**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

Факультет прикладной математики и информатики

Кафедра информационных систем управления

Анищенко Арсений Игоревич

**Отчет**

**по учебной практике**

студента 1 курса 2 группы

**Преподаватель**

Баранов Геннадий Аркадьевич

Минск

2017

Содержание

[Лабораторная работа №1. Создание HTML-страницы. 3](#_Toc484854840)

[Лабораторная работа №2. Создание игры “Жизнь”. 5](#_Toc484854841)

[Лабораторная работа №3. Реализация прокрутки таблиц. 6](#_Toc484854842)

[Лабораторная работа №4. Добавление страниц к сайту из л/р №1. 8](#_Toc484854843)

[Лабораторная работа №5. Создания калькулятора на WinApi. 11](#_Toc484854844)

[Лабораторная работа №6. Рисование мышью. 14](#_Toc484854845)

[Лабораторная работа №7. Создание иерархии классов на C#. 17](#_Toc484854846)

[Лабораторная работа №8. Принципы SOLID. 19](#_Toc484854847)

[Лабораторная работа №9. Реализация игры “Змейка”. 19](#_Toc484854848)

[Лабораторная работа №10. Создание Web-приложения. 21](#_Toc484854849)

[Список используемой литературы 23](#_Toc484854850)

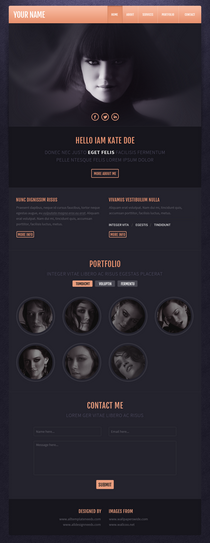
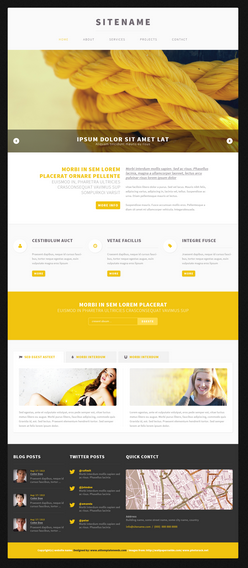
[Приложение 24](#_Toc484854851)

Лабораторная работа №1. Создание HTML-страницы.

**Условие:**

Требовалось создать HTML-страницу по одному из предоставленных примеров, используя отдельный CSS-файл для стилей, не используя таблицы и javascript и стараясь по возможности использовать семантические теги.

Пример предоставленной страницы:



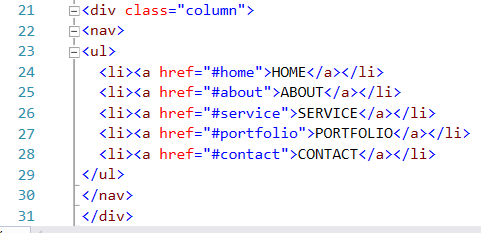


1 Лаб. 1 примеры

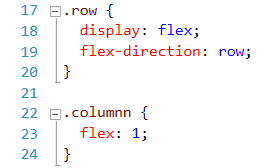
**Решение:**

Для выполнения данного задания я выбрал третий пример из выше предложенных. Для оптимального решения задачи по оформлению была применена технология flexbox. Она служит для создания разметки страницы и помогает без особых усилий распределить объекты на веб-странице.

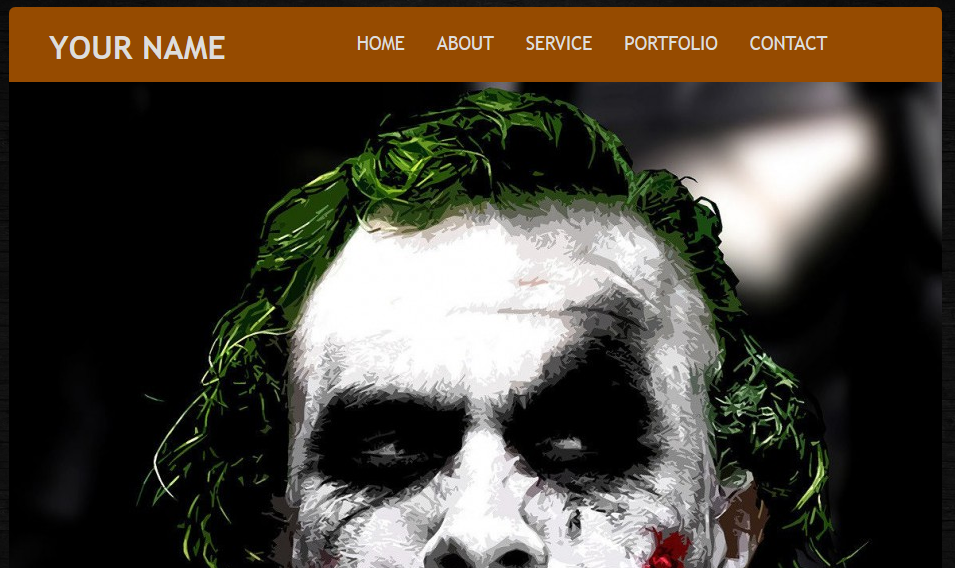
В качестве примера рассмотрим создание навигации на странице.



