Главное управление образования Гомельского облисполкома

Учреждение образования

«Гомельский государственный машиностроительный колледж»

Учебная дисциплина

«Конструирование программ и языки программирования»

**Инструкция**

по выполнению лабораторной работы №1

«Создание, компилирование, отладка и выполнение проектов

в интегрированной среде разработки»

Гомель 2018

Составитель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Гаврилова В.М.

Обсуждено и одобрено на заседании цикловой комиссии «Программное обеспечение информационных технологий»

Протокол от «01» сентября 2018 № 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.Ю. Лавская

**Лабораторная работа №1**

**Тема работы: «Создание, компилирование, отладка и выполнение проектов в интегрированной среде разработки»**

1. **Цель работы**

Обучиться созданию, компиляции, отладке и выполнению проектов в интегрированной среде разработки.

1. **Задание**
2. Выполните задания на основе рассмотренной программы.

2 Напишите программу, которая выводила бы на консоль название специальности, на которой Вы учитесь, номер группы, Ваши фамилию, имя и отчество в разных строках дисплея (консоли).

1. Проверьте программу без ключевого слова **void** для функции **main()**.
2. **Оснащение работы**

Технические средства обучения:

* IBM – совместимый компьютер;

Электронные средства обучения:

* MS Visual Studio;
* MS Office.

1. **Основные теоретические сведения**

Среда разработки Visual Studio.NET предоставляет мощные и удобные средства написания, корректировки, компиляции, отладки и запуска приложений, использующих .NET-совместимые языки. Корпорация Microsoft включила в платформу средства разработки для четырех языков: С#, VB.NET, С++ и J#.

Платформа .NET является открытой средой. Это значит, что компиляторы для нее могут поставляться и сторонними разработчиками. К настоящему времени разработаны десятки компиляторов для .NET, например, Ada, COBOL, Delphi, Eiffel, Fortran, Lisp, Oberon, Perl и Python.

Приложение в процессе разработки называется проектом. Проект объединяет все необходимое для создания приложения: файлы, папки, ссылки и прочие ресурсы. Среда Visual Studio.NET позволяет создавать проекты различных типов, например:

**•** Windows-приложение - использует элементы интерфейса Windows, включая формы, кнопки, флажки и пр.;

**•** консольное приложение - выполняет вывод «на консоль», то есть в окно командного процессора;

**•** библиотека классов - объединяет классы, которые предназначены для использования в других приложениях;

**•** веб-сервис - компонент, методы которого могут вызываться через Интернет.

Консольные приложения наилучшим образом подходят для изучения языка, так как в них не используется множество стандартных объектов, необходимых для создания графического интерфейса.

Visual Studio 2010 доступна в следующих вариантах:

**•** Express - бесплатная среда разработки, включающая только базовый набор возможностей и библиотек;

**•** Professional - поставка, ориентированная на профессиональное создание программного обеспечения, и командную разработку, при которой созданием программы одновременно занимаются несколько человек;

