная кнопка может реагировать на щелчок кнопкой мыши – событие, двойной щелчок мышью – событие `DbClick`.



Events

Рис. 1.20

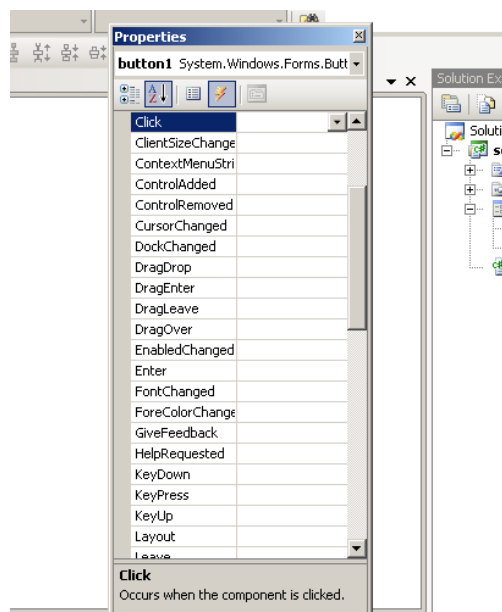


Рис. 1.21

В табл. 1 приведены некоторые события, возникающие в результате действий пользователя.

Таблица 1

Click	Щелчок кнопкой мыши
DoubleClick	Двойной щелчок кнопкой мыши
MouseDown	Нажатие кнопки мыши
MouseUp	Отпускание нажатой кнопки мыши
MouseMove	Перемещение указателя мыши
KeyPress	Нажатие клавиши
KeyDown	Нажатие клавиши. События KeyDown и KeyPress – это чередующиеся, повторяющиеся события, которые происходят до тех пор, пока не будет отпущена удерживаемая клавиша (в этот момент происходит событие KeyUp)
KeyUp	Отпускание нажатой клавиши
TextChanged	Признак, указывающий, изменился ли текст, находящийся в поле редактирования (изменилось значение свойства Text)
Load	Загрузка формы. Функция обработки этого события обычно используется для инициализации переменных, выполнения подготовительных действий
Paint	Событие происходит при появлении окна на экране в начале работы про-граммы, после появления части окна, которая, например, была закрыта другим окном
Enter	Получение фокуса элементом управления
Leave	Потеря фокуса элементом управления

Следует понимать, что одни и те же действия, но выполненные над разными объектами, вызывают разные события.

Реакция на событие реализуется как *функция обработки события*, которую программист должен написать

Выбираем событие Click для объекта Button1 (синяя кнопка) и делаем двойной клик по нему (рис. 1.22).

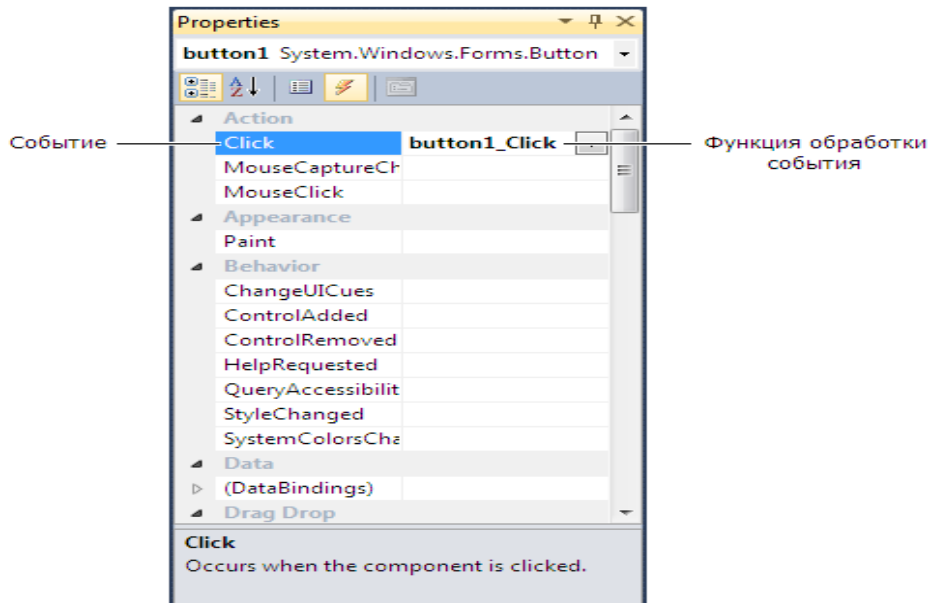


Рис. 1.22

В результате этих действий в модуль формы (cs-файл) будет добавлена функция (метод класса формы) обработки события и станет доступным окно редактора кода (рис. 1.23), в котором можно набирать код программы, реализующий функцию обработки события. В нашем случае система создала функцию **void button1_Click(...)**, Программисту осталось только добавить тело функции внутри фигурных скобок после имени функции. (Не рекомендуется ничего удалять или редактировать в строках, которые создала система автоматически. Программист должен только дописывать строки кода внутри функций.)

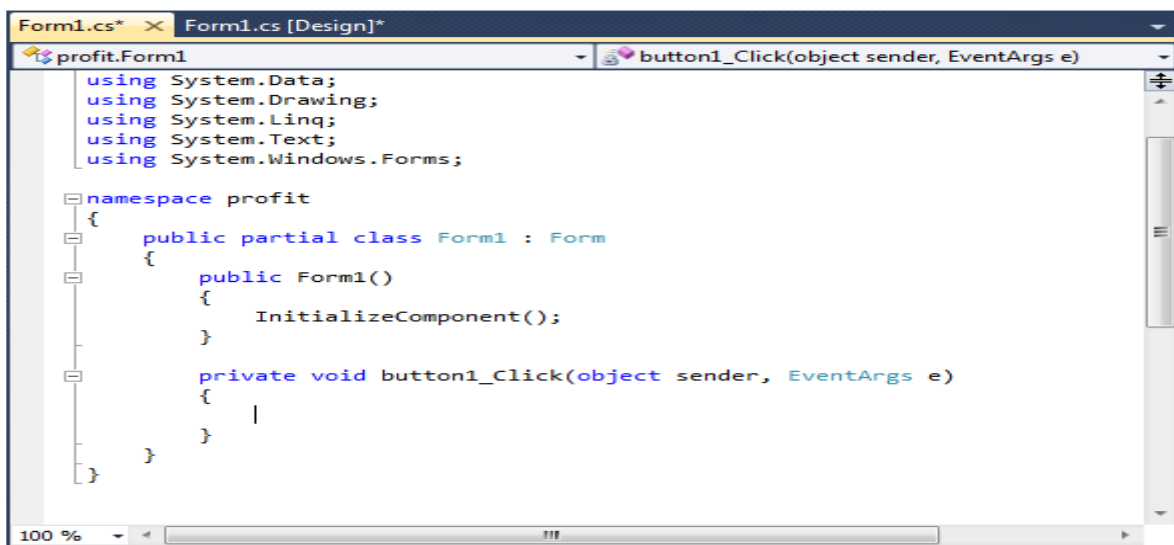


Рис. 1.23

Функция обработки события Click для кнопки button1 будет следующая:

```
button1.Width= button1.Width + 5;
button1.Height = button1.Height + 5;
```

Для удобства пользователя среда разработки подсказывает имена свойств и объектов, вам остается только начать набирать текст и выбрать нужное из предложенного списка и нажать Enter (рис. 1.24).

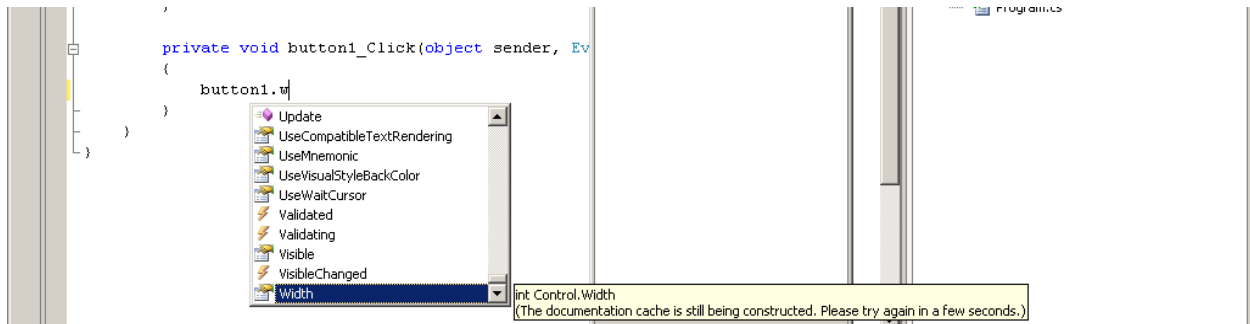
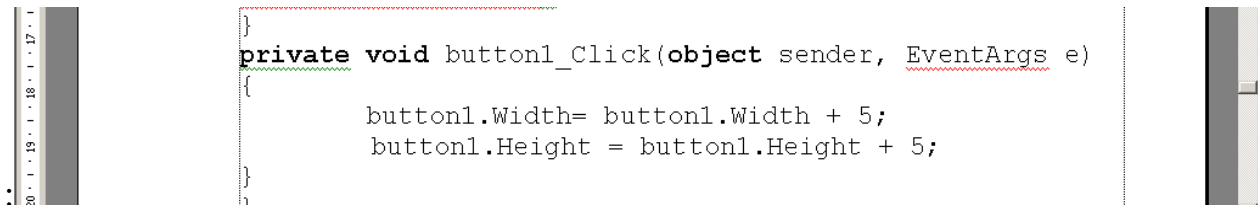


Рис. 1.24

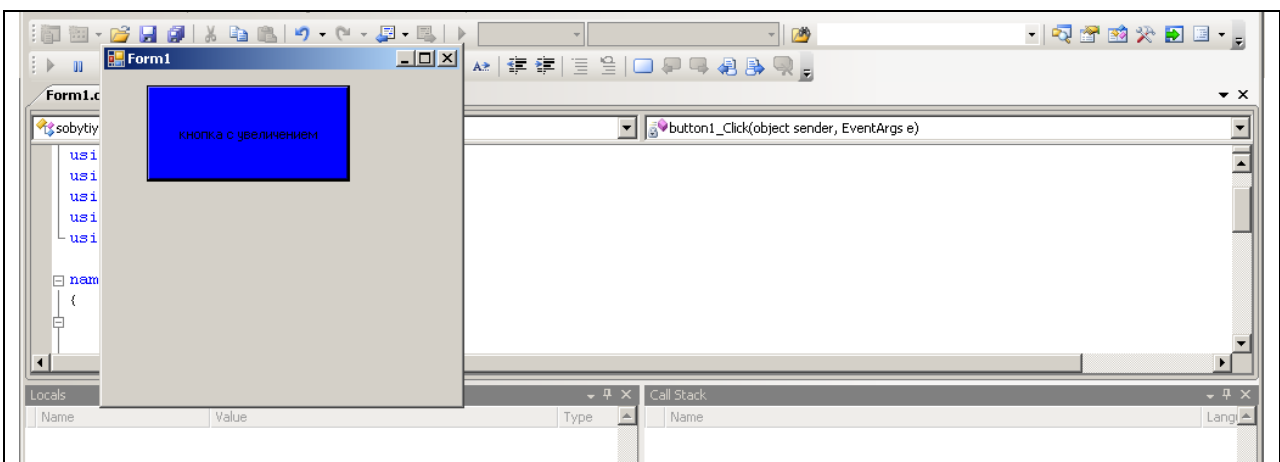
В результате получится так:



Когда текст набран, сохраняем проект и далее нажимаем кнопку Debug для запуска проекта.



Получится так:



Теперь можно проверить результат работы. Нажимаем на синюю кнопку, и она увеличивается на 5 пикселей при каждом нажатии

СТРУКТУРА ПРОЕКТА

Проект представляет собой совокупность файлов, которые компилятор использует для создания выполняемого файла. Структура проекта отображается в окне **Solution Explorer** (рис. 1.25) (solution – решение, так часто называют приложение, обеспечивающее *решение* некоторой задачи). В окне **Solution Explorer** перечислены файлы, образующие проект. Это окно удобно использовать для быстрого доступа к нужному элементу проекта.

Основными элементами проекта являются:

- главный модуль приложения (файл Program.cs);
- модули форм.

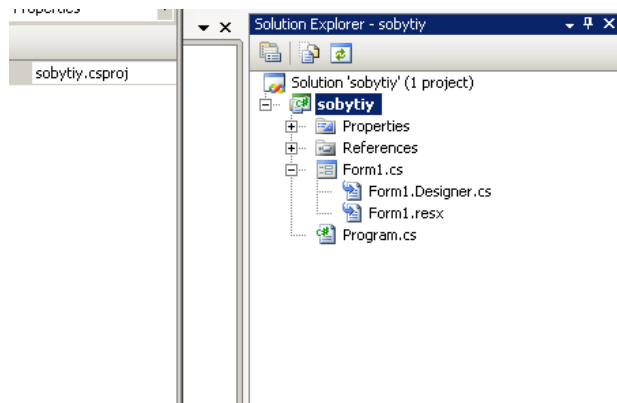
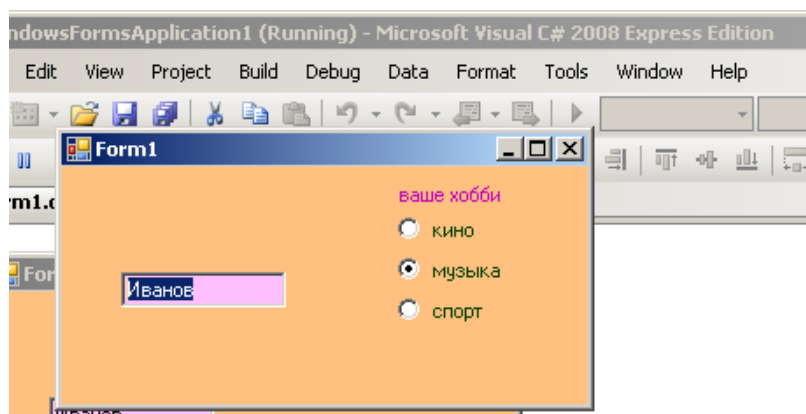


Рис. 1.25

Варианты заданий для лабораторной работы № 1

1. Создать первый проект, как указано на рисунке.



Выполнить 2 индивидуальных задания к лабораторной работе № 1, расположенные ниже. Номер варианта совпадает с вашим номером в журнале.

Варианты индивидуального задания

Задание № 1. Тема: «Изучение свойств компонентов»

1 Разместить на форме компоненты Метка и Поле для ввода. В метке – Ваше имя и отчество, установить шрифт Arial, курсив, 14, пурпурный цвет. Разрешить перенос строк в метке так чтобы отчество было на второй строке метки. В поле ввода – ваша фамилия.

2 Разместить на форме компоненты Метка и Поле для ввода. В метке – Ваше имя и отчество. Разрешить перенос строк в метке так, чтобы отчество было на второй строке метки. В поле ввода – ваша фамилия. Цвет фона в поле ввода – синий. Шрифт текста установить Arial, курсив, 14, пурпурный цвет.

3 Разместить на форме компоненты Метка и Поле для ввода. В метке – Ваше имя и отчество. В поле ввода – ваша фамилия. Цвет фона в поле ввода – желтый. Шрифт текста установить Arial, подчеркнутый, 4, зеленый цвет.

4 Разместить на форме компоненты Метка и Поле для ввода. В метке – Ваше имя и отчество. Цвет текста в метке – красный. В поле ввода – ваша фамилия. Шрифт текста установить Arial, подчеркнутый, 14, зеленый цвет. Сделать поле недоступным для ввода вовремя запуска приложения.

5 Разместить на форме компоненты Метка и Поле для ввода. В метке – Ваше имя и отчество, установить шрифт Times New Roman. Выравнивание текста в метке по центру. Разрешить перенос строк в метке так, чтобы отчество было на второй строке метки. В поле ввода – ваша фамилия на зеленом фоне.

6 Разместить на форме компоненты Метка и Поле для ввода. В метке – Ваше имя и отчество. Разрешить перенос строк в метке так, чтобы отчество было на второй строке метки. Выравнивание текста в метке по центру. В поле ввода – ваша фамилия. Установить невозможность редактирования текста в поле во время выполнения программы.

7 Разместить на форме компоненты Метка и Кнопка. В метке – Ваше имя. Выравнивание текста в метке по центру. На кнопке – Ваша фамилия. Установить недоступность кнопки для нажатия.

8 Разместить на форме компоненты Метка и две Кнопки. В метке – Ваше имя. На кнопке № 1 Ваше отчество. На кнопке № 2 – Ваша фамилия.

9 Разместить на форме три Кнопки. На кнопке № 1 – Ваше имя. На кнопке № 2 – Ваше отчество. На кнопке № 3 – Ваша фамилия. Цвет кнопок задать красный, желтый, зеленый.

10 Разместить на форме три Кнопки. На кнопке № 1 – Ваше имя. На кнопке № 2 – Ваше отчество. На кнопке – № 3 Ваша фамилия. Установить кнопку № 2 невидимой. Цвет текста на кнопках задать красный, желтый, зеленый.

11 Разместить на форме три Кнопки. На кнопке № 1 – Ваше имя. На кнопке № 2 – Ваше отчество. На кнопке № 3 – Ваша фамилия. Установить кнопку № 2 недоступной для нажатия. Разместить на форме Метку. Разрешить перенос строк в метке так чтобы произвольный текст располагался в две строки.

12 Разместить на форме компоненты Метка и 3 радиокнопки внутри Group Box. Текст в компонентах: Имя Фамилия, Отчество, Специальность. Метку сделать невидимой.

13 Создать компонент «рамка группы радиокнопок», обозначив его как «переключатели». Разместить в пределах «рамки группы переключателей» три компонента «переключатель», обозначив их соответственно «Переключатель 1», «Переключатель 2», «Переключатель 3». Для второго переключателя установить свойство, означающее, что он выбран по умолчанию.

14 Цвет форма установить зеленый, в заголовке формы написать «Лабораторная № 1». Создать компонент «поле редактирования». Написать в нем свою фамилию. Установить не возможность редактирование текста во время запуска программы.

15 Создать компонент «поле редактирования». Написать в нем свою фамилию. Установить не возможность редактирование текста во время запуска программы. Цвет фона в поле – синий. Создать компонент Кнопка, написать на ней Ваше имя. Установить кнопку недоступной для нажатия.

16 Разместить на форме компоненты Метка и Поле для ввода. В метке – Ваше имя и отчество. В поле ввода – ваша фамилия. Цвет фона в поле ввода – желтый. Шрифт текста установить Arial, подчеркнутый, 12, зеленый цвет.

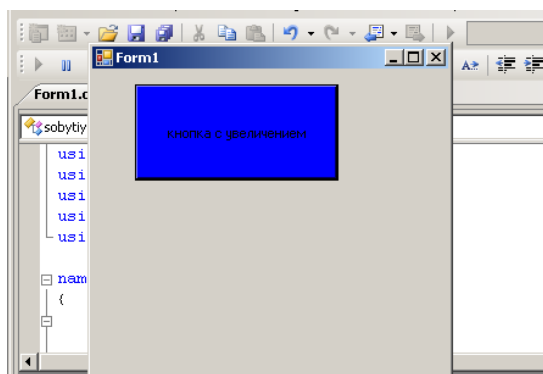
17 Создать компонент «поле редактирования». Написать в нем свою фамилию. Цвет фона в поле – желтый, текст размером 12. Создать компонент Кнопка, написать на ней Ваше имя.