2. Листинг. Навигация



3 Листинг. Использование flexbox



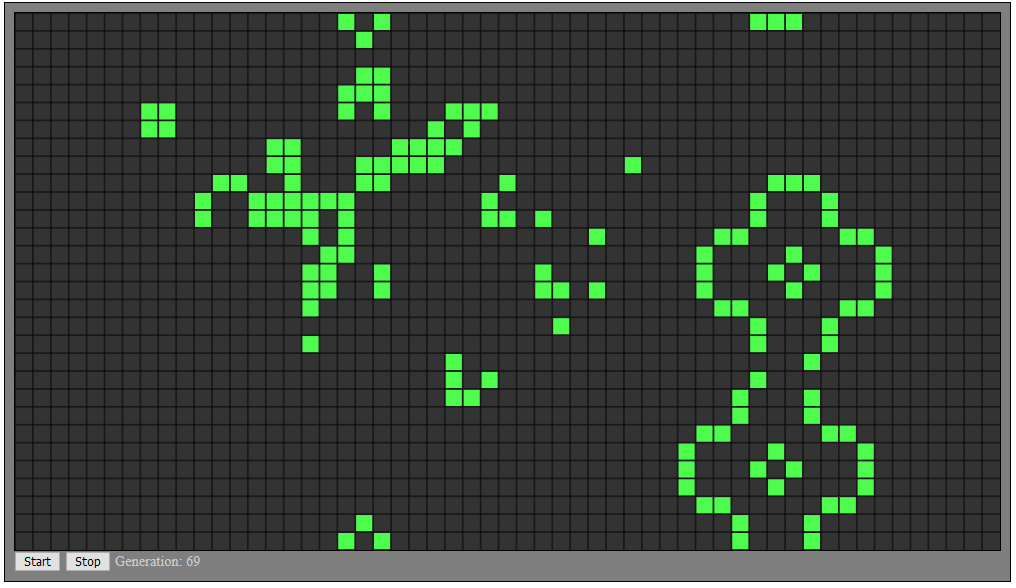
4. Выполненое задание. Навигация

Лабораторная работа №2. Создание игры “Жизнь”.

**Условие:**

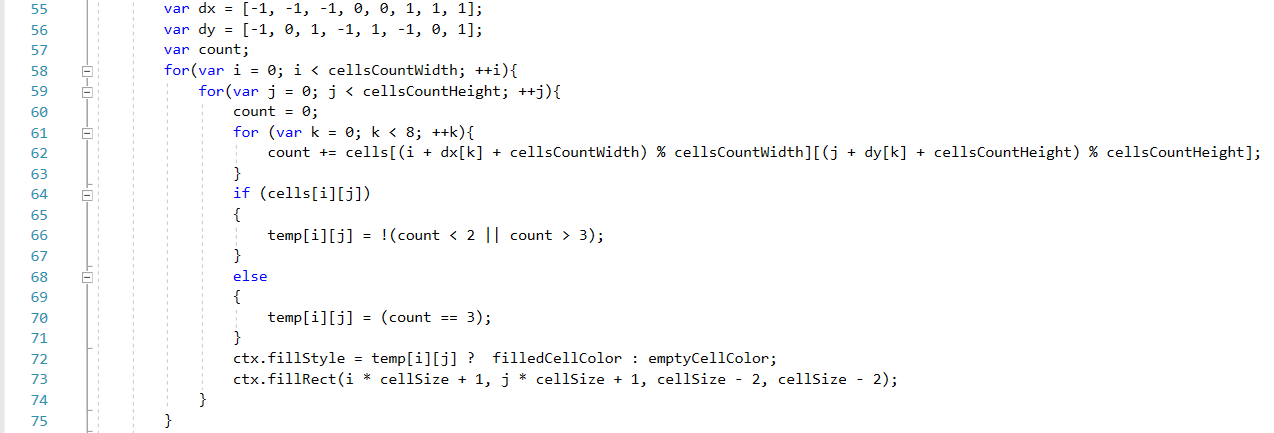
Требовалось написать JS-код для реализации игры “Жизнь” при имеющихся HTML и CSS файлах. Рекомендовалось использовать JQuery.

**Решение:**

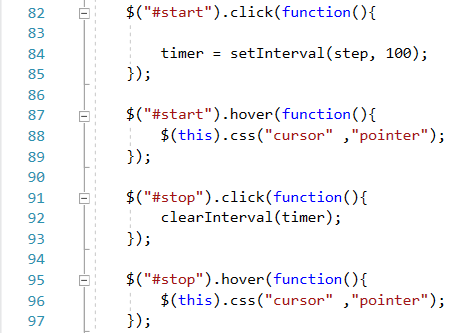


5. Игра "жизнь"

Реализация шага игры. Массивы dx и dy хранят все возможные смещения относительно текущей клетки. Клетка становится «живой», если у нее ровно 3 «живых» соседа и «умирает», если у нее меньше 2х или больше 3х «живых» соседей.



