МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт нанотехнологий, электроники и приборостроения

Кафедра электрогидроакустической и медицинской техники

Методическое руководство

к выполнению лабораторных работ
по курсу
Информационные технологии
Работа с основами HTML и
Microsoft Visual Studio Community 2015

Для студентов направлений 12.03.01, 12.03.04,
21.03.02, 17.03.01, 26.05.04, 12.04.01, 12.04.04

Издательство Южного федерального университета
Ростов–на–Дону
2017

УДК 002.001; 002:001.8
Составители: Вишневецкий В.Ю., Старченко И.Б., Ледяева В.С., Петров И.А.
Работа с основами языка HTML и Microsoft Visual Studio Community 2015. Методическое руководство к выполнению лабораторных работ по курсу «Информационные технологии». – Ростов–на–Дону: Изд–во ЮФУ, 2016. – 33 с.

Данное методическое руководство предназначено для проведения лабораторных и практических занятий по курсу «Информационные технологии», а также для самостоятельного изучения работы приложений в среде операционной системы MS Windows. В руководстве содержится описание 2-х лабораторных работ и 4-х заданий, которые охватывают основные приемы работы с языком верстки HTML и MS Visual Studio Community 2015. В конце работы содержится перечень навыков, которые студент должен приобрести после ее выполнения.
Методические указания разработаны на модульной основе и могут быть использованы для дистанционного образования в локальной вычислительной сети ИТА ЮФУ, а также в глобальной сети Интернет (inep.sfedu.ru).
Методическое руководство предназначено для подготовки бакалавров по направлениям 12.03.01, 12.03.04, 21.03.02, 17.03.01, инженеров по специальности 26.05.04 и магистров по направлениям 12.04.01, 12.04.04.

Ил. 24.
Рецензент Д.В. Бурьков, канд. техн. наук, доцент кафедры ЭиМ ИТА ЮФУ.

Цель модуля – научиться создавать простейший гипертекстовый документ средствами текстового редактора Блокнот. Научиться использовать теги форматирования шрифта и абзаца.

ЛАБОРАТОРНАЯ РАБОТА №5. ОСНОВЫ HTML

ЧАСТЬ 1. СОЗДАНИЕ ПРОСТЕЙШЕГО HTML-ДОКУМЕНТА. ФОРМАТИРОВАНИЕ ШРИФТА И АБЗАЦА

1. Язык HTML был разработан примерно в 1986-1991 годах британским ученым Тимом Бернерсом-Ли.

HTML – это гипертекстовый язык разметки (HyperText Markup Language), который используется для создания документов в Интернет(веб-страниц).

При помощи HTML создается веб-страница, содержащая все нужные элементы. На HTML странице можно разместить простой текст, выделить его жирным или курсивом, вставить ссылку, таблицу, нумерованный или ненумерованный список, картинки, разбить текст на абзацы и разделы, задавать разделам заголовки.

Также на HTML страницу можно вставить форму с текстовыми полями, кнопками, выбором вариантов из списка, checkbox'ами и radio элементами.

В помощь HTML пришли CSS и JavaScript. CSS используется для стилизации HTML элементов. При помощи CSS можно не только изменять цвет текста в блоке, но и делать геометрические фигуры (треугольник, трапецию), рисовать (сердечки, котиков), делать разнообразную анимацию (слайдеры картинок, прелоадеры, падающий снег).

Текстовый документ, содержащий язык разметки HTML, имеет расширение **.html** или **.htm** и открывается браузерами. На данный момент самыми популярными браузерами являются Google Chrome, Mozilla Firefox, Opera, Internet Explorer и Safari.

2. Каждый HTML документ должен иметь начало и конец которого обозначаются тегами <html> и </html>. Внутри документа есть два основных блока – <head> и <body>.

```
<html>
<head></head>
<body></body>
</html>
```

Начала каждого блока блока обозначается открывающим тегом (<head>), а конец - закрывающим (</head>).

В первой части документа, в блоке <head>, содержатся элементы, описывающие документ. Они не отображаются в браузере.

В блоке <head> могут содержаться:

- заголовок документа <title>, который устанавливает заголовок страницы в браузере
- мета теги названию, описания и ключевых слов документа <meta>
- ссылка на стилевой файл документа <link>
- ссылки на JavaScript файлы, использующиеся в документе (тег <script>)
- описание автора документа

Второй раздел <body> содержит непосредственно элементы страницы.

```
<html>
  <head>
    <title>Моя первая страница</title>
  </head>
  <body>
    <h1>Заголовок</h1>
    <p>Описание</p>
  </body>
</html>
```

HTML тег – это элемент HTML разметки страницы. Теги бывают открывающими (<html>) и закрывающими (</html>). Некоторые теги не имеют закрывающего – . Также теги бывают блочными и строчными. Блочные теги начинаются с новой строки и переводят следующий элемент на новую строку - <p>, . Строчный элементы отображаются в той же строке – , <b>. При расположении элементов на HTML странице, элементы будут следовать друг за другом, отображаясь сбоку или снизу от предыдущего. Чем ниже описан элемент на HTML странице, тем ниже он будет отображаться.

На расположение элемента и его соседей также влияют размеры элемента (width ширина и height высота), его внешние(margin) и внутренние(padding) отступы и границы(border).

Фактический размер элемента равен сумме его высоты (ширины), вертикальных(горизонтальных) внутренних отступов и границ. На расположение соседних элементов также влияют его внешние отступы. Но следование элементов можно изменять при помощи различных свойств.

Так, например, свойство float отображает блочные элементы в том же ряду. При помощи свойства position элемент можно вытащить из потока и разместить в любом месте страницы, поверх всех элементов и т.д.

Блок <body> может содержать такие HTML теги:

- <table> - табличка
- <a> - ссылка
- <img> - картинка
- - блочный элемент, без оформления
- <p> - параграф, абзац
- - сточный элемент, без оформления

- <form> - форма
- , <ol> - списки
- <input>, <textarea>, <select> - элементы формы
- <h1> - <h6> - заголовки
- <b>, <i>, <u>, - строчные элементы для оформления текста - жирный, курсив
- <audio>, <video>, <canvas> - мультимедийные элементы

ЧАСТЬ 2. СОЗДАНИЕ ПРОСТЕЙШЕГО HTML-ДОКУМЕНТА. ФОРМАТИРОВАНИЕ ШРИФТА И АБЗАЦА

1. Создать простой текстовый файл.

2. Запустить редактор **Блокнот**, ввести текст.

Приветствую Вас на моей первой web-страничке!

3. Сохранить файл в созданной папке. При сохранении, в окне диалога **Сохранить как…** в строке **Тип файла:** выбрать вариант **Все файлы (*.*)** , а в строке **Имя файла** задать имя с расширением .**html**, например **1_name.html** (где **name** – ваше Фамилия И.О.)

Закрыть документ, найти его пиктограмму в окне **Мой компьютер** или в окне программы **Проводник.**

4. Открыть файл. Проанализировать, *с помощью какого приложения отображается файл* и как выглядит введенная фраза.

5. Ввести теги, определяющие структуру **html**-документа:

С помощью контекстного меню открыть файл с помощью редактора **Блокнот**. Ввести приведенные ниже теги (не обязательно вводить их прописными), в разделе заголовка документа (между тегами <**TITLE**> </**TITLE**>) указать свою фамилию.

<HTML>
<HEAD> <TITLE> Фамилия </TITLE>
</HEAD>
<BODY>
Приветствую Вас на моей первой web-страничке!
</BODY>
</HTML>

6. Сохранить документ под тем же именем, обновить его отображение в браузере (выполнить **Вид/Обновить** или нажать кнопку **Обновить** на панели инструментов). Проанализировать произошедшие изменения в отображении документа.