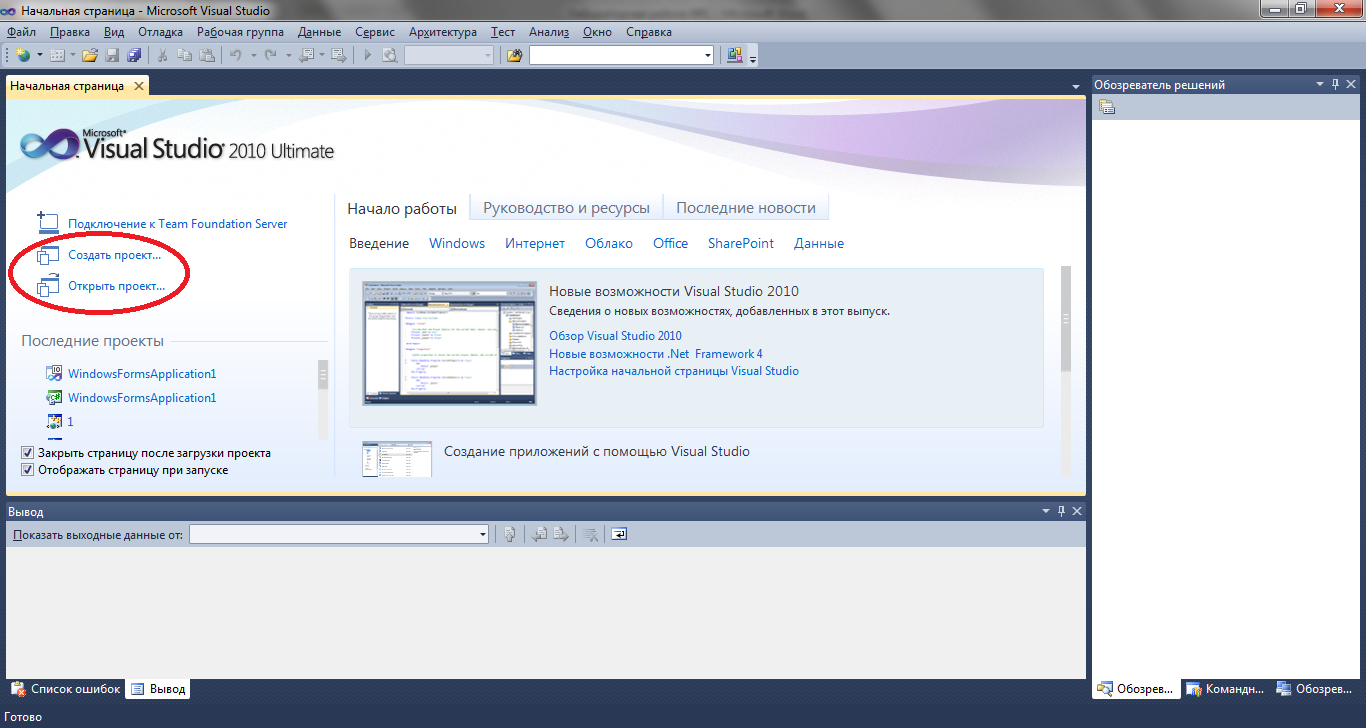
**•** Premium - издание, включающее дополнительные инструменты для работы и исходным кодом программ и создания баз данных;

**•** Ultimate - наиболее полное издание Visual Studio, включающие все доступные инструменты для написания, тестирования, отладки и анализа программ, а также дополнительные инструменты для работы с базами данных и проектирования архитектуры ПО.

Отличительной особенностью среды Microsoft Visual Studio 2010 является то, что она поддерживает