6 Листинг. Реализация шага игры



7 Листинг. Элементы управления

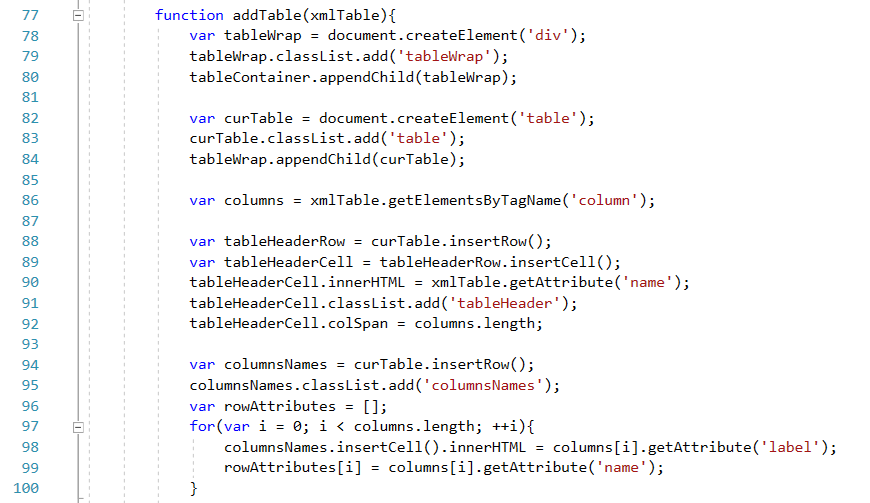
Лабораторная работа №3. Реализация прокрутки таблиц.

**Условие:**

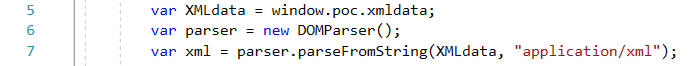
Требовалось с помощью JavaScript сделать HTML-страницу, на которой будут расположены в ряд таблицы. Информацию для таблиц нужно было брать из XML-файла и парсить в JavaScript. При изменении количества или содержимого таблиц страница должна была все равно работать. Должна была быть добавлена возможность прокрутки таблиц, если их слишком много, и они не помещаются на одну страницу. Также при изменении размера окна количество страниц также должно было меняться соответственно. Должна была быть возможность перейти на любую страницу из существующих в любой момент. Требовалось использовать лишь чистый JavaScript без дополнительных библиотек.

**Решение:**

Генерировать разметку html-страницы можно при помощи методов .createElement и .appendChild.



8 Листинг. Создание таблицы



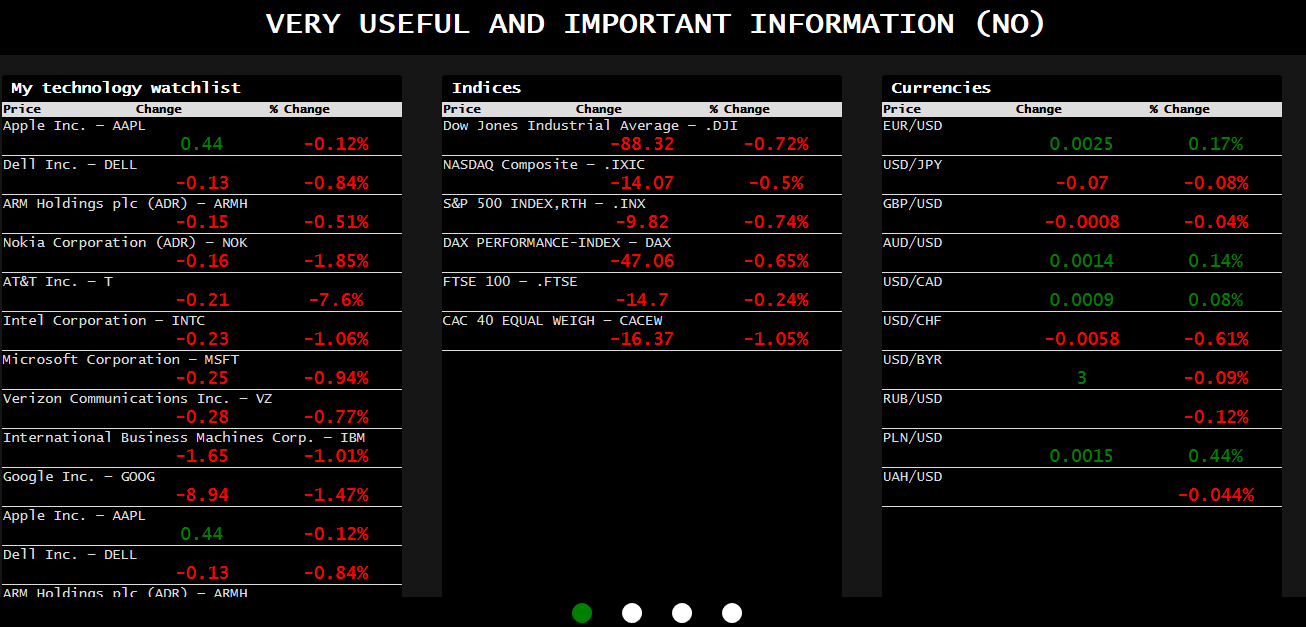
9 Листинг. XMLdata

**Итоговый результат:**

Внизу расположены радио-кнопки для переключения между блоками страниц. Данную функцию можно осуществить при помощи прокрутки мышью. Количество таблиц меняется в зависимости от размера страницы. По центру расположены таблицы, данные в которых получаются из XML-файла с помощью парсера.