7. Отредактировать документ:

Добавить после текста **«Приветствую Вас на моей первой web-страничке!»** текст подписи:

Студент группы NNN Фамилия Имя

Сохранить документ (но не закрывать) и обновить его просмотр в браузере.

8. Используя одиночный тег **
, отредактировать документ так, чтобы подпись начиналась с новой строки, а **Фамилия Имя – в следующей строке. Просмотреть в браузере новый вариант. ***Зафиксировать результат!***

Примечание! После каждого изменения документ нужно сохранять, а просмотр в браузере начинать с обновления загрузки документа с помощью кнопки ***Обновить*** *на панели инструментов.*

9. Оформить фрагменты текста с помощью стилей **Заголовков**:

Первую строку документа оформить стилем **Заголовок 1-го уровня** с помощью парного тега **<H1> …</H1>**. Вторую строку оформить как **Заголовок 6-го уровня**, а третью как **Заголовок 4-го уровня**.

10. Поменять стиль оформления первой строки на **Заголовок 2 уровня**, второй строки - на **Заголовок 5 уровня**, последней строки - на **Заголовок 3-го уровня**. ***Зафиксировать результат***.

Выполнить форматирование шрифта:

После строки **Фамилия Имя** добавить еще одну строку текста

Нас утро встречает прохладой

Оформить фразу по приведенному ниже образцу.

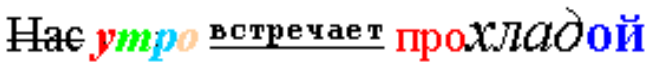


Рис.1. Образец оформления текста

В слове УТРО все буквы должны иметь **разные цвета**. В слове ПРОХЛАДОЙ оформить буквы ПРО – **красным** цветом, ОЙ – **синим**.

11. Оформить строку с подписью (**Студент группы NNN Фамилия Имя**) **курсивом**, размер шрифта задать относительным изменением. Использовать теги **<FONT SIZE=«+2»>** и **<I>**

Просмотреть полученный документ в браузере. ***Зафиксировать результат***.

12. Выполнить форматирование абзацев:

Создать новый документ **2_name.html**, сохранить его в той же рабочей папке.

Ввести текст (использовать копирование текста из документа **1_name.html**):

<HTML>
<HEAD> <TITLE> Фамилия </TITLE>
</HEAD>
<BODY>
**Приветствую Вас на моей второй web-страничке!
 Монолог Гамлета**
</BODY>
</HTML>

Выровнять текст **по центру**.

13. Ввести текст:

Быть иль не быть - вот в чем вопрос. Что благороднее: сносить удары неистовой судьбы - иль против моря невзгод вооружиться, в бой вступить. И все покончить разом...

Оформить выравнивание абзаца **по ширине**.

14. Ограничить абзац горизонтальными разделительными линиями сверху и снизу, используя тег <HR>.

Скопировать монолог и разбить его на абзацы. Выровнять **по центру**.

Быть иль не быть - вот в чем вопрос.
Что благороднее: сносить удары
Неистовой судьбы - иль против моря
Невзгод вооружиться, в бой вступить
И все покончить разом...

Сохранить документ.

Просмотреть документ в окне браузера, изменяя размер окна. ***Зафиксировать результат***

15. Выполнить оформление списков:

Создать новый документ **3_name.html**, сохранить его в той же рабочей папке жесткого диска.

Ввести текст:

<HTML>
<HEAD> <TITLE> Фамилия </TITLE>
</HEAD>
<BODY>
Приветствую Вас на моей третьей web-страничке!

</BODY>
</HTML>

16. Дополнить текст документа (между тегами <BODY>…</BODY>) следующим текстом:

Я знаю, как оформлять:
Шрифты,
Заголовки,
Абзацы

17. Оформить три последние строки как **список нумерованный**. Для этого использовать следующую конструкцию тегов:

<OL>
<LI> Шрифты, </LI>
<LI> Заголовки, </LI>
<LI> Абзацы </LI>
</OL>

18. Поменять оформление списка на **список маркированный**. Использовать теги <UL>, </UL>.

Создать «смешанный» список:

Я знаю, как оформлять:

1. **Шрифты**
   - **Размер**
   - **Цвет**
   - **Гарнитуру**
   - **Индексы**
2. **Заголовки**
   - **От 1-го до 6-го уровня**
3. **Абзацы**
   - **Выравнивание**
   - **Разрыв строк внутри абзаца**
   - **С использованием переформатирования**.

Просмотреть документ в окне браузера. ***Зафиксировать результат.***

ЧАСТЬ 3. СОЗДАНИЕ И ФОРМАТИРОВАНИЕ ТАБЛИЦ

1. Создать таблицу по приведенному образцу, сохранить документ под именем **tabl_name.html.** Сверху над таблицей разместить заголовок **Таблица №1.**

При отображении таблицы в браузере должны удовлетворяться следующие условия:

- таблица должна быть выравнена по центру и быть правильной (симметричной) формы;
- в центральной ячейке поместить символ ***** (звездочка), остальные ячейки должны быть пустыми.

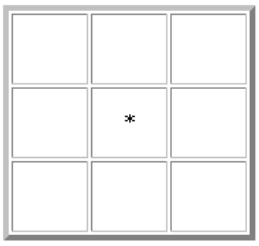


Рис.2. Пример оформления таблицы

Примечание! Для отображения пустых ячеек в них нужно поместить символьный примитив пробела *** Зафиксировать результат***.

2. Создать новый HTML-документ - **rasp_name.html** с расписанием занятий.

Документ должен начинаться заголовком:

Расписание занятий гр. NNN на весенний семестр 201_ г.

Первая строка таблицы должна быть оформлена как заголовки полей (с использованием тегов **<TH>**).

Таблица по ширине должна занимать полный размер окна. Ширину отдельных столбцов задать в относительных единицах (в **%**), с тем, чтобы при изменении ширины окна пропорции таблицы сохранялись.

3. Просмотреть созданный документ в браузере при различных размерах окна и различных настройках размера шрифта.

Выполнить цветовое оформление каждой ячейки меню. ***Зафиксировать результат.***

Таблица 1

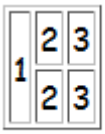
Пример расписания занятий на весенний семестр

	8:00-9:35	9:50-11:25	11:55-13:30	13:45-15:20	15:50-17:25	17:40-19:15	19:30-21:05
Понедельник		Иванов И.И. Математика (лек.) Г-000					
Вторник			Сидоров А.И. Математика (прак.) Г-000				
Среда				Иванов И.И. Математика (лек.) Г-000			
Четверг	Федорова М.Г. История (лек.) Г-000						
Пятница		Куцов Ф.Т. История (прак.) Г-000		Артемов А.А. Физика (лек.) Г-000			
Суббота							
Воскресенье							

Таблица 5

Теги оформления таблиц

Определить таблицу	<TABLE></TABLE>	Пример <TABLE border=”1” align =«CENTER» width=«50%» > <TR> <TABLE> <TH >Товар</TH> <TH>Цена</TH> </TR> <TR> <TD>Радиотелефон</TD> <TD>2000 </TD>
Окантовка таблицы	<TABLE BORDER =”?” </TABLE>	
Строка таблицы	<TR> </TR>	
Выравнивание	<TR ALIGN=left | right | center | middle | bottom >	
Ячейка таблицы	<TD></TD>	
Выравнивание по горизонтали	<TD ALIGN=LEFT | RIGHT | CENTER></TD>	
Выравнивание по вертикали	<TD VALIGN = TOP|MIDDLE| BOTTOM></TD>	