работу с несколькими языками программирования и программными платформами. Поэтому, перед тем, как начать создание программы на языке С#, необходимо выполнить несколько подготовительных шагов по созданию проекта и выбора и настройки компилятора языка С# для трансляции исходного кода.

1. **Порядок выполнения работы**

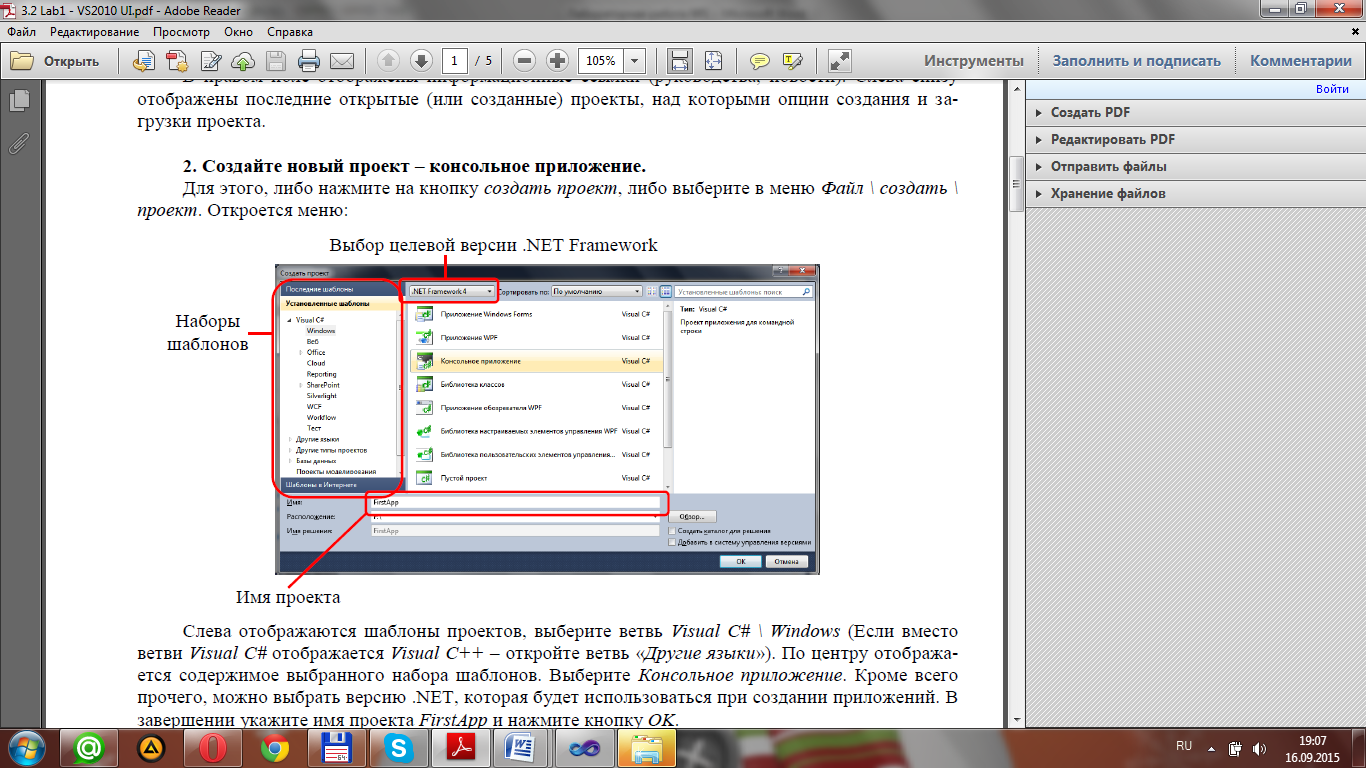
**Запустите среду разработки:** меню *Пуск \ Программы \ Microsoft Visual Studio 2010 \Microsoft Visual Studio 2010*. Появится следующее окно:



В правом поле отображены информационные ссылки (руководства, новости). Слева снизу отображены последние открытые (или созданные) проекты, над которыми опции создания и загрузки проекта.

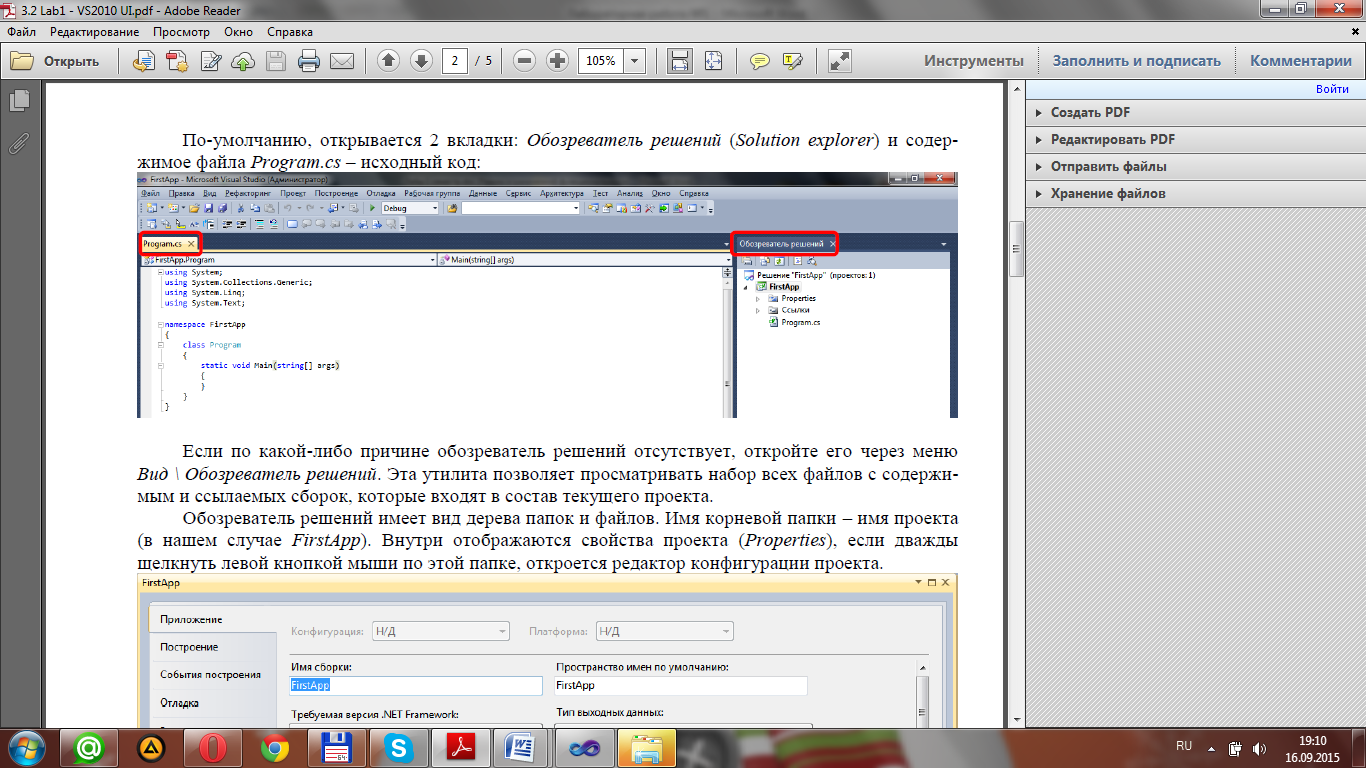
**2. Создайте новый проект – консольное приложение.**