10 Таблицы при уменьшении



11 Таблицы при увеличении

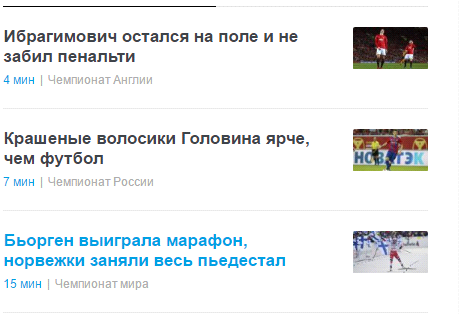
Лабораторная работа №4. Добавление страниц к сайту из л/р №1.

**Условие:**

Требовалось добавить 2 страницы к странице из лабораторной работы №1:

- Первая страница должна представлять из себя простую, статичную HTML-страницу, которую можно открыть из главной страницы через меню.

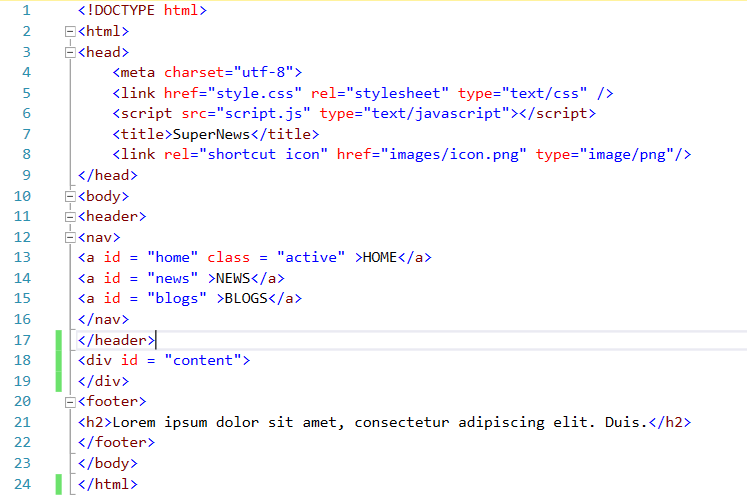
- Вторая страница должна содержать список новостей, по которым можно осуществлять поиск. Новости должны выглядеть примерно таким образом:



Информация для новостей содержалась в JSON.

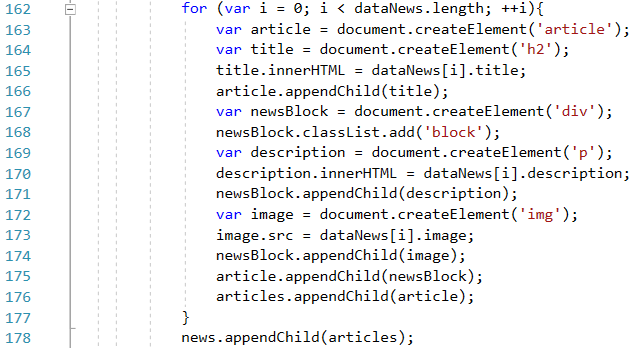
**Решение:**

В данном задании я отошёл от условия и создал новый HTML файл.

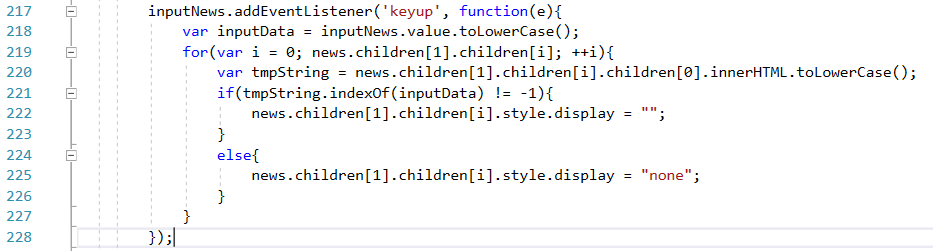


12 Листинг. HTML файл

Не трудно видеть, что его недостаточно для выполнения поставленной задачи. Дело в том, что вся информация генерируется при помощи javascript и отображается в браузере. На новом сайте есть три страницы. Первая – главная, вторая и третья эмитируют некоторые новостные ленты. Для создания данных страницы требуется уметь парсить JSON, для чего можно использовал встроенный в JavaScript метод JSON.Parse(…). Далее на второй и третьей страницах реализован поиска с помощью регистрации нажатия на клавиши клавиатуры и дальнейшей их обработки. Можно получить содержимое строки поиска и, пробегаясь по всем новостям, искать вхождения содержимого строки поиска в текст новости. При отсутствии вхождения новость скрывать с помощью атрибута display: none.