Установка ширины ячейки (в пикселях или %)	<TD WIDTH=«?»></TD>	</TR> </TABLE> **Товар** | **Цена** Радиотелефон | 2000
Заливка цветом ячейки	<TD BGCOLOR = «# цвет»> </TD>	<TD BGCOLOR = «#FF0000»> </TD> красный цвет
Объединение ячеек	<TD ROWSPAN=”ЧИСЛО”></TD>	<tr> <td rowspan="2">1</td> <td>2</td> <td>3</td> </tr> <tr> <td>2</td><td>3</td> </tr> 
Заголовок столбца или строки	<TH>текст </TH>	Текст в ячейке выравнивается по центру, устанавливается жирный шрифт

В результате выполнения работы необходимо знать следующее:

1. Что такое HTML
2. Создание простейшей html страницы
3. Определение тега
4. Основные теги html верстки
5. Способы задания цвета

Содержание отчета

1. Часть 2, пункт 8 – Обновленный документ
2. Часть 2, пункт 9 – Заголовки различных уровней
3. Часть 2, пункт 11 – Отформатированный текст
4. Часть 2, пункт 14 – Выравнивание текста в документе
5. Часть 2, пункт 18 – Создание «смешанного» списка
6. Часть 3, пункт 1 – Создание таблицы
7. Часть 3, пункт 3 – Цветовое оформление таблицы
8. На отдельном листе распечатка всего кода ваших HTML страниц

Оценка	Отлично	Хорошо	Удовлетворительно	Неудовлетворительно
Балл	1 – 0,85	0,84 – 0,71	0,7 – 0,6	менее 0,6

ЛАБОРАТОРНАЯ РАБОТА №6.
РАБОТА В MICROSOFT VISUAL STUDIO COMMUNITY 2015

ЧАСТЬ 1. ОЗНАКОМЛЕНИЕ СО СРЕДОЙ РАЗРАБОТКИ. ОСНОВНЫЕ ЭЛЕМЕНТЫ ИНТЕРФЕЙСА ПРИ СОЗДАНИИ ПРИЛОЖЕНИЯ

1. Включить компьютер и загрузить **Windows**.

2. На рабочем столе запустить средство разработки ПО **Microsoft Visual Studio Community 2015**. На экране окно, изображенное на рисунке 1.

Microsoft Visual Studio – линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств.

Visual Studio можно использовать для создания различных типов приложений, от простых приложений для магазина и игр для мобильных клиентов до больших и сложных систем, обслуживающих предприятия и центры обработки данных. Вы можете создавать:

1. приложения и игры, которые выполняются не только на платформе Windows, но и на Android и iOS;
2. веб-сайты и веб-службы на основе ASP.NET, JQuery, AngularJS и других популярных платформ;
3. приложения для самых разных платформ и устройств, включая, но не ограничиваясь: Office, Sharepoint, Hololens, Kinect и "Интернета вещей";
4. игры и графические приложения для разных устройств Windows, включая Xbox, с поддержкой DirectX.

По умолчанию Visual Studio обеспечивает поддержку C#, C и C++, JavaScript, F# и Visual Basic. Visual Studio хорошо работает и интегрируется со сторонними приложениями.

Отдельные файлы можно изменить в текстовом редакторе, встроенном в Visual Studio. Для открытия файла необходимл последовательно выбрать **Файл | Открыть | Файл.** Весь код в файле выделяется цветом. Для настройки цветовой схемы введите "Цвета" на панели быстрого запуска. Одновременно можно открыть большое число окон текстового редактора в виде вкладок. Текстовый редактор также можно запустить в полноэкранном режиме.

3. На рис.1 представлено окно средства разработки ПО **Microsoft Visual Studio Community 2015** сразу после ее запуска.

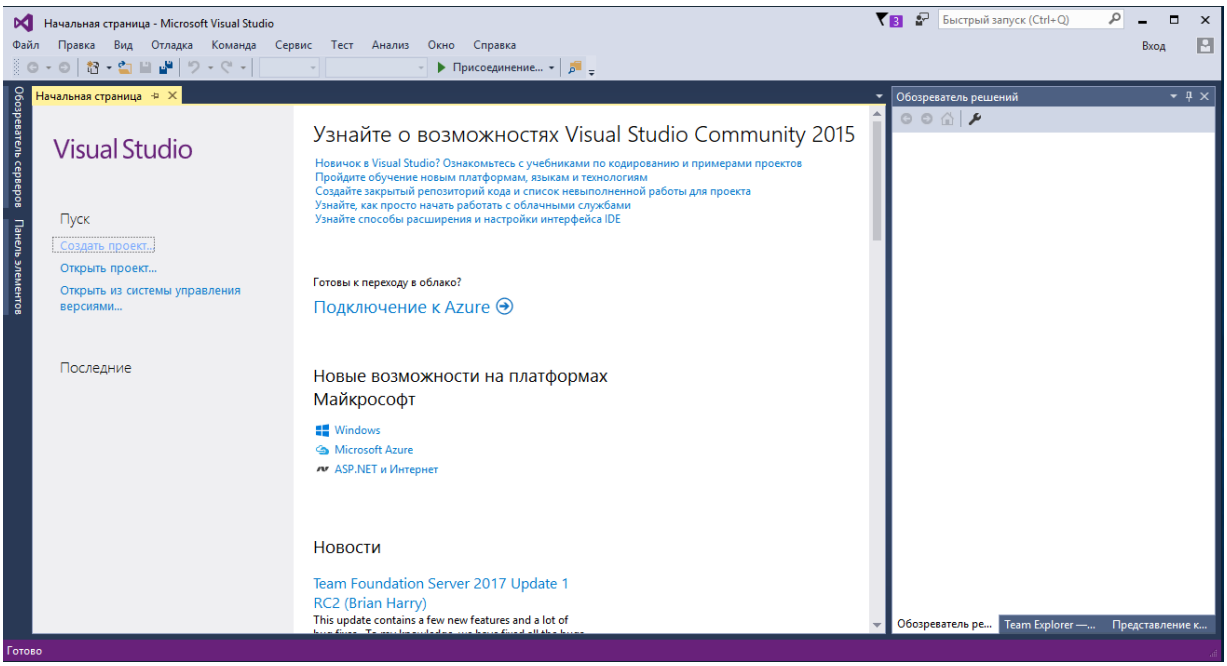


Рис. 1. Окно средства разработки ПО **Microsoft Visual Studio Community 2015**

4. В верхней части окна находятся панели инструментов, кнопки и другие элементы, которые существенно ускоряют доступ к функциям программы. В левой части окна расположен список. В нем перечислены функции создания новых и открытия уже созданных проектов. Ниже будут показаны ранее созданные проекты

5. Нажмите на надпись **Создать проект**, чтобы открыть окно установленных шаблонов, показанное на рис. 2.