Для этого, либо нажмите на кнопку *создать проект*, либо выберите в меню *Файл \ создать \ проект*. Откроется меню:



Слева отображаются шаблоны проектов, выберите ветвь *Visual C# \ Windows* (Если вместо ветви *Visual C#* отображается *Visual C++* – откройте ветвь «*Другие языки*»). По центру отображается содержимое выбранного набора шаблонов. Выберите *Консольное приложение*. Кроме всего прочего, можно выбрать версию .NET, которая будет использоваться при создании приложений. В завершении укажите имя проекта *FirstApp* и нажмите кнопку *OK*.

По-умолчанию, открывается 2 вкладки: *Обозреватель решений* (*Solution explorer*) и содержимое файла *Program.cs* – исходный код:



Если по какой-либо причине обозреватель решений отсутствует, откройте его через меню *Вид \ Обозреватель решений*. Эта утилита позволяет просматривать набор всех файлов с содержимым и ссылаемых сборок, которые входят в состав текущего проекта.

Обозреватель решений имеет вид дерева папок и файлов. Имя корневой папки – имя проекта (в нашем случае *FirstApp*). Внутри отображаются свойства проекта (*Properties*), если дважды щелкнуть левой кнопкой мыши по этой папке, откроется редактор конфигурации проекта.



В этом окне можно устанавливать различные параметры безопасности, назначать сборке имя, развертывать приложение, выставлять необходимые для приложения ресурсы и конфигурировать события, которые должны происходить перед и после компиляции сборки.

*На данном этапе менять ничего не надо.*

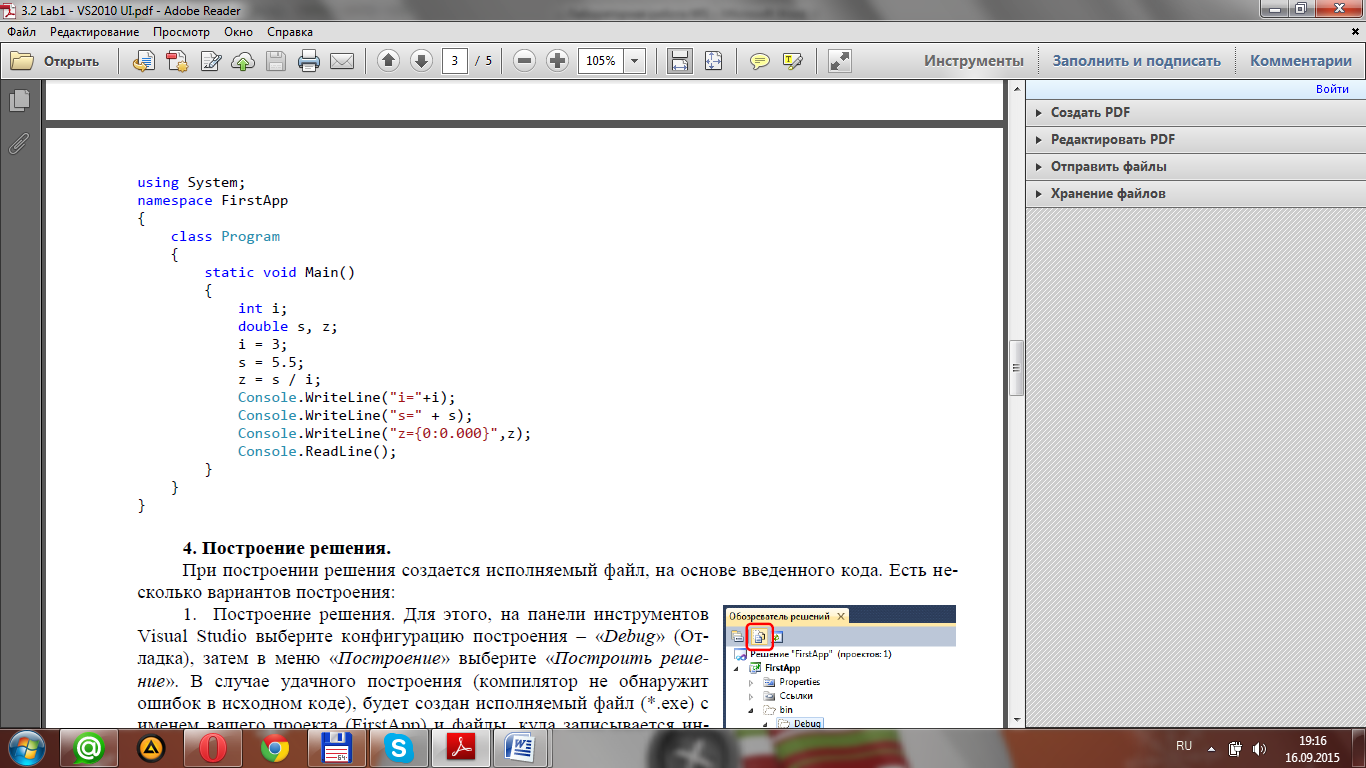
Если раскрыть папку свойств (нажав на треугольник слева от имени папки), получим доступ к файлу – «AssemblyInfo.cs», в котором содержится информация о сборке.