18 Создать компонент «поле редактирования». Написать в нем свою фамилию. Цвет текста красный, размер 12. Создать компонент «рамка группы радиокнопок». Разместить в пределах «рамки группы» три радио кнопки, обозначив их соответственно «Математика», «Информатика», «Физика». Для второго переключателя установить свойство, означающее, что он выбран по умолчанию.

19 Разместить на форме компоненты Метка и Поле для ввода. В метке – Ваше имя и отчество, установить шрифт Times New Roman, курсив, 16, зеленый цвет. Разрешить перенос строк в метке так чтобы отчество было на второй строке метки. В поле ввода – ваша фамилия. Установить невозможность редактирования текста в поле во время выполнения программы.

Задание № 2. Тема: «Изменение свойств компонентов при обработке события»

Создать второй проект с увеличивающейся синей кнопкой с использованием событий (см. лекции), как указано на рисунке.



1 Разместить на форме метку и кнопку. По нажатии кнопки должен появиться текст в метке с вашей фамилией, цвет фона метки изменится на красный.

2 Разместить на форме метку и кнопку. По нажатии кнопки должен появиться текст в метке с вашей фамилией, цвет фона формы – голубой.

3 Разместить на форме метку и кнопку. На метке отобразить текст «Добрый день». При нажатии кнопки цвет текста должен измениться на синий, а размер текста – увеличиться на 3.

4 Разместить на форме метку и кнопку. На метке отобразить текст «Добрый день». При нажатии кнопки цвет кнопки должен измениться на синий, а цвет фона метки должен измениться на желтый

5 Разместить на форме метку и кнопку. На метке отобразить текст «Добрый день». При нажатии кнопки размер текста должен увеличиться на 4, а цвет фона метки должен измениться на желтый

6 Разместить на форме метку и кнопку. На метке отобразить текст «Добрый день». При нажатии кнопки цвет фона метки должен измениться на синий, а размер метки по высоте увеличиться на 10.

7 Разместить на форме метку и кнопку. На метке отобразить текст «Добрый день». При нажатии кнопки цвет формы должен измениться на синий, а метка должна переместиться на 5 пикселей вправо.

8 Разместить на форме метку и кнопку. На метке отобразить текст «Добрый день». При нажатии кнопки цвет формы должен измениться на синий, метка должна переместиться вниз на 10 пикселей

9 Разместить на форме метку и кнопку. На метке отобразить текст «Добрый день». При нажатии кнопки цвет формы должен измениться на красный, а размер формы – увеличиться на 5 по высоте и ширине.

10 Разместить на форме поле ввода и кнопку. В заголовке формы поменять текст на вашу фамилию. При нажатии кнопки поле ввода должно перемещаться на 10 пикселей вниз, а размер формы – увеличиться на 5 по высоте и ширине.

11 Разместить на форме поле ввода и кнопку. В заголовке формы поменять текст на вашу фамилию. При нажатии кнопки цвет текста в поле ввода должен измениться на красный, а размер текста – увеличиться на 5.

12 Разместить на форме поле ввода и кнопку. В заголовке формы поменять текст на вашу фамилию. При нажатии кнопки поле ввода должно перемещаться вправо на 5 пикселей, а сам текст измениться на ваше имя.

13 Разместить на форме поле ввода и кнопку. В поле установить текст «привет!». При нажатии на кнопку цвет фона поля должен измениться на голубой, а размер текста в поле – увеличиться на 3.

14 Разместить на форме поле ввода и кнопку. В поле установить текст «привет!». При нажатии на кнопку цвет кнопки должен измениться на голубой, а размер поля по высоте увеличиться на 7.

15 Разместить на форме поле ввода и кнопку. В поле установить текст «привет!». При нажатии на кнопку цвет фона в поле должен измениться на голубой, цвет кнопки должен измениться на желтый,

Задание № 3 Тема: «Передвижение компонентов»

1 Разместить на форме поле ввода и кнопку. При нажатии на кнопку размер поля ввода должен увеличиваться на несколько пикселей.

2 Разместить на форме поле ввода и кнопку. При нажатии на кнопку положение поля ввода должно сместиться вправо на несколько пикселей.

3 Разместить на форме поле ввода и кнопку. При нажатии на кнопку положение поля ввода должно сместиться влево на несколько пикселей.

4 Разместить на форме поле ввода и кнопку. При нажатии на кнопку положение поля ввода должно сместиться вверх на несколько пикселей.

5 Разместить на форме поле ввода и кнопку. При нажатии на кнопку положение поля ввода должно сместиться вниз на несколько пикселей.

6 Разместить на форме поле ввода и кнопку. При нажатии на кнопку положение поля ввода должно сместиться по диагонали вниз на несколько пикселей.

7 Разместить на форме поле ввода и кнопку. При нажатии на кнопку положение поля ввода должно сместиться по диагонали вверх на несколько пикселей.

8 Разместить на форме поле ввода и кнопку. При нажатии на кнопку размер поля ввода должен уменьшаться на несколько пикселей.

9 Разместить на форме 2 кнопки. При нажатии на кнопку размер второй кнопки должен увеличиваться на несколько пикселей.

10 Разместить на форме кнопку. При нажатии на кнопку ее положение должно сместиться вправо на несколько пикселей.

11 Разместить на форме кнопку. При нажатии на кнопку ее положение должно сместиться влево на несколько пикселей.

12 Разместить на форме кнопку. При нажатии на кнопку ее положение поля ввода должно сместиться вверх на несколько пикселей.

13 Разместить на форме кнопку. При нажатии на кнопку ее положение должно сместиться вниз на несколько пикселей.

14 Разместить на форме кнопку. При нажатии на кнопку ее положение должно сместиться по диагонали вниз на несколько пикселей.

15 Разместить на форме кнопку. При нажатии на кнопку ее положение должно сместиться по диагонали вверх на несколько пикселей.

16 Разместить на форме и кнопку. При нажатии на кнопку ее размер должен уменьшаться на несколько пикселей.

Лабораторная работа № 2

ФУНКЦИИ ПРЕОБРАЗОВАНИЯ ТЕКСТА В ЧИСЛО

Значением свойства `Text` всех компонент среды `visual C` является строка текста, т. е. свойство `Text` строкового типа, поэтому для преобразования строк в числа используются принадлежащие *пространству имен* `System.Convert` функции:

ToInt32 для преобразования строки в целое число.

ToDouble для преобразования строки в вещественное число и

Например:

```
Int x = System.Convert.ToInt32(textBox2.Text);
```

```
Double sum = System.Convert.ToDouble(textBox1.Text);
```

Следует обратить внимание, что функция `ToDouble` возвращает результат только в том случае, если строка, переданная ей в качестве параметра, является изображением дробного числа, что предполагает использование *запятой* в качестве десятичного разделителя (при стандартной для России настройке операционной системы). Аналогично, параметр функции `ToInt32` должен представлять собой строку, являющуюся изображением целого числа.

Другие функции преобразования строк:

Функция	Значение
<code>ToSingle(s)</code> , <code>ToDouble(s)</code>	Дробное типа <code>Single</code> , <code>Double</code>
<code>ToByte(s)</code> , <code>ToInt16(s)</code> , <code>ToInt32(s)</code> , <code>ToInt64(s)</code> ,	Целое типа <code>Byte</code> , <code>Int16</code> , <code>Int32</code> , <code>Int64</code>

ФУНКЦИЯ ПРЕОБРАЗОВАНИЯ ЧИСЛА В СТРОКУ ТЕКСТА

Для преобразования числа в строку используется функция (метод) **ToString**.

Например:

```
Int x=23;
```

```
label3.Text = x.ToString("n");
```

Параметр метода `ToString` задает формат строки-результата: «n» – числовой (от англ. *number*). «c» – финансовый (от англ. *currency*); Следует обратить внимание, что при использовании финансового формата после числового значения выводится обозначение денежной единицы (в соответствии с настройкой операционной системы). В табл. 2. приведены возможные форматы представления числовой информации.

Форматы представления чисел

Параметр функции ToString	Формат	Пример
"c"	Currency – финансовый (денежный). Используется для представления денежных величин. Обозначение денежной единицы, разделитель групп разрядов, способ отображения отрицательных чисел определяют соответствующие настройки операционной системы	55 055,28 р.
"e"	Scientific (exponential) – научный. Используется для представления очень маленьких или очень больших чисел. Разделитель целой и дробной частей числа задается в настройках операционной системы	5,50528+E004
"f"	Fixed – число с фиксированным десятичным разделителем. Используется для представления дробных чисел. Количество цифр дробной части, способ отображения отрицательных чисел определяют соответствующие настройки операционной системы	55 055,28
"n"	Number – числовой. Используется для представления дробных чисел. Количество цифр дробной части, символ-разделитель групп разрядов, способ отображения отрицательных чисел определяют соответствующие настройки операционной системы	55 055,28
"g"	General – универсальный формат. Похож на Number, но разряды не разделены на группы	55055,275
"r"	Roundtrip – без округления. В отличие от формата N, этот формат не выполняет округления (количество цифр дробной части зависит от значения числа)	55 055,2775

Для иллюстрации этих функций создадим *приложение*, позволяющее посчитать доход по вкладу (рис. 2.1).

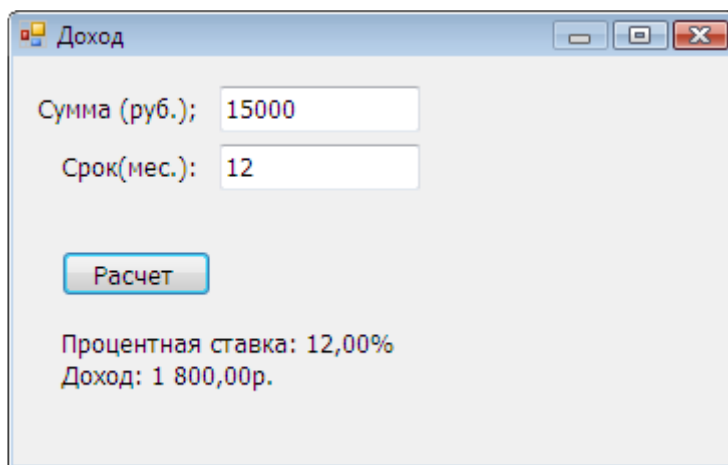
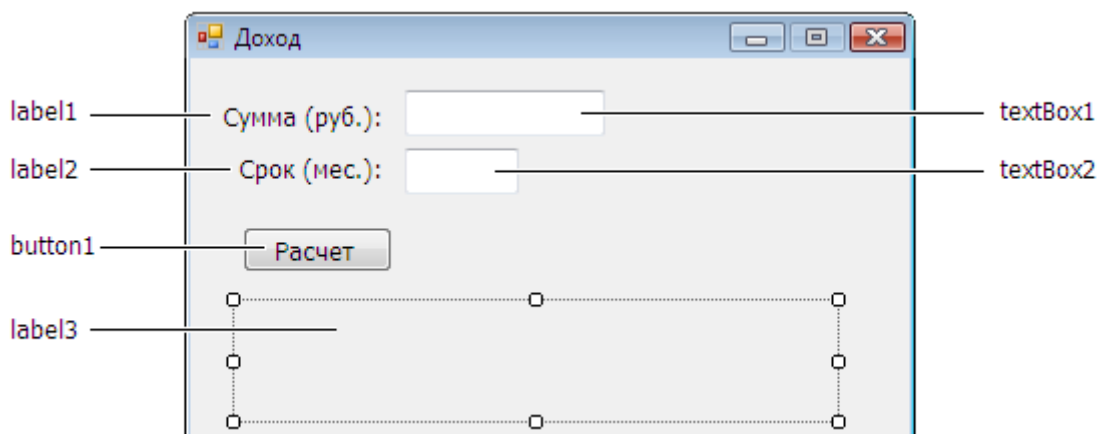


Рис. 2.1. Окно программы «Доход»

Создадим новый проект как всегда (В меню **File** команд **New Project**. В окне **New Project** тип приложения – **Windows Forms Application – Visual C#**. В поле **Name** имя проекта – **profit**).

Расположим на форме два компонента Label1 Label2. В свойстве text этих компонентов напишем соответственно «Сумма (руб.)» и «Срок вклада (мес.)» Также расположим 2 компонента TextBox для ввода в них суммы вклада и срока вклада (свойства TextBox и Label см. в 1-м проекте) Все объекты расположите, как показано на рис. 2.1. Добавьте кнопку Button1 с надписью «доход» (свойства Button см. в лекции 2 проект «события»)

И разместим еще один компонент Label 3 для вывода результата, должно получиться так:



Далее создаем событие, связанное с кнопкой, как во втором проекте. Выделяем кнопку переходим на вкладку Events (события) выбираем событие Click два раза щелкаем по нему система открывает файл *.cs с кодом для функции `void Button1_Click(...)` { }

Функция `button1_Click` должна вычислить доход по вкладу и вывести результат расчета в поле компонента **label3**. Исходные данные (сумма и срок вклада) вводятся из полей редактирования `textBox1` и `textBox2` путем обращения к свойству `Text`. Значением свойства `Text` является строка, которая находится в поле редактирования. Свойство `Text` строкового типа, поэтому для пре-

образования строк в числа используются принадлежащие *пространству имен* System.Convert функции ToDouble и ToInt32.

Например:

```
Double sum = System.Convert.ToDouble(textBox1.Text);
Int period = System.Convert.ToInt32(textBox2.Text);
```

Вычисленные значения процентной ставки и величины дохода выводятся в поле label3 путем присваивания значения свойству Text строкового типа . Для преобразования дробного числа в строку используется функция (метод) ToString.

Например:

```
label3.Text = percent.ToString("n");
```

Исходя из этого код программы будет следующий:

```
private void button1_Click(object sender, EventArgs e)
{
double sum; // сумма
int period; // срок
double percent; // процентная ставка
double profit; // доход
sum = System.Convert.ToDouble(textBox1.Text);
period = System.Convert.ToInt32(textBox2.Text);
if (sum < 10000)
percent = 8.5;
else
percent = 12;
profit = sum * (percent/100/12) * period;
label3.Text = "Процентная ставка: " + percent.ToString("n") + "%\n" +
"Доход: " + profit.ToString("c");
}
```

СООБЩЕНИЯ КОМПИЛЯТОРА ОБ ОШИБКАХ

Компилятор генерирует выполняемую программу (exe-файл) только в том случае, если в исходной программе нет ошибок.

Если в программе есть ошибки, то программист должен их устранить. Процесс устранения ошибок носит итерационный характер. Обычно сначала устраняются наиболее очевидные ошибки, например, объявляются необъявленные переменные, затем, после выполнения повторной компиляции, – остальные.

В табл. 3. приведены сообщения компилятора о типичных ошибках.

Сообщения компилятора об ошибках

Сообщение компилятора	Вероятная причина ошибки
The name <i>идентификатор</i> does not exist in the current context (В текущем контексте имя не существует)	1. Используемая в программе переменная не объявлена. 2. Ошибка при записи имени переменной. Например, объявлена переменная sum, а в тексте программы написано: Sum
Cannot implicitly convert type <i>type1</i> to <i>type2</i> (Невозможно преобразовать значение типа <i>type1</i> в значение типа <i>type2</i>) Пример: Cannot implicitly convert type 'double' to 'int'	В инструкции присваивания тип выражения не соответствует типу переменной, которой присваивается значение. Например, если переменные n и m целого типа, то инструкция n = m/12 неверная, т. к. выражение m/12 дробное
Use of unassigned local variable (Используется локальная переменная, которой не присвоено начальное значение)	В программе нет инструкции, присваивающей переменной начальное значение
<i>Пространство_имен</i> does not contain definition for <i>Идентификатор</i> (Пространство имен <i>Пространство_имен</i> не содержит определение идентификатора <i>Идентификатор</i>)	Неправильно (например, не в том регистре) записано имя пространства имен или идентификатор (например, имя функции) ему принадлежащий. Пример: System.Convert.ToInt32(textBox1.Text) – неправильно записано имя функции. Должно быть ToInt32
',' expected (ожидается символ "точка с запятой")	После инструкции нет символа "точка с запятой"

Следует обратить внимание на то, что компилятор языка C# различает прописные и строчные буквы. Запись имени переменной или функции "не в том регистре" – типичная причина ошибок в программе.

ПРЕДУПРЕЖДЕНИЯ

В программе могут быть не только ошибки, но и неточности. Например, инструкция присваивания целой переменной дробного значения формально является верной. Присвоить значение переменной можно, но что делать с дробной частью? Отбросить или округлить? Что хотел сделать программист, записав эту инструкцию?

При обнаружении в программе неточностей компилятор выводит предупреждения – Warnings. Например, при обнаружении не объявленной, но не используемой переменной выводится сообщение: unreferenced local variable. Действительно, зачем объявлять переменную и не использовать ее?

В табл. 4 приведены предупреждения и подсказки компилятора о типичных неточностях в программе.

Таблица 3

Предупреждения и подсказки компилятора

Сообщение	Причина
The variable ... is declared but never used	Переменная объявлена, но не используется
The variable ... is assigned but its value is never used	Переменной присвоено значение, но оно не используется

Варианты заданий для лабораторной работы № 2

1. Создать проект, позволяющий посчитать доход по вкладу, как указано на рис. 2.2.

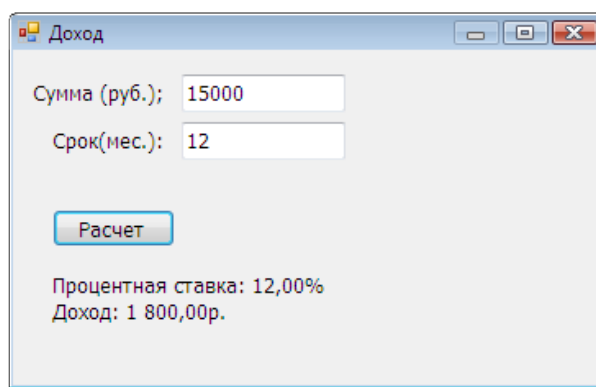


Рис. 2.2. Окно программы «Доход»

Выполнить три индивидуальных задания, расположенные ниже. Номер варианта совпадает с Вашим номером в журнале.

Варианты индивидуального задания

Задание № 1

1 Составить программу, находящую периметр равнобедренного треугольника по его основанию a и высоте h , проведенной к основанию (a и h – вещественные). Для нахождения боковой стороны b треугольника использовать теорему Пифагора: $b^2 = (a/2)^2 + h^2$.

2 Длина выражена в сантиметрах. Выразить ее в дюймах. (1 дюйм=2.5 см)

3 Составить программу, находящую площадь кольца, заключенного между двумя окружностями с общим центром и радиусами R_1 и R_2 (R_1 и R_2 – вещественные, $R_1 > R_2$). Воспользоваться формулой площади круга радиуса R : $S = \pi * R^2$.

4 Составить программу, находящую величину угла в радианах R , если дана его величина D в градусах (D – вещественное число, $0 < D < 360$). Воспользоваться следующим соотношением: $R = \pi D/180^\circ$.

5 Приобрели A шт. книг по цене B руб. за шт. и C шт. тетрадей по цене D руб. за шт. Определить стоимость всего товара.

6 Дано: a и b – стороны прямоугольника. Найти его площадь и периметр. Найти площадь прямоугольного треугольника, построенного на катетах a и b (площадь прямоугольного треугольника рассчитывается как половина произведения сторон катетов).

7 Найти среднеарифметическое трех чисел x , y , z .

8 Известен объем информации в байтах. Выразить его в мегабайтах и гигабайтах.

9 Дано R – радиус окружности. Найти длину окружности (длина окружности прямо пропорциональна удвоенному произведению Π на радиус окружности).