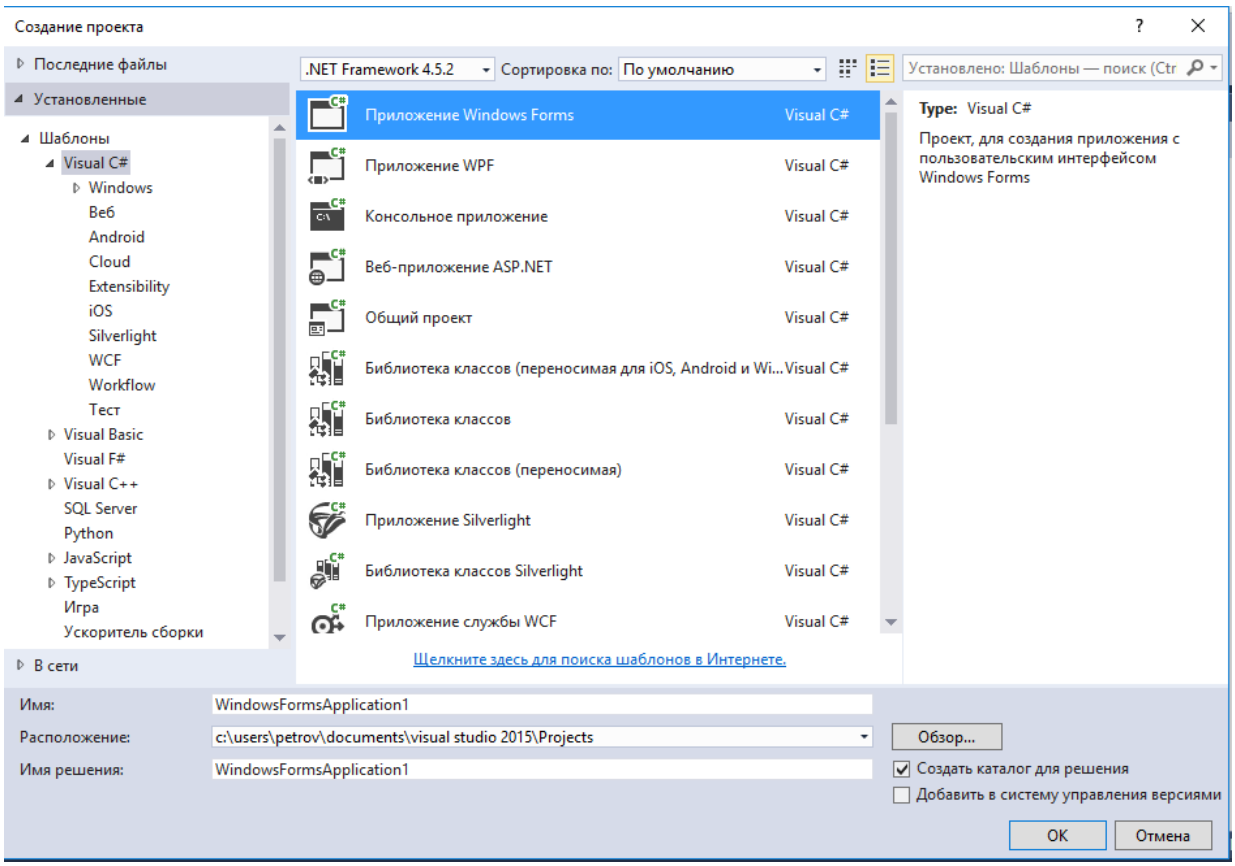


Рис. 2. Окно выбора установленных шаблонов

6. Разработка приложений ведется на языке Объектно-Ориентированного Программирования **C#**, который прост в освоении и позволяет разрабатывать сложные программные продукты.

7. В зависимости от выбранного вами шаблона средство разработки ПО **Microsoft Visual Studio Community 2015** подготовит проект для начала разработки программного обеспечения:

✓ **Приложение Windows Forms** - интерфейс программирования приложений (**API**), отвечающий за графический интерфейс пользователя и являющийся частью **Microsoft.NET Framework**. Данный интерфейс упрощает доступ к элементам интерфейса **Microsoft Windows** за счет создания обёртки для существующего **Win32 API** в управляемом коде. Приложение **Windows Forms** представляет собой событийно-ориентированное приложение, поддерживаемое **Microsoft .NET Framework**. В отличие от пакетных программ, большая часть времени тратится на ожидание от пользователя каких-либо действий, как, например, ввод текста в текстовое поле или клика мышкой по кнопке.

✓ **Приложение WPF (Windows Presentation Foundation)** - В основе данного шаблона лежит векторная система визуализации, не зависящая от разрешения устройства вывода и созданная с учётом возможностей современного графического оборудования. **WPF** предоставляет средства для создания визуального интерфейса, включая язык **XAML (Extensible Application Markup Language)**, элементы управления, привязку данных,

макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление. Графической технологией, лежащей в основе **WPF**, является **DirectX**, в отличие от **Windows Forms**, где используется **GDI/GDI+**. Производительность **WPF** выше, чем у **GDI+** за счёт использования аппаратного ускорения графики через **DirectX**.

✓ **Консольное приложение.** Данный тип шаблона позволяет создать приложение, которое принимает входные и отправляет выходные данные на консоль, которая также называется командной строкой. Можно создать консольные приложения для выполнения различных задач - от простейших до самых сложных. Консольное приложение можно также использовать для демонстрации функциональных возможностей, которые позже будут реализованы в приложении **Windows**.

✓ **Библиотека классов**. Данный шаблон позволяет создать библиотеку, выполняющую определенную функцию, которая далее будет использоваться в различных приложениях.

✓ **Приложения обозревателя WPF(XBAP)** объединяет возможности веб-приложений и многофункциональных клиентских приложений. Как веб-приложения, **XBAP** можно разворачивать на веб-сервере и запускать из браузера. Как многофункциональные клиентские приложения, **XBAP** могут воспользоваться преимуществами возможностей **WPF**. Разработка **XBAP** также аналогична многофункциональной клиентской разработке. В этой теме предоставляется простое, высокоуровневое введение в разработку **XBAP**, а также отмечается, где разработка **XBAP** отличается от стандартной многофункциональной клиентской разработки.

✓ **Пустой проект.** Создается пустой проект, с возможностью интеграции любого из перечисленных методов написания приложений.

8. В графе **Имя проекта** задается имя проекта, и разрабатываемого вами приложения.

ЗАДАНИЕ 1. СОЗДАНИЕ И ПОДГОТОВКА ПРОЕКТА

1. Запустите средство разработки ПО **Microsoft Visual Studio Community 2015.**

2. Нажмите кнопку **Создать проект**. Из предоставленных шаблонов выберите **Приложение Windows Forms**, задайте имя вашему проекту, указав **Имя**, **Фамилию** и **Группу**. ***Зафиксировать результат.***

Перед вами откроется окно работы с вашим проектом вашего проекта вида, изображенного на рис. 3

В центральной части расположена **Область редактирования**, на которой отображается внешний вид разрабатываемого приложения.

Слева находится **Панель элементов,** на ней расположены основные элементы управления, которые могут использоваться при дальнейшей разработке.

В правом верхнем углу располагается **Обозреватель решений**. В одном проекте может находиться несколько одновременно разрабатываемых приложений и их структура.

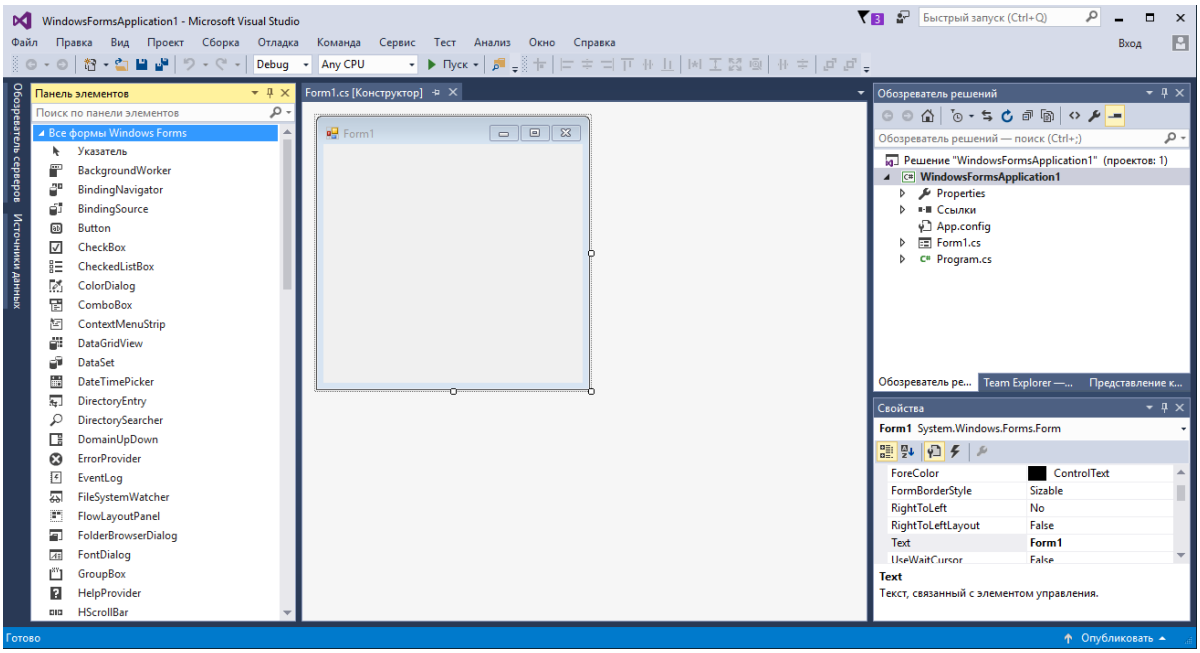


Рис. 3. Окно средства разработки ПО **Microsoft Visual Studio Community 2015**

Ниже находится панель **Свойства**, на которой отображаются свойства разрабатываемого приложения и\или элемента управления.