13 Листинг. Создание новостей



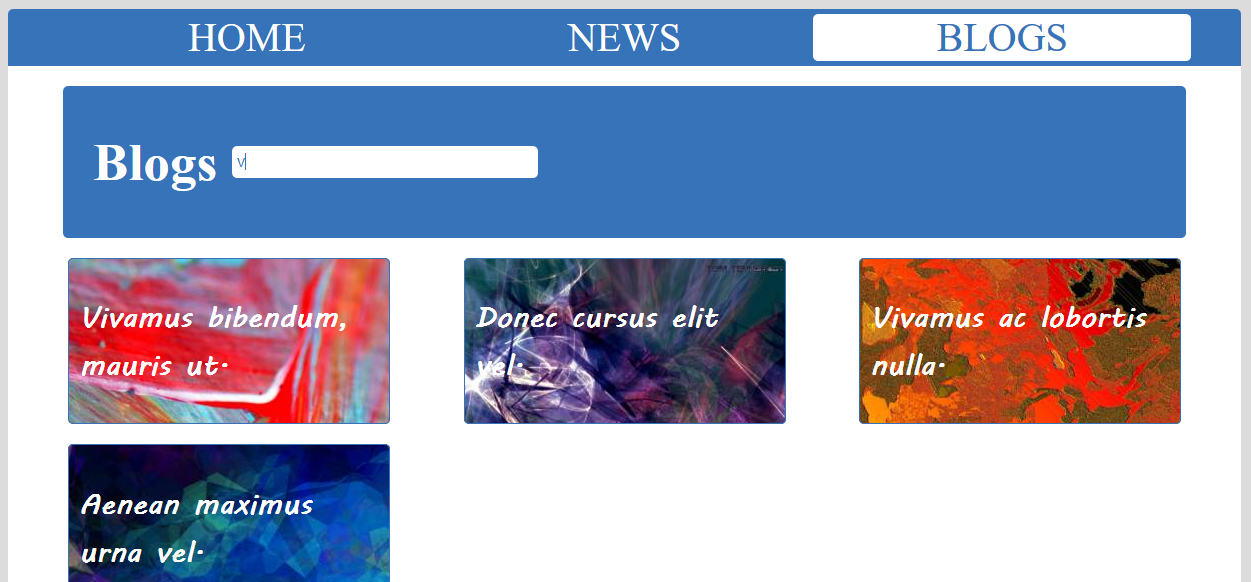
14 Листинг. Поиск

**Итоговый результат:**

Текс для создания новостей бралась с ресурса [www.lipsum.com](http://www.lipsum.com).



15 Страница новостей



16 Страница блогов



17 Снимок экрана

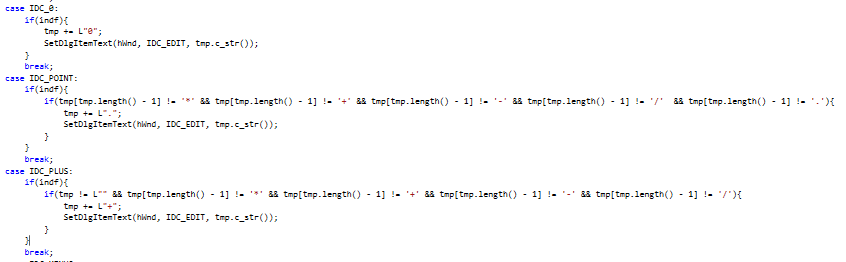
Лабораторная работа №5. Создания калькулятора на WinApi.

**Условие:**

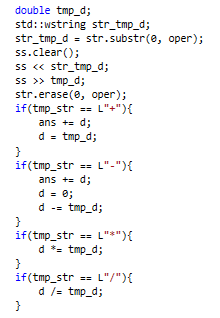
Требовалось разработать приложение “Калькулятор целых чисел” на WinApi. Было необходимо использовать редактор ресурсов, сделать главное окно приложения диалоговым, а также проверять на корректность вводимые данные. Также требовалось добавить меню с возможностью загрузить из файла строки списка результатов вычислений и возможностью записать их в файл из формы приложения.

**Решение:**

В качестве среды разработки я использовал Visual Studio 2010 и создавал проект с каркасом WinApi приложения. Далее создал функцию для обработки диалогового окна, которое в данной задаче требуется сделать главным. На данном диалоговом окне я разместил кнопки, визуально напоминающие структуру обычного калькулятора. Работой с целыми числами было решено не ограничиваться, поэтому мой калькулятор работает с вещественными числами. Запись их происходит при помощи кнопки «.». Также было решено не ограничиваться выполнением лишь одной арифметической операции за раз. Данный калькулятор имеет возможность вычислять целые выражения.



18 Листинг. Пример обработки событий кнопок

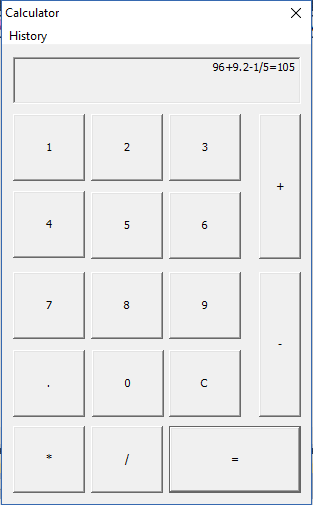


19 Листинг. Фрагмент алгоритма вычисления

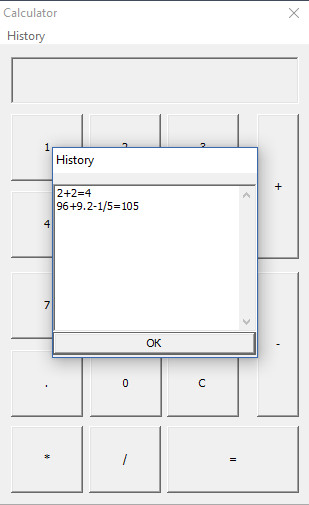
Выражения для вычисления записывается при помощи соответствующих кнопок, и заносится в editbox, и хранится в формате wstring. Затем при нажатии кнопки «равно» данная строка отправляется в функцию, которая реализует алгоритм вычисления. Функция в свою очередь возвращает значения в формате wstring. Данное значение добавляем к исходной строке: пользователь видит результат. Для того, чтобы не забыть значения выражения в меню приложения есть история, где можно просмотреть результат предыдущих вычислений.