10 На базу завезли C кг арбузов по цене 2 руб. за килограмм и B кг дынь по цене 5 руб. за килограмм. Определить, сколько всего стоят арбузы и дыни.

11 Дано a и b – стороны прямоугольника. Найти его площадь и периметр.

12 Дано значение веса. Перевести значение веса, выраженное в граммах, в унции (1 унция = 28.3 г)

13 Составить программу, находящую площадь круга радиусом R (R – вещественное) по заданному радиусу. Площадь круга радиуса R вычисляется по формуле $S = \pi * R^2$.

14 На базу завезли A кг яблок по цене 20 руб. за килограмм и B кг груш по цене 30 руб. за килограмм. Определить, сколько всего стоят яблоки и груши.

15 Составить программу, вычисляющую среднее арифметическое $A_{Meap} = (X + Y)/2$ и среднее геометрическое $G_{Meap} = \text{Sqrt}(X * Y)$ двух положительных чисел X и Y .

16 Составить программу, вычисляющую по стороне a равностороннего треугольника его периметр $P = 3 * a$ и площадь $S = a^2 * \text{Sqrt}(3)/4$.

17 Составить программу, определяющую по времени T (в секундах) содержащееся в нем количество часов H , минут M и секунд S (T – входная, H , M и S – выходные переменные целого типа).

Задание № 2. Тема: «Целочисленное деление»

1 Ввести два числа целого типа a и b . Найти остаток от деления a на b .

2 Ввести два числа целого типа a и b . Поделить нацело b на a (целочисленное деление).

3 Ввести число целого типа a (a – не менее чем двухзначное). Найти остаток от деления a на 10.

4 Поделить нацело сумму a и b на разность c и d .

5 Дано Q целое. Найти остаток от деления Q на 3.

6 Дано Q целое. Найти целую часть от деления Q на 3.

7 Дано R целое. Найти целую часть от деления R на 2.

- 8 Ввести 3 числа целого типа a , c и v . Поделить нацело c на сумму чисел v и a . (целочисленное деление).
- 9 Найти целую часть от деления суммы целых чисел A и B на число D .
- 10 Ввести числа целого типа A , B , C . Найти остаток от деления суммы целых чисел A и B на число C .
- 11 Найти остаток от деления суммы a и v на сумму c и d .
- 12 Дано Q целое. Найти остаток от деления Q на 2.
- 13 Ввести числа целого типа A , B , C . Найти целую часть от деления числа A на сумму чисел B и D .
- 14 Ввести числа целого типа A , B , C . Найти остаток от деления разности чисел A и B на число C .
- 15 Ввести 2 числа C и D . Найти остаток от деления целого числа C на число D .

Задание № 3

- 1 Пользователь вводит текст на русском языке. Вычислить количество слов, начинающихся на «м». Количество слов «Компьютер» или «компьютер», а также количество предложений.
- 2 Пользователь вводит текст на русском языке. Заменить в тексте слова «ПК» на «компьютер», подсчитав их количество.
- 3 Пользователь вводит текст на русском языке. Вывести исходный текст, заменив в нем слово «Иванов И.И.» на «Сидоров А.А.». Заменить круглые скобки на фигурные, подсчитав их количество.
- 4 Пользователь вводит текст. Вывести исходный текст, заменив в нем слово «Pascal» на «C++», подсчитав их количество. Вычислить количество слов «компьютер».
- 5 Пользователь вводит текст на русском языке. Вывести исходный текст, заменив в нем слово «плохо» на «хорошо». Вычислить количество всех слов.
- 6 Пользователь вводит текст на русском языке. Вычислить количество слов, начинающихся на «А». Количество слов «мало» или «Мало». Заменить в тексте слова «доллар» на «рубль».
- 7 Пользователь вводит текст на русском языке. Заменить в тексте слова «Максимальный» на «Наибольший». Удалить все слова «Иванов И.И.». вычислить количество предложений.
- 8 Пользователь вводит текст на русском языке. Заменить в тексте слова «кризис» на «проблема», подсчитав их количество. Удалить все слова «компьютер»,
- 9 Пользователь вводит текст на русском языке. Вывести исходный текст, заменив в нем квадратные скобки на круглые. Вычислить количество всех слов и количество появления слова «обучаемый».
- 10 Пользователь вводит текст на русском языке. Вывести исходный текст, заменив в нем слово «проблема» на «задача». Удалить все слова «Иванов И.И.».

11 Пользователь вводит текст. Вывести исходный текст, заменив в нем слово «три» на «удовлетворительно». Вычислить количество слов, начинающихся на «к».

12 Пользователь вводит текст на русском языке. Вывести исходный текст, заменив в нем слово «дублирование» на «копирование». Вычислить количество всех слов.

13 Пользователь вводит текст на русском языке. Вывести исходный текст, заменив в нем слово «четыре» на «хорошо». Вычислить количество всех слов и предложений. Заменить все скобки на пробелы.

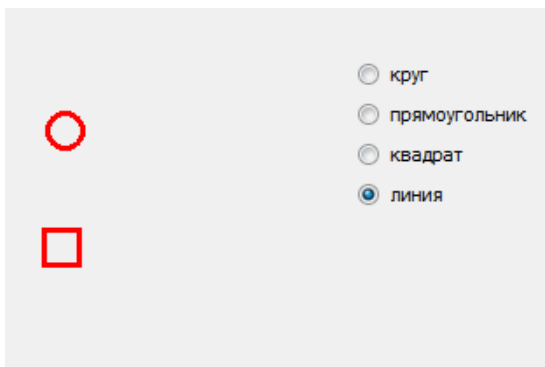
14 Пользователь вводит текст. Вывести исходный текст, заменив в нем слово «Pascal» на «C++». Удалить символы «*». Заменить все цифры на пробелы.

15 Пользователь вводит текст на русском языке. Вывести исходный текст, заменив в нем слово «дублирование» на «копирование». Вычислить количество всех слов.

Лабораторная работа № 3

ИСПОЛЬЗОВАНИЕ ВИЗУАЛЬНЫХ КОМПОНЕНТОВ И НАСТРОЙКА ИХ СВОЙСТВ

ИСПОЛЬЗОВАНИЕ РАДИОКНОПОК



Основное свойство радиокнопки `radioButton` – это свойство **Checked=true** кнопка выбрана – `false` –кнопка выключена. В зависимости от этого свойства выполняется тот или другой код.

Например:

```
if (radioButton2.Checked == true)
{
    textBox2.Text = Convert.ToString(i,
```

```
16); //например перевод в 16 систему исчисления
```

```
}
```

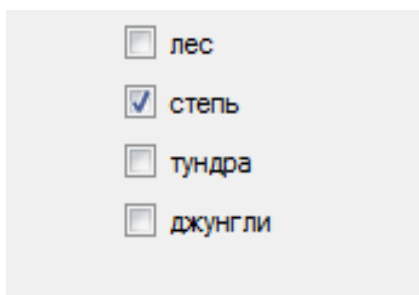
```
else if (radioButton3.Checked == true)
```

```
{
```

```
    textBox2.Text = Convert.ToString(i, 8);
```

```
}
```

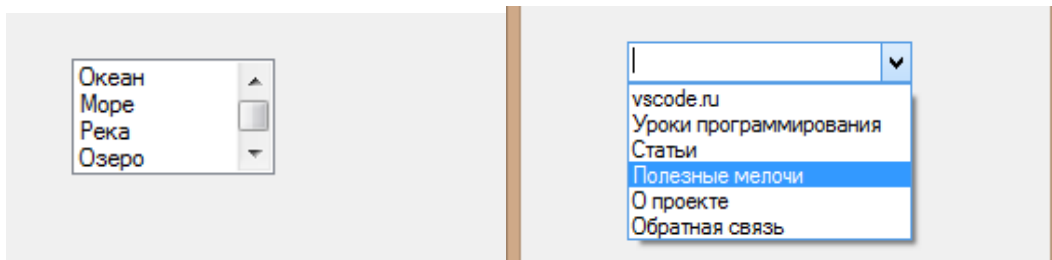
Например:



```
if ((radioButton1.Checked) && (radioButton1.Text == k)) f++;
```

Те же самые свойства у компонента **CheckBox**.
Checked

ИСПОЛЬЗОВАНИЕ СПИСКОВ LISTBOX И COMBOBOX



Компонент **listBox** позволяет запомнить в себе список текстовых элементов, и предоставить выбор одного из элементов для пользователя. Свойство **Items** хранит элементы списка. Элементы в список **listBox** можно добавлять на этапе конструирования формы, для этого нажать на троеточку на против свойства **Items** и в появившемся окне набрать элементы списка

1) метод **Add** используется для добавления программно элемента в список **listBox**. Элементы в список **listBox** можно добавлять как на этапе конструирования формы так и программно. Например:

```
listBox1.Items.Add("Лес");
listBox1.Items.Add("Степь ");
listBox1.Items.Add("Озеро");
listBox1.Items.Add("Море");
listBox1.Items.Add("Океан");
```

2) с помощью свойства **SelectedIndex** определяется Выбранный элемент и тогда можно выполнять те или иные действия в зависимости от выбранного пользователем пункта в списке. Нумерация как всегда с **0**. Значение = **-1** означает, что ничего не выбрано.

```
if (listBox1.SelectedIndex==2) {производим действия если выбран 2-й пункт }
или
```

```
listBox1.SelectedIndex = 2; – выделить 3-ий пункт
```

3) В свойстве **SelectedItem** хранится выделенный элемент списка

```
if(listBox1.SelectedIndex != -1) Tx=listBox1.SelectedItem.ToString();
```

 Выводим сообщение с указанием выбранного в списке пункта

```
MessageBox.Show(this, "Вы выбрали " + listBox1.SelectedItem);
```

4) свойство **Text** хранит содержимое текста только выбранного пункта
 (пока одно и то же):

```
MessageBox.Show(listBox1.Text);
```

5) количество элементов в списке можно получить через **listbox1.Items.Count**.

6) свойство **Items[1]** хранит элемент списка с указанным номером;

7) `Tt=listbox1.Items[index].ToString();`

Пример 1: Выделим `listBox1` и выполним по нему 2-й щелчок. Появится обработчик события `SelectedIndexChanged` (которое происходит, когда пользователь выделяет что-то в списке). Сделаем так чтобы при выборе в списке изменялось сообщение в `label2`.

```
private void listBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    if (listBox1.SelectedIndex == 0) label2.Text = " вы выбрали " +
listBox1.SelectedItem; //или listBox1.Items[0];
    if (listBox1.SelectedIndex == 1) label2.Text = " вы выбрали " +
listBox1.Items[1];
    if (listBox1.SelectedIndex == 2) label2.Text = " вы выбрали " +
listBox1.Items[2];
}
```

8) метод **Insert** позволяет вставить новый пункт в любое место списка.

```
listBox1.Items.Insert(1,"Лесостепь");
```

9) Для удаления пунктов из списка можно воспользоваться одним из следующих способов:

1 способ

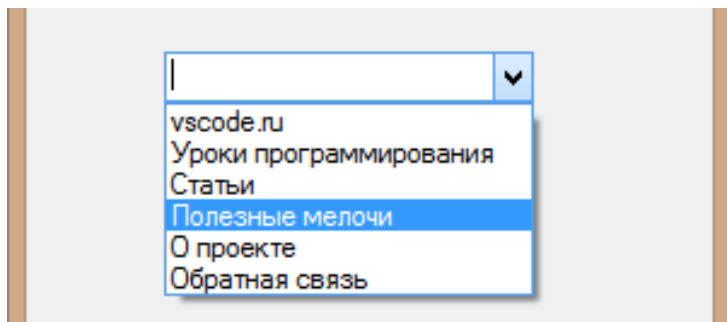
```
try
{
    listBox1.Items.RemoveAt(0);
}
catch
{
}
```

2 способ

```
listBox1.Items.Remove(listBox1.SelectedItem);
```

3 способ

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



Элемент управления **ComboBox** также позволяет запомнить в себе список текстовых элементов, и предоставить выбор одного из элементов для пользователя. Но в отличие от `listBox1` в **ComboBox** можно включить возможность редактирования, то

есть ввода элементов списка пользователем во время работы приложения. Остальные методы совпадают с описанными выше для `listBox1`.

Items – коллекция элементов списка

SelectedIndex – выделенный элемент ComboBox. `SelectedIndex`

Items[i]; – элемент под номером *i*. `ComboBox.Items[0]`;

Text – вводимый элемент ComboBox. `Text`


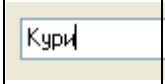
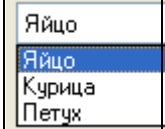
Count – количество элементов `ComboBox.Items.Count`.

Add – добавить `ComboBox.Items.Add(str)`,

Remove – удалить `ComboBox.Items.Remove(ComboBox.SelectedItem)`;

Insert – вставить

Основное поведение (возможность редактирования) ComboBox зависит от свойства **DropDownStyle**. Имеется 3 варианта значения этого свойства:

Simple		Текстовое поле ввода, в котором может быть отображен один элемент из списка. Выпадающий список отключен, пользователю выбрать элемент нельзя, однако программно это сделать можно. Окно ComboBox в этом режиме работает как простое однострочное текстовое окошко ввода, так что пользователь может отредактировать текст текущего элемента.
DropDown (по умолчанию)		Пользователь может выбрать любой элемент из списка с помощью выпадающего списка. Кроме того, разрешено редактирование (ввод) текста в текущий элемент списка. Тогда Вводимое значение находится в свойстве <code>comboBox1.Text</code>
DropDownList		Стандартное поведение выпадающего списка. Пользователь может выбрать любой элемент <code>ComboBox.Items</code> из выпадающего списка. Редактирование текущего элемента запрещено.

Добавлять элементы можно методом `ComboBox.Items.Add(str)`, удалять методами **Remove** и **RemoveAt**.

Элементы в списке программно доступны через свойство `ComboBox.Items`. Количество элементов в списке можно получить через `ComboBox.Items.Count`.

```
For (i=0; i< ComboBox1.Items.Count; i++)
```

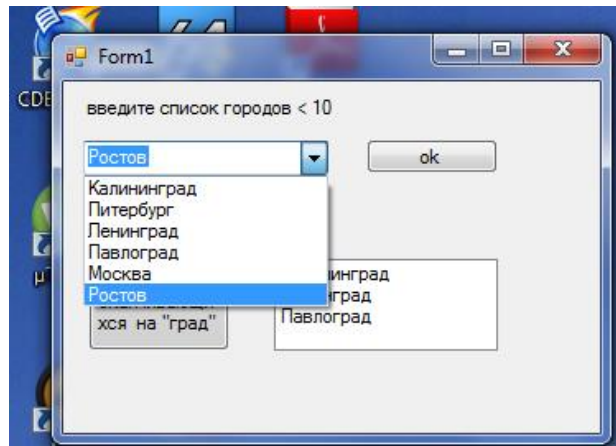
```
    Mass[i]= convert.ToInt32(ComboBox1.Items[i]);
```

элемент **Items** можно программно поменять:

```
ComboBox1.Items [i]="New ";
```

Пример 2

Ввести список городов через компонент `ComboBox1` и сохранить его в массив строк. Переписать в другой список `listBox1` все города, заканчивающиеся на «град».



```

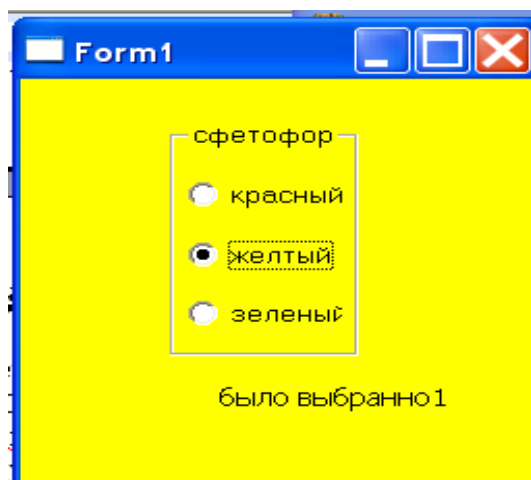
public partial class Form1 : Form
{
    string[] a = new string[10]; выделяем память для глобального массива строк
    int k = 0; - счетчик городов
    public Form1()
    {
        InitializeComponent();
    }

    private void button1_Click(object sender, EventArgs e)
    {
        comboBox1.Items.Add(comboBox1.Text); добавляем элемент в список из
        введенного поля
        a[k] = comboBox1.Text; - сохраняем в массив
        k++; - подсчитываем количество городов
    }
    private void button2_Click(object sender, EventArgs e)
    {
        int lk; string t;
        for (int i = 0; i < k; i++)
        {
            lk = a[i].Length; - длина слова
            t = a[i].Substring(lk - 4, 4); -выделяем подстроку
            //if (a[i][lk-1] == 'k')
            if( t=="град") если подстрока совпадает с град
            {
                listBox1.Items.Add(a[i]); -то добавляем в другой список
            }
        }
    }
}

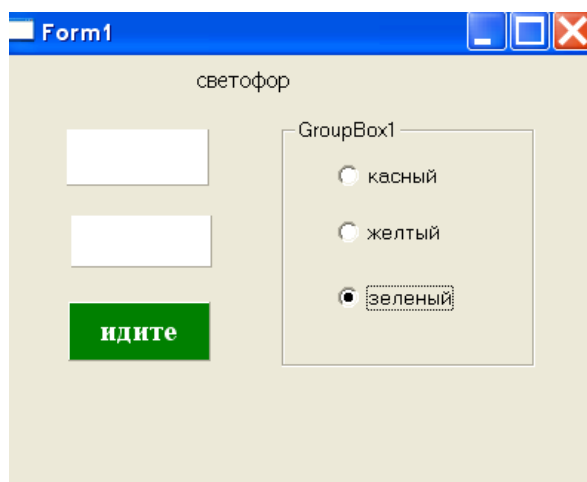
```

Варианты заданий для лабораторной работы № 3

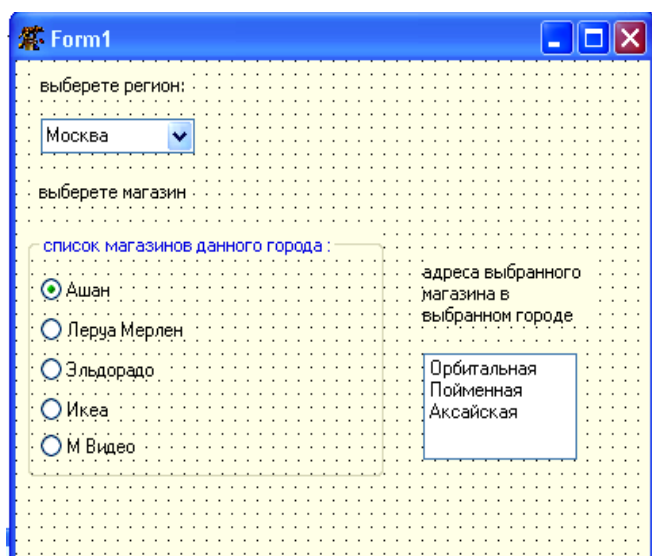
Тема: «Использование компонентов RadioButton, ListBox, ComboBox»



Задание № 1 для вариантов 1, 4, 7, 10, 13, 16: Создать приложение СВЕТОФОР в котором цвет формы меняется по выбору радио кнопки с названием кнопки. (для обращения к цвету форму использовать: `this.BackColor=....`)



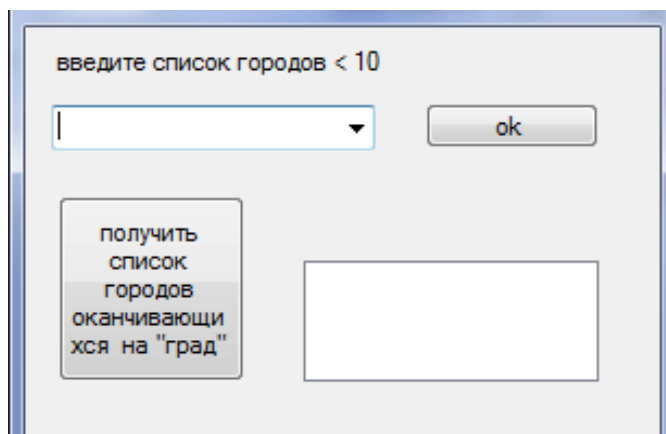
Задание № 1 для вариантов 2, 5, 8, 11, 14: Создать приложение СВЕТОФОР в котором цвет компонентов Panel1, Panel2, Panel3 меняется по выбору радиокнопки с названием кнопки.