3. Нажмите левой кнопкой на будущее окно вашего приложения в области редактирования. И начните набирать название окна для вашего будущего приложения. На рис.4 показана область, в которой будет происходить изменение названия.

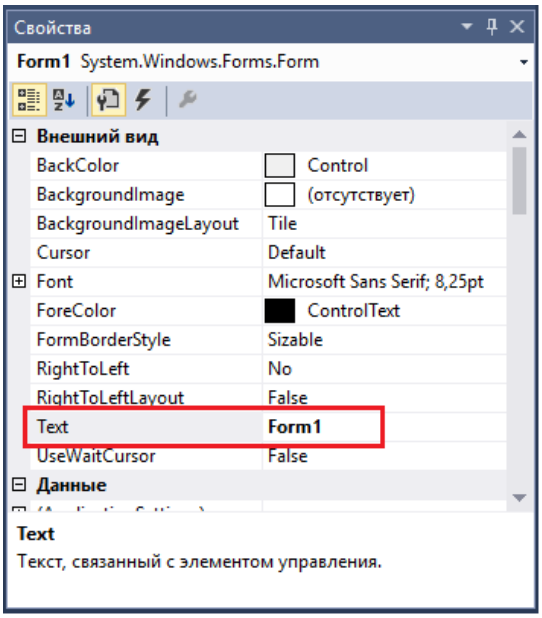


Рис. 4. Название окна разрабатываемого приложения

4. Из панели элементов, указанной на рис.5 перетащить на рабочее окно приложения элемент **Button**, после чего сразу начать набирать текст, который будет отображаться на кнопке. ***Зафиксировать результат.***

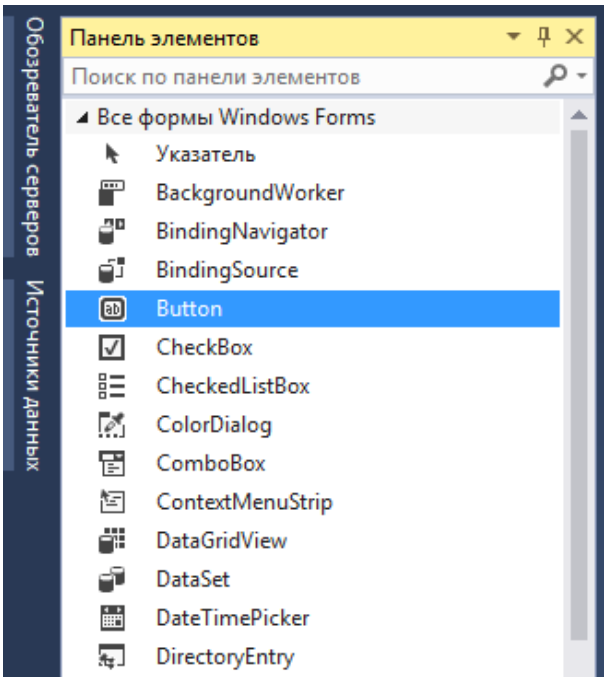


Рис. 5. Элемент управления Button

5. Точно так же поступить с элементом управления **TextBox,** оставить поле пустым. Расположить все элементы в любом положении на рабочем поле приложения. ***Зафиксировать результат.***

Примерный вид приложения изображен на рис. 6.

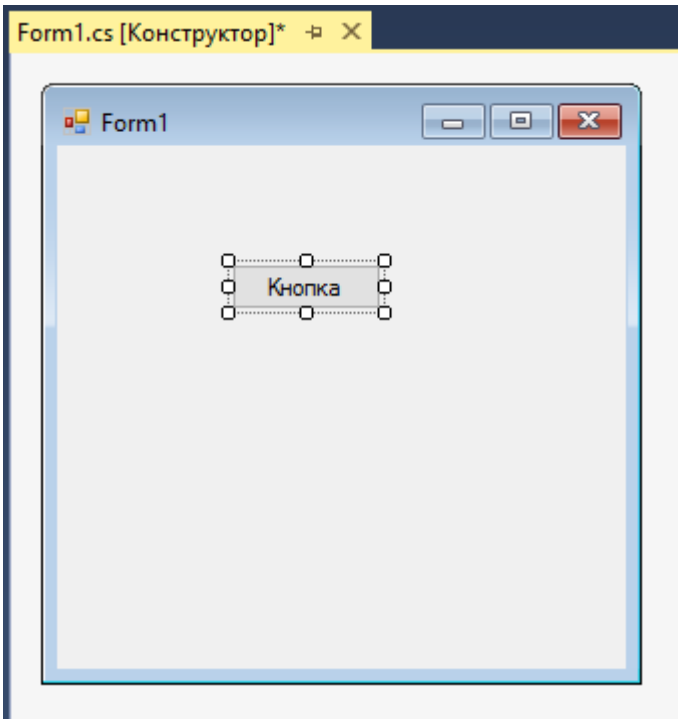


Рис. 6. Вид разрабатываемого приложения

Примечание! Все элементы управления, перенесенные в ***Область редактирования*** *имеют свою ссылку. Например, кнопка имеет ссылку* ***button1****, а текстовое поле* ***textBox1****. Данные ссылки можно изменить в окне свойства.*

6. Двойной клик на элементе управления **Кнопка** откроет редактор исходного кода программы. Автоматически создался обработчик события при нажатии кнопки. Если написать исполняемый код внутри этого обработчика, между **{**и**}**, то выполняться он будет только после нажатия на кнопку.

7. Напишите внутри обработчика нажатия кнопки текст:

textBox1.Text = “Любой текст”;

и

MessageBox.Show(textBox1.Text);

Примечание! Точка с запятой в конце строчки обязательна!

8. Волнистые линии предупреждают об ошибках или потенциальных проблемах в коде в режиме реального времени по мере ввода, что позволяет исправлять их немедленно, не дожидаясь обнаружения ошибок во время компиляции или выполнения. Если навести указатель мыши на волнистую линию, на экран будут выведены дополнительные сведения об ошибке. Кроме того, в поле слева может появляться значок лампочки с предложениями по устранению ошибки. Пример ошибки в написании кода представлен на рисунке 7.

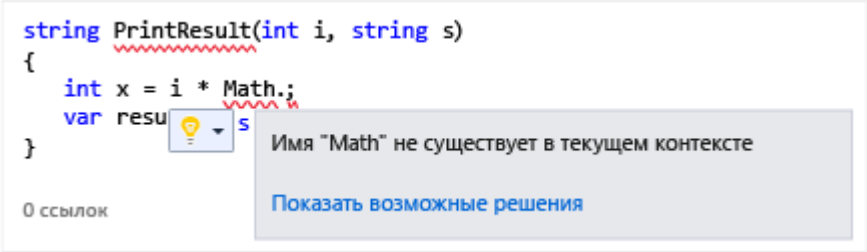


Рис. 7. Пример ошибки в написании кода

После проект необходимо собрать и запустить, выполнить это действие можно нажав кнопку, изображенную на рисунке 8.