**Итоговый результат:**

В итоге с помощью WinApi я создал приложение “Калькулятор”, имеющее достаточный функционал для использования его при простейших расчётах.



20 Калькулятор



21 История вычислений

Лабораторная работа №6. Рисование мышью.

**Условие:**

Требовалось создать WinApi приложение, в котором мышь будет оставлять след на экране при зажатой левой клавише мыши. Остальные требования:

Сохранять рисунок при перерисовке окна.

В заголовке окна указать свою фамилию.

B меню задавать цвет линии (диалог выбора цвета), толщину и тип линии (модальное пользовательское окно).

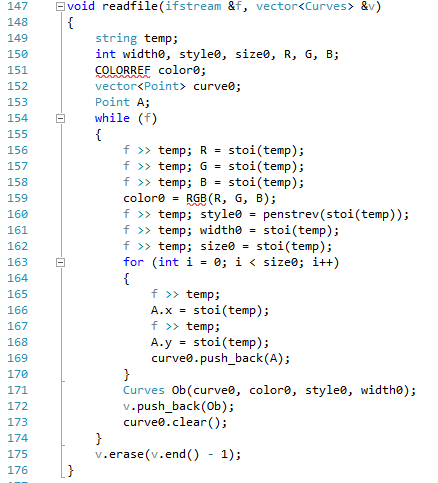
В строке состояния отображать информацию об текущих атрибутах линии, которой рисуем.

Рисунок сохранять в файле. Диалог выбора файла. Рисунок сохраняется как набор точек и атрибутов линий, соединяющих их.

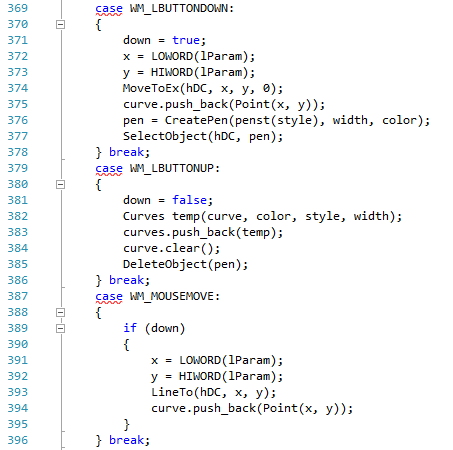
Загружать рисунок из файла с возможностью его модификации.

**Решение:**

Идея реализации состоит в том, чтобы использовать специальный класс для хранения и работы с данными отрисовки линий. Необходимо поддерживать функции чтения из файла и записи в файл. Также использовались стандартные формы выбора файла, сохранения файла и палитры цветов. Стиль «карандаша» выбирается при помощи радио-кнопок.



22 Листинг. Чтение файла



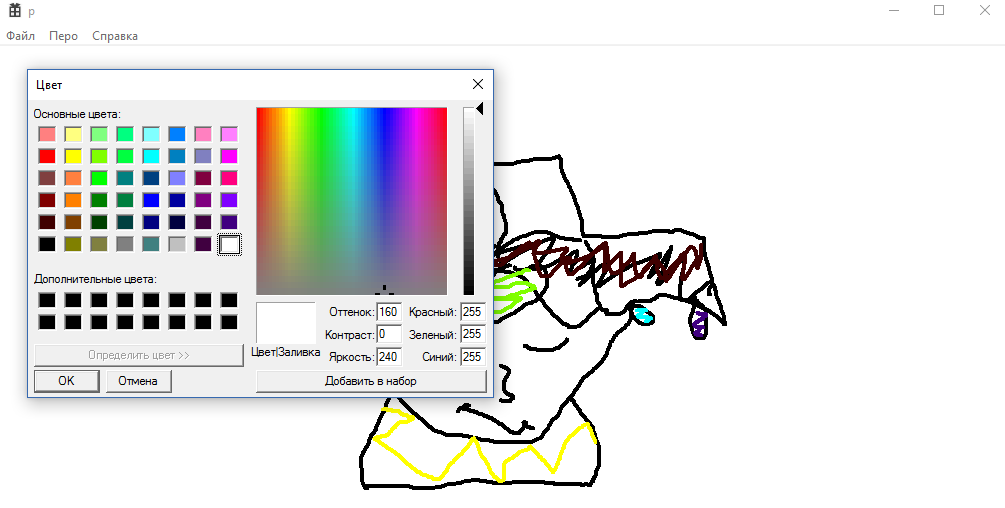
23 Листинг. Обработка событий мыши

**Итоговый результат:**

Было создано приложение почти эквивалентное оригинальному Paint.