Задание № 1 для вариантов № 3, 6, 9, 12, 15. «Сеть магазинов» Расположить компоненты ComboBox, Radiobutton, GroupBox, ListBox, как показано на рисунке. Пользователь выбирает регион (3–4 города в списке ComboBox,). После выбора города становится видимой и доступной та группа радиокнопок со списком магазинов, которая соответствует выбранному городу (количество магазинов в каждом городе 3–4). Для этого создать 3–4 невидимых GroupBox с ра-

диокнопками (Visible). Далее в появившейся радио группе пользователь выбирает магазин и становится виден список адресов данного магазина в данном городе.

Задание № 2. Тема: «Массив текстовых строк и компонент **ComboBox**»



Во всех вариантах Ввод массива строк оформить через компонент `ComboBox1`. Измененный список выводить в другой `ComboBox`. Вывод других результатов вычислений выводить в компонент `Label` или `TextBox`.

- 1 Задан список из десяти городов. (*массив [] string*) Подсчитать количество названий, которые оканчиваются буквой «В».
- 2 Задан список из десяти городов. (*массив [.] string*). Найти порядковые номера городов, в названии которых вторая буква «о».
- 3 Задан список из десяти городов. (*массив [.] string*) Поменять местами название последнего города таблицы и последнего города, начинающегося с буквы «к».
- 4 Задан список из десяти городов. (*массив [.] string*). Найти количество городов, название которых заканчивается сочетанием букв «град».
- 5 Задан список из 10 имен. (*массив [.] string*). Найти порядковый номер имени, в названии которого максимальное число букв.
- 6 Задан список из 20 названий горных вершин. (*массив [.] string*). Найти порядковый номер вершины, в названии которой максимальное число букв.
- 7 Задан список из 20 фамилий. (*массив [.] string*). Присвоить переменной *st* самую длинную фамилию
- 8 Задан список из 20 фамилий. (*массив [.] string*). Найти порядковый номер фамилии, в которой минимальное число букв.
- 9 Задан список из 20 фамилий. (*массив [.] string*). Найти количество букв в самой длинной фамилии.
- 10 Задан список из 20 фамилий. (*массив [.] string*). Поменять местами названия самого длинного и самого короткого слова.
- 11 Задан список из 20 названий горных вершин. (*массив [.] string*). Найти количество букв в самом коротком названии.
- 12 Задан список из десяти городов. (*массив [.] string*)). Найти порядковые номера городов, начинающихся с буквы «Н».
- 13 Задан список из десяти городов. (*массив [.] string*) Подсчитать количество названий, которые начинаются на букву «А».
- 14 Задан список из десяти городов. (*массив [.] string*) Найти порядковые номера городов, которые оканчиваются буквой «к».

- 15 Задан список из десяти городов (*массив [.] string*). Поменять местами названия любых двух городов, заканчивающихся буквой «а».
- 16 Задан список из десяти городов (*массив [.] string*). Присвоить переменной *g* название города с максимальным числом букв.
- 17 Задан список из десяти городов (*массив [.] string*). Поменять местами названия самого длинного и самого короткого слова.
- 18 Задан список из 10 имен (*массив [.] string*). Найти количество букв в самом длинном имени.
- 19 Задан список из 20 названий горных вершин (*массив [.] string*). Присвоить переменной *st* самое короткое название
- 20 Задан список из десяти городов (*массив [.] string*). Поменять местами названия любых двух городов, начинающихся с буквы «а».
- 21 Задан список из десяти городов (*массив [.] string*). Поменять местами название первого города таблицы и последнего города, начинающегося с буквы «К».
- 22 Задан список из десяти городов (*массив [.] string*). Подсчитать количество названий городов, которое содержит более 4 букв.
- 23 Задан список из десяти городов (*массив [.] string*). Найти порядковые номера городов, в названии которых по 7 букв.
- 24 Задан список из десяти городов (*массив [.] string*). Присвоить переменной *st* название последнего из городов, которое содержит более 4 букв.
- 25 Задан список из десяти городов (*массив [.] string*). Поменять местами названия первого города и любого другого, которое содержит более семи букв.
- 26 Задан список из десяти городов (*массив [.] string*). Найти все порядковые номера городов, название которых заканчивается сочетанием букв «бург».
- 27 Задан список из десяти городов (*массив [.] string*). Поменять местами названия двух городов, названия которых оканчиваются сочетанием букв «ск».
- 28 Задан список из 5 имен девочек (*массив [.] string*). Присвоить переменной *d* имя с наименьшим числом букв.
- 29 Задан список из десяти городов (*массив [.] string*). Найти порядковый номер города, в названии которого минимальное число букв.
- 30 Задан список из десяти городов (*массив [.] string*). Найти порядковый номер города, в названии которого максимальное число букв.
- 31 Задан список из 20 названий горных вершин (*массив [.] string*). Поменять местами названия самого длинного и самого короткого слова.

Задание № 3. Тема: «Сортировка массива строк и сжатие массива»

Исходный массив вводится в `ComboBox1`. Переписать сжатый массив во второй компонент `ComboBox` или `ListBox2` после нажатия 1-й кнопки. Далее с помощью радиокнопок пользователь выбирает способ сортировки – по возрастанию или по убыванию. Результат сортировки выводится в `ListBox3`, который должен быть не видим до выбора сортировки.

1 Дан список фамилий сотрудников. Переписать в другой список только фамилии, чья длина больше 7 букв. Затем упорядочить по алфавиту второй список.

2 Дан список фамилий сотрудников. Переписать в другой список только те фамилии, которые заканчиваются на «а». Затем упорядочить по алфавиту второй список.

3 Задан список из десяти городов (*massiv [.] string*). Переписать в другой список только те города, в названии которых есть по две буквы «а». Затем упорядочить по алфавиту второй список.

4 Задан список из десяти городов (*massiv [.] string*). Переписать в другой список только те города, в названии которых есть по две буквы «с». Затем упорядочить по алфавиту второй список.

5 Задан список из десяти городов. (*massiv [.] string*). Переписать в другой список только те города, в которых есть ровно по 3 буквы «о». Затем упорядочить по алфавиту второй список.

6 Задан список из десяти городов. (*massiv [.] string*). Переписать в другой список только те города, в названии которых нет буквы «р». Затем упорядочить по алфавиту второй список.

7 Задан список из 10 имен девочек. (*massiv [.] string*). Переписать в другой список только те имена, в которых есть хотя бы 1 буква «р». Затем упорядочить по алфавиту второй список.

8 Дан список фамилий сотрудников. Переписать в другой список только те фамилии, которые заканчиваются на «в». Затем упорядочить по алфавиту второй список методом «пузырька».

9 Дан список фамилий сотрудников. Переписать в другой список только те фамилии, в которых вторая буква «л». Затем упорядочить по алфавиту второй список методом «пузырька».

10 Дан список фамилий сотрудников. Переписать в другой список только те фамилии, которые оканчиваются на «ов». Затем упорядочить по алфавиту второй список методом «пузырька».

11 Дан список из 10 городов. Переписать в другой список только те города, в которых третья буква «к». Затем упорядочить по алфавиту второй список методом «пузырька».

12 Дан список из 10 городов. Переписать в другой список только те города, которые заканчиваются на «в». Затем упорядочить по алфавиту второй список.

13 Задан список из 20 названий горных вершин (*massiv [.] string*). Переписать в другой список только те горы, в названии которых есть более одной буквы «н». Затем упорядочить по алфавиту второй список.

14 Задан список из десяти названий горных вершин. (*massiv [.] string*). Переписать в другой список только те горы, в названии которых есть хотя бы одна буква «т». Затем упорядочить по алфавиту второй список.

15 Задан список из десяти названий горных вершин. (*massiv [.] string*). Переписать в другой список только те горы, в названии которых нет буквы «р». Затем упорядочить по алфавиту второй список.

16 Задан список из 10 имен названий горных вершин. (*massiv [.] string*). Переписать в другой список только те горы, в которых есть хотя бы 1 буква «р». Затем упорядочить по алфавиту второй список.

17 Задан список из десяти названий горных вершин. (*massiv [.] string*). Переписать в другой список только те горы, в названии которых, есть не менее 3 букв «а». Затем упорядочить по алфавиту второй список.

18 Дан список из 10 городов. Переписать в другой список только те города, чье название длиннее 4 букв. Затем упорядочить по алфавиту второй список.

19 Задан список из 10 имен девочек (*massiv [.] string*). Переписать в другой список только те имена, в которых есть ровно 1 буква «а». Затем упорядочить по алфавиту второй список.

20 Задан список из 10 имен девочек (*massiv [.] string*). Переписать в другой список только те имена, в названии которых есть по две буквы «а». Затем упорядочить по алфавиту второй список.

21 Задан список из десяти городов (*massiv [.] string*). Переписать в другой список только те города, в названии которых есть хотя бы одна буква «т». Затем упорядочить по алфавиту второй список.

22 Задан список из 10 имен девочек (*massiv [.] string*). Переписать в другой список только те имена, в названии которых есть не менее двух букв «Н». Затем упорядочить по алфавиту второй список.

23 Задан список из десяти городов (*massiv [.] string*). Переписать в другой список только те города, в названии которых есть не менее 3 букв «о». Затем упорядочить по алфавиту второй список.

24 Задан список из десяти городов (*massiv [.] string*). Переписать в другой список только те города, в названии которых есть более одной буквы «н» Затем упорядочить по алфавиту второй список.

25 Задан список из 20 названий горных вершин (*massiv [.] string*). Переписать в другой список только те горы, в названии которых есть более одной буквы «н» Затем упорядочить по алфавиту второй список.

26 Задан список из десяти названий горных вершин. (*massiv [.] string*). Переписать в другой список только те горы, в названии которых есть хотя бы одна буква «т» Затем упорядочить по алфавиту второй список.

27 Задан список из десяти названий горных вершин. (*massiv [.] string*). Переписать в другой список только те горы, в названии которых нет буквы «р». Затем упорядочить по алфавиту второй список.

28 Задан список из 10 имен названий горных вершин. (*massiv [.] string*). Переписать в другой список только те горы, в названии которых есть хотя бы 1 буква «р». Затем упорядочить по алфавиту второй список.

29 Задан список из десяти названий горных вершин. (*massiv [.] string*). Переписать в другой список только те горы, в названии которых есть не менее 3 букв «а». Затем упорядочить по алфавиту второй список.

Лабораторная работа № 4

ОБРАБОТКА СОБЫТИЙ ДЛЯ ФОРМАТИРОВАНИЯ ТЕКСТА В МНОГОСТРОЧНОМ ПОЛЕ ВВОДА

ФУНКЦИИ ОБРАБОТКИ СТРОК STRING В C

1) Свойство **Length**. Возвращает длину строки. Пример:

```
s="qqq";
int k=s.Length; // В k запишется 3
```

2) Метод **Substring**. Позволяет извлечь из строки подстроку (скопировать). Параметры: первый – с какого места извлекаем (нумерация с нуля) и второй – сколько символов извлекаем. Пример:

```
String s1="abcdefg", s2;
s2=s1.Substring(3, 2); // В s2 запишется "de"
```

3) Метод **Insert**. Вставляет в строку другую строку. Первый параметр – это куда вставляем (нумерация, как всегда, с нуля), второй – что за строчку вставляем. Пример:

```
String s1="abcdefg", s2;
s2=s1.Insert(1, "xyz"); // в (s2); будет "axyzbcdefg"
```

4) Метод **IndexOf**. Позволяет найти в строке подстроку. Этот метод возвращает номер позиции, на котором в строке находится передаваемая в качестве параметра подстрока. Если такой подстроки нет, то возвращается -1. Пример:

```
String s1="abcabcab", s2="bc", s3="zzz";
Int k=s1.IndexOf(s2); //будет =1
Int p=s1.IndexOf(s3); //будет = -1
```

5) Метод **Remove**. Удаляет часть строки. *Первый параметр* – индекс в строке, начиная с которого надо удалить символы. Второй параметр – сколько символов надо удалить.

```
text = text.Remove(0, 2);
```

6) **Compare**. Метод, сравнивающий две строки. Возвращает 0, если строки равны, отрицательное значение, если первая строка меньше второй и положительное значение, если первая строки больше второй (больше и меньше в алфавитном смысле, разумеется). Пример:

```
String s1="arbour", s2="ace", s3="azote";
(String.Compare(s1, s1)); //Выдаст 0, т. к. "arbour" равно "arbour".
```

7) **Equals**. Метод, возвращает true, если строки равны, false – если не равны. Пример:

```
String s1="qqq", s2="www";
(String.Equals(s1, s2).ToString());
```

ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ ФУНКЦИЙ ДЛЯ ОБРАБОТКИ СТРОК

Кнопка **button1** позволяет заменить в тексте из **textBox1** все слова “**Sum**” на 4 звездочки “****”. Приводим текст для кнопки:

```
private void button1_Click(object sender, EventArgs e)
{
    string t = textBox1.Text;
    int k=t.IndexOf("sum");- находим где есть слово sum
    while (k > 0)
    {
        t = t.Remove(k, 3);- удаляем слово sum
        t = t.Insert(k, "****"); – вставляем слово ****
        k = t.IndexOf("sum"); – находим где еще есть слово sum
    }
    textBox3.Text = t;
}
```

Кнопка **button2** позволяет вычислить количество слов “**Sum1**” в тексте из **textBox1** Приводим текст для кнопки:

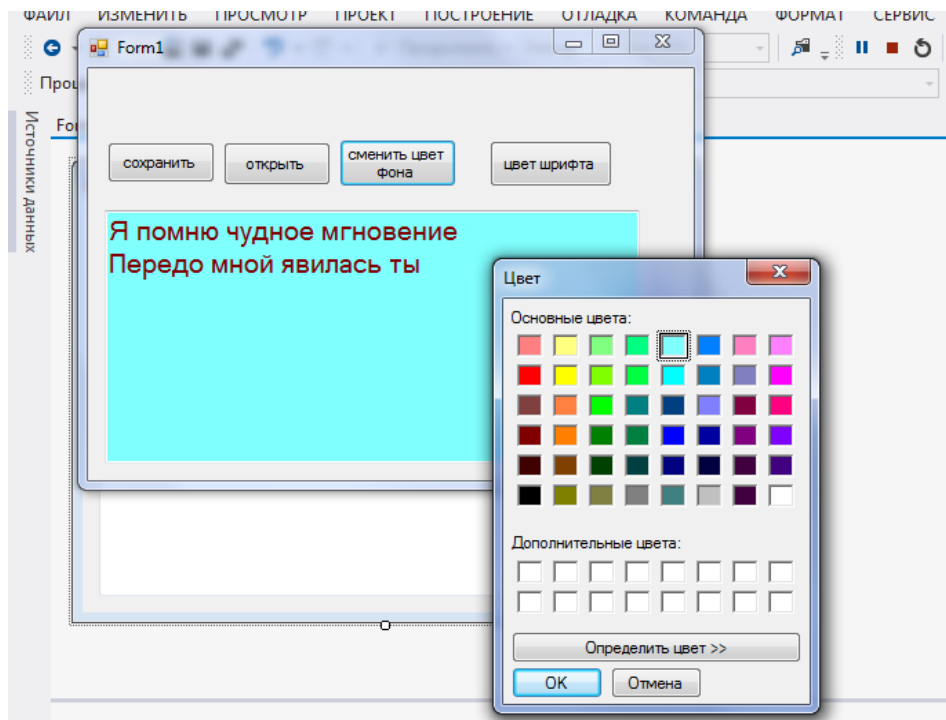
```
private void button2_Click(object sender, EventArgs e)
{
    int k = 0;
    string t2, t = textBox1.Text;

    for (int i = 0; i < t.Length-4; i++) – в цикле до конца строки-4
    {
        t2 = t.Substring(i, 4); – выделяем подстроку из 4 символов
        if (t2 == "sum1") k++; – если выделенные символы совпадают с иско-
        МЫМ СЛОВОМ , то
    }
    textBox3.Text = k.ToString();
}
}
```

ОБЪЯВЛЕНИЯ ОДНОМЕРНОГО МАССИВА

```
Int [] a=new int[10];
String[] a =new string[10];
```

Работа с окнами диалога. Работа с внешними текстовыми файлами.
Пример простейшего текстового редактора



В среде MS Visual C есть несколько невизуальных компонентов, создающих окна диалога:

- **colorDialog** – окно диалога для выбора **цвета** / основное свойство **Color** – выбранный цвет
- **fontDialog** – окно диалога для выбора параметров **текста** / основное свойство **Font**
- **saveFileDialog** – окно диалога для выбора пути **сохраняемого файла** / основное свойство **FileName** – путь и имя выбранного файла
- **openFileDialog** – окно диалога для выбора пути **открываемого файла** / основное свойство тоже **FileName** – путь и имя выбранного файла

У всех четырех объектов есть метод **ShowDialog()**, который открывает окно диалога и возвращает код нажатой пользователем кнопки (ОК) или (Отмена – Cansel). В следующем операторе if происходит открытие окна и проверка какая кнопка нажата (ОК):

```
if (colorDialog1.ShowDialog() == DialogResult.OK)
```

```
private void button3_Click(object sender, EventArgs e) –сменить цвет фона
{ // открытие окна и проверка нажата ли кнопка ОК:
```

```

if (colorDialog1.ShowDialog() == DialogResult.OK)
    {// установить цвет фона TextBox таким как в окне диалога:
    textBox1.BackColor = colorDialog1.Color;
    }
}

```

```

private void button4_Click(object sender, EventArgs e)
    {//открытие окна и проверка нажата ли кнопка ОК:
    if (colorDialog2.ShowDialog() == DialogResult.OK)
        // установить цвет текста таким как в окне диалога:
        textBox1.ForeColor = colorDialog2.Color;
    }
}

```

ОСОБЕННОСТИ РАБОТЫ С ВНЕШНИМИ ТЕКСТОВЫМИ ФАЙЛАМИ В C#

1) Для работы с файловыми потоками необходимо подключить библиотеку using System.IO; в начале файла.