Сборка проекта означает компиляцию исходного кода и выполнение действий, необходимых для создания исполняемого файла. В разных языках предусмотрены разные операции сборки, а для обычных веб-сайтов сборка вообще не выполняется. Но независимо от типа проекта меню "Сборка" — это стандартное расположение этих команд. Чтобы скомпилировать и запустить код одним нажатием клавиши, воспользуйтесь клавишей F5.

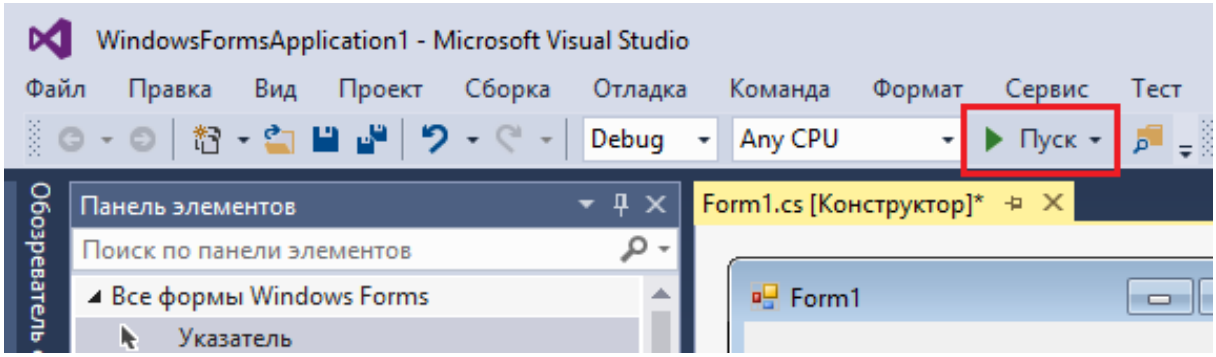


Рис. 8. Кнопка компиляции и запуска

8. После запуска вашего приложения наберите любой текст и нажмите на элемент управления **Button.** ***Зафиксировать результат.***

ЧАСТЬ 2. ОСНОВЫ ЯЗЫКА C#

1. Переменные в **C#** объявляются с определенным типом данных и имени. Первым делом необходимо указать тип переменной: **int, string, double, bool** или другой из более чем 20 различных типов данных. Тип указывает, помимо всего прочего, точный объем памяти, который следует выделить для хранения значения при выполнении приложения. При преобразовании переменной из одного типа в другой язык **C#** следует определенным правилам. Общий вид написания переменной: **Тип Имя = Значение**.

Пример:
int answer = 42; (тип – целое число, имя – answer, значение – 42)
string greeting = "Hello, World!"; (тип – строка, имя – greeting, значение – Hello, World!)
double bigNumber = 3,7; (тип – дробное число, имя – bigNumber, значение – 3,7, с точностью до 16 знаков)
float Number = 1,1; (тип – дробное число, имя – Number, значение – 1,1, с точностью до 7 знаков)

Примечание! Строковые значения указываются в кавычках “и”.

2. Операторы. Самыми простыми операторами в **C#** служат арифметические функции, которые указаны в таблице 1.

Таблица 1

Простые операторы C#

Оператор	Назначение
=	Присваивание значения.
==	Проверка на равенство.
*, /, %	Умножение, деление, остаток от деления

+, -	Сложение, вычитание
&	Логическое И
^	Логическое исключающее ИЛИ
|	Логическое ИЛИ
v++	Увеличение переменной v на 1.
v+=n	Увеличение переменной v на n.
v*=n	Умножение переменной v на n.
v-=n	Вычитание n из переменной v.
!=	Проверка на неравенство.
>	Больше.
<	Меньше.
>=	Больше или равно.
<=	Меньше или равно.
&&	Условное И.
||	Условное ИЛИ.
!	Условное НЕТ.

Пример:

```
int x = 100;
if (x == 100)
{
   MessageBox.Show("X равно 100");
}
```

Целочисленной переменной **X** присваивается значение 100, далее идет проверка, с помощью функции **if** и оператора проверки на равенство, что если **X** равняется **100**, то вывести соответствующее сообщение.

```
int x = 10;
int y = 15;
if (x>y)
{
   MessageBox.Show("X больше Y");
}
else
{
   MessageBox.Show("X меньше Y");
}
```

Вводим две переменные **X** и **Y**, с произвольными значениями, дальше проводим проверку, и **если X** больше **Y**, то в диалоговом окне выведется соответствующая запись. Функция **else** обозначает действие **иначе**, и если значение **X** меньше, то в диалоговом окне выведется, что **X** меньше **Y**.

Ниже приведен пример арифметических действий в **C#.**

```
int x = 1;
int y = x + 10 * 100;
int z = (x + 10) * 100;
```

3. Массивы являются коллекциями объектов одного типа. Поскольку длина массивов практически не ограничена, они могут использоваться для хранения тысяч или даже миллионов объектов, но размер массива должен быть указан при его создании.

Массив является индексированной коллекцией объектов. Отсчёт индекса начинается с нуля. Одномерный массив объектов объявляется следующим образом: **Тип[] Имя = {Значение, Значение, Значение и т.д.}.**

Пример:

```
int[] array = {1, 3, 5, 7, 9};
string[] days = {"Пн", "Вт", "Ср", "Чт", "Пт", "Сб", "Вс"};
```

Примечание! Обращение к первому элементу массива выглядит следующим образом: ***days[0]****, если вывести в диалоговое окно* ***MessageBox.Show(days[0])****, то результатом, в диалоговом окне отобразится* ***Пн.***

int l = array.Length; - результатом выполнения данного кода будет присвоения переменной l значения длины массива(количество элементов в массиве).

4. Оператор foreach часто используется для доступа к каждому элементу, хранимому в массиве.

Пример:

```
int[] numbers = { 4, 5, 6, 1, 2, 3, -2, -1, 0 };
foreach (int i in numbers)
{
   MessageBox.Show(i.ToString());
}
```

Данный код выведет в диалоговом окне поочередно все элементы массива.

Примечание! Функция ***ToString()*** *переводит любое значение в строку, что позволяет отобразить ее в текстовых полях.*

5. Циклом называется один или несколько операторов, повторяющихся заданное число раз или до тех пор, пока не будет

выполнено определенное условие. Выбор типа цикла зависит от задачи программирования и личных предпочтений кодирования.

Пример:

```
int[] array = {0, 1, 2, 3, 4, 5};
for (int i=0; i<array.Length; i++)
{
   MessageBox.Show(array[i].ToString());
}
```

6. Комментарии к коду - пояснения к исходному тексту программы, находящиеся непосредственно внутри комментируемого кода. Для комментирования используется символ «//» и последующий непосредственно внутри кода. С точки зрения компилятора или интерпретатора, комментарии - часть текста программы, не влияющая на её семантику. Комментарии не оказывают никакого влияния на результат компиляции программы или её интерпретацию.

Пример:

```
int i = 5; //присвоено значение 5
```

ЗАДАНИЕ 2. НАХОЖДЕНИЕ СРЕДНЕАРИФМЕТИЧЕСКОГО ЗНАЧЕНИЯ ЦЕЛОЧИСЛЕННОГО МАССИВА

1. Создайте целочисленный массив, заполненный произвольными числами и который, по нажатию кнопки выводится в текстовое поле **TextBox**. ***Зафиксируйте результат.***

Пример:

```
int[] array = {0, 1, 2, 3, 4, 5};
for(int i = условие начала; i<условие конца; i++)
{
        textbox(номер).Text += array[i].ToString();
}
```

2. Задать переменную, являющейся суммой всех элементов заданного массива. Для этого необходимо создать цикл, с каждым проходом которого, в данную переменную будет суммироваться каждый элемент массива.