24 Приложение Paint



25 Форма цветовая палитра

Лабораторная работа №7. Создание иерархии классов на C#.

**Условие:**

Требовалось создать иерархию классов и прописать ключевые методы для компьютерной игры без реализации полного функционала на языке C#. Суть игры:

Игрок может передвигаться по прямоугольному полю размером Width на Height.

На поле располагаются бонусы (яблоко, вишня и т.д.), которые игрок может подобрать для поднятия каких-либо характеристик. Некоторые бонусы могут менять свое состояние со временем (например, яблоко может «спеть», увеличивая здоровье, добавляемое при «съедании»)

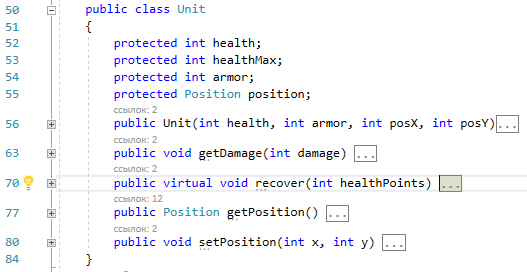
За игроком охотятся монстры (волки, медведи и т.д.), которые могут передвигаться по карте по какому-либо алгоритму (алгоритм и другие характеристики зависят от типа монстра).

Клетки поля обладают разной проходимостью для разных типов монстров. Например, вода непроходима для игрока, но преодолима летающим монстром, трава замедляет движение и т.д.

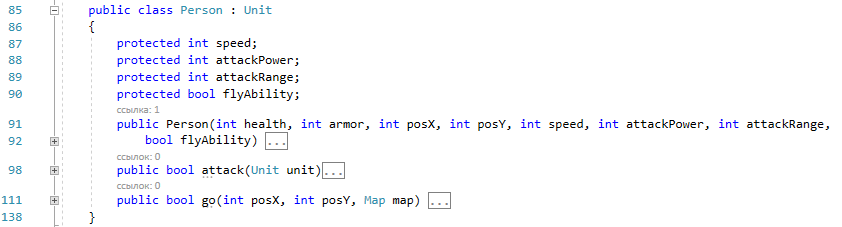
Цель игры – собрать все бонусы и не быть “съеденным” монстрами.

**Решение:**

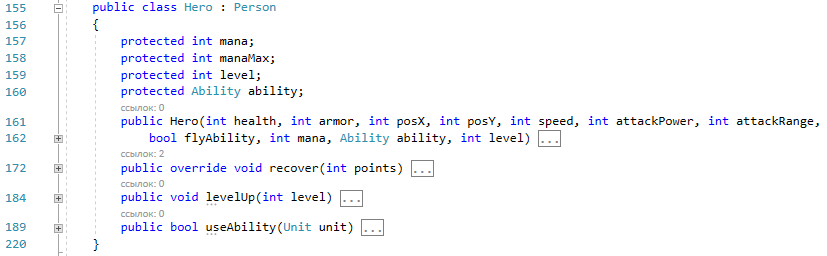
Для решения данной задачи была создана вспомогающая структура Position, enum BuildingType и Ability. Далее были разработаны игровые классы: Map, Unit. От класса Unit были созданы два дочерних класса Person и Building, обладающие особенными свойствами. И от класса Person Был создан дочерний класс Hero, обладающий дополнительными свойствами.



26 Листинг. Класс Unit



27 Листинг. Класс Person



28 Листинг. Класс Hero

Лабораторная работа №8. Принципы SOLID.

**Условие:**

Требовалось провести рефакторинг кода на C#, в котором были нарушены некоторые принципы SOLID. Давались вспомогательные лекции по SOLID и рекомендации по тому, какие части кода следует рефакторить и какие методы следует применять в конкретных ситуациях.

**Решение:**

В основном процесс решения данной задачи заключается в нахождении проблемных участков кода, то есть тех участков, которые подлежали рефакторингу, определении принципа SOLID, нарушенного в данном участке, и применении соответствующего этому принципу метода рефакторинга или применения паттерна проектирования, подходящего в данной ситуации.

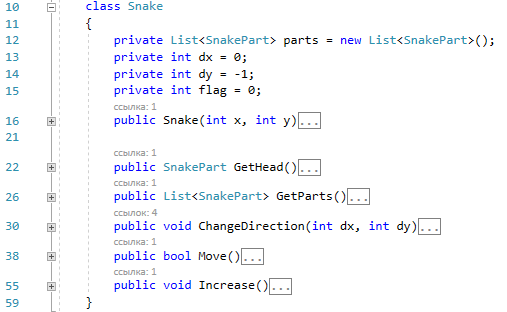
Лабораторная работа №9. Реализация игры “Змейка”.

**Условие:**

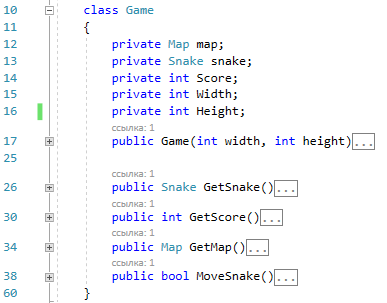
Требовалось создать игру “Змейка” с помощью Windows Forms на C#. Приложение должно было состоять их трех экранов: главного меню, непосредственно самого экрана игры и экрана рекордов. Также рекомендовалось прописать иерархию классов, продумать систему навигации для экранов и использовать MVP паттерн.