2) Файловый поток для записи создается с помощью оператора:

```

StreamWriter имя_файлового_потока = new
StreamWriter("имя_текстового_файла");

```

3) Файловый поток для чтения создается с помощью оператора:

```

StreamReader имя_файлового_потока = new
StreamReader("имя_текстового_файла");

```

4) Закрыть поток

```

имя_файлового_потока .Close();

```

5) Записать в файловый поток

```

имя_файлового_потока . WriteLine(записываемый_текст);

```

6) Считать из файлового потока все до конца файла

```

переменная_string = имя_файлового_потока . ReadToEnd();

```

7) Считать из файлового потока до конца строки

```

переменная_string = имя_файлового_потока . ReadLine();

```

8) Определить конец файла

```

while(! fr.EndOfStream)

```

```

using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;

```

```

namespace tekst1
{
    public partial class Form1 : Form
    {
        public Form1()
        {

```

```

InitializeComponent();
}

private void button1_Click(object sender, EventArgs e) // сохранить
{
    if (saveFileDialog1.ShowDialog() == DialogResult.OK) // открываем окно
        диалога
        {
            StreamWriter fw = new StreamWriter(saveFileDialog1.FileName); // со-
            здали файловый поток и связали его с внешним текстовым файлом имя кото-
            рого выберет пользователь в окне saveFile
            fw.WriteLine(textBox1.Text); // записать текст в файл
            fw.Close(); // закрыть файловый поток
        }
}
private void button2_Click(object sender, EventArgs e) // открыть
{
    if (openFileDialog1.ShowDialog() == DialogResult.OK) // открываем окно
        диалога
        {
            StreamReader fr = new StreamReader(openFileDialog1.FileName); // со-
            здать файловый поток и связать его с файлом из переменной FileName
            textBox1.Text = fr.ReadToEnd(); // читать все до конца
            // for(int i=1;i<5;i++) textBox1.Text += fr.ReadLine(); читать по стро-
            кам
            //или в цикле пока не конец файла while(! fr.EndOfStream) { textBox1.Text +=
            fr.ReadLine(); }

            fr.Close();
        }
}
}

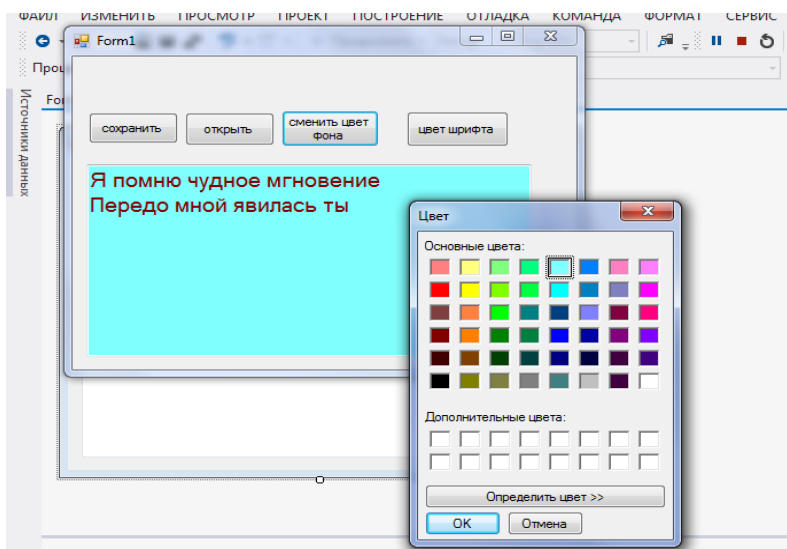
```

Варианты заданий для лабораторной работы № 4

Тема: «Обработка текста в многострочном TextBox»

Задание № 1. Создать простейший текстовый редактор с использованием окон диалога **colorDialog fontDialog saveFileDialog openFileDialog**

Задание № 2. Во всех вариантах в созданный текстовый редактор добавить кнопку для выполнения индивидуального задания.



1 Дан текст в компоненте TextBox. Вычислить количество предложений в нем (не использовать substring, insert, indexof...).

2 Дан текст в компоненте TextBox. Определить, в каких позициях в нем встречаются символы «;» (не использовать substring, insert, indexof...) (другими словами: Вывести на экран номера позиций в которых встречается символ «;»).

3 Дан текст в компоненте TextBox. Вывести на экран номера позиций в которых встречается цифры (не использовать substring, insert, indexof...).

4 Дан текст в компоненте TextBox. Определить, в каких позициях в нем начинается каждое новое предложение (сначала найти позиции точек) (не использовать substring, insert, indexof...).

5 Дан текст в компоненте TextBox. Сколько раз в нем встречается символ «=» (не использовать substring, insert, indexof...).

6 Дан текст в компоненте TextBox. Вывести на экран номера позиций в которых встречается русские буквы (не использовать substring, insert, indexof...).

7 Дан текст в компоненте TextBox. Определить, в каких позициях в нем начинается каждое новое слово (сначала найти позиции пробелов) (не использовать substring, insert, indexof...).

8 Дан текст в компоненте TextBox. Сколько раз в нем встречается латинские буквы (не использовать substring, insert, indexof...).

9 Дан текст в компоненте TextBox. **Insert.** Переставить в нем первую букву первого слова и первую букву второго слова (Сначала найти номер первого пробела). (не использовать substring, indexof...).

10 Дан текст в компоненте TextBox. Сколько раз в нем встречается символ «@» (не использовать substring, insert, indexof...)

11 Дан текст в компоненте TextBox. Вывести на экран номера позиций в которых встречается символ «@» (не использовать substring, insert, indexof...)

12 Дан текст в компоненте TextBox. Сколько раз в нем встречается цифры (не использовать substring, insert, indexof...).

13 Дан текст в компоненте TextBox. Вычислить количество слов в нем (не использовать substring, insert, indexof...).

14 Дан текст в компоненте `TextBox`. Сколько раз в нем встречается русские буквы (не использовать `substring`, `insert`, `indexOf`...).

15 Дан текст в компоненте `TextBox`. Вывести на экран номера позиций в которых встречается латинские буквы (не использовать `substring`, `insert`, `indexOf`...).

16 Пользователь вводит текст на русском языке. Вывести исходный текст, заменив в нем квадратные скобки на круглые. Вычислить количество слов, а также количество предложений.

17 Пользователь вводит текст на русском языке. Вывести исходный текст, заменив в нем квадратные скобки на круглые. Вычислить количество слов, а также количество предложений. А также удалить символы `@`.

18 Пользователь вводит текст на русском языке. Вывести исходный текст, заменив в нем квадратные скобки на круглые. Вычислить количество слов, а также количество предложений. А также удалить круглые скобки.

19 Дан текст в компоненте `TextBox1`. В другой компонент `TextBox2` переписать текст, включив в него символы исходного, кроме символов пробелов, точек и запятых (использовать склейку `«+»`) (не использовать `substring`, `insert`, `indexOf`...).

20 Дан текст в компоненте `TextBox1`. В другой компонент `TextBox2` переписать текст, включив в него символы исходного, кроме символов пробелов, точек и запятых (использовать склейку `«+»`) (не использовать `substring`, `insert`, `indexOf`...).

21 Дан текст в компоненте `TextBox1`. В другой компонент `TextBox2` переписать текст, включив в него символы данного, расположенные на четных позициях. (использовать `repeat`) (не использовать `substring`, `insert`, `indexOf`...).

22 Дан текст в компоненте `TextBox1`. В другой компонент `TextBox2` переписать только буквы латинского алфавита и пробелы (не использовать `substring`, `insert`, `indexOf`...).

23 Дан текст в компоненте `TextBox1`. В другой компонент `TextBox2` переписать только цифры и символы арифметических операций (не использовать `substring`, `insert`, `indexOf`...).

24 Дан текст в компоненте `TextBox1`. В другой компонент `TextBox2` включить символы данной последовательности, расположенные на нечетных позициях (не использовать `substring`, `insert`, `indexOf`...).

25 Дан текст в компоненте `TextBox1`. В другой компонент `TextBox2` образовать новую последовательность, удвоив каждый символ `«=»` и пропустив пробелы (не использовать `substring`, `indexOf`...).

26 Дан текст в компоненте `TextBox1`: 4 слова в 4 разных строках. В другой компонент `TextBox2` образовать новую последовательность символов, состоящую из вторых букв каждого слова (использовать склейку `«+»`) (не использовать `substring`, `insert`, `indexOf`...).

27 Дан текст в компоненте `TextBox1`: 3 слова – ваши имя, отчество, фамилия в 3 разных строках. В другой компонент `TextBox2` образовать новую символьную переменную, хранящую только ваши инициалы (использовать склейку `«+»`) (не использовать `substring`, `insert`, `indexOf`...).

28 Дан текст в компоненте `TextBox1`: 3 слова в 3 разных строках. В другой компонент `TextBox2` образовать новую последовательность символов, состоящую из последних букв каждого слова (использовать склейку «+») (не использовать `substring`, `insert`, `indexOf`...).

29 Дан текст в компоненте `TextBox1` 3 слова в 3 разных строчках. В другой компонент `TextBox2` записать последовательность символов, состоящую из первых букв каждого слова (использовать склейку «+») (не использовать `substring`, `insert`, `indexOf`...).

30 Дан текст в компоненте `TextBox1` 3 слова в 3 разных строчках. В другой компонент `TextBox2` записать последовательность символов, состоящую из первых букв каждого слова (использовать склейку «+») (не использовать `substring`, `insert`, `indexOf`...).

31 Дан текст в компоненте `TextBox1` 3 слова – ваши Имя, Отчество, Фамилия в 3 разных строках. В другой компонент `TextBox2` переписать текст, хранящую полностью «имя отчество фамилию» (использовать склейку «+») (не использовать `substring`, `insert`, `indexOf`...).

32 Дан текст в компоненте `TextBox1` из двух восьмибуквенных слов. В другой компонент `TextBox2` образовать последовательность букв, в которой должны чередоваться буквы первого и второго слова (не использовать `substring`, `insert`, `indexOf`...)

33 Дан текст в компоненте `TextBox1` 3 слова. В другой компонент `TextBox2` переписать новый текст, в котором должны чередоваться буквы первого, второго и третьего слова (не использовать `substring`, `insert`, `indexOf`...).

34 Дан текст в компоненте `TextBox1`: 3 строки – фамилия, имя и отчество учащегося. В другой компонент `TextBox2` оставить только фамилию и инициалы.

Задание № 3. Тема: «Использование функции `substring`, `insert`, `indexOf`».

Для всех вариантов в созданный выше текстовый редактор добавить кнопку для выполнения индивидуального задания

1 Дан текст в компоненте `TextBox1`. Вычислить количество слов «Компьютер» или «компьютер», а также количество предложений.

2 Дан текст в компоненте `TextBox1`. Вычислить количество слов «ПК». Удалить символы #/.

3 Дан текст в компоненте `TextBox1`. Вывести исходный текст, заменив в нем квадратные скобки на круглые. Вычислить количество появления слова «обучаемый».

Дан текст в компоненте `TextBox1`. Вычислить количество слов «Pascal». Заменить все скобки на «*». Заменить все латинские прописные буквы на соответствующие заглавные.

4 Дан текст в компоненте `TextBox1`. Вычислить количество слов «дублирование», а также удалить фигурные скобки.

5 Дан текст в компоненте TextBox1. Вычислить количество слов «проблема», а также заменить все цифры на «*».

6 Дан текст в компоненте TextBox1. Вычислить количество слов, начинающихся на «м». Количество слов «Компьютер» или «компьютер», а также количество предложений.

7 Дан текст в компоненте TextBox1. Вычислить количество слов «война», а также заменить все латинские буквы на «*».

8 Дан текст в компоненте TextBox1. Вычислить количество слов «Pascal», удалить символы «*».

9 Дан текст в компоненте TextBox1. Вычислить количество слов «плохо», а также заменить все русские заглавные буквы на «*».

10 Дан текст в компоненте TextBox1. Вычислить количество слов, начинающихся на «м». Количество слов «мало» или «Мало», а также количество предложений.

11 Дан текст в компоненте TextBox1. Вычислить количество слов «кризис». Заменить все русские заглавные буквы на соответствующие прописные.

12 Дан текст в компоненте TextBox1. Вычислить количество слов «удовлетворительно». Заменить все цифры на 0.

13 Дан текст в компоненте TextBox1. Вычислить количество слов «хорошо». Вычислить количество всех слов. Заменить все латинские заглавные буквы на соответствующие прописные.

Задание № 4. Тема: «Исправление опечаток в тексте».

Для всех вариантах в созданный текстовый редактор добавить кнопку для выполнения индивидуального задания

1 Исходный текст набран с ошибками: иногда отсутствуют пробелы после точек. Вставить 1 пробел после каждой точки, если он отсутствует перед следующим предложением. А также вычислить количество слов и предложений.

2 Исходный текст набран с ошибками: Вывести исходный текст, заменив в нем строчные буквы, следующие за точкой и произвольным количеством пробелов на прописные буквы. А также вычислить количество слов и предложений.

3 Исходный текст набран с ошибками: иногда отсутствуют пробелы после запятых. Вставить 1 пробел после каждой запятой, если он отсутствует перед следующим словом. А также вычислить количество слов «Информатика».

4 В тексте заменить цифры на их словесные названия (использовать Case)

5 Исходный текст набран с ошибками: после слова может находиться один или более пробелов перед точкой (или нет). Вывести исходный текст, убрав в нем эти пробелы перед точкой (между словом и точкой). А также удалить символы_ «#».

6 Исходный текст набран с ошибками: иногда отсутствуют пробелы после точек. Вставить 1 пробел после каждой точки, если он отсутствует перед

следующим предложением. а также вычислить количество предложений. А также удалить квадратные скобки.

7 Исходный текст набран с ошибками. Вывести исходный текст, заменив в нем строчные буквы, следующие за точкой и одним пробелом, на прописные буквы. Вычислить количество слов, начинающихся на букву «А».

8 Пользователь вводит текст на русском языке. Вывести исходный текст, заменив в нем все заглавные (прописные) буквы на строчные. Вычислить количество слов, а также количество предложений. А также удалить символы_ «@».

9 Исходный текст набран с ошибками: Выражения, заключенные в скобки. имеют один пробел в начале и в конце. Вывести исходный текст, убрав в нем пробелы после открывающейся скобки, а также перед закрывающейся скобкой

10 Исходный текст набран с ошибками. Вывести исходный текст, заменив в нем строчные буквы, следующие за точкой и произвольным количеством пробелов на прописные буквы. А также вычислить количество слов «ПК».

11 Исходный текст набран с ошибками: после слова может находиться один или более пробелов перед запятой (или нет). Вывести исходный текст, убрав в нем пробелы перед запятой (между словом и запятой). А также удалить символы_ «-».

12 Исходный текст набран с ошибками: Выражения, заключенные в скобки, имеют один или более пробелов вначале и в конце (или нет). Вывести исходный текст, убрав в нем пробелы после открывающейся скобки, а также перед закрывающейся скобкой

13 Исходный текст набран с ошибками: после слова может находиться один пробел перед запятой или нет. убрать в тексте эти пробелы перед запятой (между словом и запятой). А также удалить символы_ «\$».

14 Исходный текст набран с ошибками: некоторые слова по ошибке начинаются не с одной первой заглавной буквы, а с двух заглавных букв. Исправить текст.

Лабораторная работа № 5

ОБРАБОТКА СТРУКТУРНОЙ ИНФОРМАЦИИ В ТАБЛИЦЕ DATAGRIDVIEW

Компонент **DataGridView** имеет следующие свойства:

- 1) Свойство **Columns** из окна «свойств» содержит коллекцию столбцов таблицы.

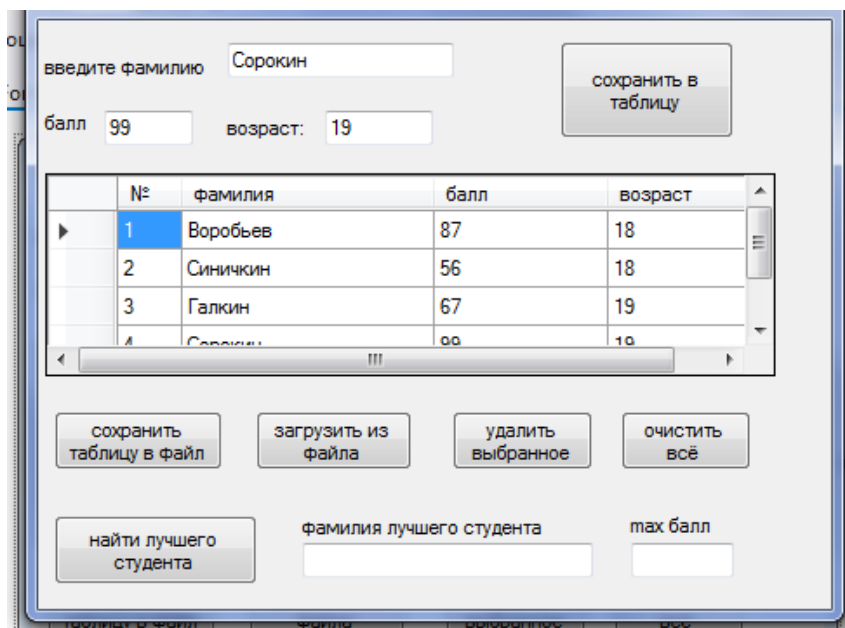
На этапе конструирования формы настройте свойства таблицы, Нажав на троеточие напротив свойство **Columns** войдите в диалоговое окно для добавления столбцов и их названий. Нажмите кнопку **Добавить**, введите название столбца (измените ширину столбца при необходимости – **Width**), и повторите столько сколько столбцов. (*Row – строка. Column – столбец*)

В программе доступны следующие методы:

- 2) `dataGridView1.Rows.Clear();` – очистить всю таблицу

- 3) `dataGridView1.Rows.Add()`; – добавить строку в таблицу
- 4) `dataGridView1.RowCount` – количество строк в таблице
- 5) `dataGridView1.ColumnCount` – количество столбцов в таблице
- 6) `dataGridView1.Rows[i].Cells[j].Value=` – содержимое ячейки из *i*-ой строки и *j*-ого столбца
- 7) `int k = dataGridView1.SelectedCells[0].RowIndex`; – позволяет узнать номер выделенной пользователем строки
- 8) `dataGridView1.Rows.RemoveAt(k)`; – удалить строку с номером *k*

Создадим форму, как указано на рисунке:



В окне кода объявим комбинированный тип

```
struct T_student
```

```
{
    public string fio;
    public int test;
    public int age;
};
```

В окне кода объявим глобальный комбинированный массив :

```
T_student[] stud = new T_student[100];
```

Объявим глобальную переменную *N* – количество записей в таблице и в массиве **stud**

```
int N=0;
```

Приведем полный текст программы:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
```

```
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO; – для работы с файлами
```

```
namespace таблица
```

```
{
    public partial class Form1 : Form
    {int N=0;
    struct T_student
    {
        public string fio;
        public int test;
        public int age;
    };
    T_student[] stud = new T_student[100];
```

```
    public Form1()
    {
        InitializeComponent();
    }
```

```
private void button1_Click(object sender, EventArgs e) –сохранить в
```

таблицу

```
{
    dataGridView1.Rows.Add();
    dataGridView1.Rows[N].Cells[0].Value = (N + 1).ToString();
    dataGridView1.Rows[N].Cells[1].Value = textBox1.Text;
    dataGridView1.Rows[N].Cells[2].Value = textBox2.Text;
    dataGridView1.Rows[N].Cells[3].Value = textBox3.Text;
    stud[N].fio = textBox1.Text;
    stud[N].test = Convert.ToInt16(textBox2.Text);
    stud[N].age = Convert.ToInt16(textBox3.Text);
    N++;
}
```

```
private void button2_Click(object sender, EventArgs e) //сохранить
```

таблицу в файл

```
{
    if (saveFileDialog1.ShowDialog() == DialogResult.OK) //
    {
        StreamWriter fw = new StreamWriter(saveFileDialog1.FileName);
```