Пример:

```
int[] array = {0, 1, 2, 3, 4, 5};
тип sum = 0;
for(int i = условие начала; i<условие конца; i++)
{
```

```
        sum += array[i];
}
```

3. Для подсчета среднеарифметического значения нам нужно поделить сумму всех элементов массива на длину массива. Для этого необходимо ввести еще одну переменную, внутри которой будет производиться счет.

Пример:
тип переменной answer = sum/длина массива;

4. Добавить в окно программы кнопку, по нажатию которой будет производиться вывод результата подсчета среднеарифметического значения всех элементов массива через диалоговое окно **MessageBox**. ***Зафиксируйте результат.***

ЧАСТЬ 3. СОРТИРОВКА МАССИВА МЕТОДОМ «ПУЗЫРЬКА»

1. Сортировка «пузырьком» - простой алгоритм сортировки. Для понимания и реализации этот алгоритм - простейший, но эффективен он лишь для небольших массивов. Алгоритм состоит из повторяющихся проходов по сортируемому массиву. За каждый проход элементы последовательно сравниваются попарно и, если порядок в паре неверный, выполняется обмен элементов.

2. Проходы по массиву повторяются. N - 1 раз или до тех пор, пока на очередном проходе не окажется, что обмены больше не нужны, что означает - массив отсортирован. Для реализации данного алгоритма понадобятся два цикла, один находится внутри другого. При каждом проходе алгоритма по внутреннему циклу, очередной наибольший элемент массива ставится на своё место в конце массива рядом с предыдущим «наибольшим элементом», а наименьший элемент перемещается на одну позицию к началу массива («всплывает» до нужной позиции как пузырёк в воде, отсюда и название алгоритма).

Пример работы алгоритма:

Возьмём массив с числами «5 1 4 2 8» и отсортируем значения по возрастанию, используя сортировку пузырьком. Выделены те элементы, которые сравниваются на данном этапе.

Первый проход:
(**5 1** 4 2 8) (**1 5** 4 2 8), сравниваются первые элементы и меняется местами.
(1 **5 4** 2 8) (1 **4 5** 2 8), меняет местами, так как 5>4
(1 4 **5 2** 8) (1 4 **2 5** 8), меняет местами, так как 5>2

(1 4 2 **5 8**) (1 4 2 **5 8**), теперь, ввиду того, что элементы стоят на своих местах (8>5), алгоритм не меняет их местами.

Второй проход:
(**1 4** 2 5 8) (**1 4** 2 5 8)
(1 **4 2** 5 8) (1 **2 4** 5 8), Меняет местами, так как 4>2
(1 2 **4 5** 8) (1 2 **4 5** 8)
(1 2 4 **5 8**) (1 2 4 **5 8**)

Теперь массив полностью отсортирован, но алгоритм не знает так ли это. Поэтому ему необходимо сделать полный проход и определить, что перестановок элементов не было.

Третий проход:
(**1 2** 4 5 8) (**1 2** 4 5 8)
(1 **2 4** 5 8) (1 **2 4** 5 8)
(1 2 **4 5** 8) (1 2 **4 5** 8)
(1 2 4 **5 8**) (1 2 4 **5 8**)

Теперь массив отсортирован и алгоритм может быть завершён.

ЗАДАНИЕ 3. СОРТИРОВКА МАССИВА МЕТОДОМ «ПУЗЫРЬКА»

1. Создайте новый проект, и расположите на рабочем поле два элемента **TextBox** и один элемент **Button**.

2. Перетащите два элемента **Label** на рабочее поле, а также подпишите их **«Исходный массив»** и **«Отсортированный массив»**. Расположив эти элементы над каждым из **TextBox.** ***Зафиксируйте результат.***

3. Создайте целочисленный массив с произвольными значениями выведите данный массив по нажатию кнопки в первый **TextBox.**

Для сортировки массива необходимо сравнение двух соседних элементов массива. Сравнение производится при помощи оператора **IF** обращение к двум соседним элементам производится при помощи двух циклов(цикл в цикле).

Пример:

```
int[] array = {7, 2, 1, 3, 5, 4};
for(int i = условие начала; i<условие конца; i++)
{
    for(int j = i+1; j<условие конца; j++)  //i+1 берется соседний от i элемент
    {
        if(array[i]<array[j])
        {
            если условие выполняется, то необходимо поменять элементы местами;
        }
    }
}
```

4. Отсортируйте созданный массив, используя алгоритм сортировки «пузырьком» и выведите отсортированный массив во второй **TextBox**. ***Зафиксируйте результат.***

Примечание! Вывести массив с помощью цикла foreach:

```
foreach (int i in array)
{
   textbox(номер).Text = textbox(номер).Text + " " + i.ToString();
}
```

ЧАСТЬ 4. ОБРАБОТЧИК ИКЛЮЧЕНИЙ
ЗАДАНИЕ 4. ПРИЛОЖЕНИЕ «КАЛЬКУЛЯТОР»

1. Создайте новый проект, и расположите на рабочем поле два элемента **TextBox** и четыре элемента **Button.**

2. Каждому элементу **Button** присвоить имя, соответствующее математическим операциям (сложение, вычитание, умножение и деление).

3. При нажатии на одну из кнопок должна производиться операция с двумя числами, введенными в элементы управления **TextBox1** и **TextBox2.**

Для того, чтобы записать значение **textBox** в переменную необходимо использовать метод **конвертации**.

Пример:

```
Тип A = Convert.To(тип в который конвертировать)(textBox1.Text);
или
double A = Convert.ToDouble(textBox1.Text);
```

4. Ответ должен выводиться через диалоговое окно **MessageBox**.

5. Если в любое из полей ввести не число, а букву, то при выполнении операции появится исключение (ошибка). При возникновении исключения общеязыковая среда выполнения (**CLR**) ищет оператор **catch**, который обрабатывает это исключение. Если блок **catch** не находится, то среда **CLR** отображает пользователю сообщение о необработанном исключении и останавливает выполнение программы. Блок **try** содержит защищенный код, который может вызвать исключение. Этот блок выполняется, пока не возникнет исключение или пока он не будет успешно завершен.

6. Внешний вид конструкции:

```
try{//Исполняемый код, который может выдать ошибку}
catch{//действие, которое выполняется, если выше появилась ошибка.
```

Пример:

```
MessageBox.Show(“Ошибка”);}
```

7. Доработать свой код, добавив обработчик исключений. ***Зафиксировать результат.***

В результате выполнения работы необходимо знать следующее:

1. Создание проекта в среде разработки
2. Основные элементы управления
3. Редактирование свойств основного окна приложения
4. Редактирования свойств элементов управления
5. Основные переменные
6. Основные операторы
7. Массивы и оператор foreach
8. Циклы
9. Обработчик исключений

Содержание отчета

1. Часть 1, задание 1, пункт 2 – Создать новый проект
2. Часть 1, задание 1, пункт 4 – Создать элемент управления Button.
3. Часть 1, задание 1, пункт 5 – Создать элемент управления TextBox.
4. Часть 1, задание 1, пункт 8 – Вывод диалогового окна с текстом и TextBox.
5. Часть 2, задание 2, пункт 1 – Вывод элементов массива в TextBox.
6. Часть 2, задание 2, пункт 4 – Вывод среднеарифметического значения массива в диалоговом окне MessageBox.
7. На отдельном листе распечатка всего исполняемого кода вашей программы
8. Часть 3, задание 3, пункт 2 – Расположение всех элементов приложения.
9. Часть 3, задание 3, пункт 4 – Результат выполнения сортировки массива.
10. Часть 4, задание 4, пункт 7 – Результат выполнения работы калькулятора.
11. На отдельном листе распечатка всего исполняемого кода вашей программы