Кроме того, в проекте присутствует папка «Ссылки» (*References*), внутри которой отображается список всех сборок, на которые в проекте были добавлены ссылки. В зависимости от типа проекта и целевой версии .Net Framework, этот список выглядит по-разному.

Ну и последний элемент – файл с исходным кодом приложения «*Program.cs*». Если по какой-либо причине вкладка (или окно) с исходным кодом не отображается, просто дважды щелкните мышкой по этому файлу.

**3. Модифицируйте исходный код.**

Поскольку в нашем случае, объекты пространств имен *System.Text, System.Linq, System.Collections.Generic* мы не будем использовать, уберите их. Кроме того, в метод Main() добавьте строки, в соответствии с текстом ниже и сохраните изменения.

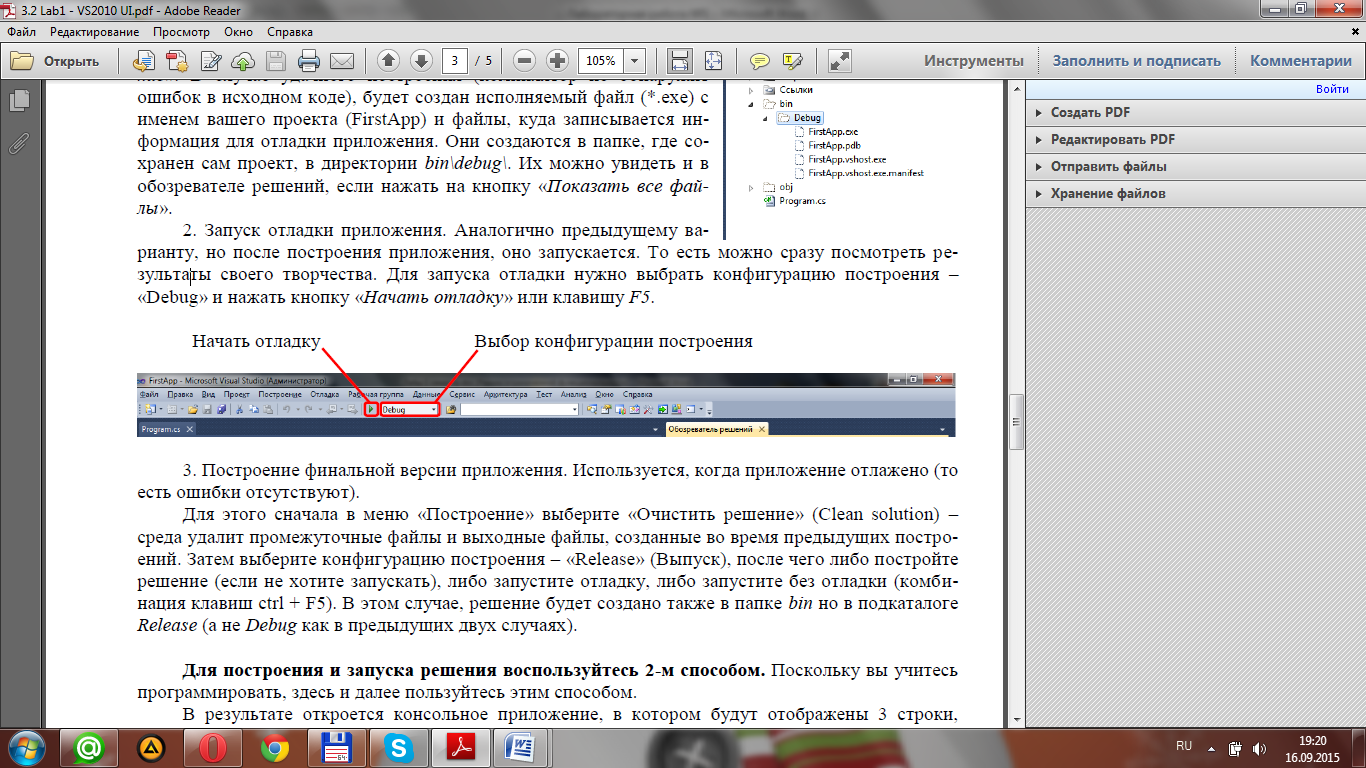


**4. Построение решения.**

При построении решения создается исполняемый файл, на основе введенного кода. Есть несколько вариантов построения:

1. Построение решения. Для этого, на панели инструментов Visual Studio выберите конфигурацию построения – «*Debug*» (Отладка), затем в меню «*Построение*» выберите «*Построить решение*». В случае удачного построения (компилятор не обнаружит ошибок в исходном коде), будет создан исполняемый файл (\*.exe) с именем вашего проекта (FirstApp) и файлы, куда записывается информация для отладки приложения. Они создаются в папке, где сохранен сам проект, в директории *bin\debug\*. Их можно увидеть и в обозревателе решений, если нажать на кнопку «*Показать все файлы*».

2. Запуск отладки приложения. Аналогично предыдущему варианту, но после построения приложения, оно запускается. То есть можно сразу посмотреть результаты своего творчества. Для запуска отладки нужно выбрать конфигурацию построения – «Debug» и нажать кнопку «*Начать отладку*» или клавишу *F5*.

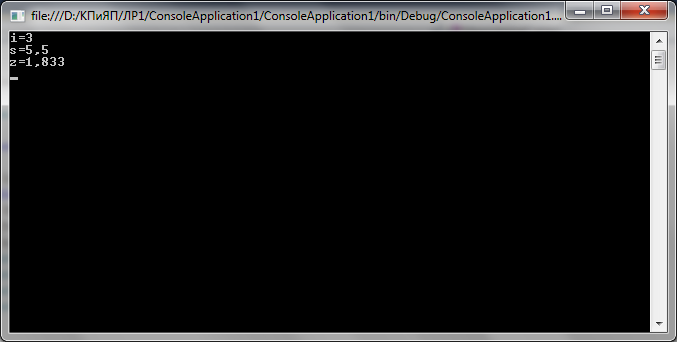


3. Построение финальной версии приложения. Используется, когда приложение отлажено (то есть ошибки отсутствуют).

Для этого сначала в меню «Построение» выберите «Очистить решение» (Clean solution) – среда удалит промежуточные файлы и выходные файлы, созданные во время предыдущих построений. Затем выберите конфигурацию построения – «Release» (Выпуск), после чего либо постройте решение (если не хотите запускать), либо запустите отладку, либо запустите без отладки (комбинация клавиш ctrl + F5). В этом случае, решение будет создано также в папке *bin* но в подкаталоге *Release* (а не *Debug* как в предыдущих двух случаях).