```

for (int i = 0; i < dataGridView1.RowCount-1; i++)
    {
        fw.WriteLine(dataGridView1.Rows[i].Cells[1].Value);
        fw.WriteLine(dataGridView1.Rows[i].Cells[2].Value);
        fw.WriteLine(dataGridView1.Rows[i].Cells[3].Value);
        //или ещё вложенный цикл :
        //for (int j=1; j<dataGridView1.ColumnCount;j++)
        // fw.WriteLine(dataGridView1.Rows[i].Cells[j].Value);
    }
    fw.Close();
}
}

private void button3_Click(object sender, EventArgs e)- загрузить из
файла
{
    if (openFileDialog1.ShowDialog() == DialogResult.OK)
    {
        StreamReader fr = new StreamReader(openFileDialog1.FileName);
//
        dataGridView1.Rows.Clear();
        int i = 0;
        while(! fr.EndOfStream)
        {
            dataGridView1.Rows.Add();
            dataGridView1.Rows[i].Cells[0].Value = (i + 1).ToString();
            for (int j = 1; j < dataGridView1.ColumnCount; j++)
            {
                dataGridView1.Rows[i].Cells[j].Value = fr.ReadLine();
            }

            stud[i].fio = (dataGridView1.Rows[i].Cells[1].Value).ToString();
            stud[i].test = Con-
vert.ToInt16(dataGridView1.Rows[i].Cells[2].Value);
            stud[i].age = Con-
vert.ToInt16(dataGridView1.Rows[i].Cells[3].Value);
            i++;
        }- конец цикла while
        fr.Close();
        N=i;
    }
}

private void button5_Click(object sender, EventArgs e)

```

```

{
    dataGridView1.Rows.Clear();
    N = 0;
}

private void button4_Click(object sender, EventArgs e)
{
    int k = dataGridView1.SelectedCells[0].RowIndex;
    dataGridView1.Rows.RemoveAt(k);
    N=N-1;
}

private void button6_Click(object sender, EventArgs e)
{
    int max = 0;
    string Famil = " ";
    for (int i = 0; i < N; i++)
    {
        if (stud[i].test > max)
        {
            max = stud[i].test;
            Famil = stud[i].fio;
        }
    }
    textBox4.Text = Famil;
    textBox5.Text = max.ToString();
}
}
}
}

```

The screenshot shows a Windows application window with the following elements:

- Input fields: "введите фамилию" (with "Сорокин" entered), "балл" (with "99" entered), and "возраст:" (with "19" entered).
- Buttons: "сохранить в таблицу", "сохранить таблицу в файл", "загрузить из файла", "удалить выбранное", "очистить всё", and "найти лучшего студента".
- Table: A table with 5 columns: "№", "фамилия", "балл", "возраст". The first row is selected (row 1, name "Воробьев", score "87", age "18").
- Output fields: "фамилия лучшего студента" (empty) and "max балл" (empty).

Рассмотрим решение этой задачи с использованием классов:

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```

```
using System.Windows.Forms;
using System.IO;- не забудьте для работы с файлами
```

```
namespace table_class_file
{
    public partial class Form1 : Form
    {
        int NN = 0;//счетчик строк в таблице
        public class T_student
        {
            public string fio;
            public int test;
            public int age;
        }

        public class T_group
        {
            public int N;// количество элементов в группе
            public T_student[] stu;//агрегация
            public T_group();// конструктор сразу с реализацией
            {
                N = 10;//пусть max количество =10
                stu = new T_student[10];
                for (int i = 0; i < N; i++)
                    stu[i] = new T_student();//выделяем память для массива объек-
тов в конструкторе
            }
            public string poisk_max_test();// метод класса сразу с реализацией
            {
                string fam = " ";
                int max = 0;
                for (int i = 0; i < N; i++)
                {
                    if (stu[i].test > max)
                    {
                        max = stu[i].test;
                        fam = stu[i].fio;
                    }
                }
                return fam;
            }
        }
    }
}
// конец метода poisk_max_test
//конец класса
```


T_group a_32 = new T_group(); //- объявление объекта класса T_group
и выделение для него памяти

```

public Form1()
{
    InitializeComponent();
}

private void button1_Click(object sender, EventArgs e) -сохранить в
таблицу
{
    dataGridView1.Rows.Add();- добавить строку
    dataGridView1.Rows[NN].Cells[0].Value = (NN + 1).ToString();-
номер строки
    dataGridView1.Rows[NN].Cells[1].Value = textBox1.Text;- фамилия
    dataGridView1.Rows[NN].Cells[2].Value = textBox2.Text; -тест
    dataGridView1.Rows[NN].Cells[3].Value = textBox3.Text; -возраст
    a_32.stu[NN].fio = textBox1.Text; -сохранить в массив (класс)
    a_32.stu[NN].test = Convert.ToInt16(textBox2.Text);
    a_32.stu[NN].age = Convert.ToInt16(textBox3.Text);
    NN++;
}

private void button2_Click(object sender, EventArgs e) -сохранить в
файл
{
    if (saveFileDialog1.ShowDialog() == DialogResult.OK)
    {
        StreamWriter fw = new StreamWriter(saveFileDialog1.FileName);
        for (int i = 0; i < dataGridView1.RowCount - 1; i++)
        {
            fw.WriteLine(dataGridView1.Rows[i].Cells[1].Value);
            fw.WriteLine(dataGridView1.Rows[i].Cells[2].Value);
            fw.WriteLine(dataGridView1.Rows[i].Cells[3].Value);
            //или ещё вложенный цикл :
            //for (int j=1; j<dataGridView1.ColumnCount;J++)
            // fw.WriteLine(dataGridView1.Rows[i].Cells[j].Value);
        }
        fw.Close();
    }
}

```

```

private void button3_Click(object sender, EventArgs e) –загрузить из
файла
{
    if (openFileDialog1.ShowDialog() == DialogResult.OK)
    {
        StreamReader fr = new StreamReader(openFileDialog1.FileName); //
        dataGridView1.Rows.Clear(); -очистить всю таблицу
        int i = 0; - счетчик строк
        while (!fr.EndOfStream) пока не конец файла
        {
            dataGridView1.Rows.Add();
            dataGridView1.Rows[i].Cells[0].Value = (i + 1).ToString();
            for (int j = 1; j < dataGridView1.ColumnCount; j++)
            {
                dataGridView1.Rows[i].Cells[j].Value = fr.ReadLine();
            }

            a_32.stu[i].fio = (dataGridView1.Rows[i].Cells[1].Value).ToString();
            a_32.stu[i].test = Con-
vert.ToInt16(dataGridView1.Rows[i].Cells[2].Value);
            a_32.stu[i].age = Con-
vert.ToInt16(dataGridView1.Rows[i].Cells[3].Value);
            i++;
        }
        fr.Close();
        NN = i;
    }
}

private void button5_Click(object sender, EventArgs e) –очистить все
{
    dataGridView1.Rows.Clear();
    NN = 0;
}

private void button4_Click(object sender, EventArgs e) –удалить выде-
ленную строку
{
    int k = dataGridView1.SelectedCells[0].RowIndex;
    dataGridView1.Rows.RemoveAt(k);
    NN = NN - 1;
}

```

```

private void button6_Click(object sender, EventArgs e) – найти лучшего студента
{
    string Fami = a_32.poisk_max_test();
    textBox4.Text = Fami;
}
}

```

ОСОБЕННОСТИ ОБЪЯВЛЕНИЯ КЛАССОВ И МАССИВОВ ОБЪЕКТОВ В C#

1) Класс описывается в самом начале файла для того чтобы это описание было доступно во всех подпрограммах-функциях. А лучше располагать описание класса в отдельном файле специально для этого предназначенном (проект->добавить класс-> Class1.cs)

```

....
using System.Windows.Forms;
namespace klass
{
    public partial class Form1 : Form
    {
        public class T_mount
        {.....

```

2) Перед каждым полем указывать уровень доступа public

```

public class T_mount
{
    public string name;
    public int height;
}

```

3) Объявить глобальную переменную-объект класса (там же, в начале файла)

3.1) `T_mount rock = new T_mount();` – так объявляется единственный объект класса `T_mount` и выделяется память для него

3.2) `T_mount[] rocks = new T_mount[10];` – так объявляется массив из объектов класса `T_mount`

4) В разделе инициализации для каждого элемента массива из объектов необходимо выделять память в цикле

```

public Form1()
{
    InitializeComponent();
    for (int i = 0; i < 10; i++)

```

`rocks[i] = new T_mount();` – для каждого элемента массива из объектов необходимо выделять память в цикле

}

5) Обращение к полям класса:

rock.name = "Эльбрус"; *или* rock.name = textBox1.text;rock.height = 11000; *или* rock.height = Convert.ToInt32(textBox2.text);

Обращение к полям класса в массиве:

Rocks[i].name = "Эльбрус"; *или* rocks[i].name = textBox1.text;Rocks[i].height = 11000; *или* rocks[i].height = Convert.ToInt32(textBox2.text);

Приведем полный текст программы:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace class_my

{

public partial class Form1 : Form

{

public class T_mount

{

public string name;

public int height;

}

T_mount rock = new T_mount(); *//- единственный объект класса**T_mount и выделяется память для него*

public const int max_n=20;

T_mount [] rocks = new T_mount[max_n]; *//- так объявляется массив из объектов класса T_mount*Int k=0; *// глобальная переменная для счета количество гор*

public Form1()

{

InitializeComponent();

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

rocks[i] = new T_mount(); *// – для каждого элемента массива из**объектов необходимо выделять память в цикле*

}

```

private void button1_Click(object sender, EventArgs e) // ввод в массив
всех гор по одной
{
    //rock.name = textBox1.Text;
    // rock.hieght = Convert.ToInt16(textBox2.Text);
    rocks[k].name = textBox1.Text;
    rocks[k]. height = Convert.ToInt16(textBox2.Text);
    k++; – сколько раз нажали кнопку
}
private void button2_Click(object sender, EventArgs e)// поиск названия
высочайшей горы
{int max=0;
String name_max;
For (int i=0;i<k;i++)
{
    If (rocks[i].height>max){ max=rocks[i].height;
                                Name_max=rocks[i].name;}
}
textBox3.text=name_max;
}

```

АГРЕГАЦИЯ ИЛИ ВЛОЖЕННЫЕ КЛАССЫ

6) Для агрегации в другом классе указать массив объектов следующим образом: `public T_student [] stu;` (*вспомните описание массива: `int [] a ;`*)

```

public class T_group
{
    public int N;
    public T_student[] stu;

```

...

7) Объявить глобальную переменную-объект агрегированного класса и выделить для неё память: `T_group a_32 = new T_group();` (см. массив `Int [] a =new int[10];`)

8) В конструкторе для агрегированного класса обязательно выделить память для объектов массива класса `T_student`

```

public T_group() // конструктор
{
    N = 100;// или другое число max допустимое
    stu = new T_student[N];
    for (int i = 0; i < N; i++)
        stu[i] = new T_student();
}

```

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace klass
{
    public partial class Form1 : Form
    {
        public class T_student
        {
            public string fio;
            public int test;
            public int age;
        }

        public class T_group
        {
            public int N;
            public T_student[] stu;
            public T_group()// конструктор
            {
                N = 10;
                stu = new T_student[N];
                for (int i = 0; i < N; i++)
                    stu[i] = new T_student();
            }
            public string poisk_max_test()
            {
                string fam=" ";
                int max = 0;
                for (int i = 0; i < N; i++)
                {
                    if (stu[i].test > max)
                    {
                        max = stu[i].test;
                        fam = stu[i].fio;
                    }
                }
            }
        }
    }
}
```

```

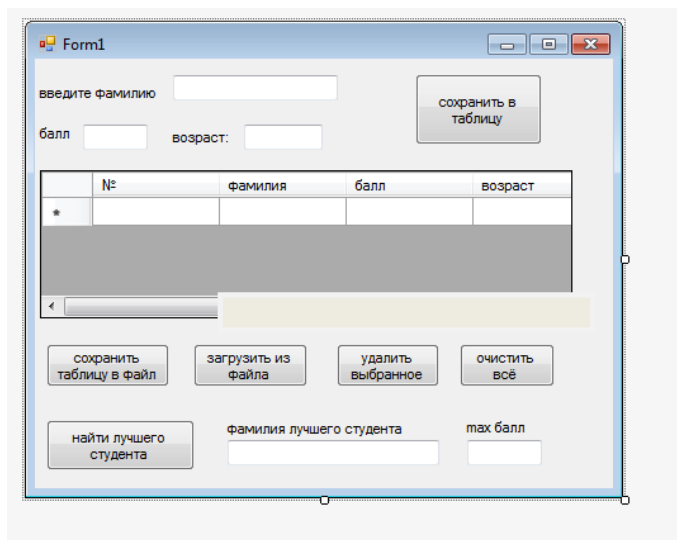
        }
    }
    return fam;
}
} // конец описания класса
// T_student stud2 = new T_student(); – так объявляется единственный
объект класса T_student и выделяется память для него (если нужно)
// T_student[] stud = new T_student[10]; – так объявляется массив из
объектов класса T_student (если нужно)
T_group a_32 = new T_group(); //- объявление объекта класса T_group
и выделение для него памяти
public Form1()
{
    InitializeComponent();
    for (int i = 0; i < 10; i++)
        stud[i] = new T_student(); – (если нужно) для каждого элемента
массива из объектов необходимо выделять память в цикле
}

private void button1_Click(object sender, EventArgs e)
{
    //stud2.fio = textBox1.Text;
    // stud2.test = Convert.ToInt16(textBox2.Text);
    stud[2].fio = textBox1.Text;
    stud[2].test = Convert.ToInt16(textBox2.Text);
}

private void button2_Click(object sender, EventArgs e)
{
    a_32.N = 4;
    a_32.stu[0].fio = textBox1.Text;
    a_32.stu[0].test = 23;
    a_32.stu[1].fio = "Петров";
    a_32.stu[1].test = 45;
    a_32.stu[2].fio = "Иванов";
    a_32.stu[2].test = 67;
    a_32.stu[3].fio = "Сидоров";
    a_32.stu[3].test = 89;
    string f = a_32.poisk_max_test(); //- вызов метода класса
    textBox2.Text = f;
}
}
}
}

```

Варианты заданий для лабораторной работы № 5



Задание № 1. Для всех вариантов описать класс и массив объектов этого класса. Ввести данные с клавиатуры через компоненты TextBox (в таблицу dataGridView или можно без таблицы). Результат вывести в Label – примерно как на рисунке. *Не использовать агрегацию (вложенные классы). Не использовать внешние текстовые файлы, а только массив объектов класса.*

1 Определить класс для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию не более чем по 10 детям через компоненты TextBox в таблицу dataGridView. Вывести средний рост мальчиков

2 Составить программу, выводящую на экран ведомость начисленной заработной платы (Ф.И.О., должность, год и дата рождения, заработная плата). Найти среднюю зарплату. И вывести фамилии с зарплатой выше средней.

3 Определить класс для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию не более чем по 20 вершинам. Вывести среднее значение высот всех вершин. Далее вывести названия всех вершин ниже среднего.

4 Составить программу, выводящую на экран ведомость начисленной заработной платы (Ф.И.О., должность, год и дата рождения, заработная плата). Вывести фамилию сотрудника с самой маленькой зарплатой.

5 Составить программу, выводящую на экран ведомость начисленной заработной платы (Ф.И.О., должность, год и дата рождения, заработная плата). Вывести фамилию сотрудника с самой большой зарплатой.

6 Определить класс для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию не более чем по 20 детям. Вывести имя самой высокой девочки.

7 Определить класс для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию не более чем по 50 вершинам. Вывести название самой низкой вершины из всех.

8 Определить класс для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию не более чем по 20 детям. Вывести средний рост девочек.

9 Определить класс для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию по 20 вершинам. Вывести среднее значение высот всех 20 вершин. Далее вывести названия всех вершин выше среднего.

10 Определить класс для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию не более чем по 20 детям. Вывести средний рост всех детей.

11 Определить класс для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию не более чем по 10 вершинам. Вывести название самой высокой вершины из всех.

12 Определить класс для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию не более чем по 20 детям. Вывести средний рост девочек. Далее вывести имена всех девочек выше среднего.

13 Определить класс для представления информации по горным вершинам, состоящей из названия вершины и ее высоты. Ввести информацию не более чем по 20 вершинам. Вывести среднее значение высот всех 20 вершин.

14 Определить класс для представления анкеты ребенка, состоящей из его имени, пола и роста. Ввести информацию не более чем по 20 детям. Вывести имя самого высокого мальчика. Вывести средний рост мальчиков. Далее вывести имена всех мальчиков ниже среднего.

15 Составить программу, выводящую на экран ведомость начисленной заработной платы (Ф.И.О., должность, год и дата рождения, заработная плата). Вывести все фамилии, начинающиеся на букву «А» и их зарплату.

16 Составить программу, выводящую на экран ведомость начисленной заработной платы (Ф.И.О., должность, год и дата рождения, заработная плата). Вывести дату рождения сотрудника с самой маленькой зарплатой.

Задание № 2

Для всех вариантов создать форму как на рисунке. *Использовать окна диалога (SaveFileDialog), сохранение в текстовый внешний файл, использовать классы*, вложенные (агрегированные) классы*.* (отмеченное * – по желанию)

Screenshot of a Windows application window for student data management. The window contains the following elements:

- Input field: "введите фамилию" with the value "Сорокин".
- Input field: "балл" with the value "99".
- Input field: "возраст:" with the value "19".
- Button: "сохранить в таблицу".
- Table with columns: №, фамилия, балл, возраст.

№	фамилия	балл	возраст
1	Воробьев	87	18
2	Синичкин	56	18
3	Галкин	67	19
4	Сорокин	99	19
- Buttons: "сохранить таблицу в файл", "загрузить из файла", "удалить выбранное", "очистить всё".
- Buttons: "найди лучшего студента".
- Text labels: "фамилия лучшего студента" and "max балл".
- Input fields for "фамилия лучшего студента" and "max балл".

Варианты индивидуального задания (продолжение Задания № 2)

1 Добавить кнопку для вычисления среднего балла всех студентов по дисциплине «Математика».

2 Добавить компонент ComboBox, в котором пользователь может выбрать Название дисциплины (интересующей его). Далее вычислить средний балл по указанной дисциплине.

3 Добавить 3 компонента CheckBox, с помощью которых пользователь может выбрать название/и/или названия групп (интересующей его). Далее вычислить средний балл (по всем предметам) по указанной группе или сразу по нескольким указанным группам

4 Добавить кнопку для: Вычислить средний балл студентов группы АИБ-2-26 по каждой дисциплине

5 Пользователь должен выбирать номер группы из выпадающего списка АИБ-2-28, АИБ-1-30, АВБ-2-25, АВБ-1-26

6 Добавить компонент ComboBox1, в котором пользователь может выбрать Название группы (3-4 шт.), (интересующей его) и ComboBox2, в котором пользователь может выбрать Название дисциплины. Далее вычислить средний балл в указанной группе по данной дисциплине.

7 Добавить компоненты RadioButton, в котором пользователь может выбрать Название группы (интересующей его). Далее вычислить средний балл по указанной группе (по всем предметам).

8 Добавить компонент ListBox, в котором пользователь может выбрать Название дисциплины (интересующей его). Далее вычислить средний балл по указанной дисциплине (по всем группам).

9 Вычислить средний балл студентов группы АИБ-2-26 по дисциплине «Информатика».

10 Добавить компонент ListBox, в котором пользователь может выбрать Название группы (интересующей его). Далее вычислить средний балл по указанной группе (по всем дисциплинам).

11 Добавить компоненты Radiobutton, в котором пользователь может выбрать Название группы (3–4 шт.) (интересующей его) и Radiobutton (2), в котором пользователь может выбрать Название дисциплины. Далее вычислить средний балл в указанной группе по данной дисциплине.

12 Вычислить средний балл всех студентов по дисциплине «Информатика».