Оценка	Отлично	Хорошо	Удовлетворительно	Неудовлетворительно
Балл	1 – 0,85	0,84 – 0,71	0,7 – 0,6	менее 0,6

Приложение 1
Титульный лист отчета

МИНОБРНАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение высшего образования
«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт нанотехнологий, электроники и приборостроения

Дисциплина «Информационные технологии»

ОТЧЕТ
о выполнении лабораторной работы № ____

по теме «Работа с Microsoft Visual Studio Community 2015»

*Фамилия, и.о. студента, № группы
_____

*Дата выполнения
_____

*Фамилия, и.о. преподавателя
_____

**Оценка задания _____ (30+50+20=100 максимум)
(уровень теоретических знаний + практический навык + отчет)

**Общая оценка _____ x _____ = _____ _____
сумма баллов | коэффициент | итоговые баллы | оценка | подпись

**Комментарий преподавателя:
_____
_____
_____

Таганрог 201__

* Заполняется студентом
** Заполняется преподавателем

Приложение 2.
Основные теги HTML-документа.
Теги форматирования шрифта и абзаца

Назначение	Вид тегов	Примечание
Общая структура документа HTML		
Тип документа	<HTML></HTML>	Начало и конец документа
Имя документа	<HEAD></HEAD>	Не отображается браузером
Заголовок	<TITLE></TITLE>	Содержимое строки заголовка окна браузера
Тело документа	<BODY></BODY>	Содержимое WEB-страницы
Структура содержания документа		
Внутренние заголовки различного уровня	<H№> текст </H№>	Где № – номер уровня заголовка (от 1 до 6). Например, <H1>…</H1> - заголовок 1-го уровня.
Заголовок с выравниванием	<H№ ALIGN=”LEFT| CENTER| RIGHT”> текст </H№>	LEFT - по левому краю, CENTER - по центру, RIGHT - по правому краю.
Форматирование абзацев		
Создание абзаца (параграфа)	<P> текст </P>	Абзацы отделяются двойным межстрочным интервалом
Перевод строки внутри абзаца	 	Одиночный тег
Выравнивание абзаца	<P ALIGN=”LEFT”>текст </P> <P ALIGN=”CENTER”>текст </P> <P ALIGN=”RIGHT”> текст</P> <P ALIGN=”JUSTIFY”> текст </P>	LEFT - по левому краю CENTER - по центру RIGHT - по правому краю JUSTIFY – по ширине
Разделительная горизонтальная линия между абзацами	<HR SIZE=«?»>	Одиночный тег. «?» - толщина линии в пикселях. Толщину линии можно не указывать.
Создание списков		
Нумерованный	<OL>элементы списка</OL>	<OL> <LI>Элемент списка 1 </LI> <LI>Элемент списка 2</LI> <LI>Элемент списка 3</LI> </OL>
Маркированный	<UL> элементы списка </UL>	
Элемент списка	<LI> элементы списка </LI>	

Приложение 3.
Форматирование шрифта

Назначение	Вид тегов	Примечание
Форматирование шрифта		
Жирный	<B> текст </B>	<B>**Жирный**</B> <I> *Курсив* </I> <U> Подчеркнутый </U> <S> ~~Перечеркнутый~~</S> ^{Верхний индекс} _{Нижний индекс}
Курсив	<I> текст </I>	
Подчеркнутый	<U> текст </U>	
Перечеркнутый	<S> текст </S>	
Увеличенный размер	<BIG> текст </BIG >	
Уменьшенный размер	<SMALL> текст </SMALL>	
Верхний индекс	^{текст}	
Нижний индекс	_{текст}	
Размер шрифта	<FONT SIZE=«?»> текст </FONT>	?- значения от 1 до 7 или относительное изменение (например, +2)
Базовый размер шрифта	<BASEFONT SIZE=«?»>	Одиночный тег ? – размер от 1 до 7; по умолчанию равен 3 и задается для всего документа в целом
Гарнитура шрифта	<FONT FASE=«название1, название2, …»> текст </FONT>	Текст оформляется первым, установленным на компьютере шрифтом из списка названий
Цвет шрифта	<FONT COLOR=”$$$$$$”> текст </FONT>	Цвет задается либо ключевым словом, либо шестнадцатеричным кодом с символом # RED –красный, #FF0000 – шестнадцатеричный код – красного цвета

Приложение 4.
Коды основных цветов

Цвет	Color's name	Шестнадцатеричный код цвета		
		Red	Green	Blue
Черный	black	00	00	00
Темно-синий	navy	00	00	80
Голубой	blue	00	00	FF
Зеленый	green	00	80	00
Темно-зеленый	teal	00	80	80
Салатный	lime	00	FF	00
Бледно-голубой	aqua	00	FF	FF
Вишневый	maroon	80	00	00
Фиолетовый	purple	80	00	80
Оливковый	olive	80	80	00
Серый	gray	80	80	80
Светло-серый	silver	C0	C0	C0
Красный	red	FF	00	00
Лиловый	fushsia	FF	00	FF
Желтый	yellow	FF	FF	00
Белый	white	FF	FF	FF

СОДЕРЖАНИЕ

1. РАБОТА В MICROSOFT VISUAL STUDIO COMMUNITY 2015……..…….....3

ЧАСТЬ 1. ОЗНАКОМЛЕНИЕ СО СРЕДОЙ РАЗРАБОТКИ. ОСНОВНЫЕ ЭЛЕМЕНТЫ ИНТЕРФЕЙСА ПРИ СОЗДАНИИ ПРИЛОЖЕНИЯ …….....……3

ЗАДАНИЕ 1. СОЗДАНИЕ И ПОДГОТОВКА ПРОЕКТА……..5

ЧАСТЬ 2. ОСНОВЫ ЯЗЫКА C#……..……...…….9

ЗАДАНИЕ 2. НАХОЖДЕНИЕ СРЕДНЕАРИФМЕТИЧЕСКОГО ЗНАЧЕНИЯ ЦЕЛОЧИСЛЕННОГО МАССИВА……12

ЧАСТЬ 3. СОРТИРОВКА МАССИВА МЕТОДОМ «ПУЗЫРЬКА»..12

ЗАДАНИЕ 3. СОРТИРОВКА МАССИВА МЕТОДОМ «ПУЗЫРЬКА».....14

ЧАСТЬ 4. ОБРАБОТЧИК ИКЛЮЧЕНИЙ…….……..…….14

ЗАДАНИЕ 4. ПРИЛОЖЕНИЕ «КАЛЬКУЛЯТОР»……...16

2. ПРИЛОЖЕНИЕ 1……..21

3. ПРИЛОЖЕНИЕ 2

4. ПРИЛОЖЕНИЕ 3

5. ПРИЛОЖЕНИЕ 4

Вишневецкий Вячеслав Юрьевич
Старченко Ирина Борисовна
Ледяева Валерия Сергеевна
Петров Игорь Андреевич

Методическое руководство

к выполнению лабораторных работ
по курсу
Информационные технологии

Работа с основами HTML и Microsoft Visual Studio Community 2015

Для студентов направления 12.03.01, 12.03.04, 21.03.02, 17.03.01, 26.05.04, 12.04.01, 12.04.04

Ответственный за выпуск	Вишневецкий В.Ю.
Редактор	Надточий З.И.
Корректор	

Подписано в печать
Формат 60x84 1/16. Печать офсетная.
Бумага офсетная.
Усл.п. л. – 2,5 Уч.–изд. л. – 2,25
Заказ № Тираж 100 экз.

Издательство Южного федерального университета
344091, г. Ростов-на-Дону, пр. Стачки, 200/1.
Тел. (863)2478051.
Отпечатано в Отделе полиграфической, корпоративной и сувенирной продукции ИПК КИБИ МЕДИА ЦЕНТРА ЮФУ.
ГСП 17А, Таганрог, 28, Энгельса,1.
Тел. (8634)371717, 371655.
33