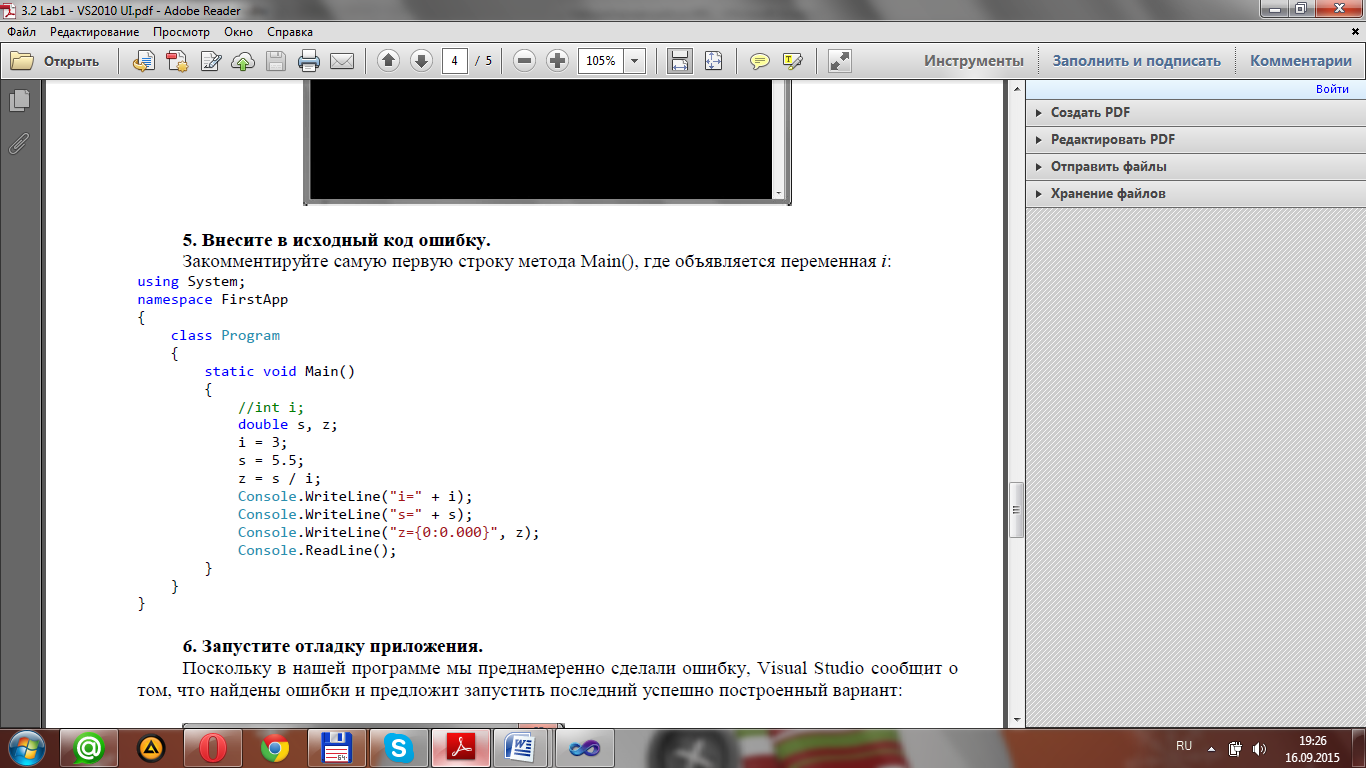
**Для построения и запуска решения воспользуйтесь 2-м способом.** Поскольку вы учитесь программировать, здесь и далее пользуйтесь этим способом.

В результате откроется консольное приложение, в котором будут отображены 3 строки, отображающие содержимое используемых переменных (i, s, z). После нажатия клавиши *Enter*, приложение завершит свою работу.



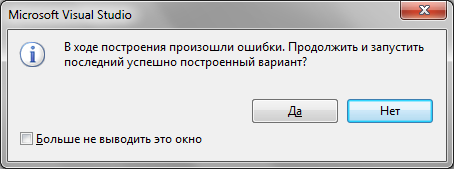
**5. Внесите в исходный код ошибку.**

Закомментируйте самую первую строку метода Main(), где объявляется переменная *i*:

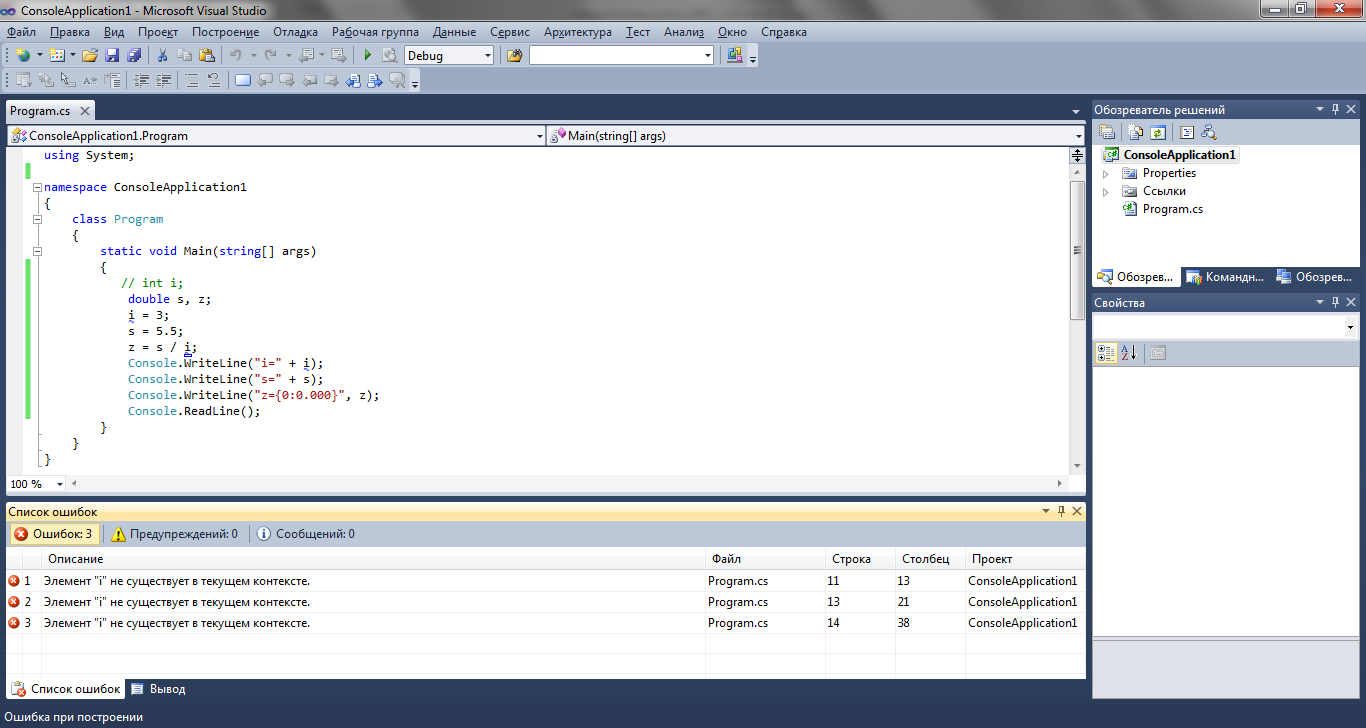


**6. Запустите отладку приложения.**

Поскольку в нашей программе мы преднамеренно сделали ошибку, Visual Studio сообщит о том, что найдены ошибки и предложит запустить последний успешно построенный вариант:



Разумеется, нужно отказаться и просмотреть какие именно ошибки были найдены компилятором. Не переживайте, если сообщение не появилось, а сразу отобразилось окно со списком ошибок. Если же список ошибок не появился, а запустился ранее собранный проект, сначала в меню «*Вид*» выберите «*Список ошибок*», а затем в меню «*Построение*» выберите «*Построить решение*». Рассмотрим список ошибок более подробно.



Для каждой ошибки отображается описание, имя проекта, имя файла, номер строки и столбца, где встретилась эта ошибка в указанном файле. Если дважды щелкнуть левой кнопкой мыши по ошибке, то курсор в файле с исходным кодом будет помещен в указанную строку и столбец. В нашем случае, все три ошибки встречаются там, где используется переменная *i*. И все они являются следствием того, что эта переменная не объявлена (мы эту строку закомментировали). Этот пример показывает, что ошибка находится не всегда в том месте, где указывает ее среда разработки. Поэтому важно уметь анализировать и правильно понимать описание ошибки. В процессе написания программ вы этому научитесь.

Если убрать символы комментария в строке

//int i;

и попытаться построить проект, то ошибок не возникнет.

1. **Форма отчета о работе**

*Номер учебной группы \_\_\_\_\_\_\_\_*

*Фамилия, инициалы обучающегося \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Дата выполнения работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Тема работы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Цель работы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Задание: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Оснащение работы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Результат выполнения работы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

1. **Контрольные вопросы и задания**

1 Какие типы проектов позволяет создавать среда Visual Studio.NET?

1. В каких вариантах доступна Visual Studio 2010?
2. Как создать новый проект?
3. Как отобразить *Обозреватель решений*?
4. Как выполнить отладку проекта?

**Рекомендуемая литература**

**Албахари, Д.** C# 5.0. Справочник. Полное описание языка / Д. Албахари, Б. Албахари : [пер. с англ. Ю.Н.Артеменко]. М., 2013.

**Гриффитс, И.** Программирование на С# 5.0 / И. Гриффитс ; [пер. с англ. И.М. Райтмана]. М., 2014.

**Павловская, Т.А.** C#. Программирование на языке высокого уровня / Т.А. Павловская. СПб., 2015.

**Стиллмен, Э.** Изучаем С# / Э. Стиллмен, Дж. Грин. СПб., 2014.