13 Добавить компоненты Radiobutton, в котором пользователь может выбрать Название дисциплины (интересующей его). Далее вычислить средний балл по указанной дисциплине

14 Вычислить средний балл всех студентов по дисциплине «Физика».

15 Добавить компонент ComboBox, в котором пользователь может выбрать Название группы (интересующей его). Далее вычислить средний балл по указанной группе.

16 Добавить 3 компонента CheckBox, с помощью которых пользователь может выбрать название/и/или названия дисциплин (интересующей его). Далее

вычислить средний балл по указанной дисциплине Или сразу по нескольким указанным дисциплинам.

Задание № 3. В проект, созданный в предыдущем здании № 2 добавить еще следующие компоненты, кнопки и действия по индивидуальному заданию. В отчете оформить как одно задание с предыдущим.

1 Создать 2-ю таблицу аналогичную таблице 1. Добавить на форму кнопку «**Поиск**» и поле ввода TextBox. Пользователь вводит последнюю букву отчества в поле TextBox и при нажатии кнопки «**Поиск**» получает список студентов с указанными отчествами в таблице 2 со всеми заполненными 6 полями

2 Создать 2-ю таблицу аналогичную таблице 1. Добавить на форму кнопку «**Поиск**» и поле ввода TextBox. Пользователь вводит первые три буквы группы в поле TextBox и при нажатии кнопки «**Поиск**» получает список студентов с указанными группами в таблице 2 со всеми заполненными 6 полями

3 Создать 2-ю таблицу аналогичную таблице 1. Добавить на форму кнопку «**Поиск**» и поле ввода TextBox. Пользователь вводит последние три буквы имени в поле TextBox и при нажатии кнопки «**Поиск**» получает список студентов с указанными именами в таблице 2 со всеми заполненными 6 полями.

4 Создать 2-ю таблицу аналогичную таблице 1. Добавить на форму кнопку «**Поиск**» и поле ввода TextBox. Пользователь вводит последнюю букву группы в поле TextBox и при нажатии кнопки «**Поиск**» получает список студентов с указанными группами в таблице 2 со всеми заполненными 6 полями.

5 Создать 2-ю таблицу аналогичную таблице 1. Добавить на форму кнопку «**Поиск**» и поле ввода TextBox. Пользователь вводит последние три буквы отчества в поле TextBox и при нажатии кнопки «**Поиск**» получает список студентов с указанными отчествами в таблице 2 со всеми заполненными 6 полями.

6 Создать 2-ю таблицу аналогичную таблице 1. Добавить на форму кнопку «**Поиск**» и поле ввода TextBox. Пользователь вводит последние три символа группы в поле TextBox и при нажатии кнопки «**Поиск**» получает список студентов с указанными группами в таблице 2 со всеми заполненными 6 полями.

7 Создать 2-ю таблицу аналогичную таблице 1. Добавить на форму кнопку «**Поиск**» и поле ввода TextBox. Пользователь вводит отчество в поле TextBox и при нажатии кнопки «**Поиск**» получает список студентов с указанным отчеством в таблице 2 со всеми заполненными 6 полями.

8 Создать 2-ю таблицу аналогичную таблице 1. Добавить на форму кнопку «**Поиск отличников**» и поле ввода TextBox. Пользователь вводит дисциплину в поле TextBox и при нажатии кнопки «**Поиск**» получает список студентов отличников по указанной дисциплине в таблице 2 со всеми заполненными 6 полями.

9 Создать 2-ю таблицу аналогичную таблице 1. Добавить на форму кнопку «**Поиск**» и поле ввода TextBox. Пользователь вводит первые три буквы фамилии в поле TextBox и при нажатии кнопки «**Поиск**» получает список студентов с указанными фамилиями в таблице 2 со всеми заполненными 6 полями.

студентов задолжников по указанной дисциплине в таблице 2 со всеми заполненными 6 полями.

21 Создать 2-ю таблицу аналогичную таблице 1. Добавить на форму кнопку «**Поиск**» и поле ввода TextVox. Пользователь вводит номер группы в поле TextVox и при нажатии кнопки «**Поиск**» получает список указанной группы в таблице 2 со всеми заполненными 6 полями.

22 Создать 2-ю таблицу аналогичную таблице 1. Добавить на форму кнопку «**Поиск**» и поле ввода TextVox. Пользователь вводит Имя в поле TextVox и при нажатии кнопки «**Поиск**» получает список студентов с указанным именем в таблице 2 со всеми заполненными 6 полями.

23 Создать 2-ю таблицу аналогичную таблице 1. Добавить на форму кнопку «**Поиск**» и поле ввода TextVox. Пользователь вводит фамилию в поле TextVox и при нажатии кнопки «**Поиск**» получает список студентов с указанной фамилией в таблице 2 со всеми заполненными 6 полями.

Задание № 4*. В проект, созданный в двух предыдущих заданиях № 2, № 3, добавить еще следующие компоненты по индивидуальному заданию. В отчете оформить как одно задание с двумя предыдущим.

1 Добавить кнопку для сортировки таблицы по возрастанию оценок по математике.

2 Добавить кнопку для сортировки таблицы по возрастанию оценок по физике.

3 Добавить кнопку для сортировки таблицы по возрастанию оценок по информатике.

4 Добавить кнопку для сортировки таблицы по убыванию оценок по математике.

5 Добавить кнопку для сортировки таблицы по убыванию оценок по физике.

6 Добавить кнопку для сортировки таблицы по убыванию оценок по информатике.

7 Добавить кнопку для сортировки таблицы по возрастанию группы.

8 Добавить кнопку для сортировки таблицы по алфавиту по имени.

9 Добавить кнопку для сортировки таблицы по алфавиту по фамилии.

10 Добавить кнопку для сортировки таблицы по алфавиту по отчеству.

11 Добавить кнопку для сортировки таблицы по возрастанию оценок по математике.

12 Добавить кнопку для сортировки таблицы по возрастанию оценок по физике.

13 Добавить кнопку для сортировки таблицы по возрастанию оценок по информатике.

14 Добавить кнопку для сортировки таблицы по убыванию оценок по математике.

15 Добавить кнопку для сортировки таблицы по убыванию оценок по физике.

16 Добавить кнопку для сортировки таблицы по убыванию оценок по информатике.

17 Добавить кнопку для сортировки таблицы по возрастанию группы.

18 Добавить кнопку для сортировки таблицы по алфавиту по имени.

19 Добавить кнопку для сортировки таблицы по алфавиту по фамилии.

20 Добавить кнопку для сортировки таблицы по алфавиту по отчеству.

21 Добавить кнопку для сортировки таблицы по возрастанию оценок по математике.

22 Добавить кнопку для сортировки таблицы по возрастанию оценок по физике.

23 Добавить кнопку для сортировки таблицы по возрастанию оценок по информатике.

24 Добавить кнопку для сортировки таблицы по убыванию оценок по математике.

25 Добавить кнопку для сортировки таблицы по убыванию оценок по физике.

26 Добавить кнопку для сортировки таблицы по убыванию оценок по информатике.

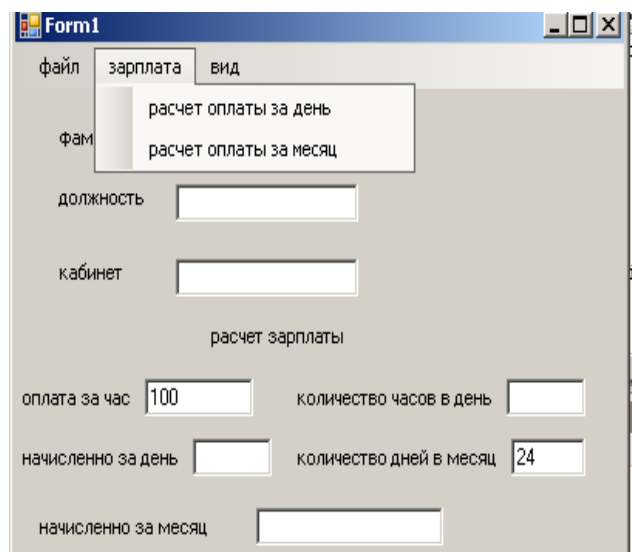
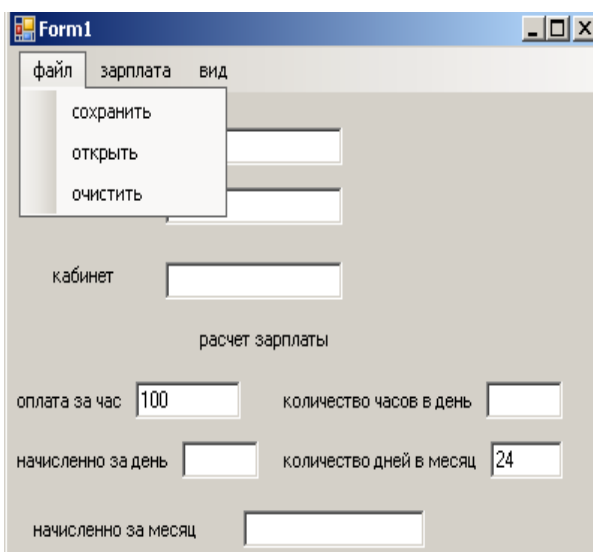
27 Добавить кнопку для сортировки таблицы по возрастанию группы.

Лабораторная работа № 6

СОЗДАНИЕ МНОГОСТРАНИЧНЫХ ПРИЛОЖЕНИЙ ИЛИ РАБОТА С НЕСКОЛЬКИМИ ФОРМАМИ

СОЗДАНИЕ МЕНЮ

В проекте, созданном в среде Visual C можно организовать меню с помощью компонента **MenuStrip**. Например, меню из 3 разделов (файл, зарплата, вид) и 2 – 3 подпунктов в каждом разделе меню (сохранить, открыть и т.д.):



Приведем коды для некоторых пунктов меню:

```
public Form1()
{
    InitializeComponent();
}

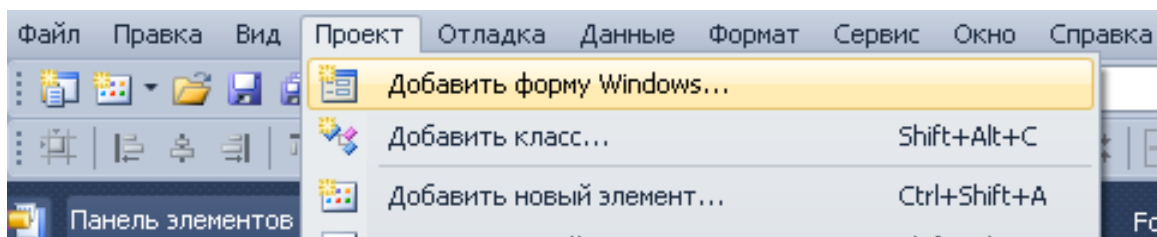
private void
расчетОплатыЗаДеньToolStripMenuItem_Click(object sender,
EventArgs e)
{
    int h=Convert.ToInt16(textBox4.Text);
    int z=Convert.ToInt16(textBox5.Text);
    z=h*z;
    textBox6.Text = z.ToString();
}

private void
расчетОплатыЗаМесяцToolStripMenuItem_Click(object sender,
EventArgs e)
{
    int a = Convert.ToInt16(textBox6.Text);
    int b = Convert.ToInt16(textBox7.Text);
    b=a*b;
    textBox8.Text = b.ToString();
}

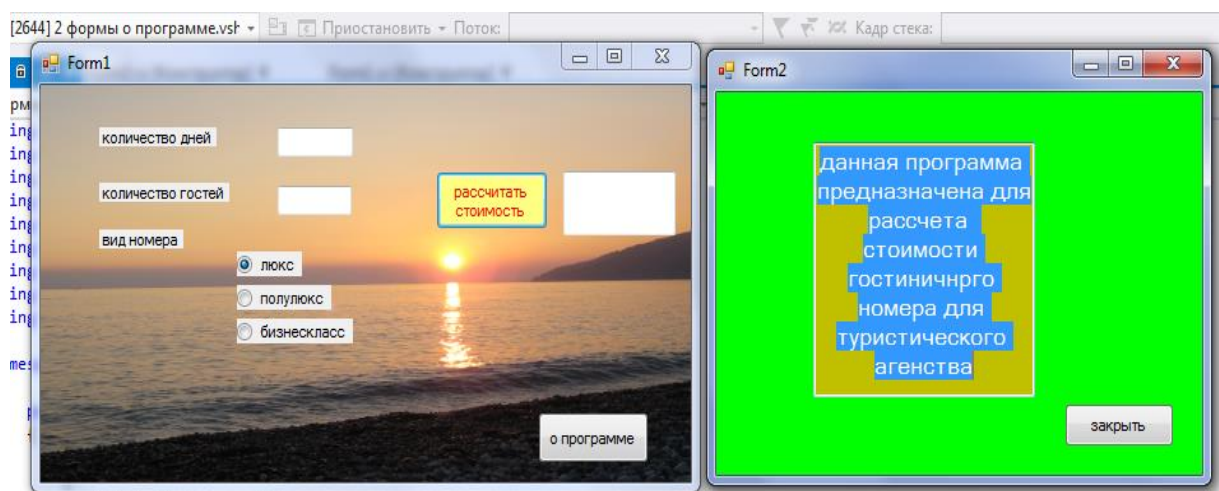
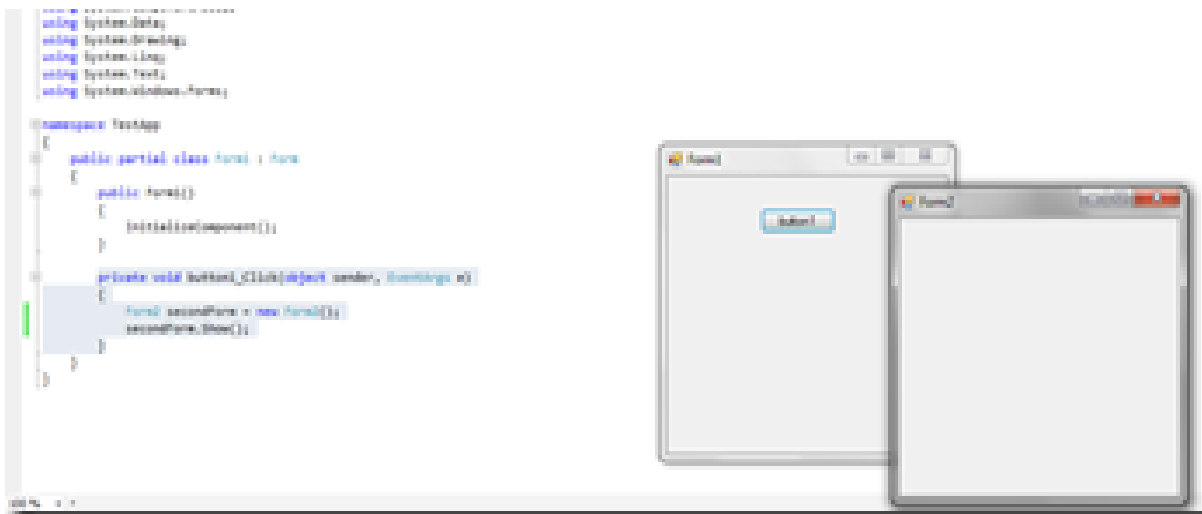
private void
шрифтДляФамилииToolStripMenuItem_Click(object sender,
EventArgs e)
{
    if (fontDialog1.ShowDialog()==DialogResult.OK)
        textBox3.Font = fontDialog1.Font;
}
}
```

РАБОТА С НЕСКОЛЬКИМИ ФОРМАМИ

Чтобы научиться работать с несколькими формами, сделать пример – вызов 2-й формы с описанием «О программе» и закрытие ее. Для создания 2-й формы необходимо зайти в меню «Проект» и выбрать «Добавить форму Win».



После чего появится:



На первой форме кнопка «О программе»

```

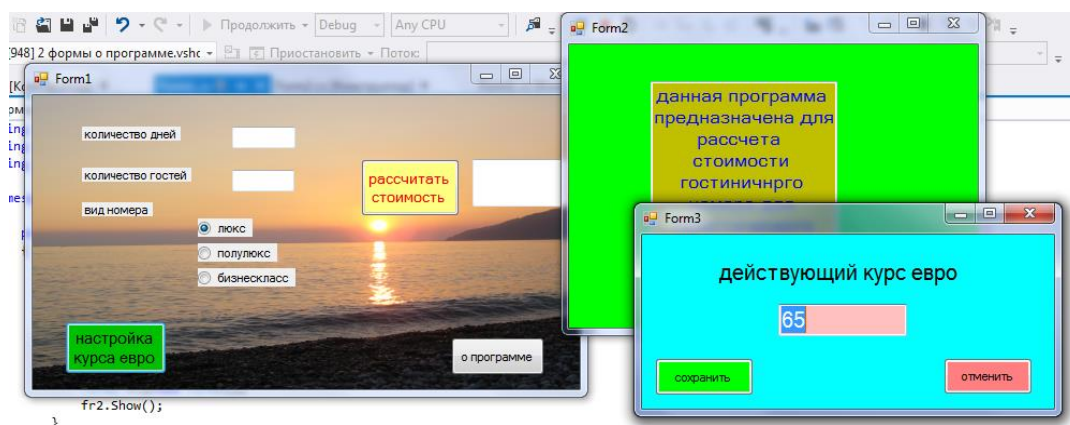
private void button1_Click(object sender, EventArgs
e) // кнопка «О программе» на 1-ой форме
{
    Form2 secondForm = new Form2(); //создать переменную
secondForm
    secondForm.Show(); // открыть форму
}
  
```


На 2-й форме кнопка «Заккрыть»:

```
private void button1_Click(object sender, EventArgs e)
{
    Hide(); //- закрыть 2-ую форму или Close();
}
```

ПЕРЕДАЧА ДАННЫХ МЕЖДУ ФОРМАМИ

Что бы научиться передавать данные во 2-ю форму и обратно, сделать пример – перейти на форму настройки (ввести текущий курс евро). На главной форме рассчитать стоимость товара.



1) На главной форме объявляем статическую переменную для хранения и обмена данными (курс евро) в 1-й форме:

```
public static int kurs_e =65;
// перед строкой
public Form1()
{
    InitializeComponent();
}
```

2) В 1-й форме создать событие по нажатию кнопки «Настройки курса»
кнопка «Настройки курса»

```
{
    Form2 Frm2 = new Form2(); // описать переменную fm2
    Frm2.ShowDialog(); // открыть 2-ую форму
}
```

3) Меняем конструктор во 2-й форме так, что бы в поле `textBox1` записывались значения из 1-й формы из глобальной статической переменной `kurs_e`;

```
public Form2()
```

```

{
    InitializeComponent();
    textBox1.Text = Form1.kurs_e.ToString();
}

```

4) По нажатию кнопки «сохранить изменения» во 2-ой форме записать следующее, чтобы в статическую переменную `kurs_e` записались новое значение из `textBox1` 2-ой формы.

```
private void button1_Click(object sender, EventArgs e)//кнопка
«сохранить» на 2-ой форме
```

```

{
    Form1.kurs_e = Convert.ToInt16(textBox1.Text);
}

```

5) По нажатию кнопки «закрыть» во 2-ой форме

```
private void button2_Click(object sender, EventArgs e)
```

```

{
    Hide(); //- закрыть 2-ую форму
}

```

6) Теперь можем использовать настроенное значение курс валют (можно несколько переменных) Для этого, например, в 1-ой форме создать событие «вычислить цену товара»

```
private void button2_Click(object sender, EventArgs e)//кнопка
вычислить цену товара
```

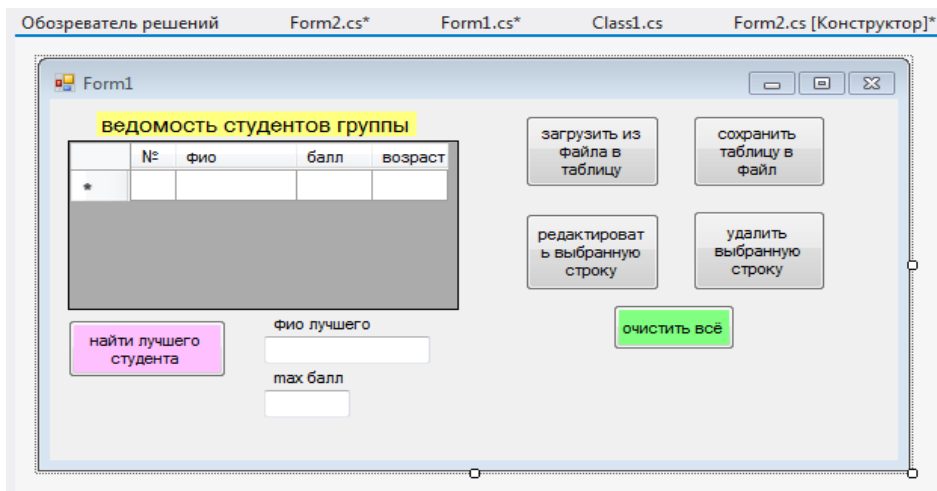
```

{ textBox5.Text= (kurs_e*120).ToString();// будет использоваться
  тот курс который настроен во 2-ой форме
}

```

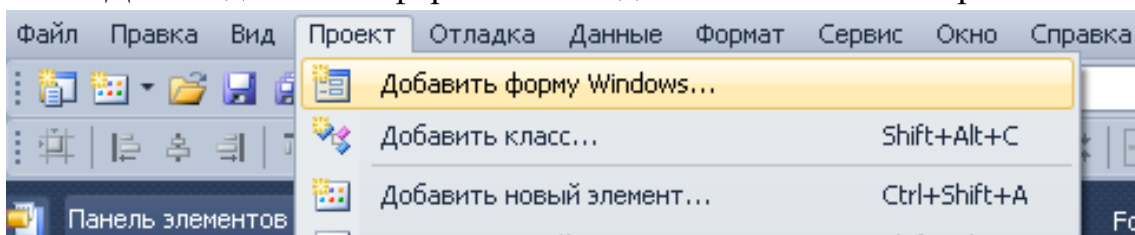
ВТОРОЙ СПОСОБ ПЕРЕДАЧИ СТРОКИ ТАБЛИЦЫ НА РЕДАКТИРОВАНИЕ ВО 2-Ю ФОРМУ

1) Пусть первая (главная)форма выглядит так:

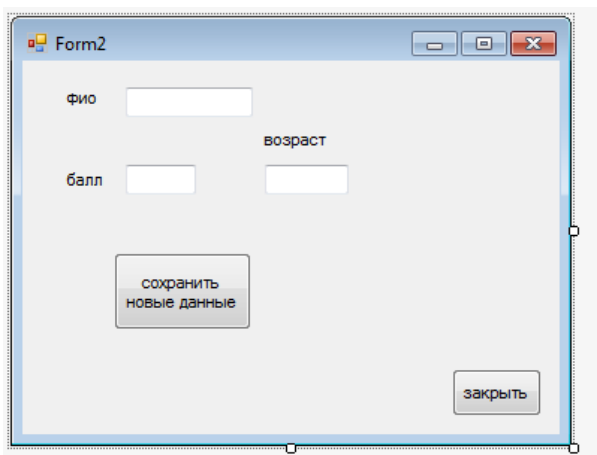


2) Создаем 2-ю форму.

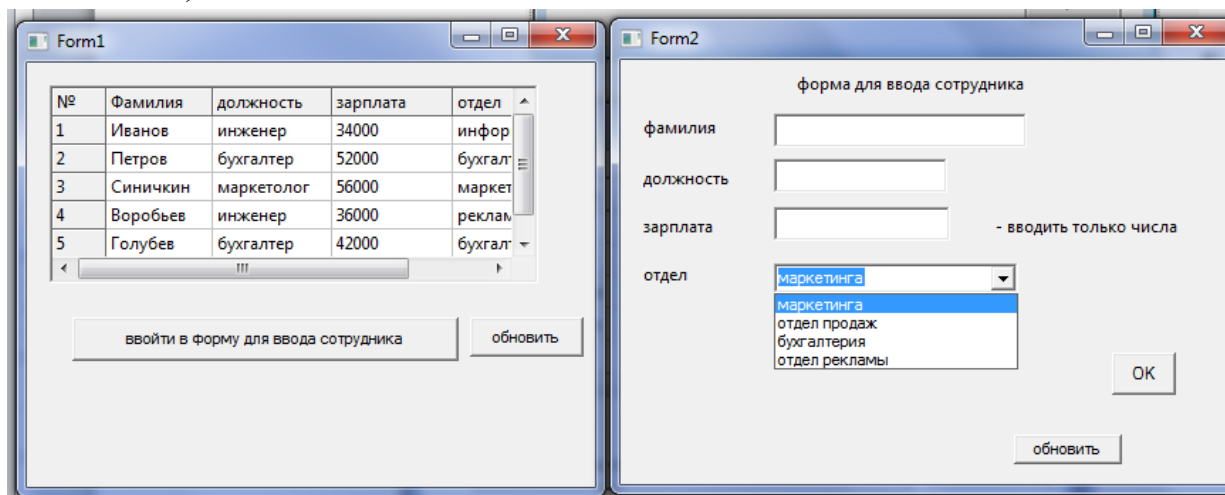
Для создания 2-й формы необходимо зайти в меню Проект



3) Размещаем поля и кнопки на ней вот так:



Данная операция может выглядеть и так: (с выбором из списка или из радиокнопок)



4) создаем класс `C_Student` с помощью команды *Проект – Добавить класс*

```
public class C_student
{
    public string fio;
    public int ball;
    public int age;
}
```

5) В главной форме объявляем объект **public static** данного класса или статическую переменную для хранения и обмена данными в 1-й форме

```
public static C_student stud = new
C_student();//объявить статический глобальный объект класса
```

б) Событие по нажатии кнопки «Редактировать выбранную строку»:

```
private void button3_Click(object sender, EventArgs e) //кнопка редактиро-
вать выбранную строку
{int k = dataGridView1.SelectedCells[0].RowIndex; //номер выделенной
строки в таблице для редактирования .
    stud.fio = dataGridView1.Rows[k].Cells[1].Value.ToString();//сохраняем
данные из выделенной строки
    stud.ball = Convert.ToInt16(dataGridView1.Rows[k].Cells[2].Value);//чтобы
они отобразились на 2-й форме
    stud.age = Convert.ToInt16(dataGridView1.Rows[k].Cells[3].Value);//при
вызове конструктора, который мы изменим позже
```

```
Form2 Frm2 = new Form2();//описать переменную fm2 для доступа к 2-й
форме
```

```
Frm2.ShowDialog();// открыть 2-ую форму
```

```
dataGridView1.Rows[k].Cells[1].Value = stud.fio;//- записать новое значе-
ние в таблицу на 1-ой форме из поля фамилия из 2-ой формы
dataGridView1.Rows[k].Cells[2].Value = stud.ball.ToString();//- записать
новое значение в таблицу на 1-ой форме из поля балл из 2-ой формы
dataGridView1.Rows[k].Cells[3].Value = stud.age.ToString();
}
```

7) Меняем конструктор во 2-й форме так, чтобы в поля **textBox1** записывались значения из 1-й формы из глобального объекта нашего класса **Form1.stud.fio**;

```
public Form2()
{
    InitializeComponent();
    textBox1.Text = Form1.stud.fio;
    textBox3.Text = Form1.stud.ball.ToString();
    textBox2.Text = Form1.stud.age.ToString();
}
```

8) По нажатию кнопки «сохранить изменения» во 2-й форме записать следующее чтобы в статический объект класса stud.fio записались новые значения из textBox

```
private void button1_Click(object sender, EventArgs e)
//сохранить
{
    Form1.stud.fio = textBox1.Text;
    Form1.stud.ball = Convert.ToInt16(textBox3.Text);
    Form1.stud.age = Convert.ToInt16(textBox2.Text);
}
```

9) По нажатию кнопки «закрыть»

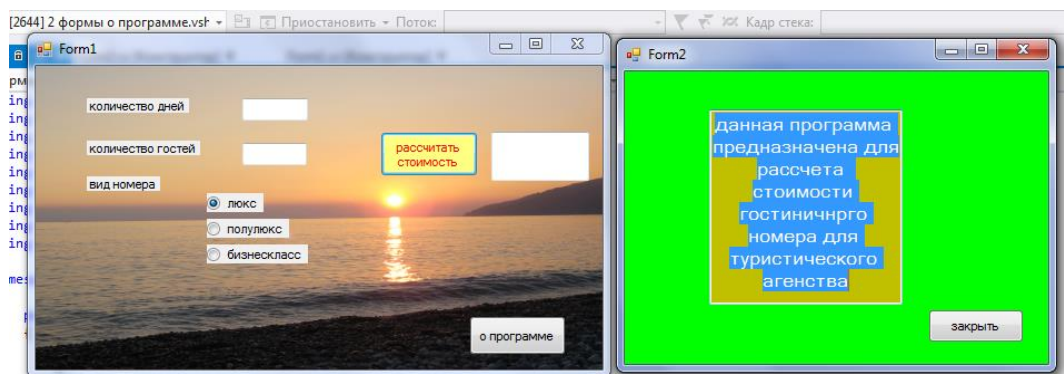
```
private void button2_Click(object sender, EventArgs e)
{
    Hide(); //- закрыть 2-ую форму
}
```

Варианты заданий для лабораторной работы № 6

Тема: «Работа с несколькими формами»

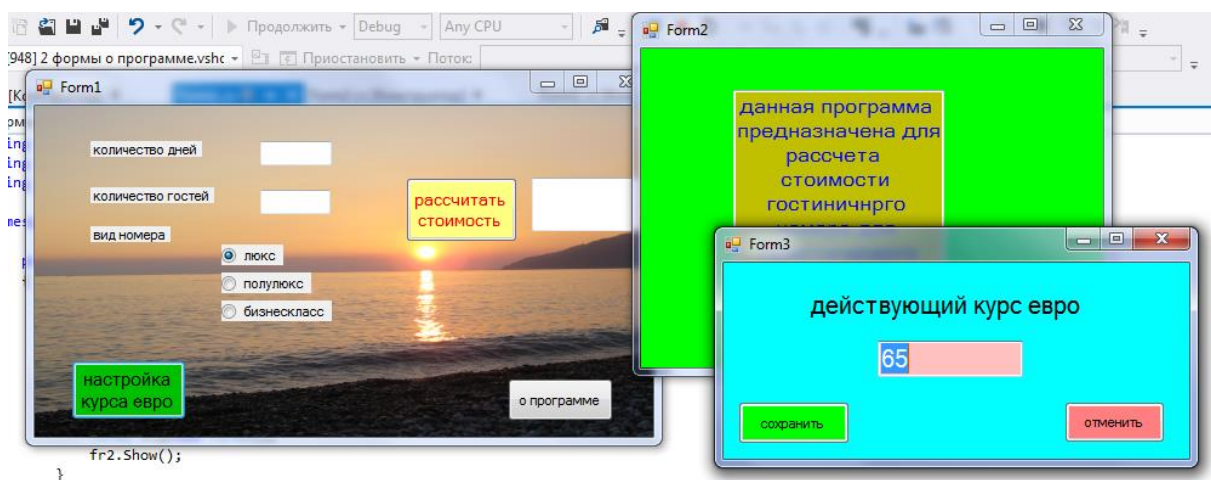
Задание № 1

Чтобы научиться работать с несколькими формами, сделать пример – вызов 2-й формы с описанием «О программе» и закрытие ее. (см. лекции)

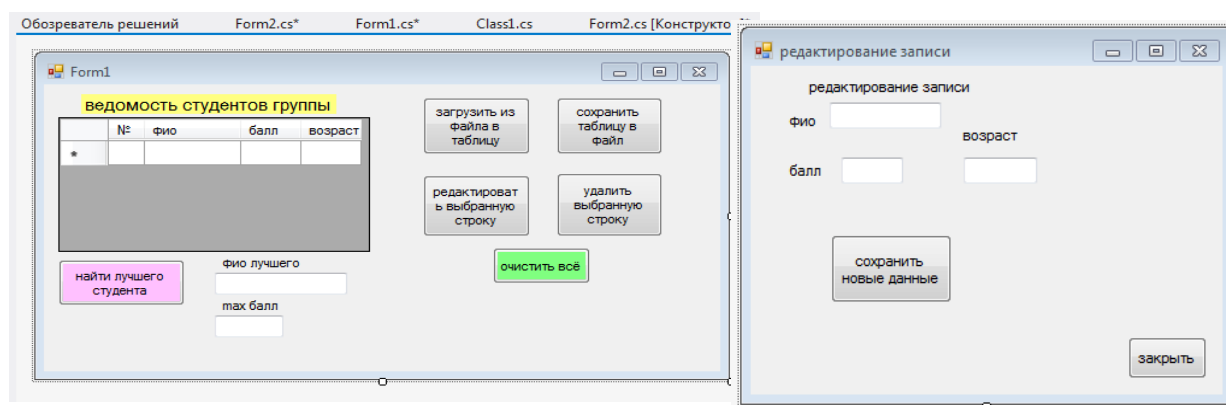


Задание № 2. Тема: «Передача данных между формами»

Чтобы научиться передавать данные во 2-ю форму и обратно, сделать пример – перейти на форму настройки (ввести текущий курс евро). На главной форме рассчитать стоимость товара, используя установленный курс евро (см. лекции). Например, по формуле: умножить стоимость дней на количество гостей, умножить на курс евро и умножить на стоимость номера (пусть люкс – 500€; полулюкс – 300€; бизнес – 200€)



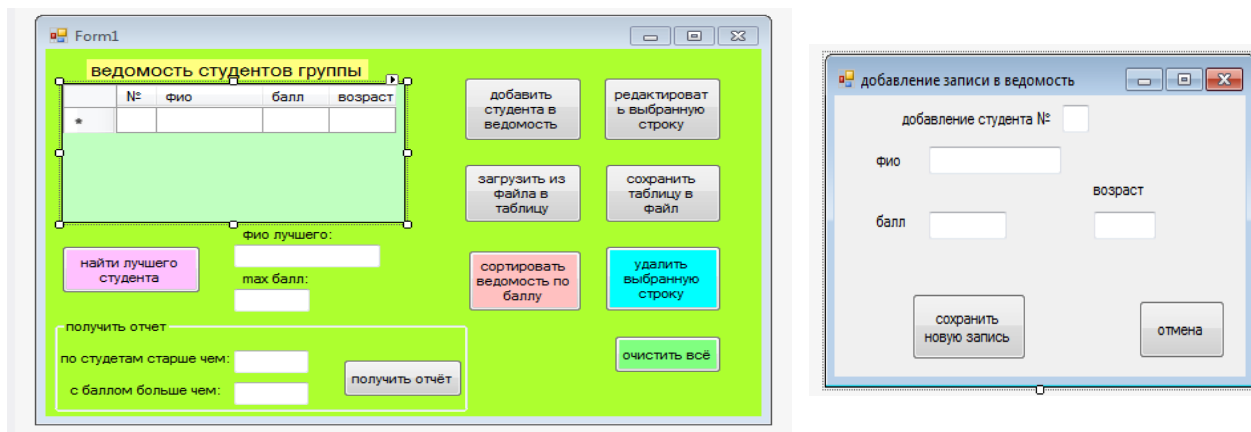
Задание № 3. Тема: «Передача строк таблицы на редактирование во 2-ю форму используя классы»



Вместо таблицы разместить таблицу другого вида, а именно

№	Фамилия	Имя	Отчество	Группа	Математика	Физика	Информатик.
1							
2							
3							
4							

Для всех вариантов на основе задания из лабораторной № 6, добавить в проект кнопку «**Редактировать Выбранную Строку**», при нажатии на которую появляется новая форма 2 «Редактирование». В ней отражается содержимое выбранной строки. После внесения изменений на форме 2 они должны сохраняться в массив и в таблицу (см. лекции). Далее добавить кнопку «**Добавить студента**» при нажатии на которую появляются другая новая форма 3 «**добавление**». После внесения изменений на форме 3 они должны сохраняться в массив и в таблицу. Использовать описание класса в отдельном модуле.



Задание № 4. В проект, созданный в предыдущем здании и в лаб. работе № 5, добавить еще действие по индивидуальному заданию:

1 Добавить кнопку для сортировки таблицы методом «пузырька» по возрастанию оценок по математике (не использовать встроенные сортировки).

2 Добавить кнопку для сортировки таблицы методом «пузырька» по возрастанию оценок по физике (не использовать встроенные сортировки).

3 Добавить кнопку для сортировки таблицы методом «пузырька» по возрастанию оценок по информатике (не использовать встроенные сортировки).

4 Добавить кнопку для сортировки таблицы методом «пузырька» по убыванию оценок по математике (не использовать встроенные сортировки).

5 Добавить кнопку для сортировки таблицы методом «пузырька» по убыванию оценок по физике (не использовать встроенные сортировки).

6 Добавить кнопку для сортировки таблицы методом «пузырька» по убыванию оценок по информатике (не использовать встроенные сортировки).

7 Добавить кнопку для сортировки таблицы методом «пузырька» по возрастанию группы (не использовать встроенные сортировки).

8 Добавить кнопку для сортировки таблицы методом «пузырька» по алфавиту по имени (не использовать встроенные сортировки).

9 Добавить кнопку для сортировки таблицы методом «пузырька» по алфавиту по фамилии (не использовать встроенные сортировки).

10 Добавить кнопку для сортировки таблицы методом «пузырька» по алфавиту по отчеству (не использовать встроенные сортировки).

11 Добавить кнопку для сортировки таблицы методом «пузырька» по возрастанию оценок по математике (не использовать встроенные сортировки).

12 Добавить кнопку для сортировки таблицы методом «пузырька» по возрастанию оценок по физике (не использовать встроенные сортировки).

13 Добавить кнопку для сортировки таблицы методом «пузырька» по возрастанию оценок по информатике (не использовать встроенные сортировки).

14 Добавить кнопку для сортировки таблицы методом «пузырька» по убыванию оценок по математике (не использовать встроенные сортировки).

15 Добавить кнопку для сортировки таблицы методом «пузырька» по убыванию оценок по физике.

16 Добавить кнопку для сортировки таблицы методом «пузырька» по убыванию оценок по информатике (не использовать встроенные сортировки).

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

- 1 **Павловская, Т.А.** С#. Программирование на языке высокого уровня / Т.А. Павловская. – СПб. : Питер, 2014. – 432 с.
- 2 **Мартынов, Н.Н.** С# для начинающих / Н.Н. Мартынов. – СПб. : Ку-диц-Пресс, 2007. – 400 с.
- 3 **Медведев, В.И.** Особенности объектно ориентированного программирования на С++/CLI, С# / В.И. Медведев. – Казань : РИЦ «Школа», 2010. – 62 с.
- 4 **Фомин, Г.В.** Введение в программирование на С# в среде MS Visual Studio / Г.В. Фомин. – Ростов н/Д : Изд-во ЮФУ, 2010. – 181 с.
- 5 **Чернов, Э.А.** Лекции по языку Visual С# / Э.А. Чернов. – М. : МАДИ, 2014. – 159 с.