**Решение:**

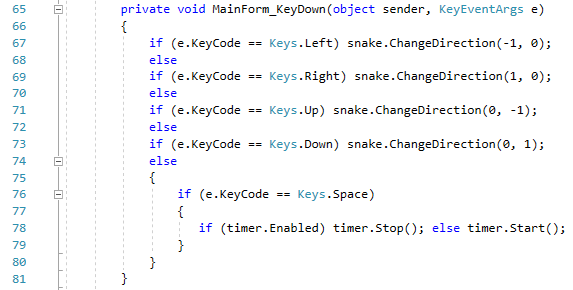
Для реализации внешнего вида приложения использовался стандартный конструктор форм, встроенный в Visual Studio 2017, содержащий все необходимые инструменты, такие как кнопки, области для текста. Для реализации движения змейки использовался таймер. Рисование осуществлялось с помощью методов Paint для соответствующий панелей. Регистрация нажатий осуществлялась с помощью средств C#.



29. Листинг. Класс Snake



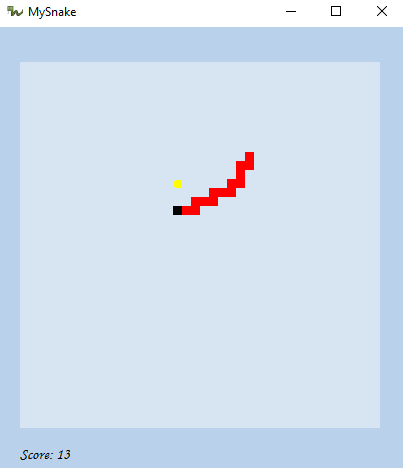
30 Листинг. Класс Game



31 Листинг. Обработка событий

**Итоговый результат:**

Было создано приложение Snake при помощи стандартной формы и языковых средств C#.



32 Приложение Snake

Лабораторная работа №10. Создание Web-приложения.

**Условие:**

Требовалось создать веб-приложение Stack overflow, позволяющее пользователю задавать вопросы, на которые другие пользователи могут дать ответы. Система вопросов и ответов о программировании.

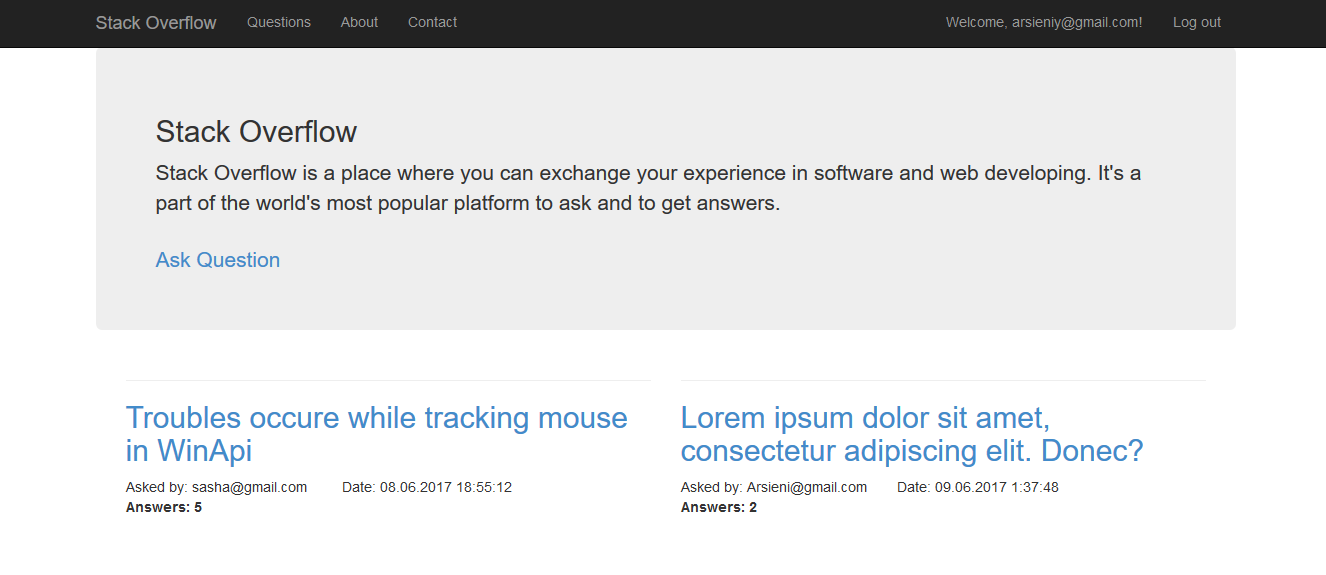
**Решение:**

Работа над приложением осуществлялась средствами ASP.NET и Entity Framework. Использовалась заготовка Web-приложения ASP.NET, в которой уже были страницы регистрации и входа в систему, а также простое оформление и некоторые дополнительные страницы. Была создана главная страница с отображением вопросов. Отдельная страница для просмотра всех вопросов и страница, где размещались ответы на конкретный вопрос. Веб-приложение взаимодействует с сервером, который в свою очередь получает информацию из базы данных для отображения на страницах сайта.

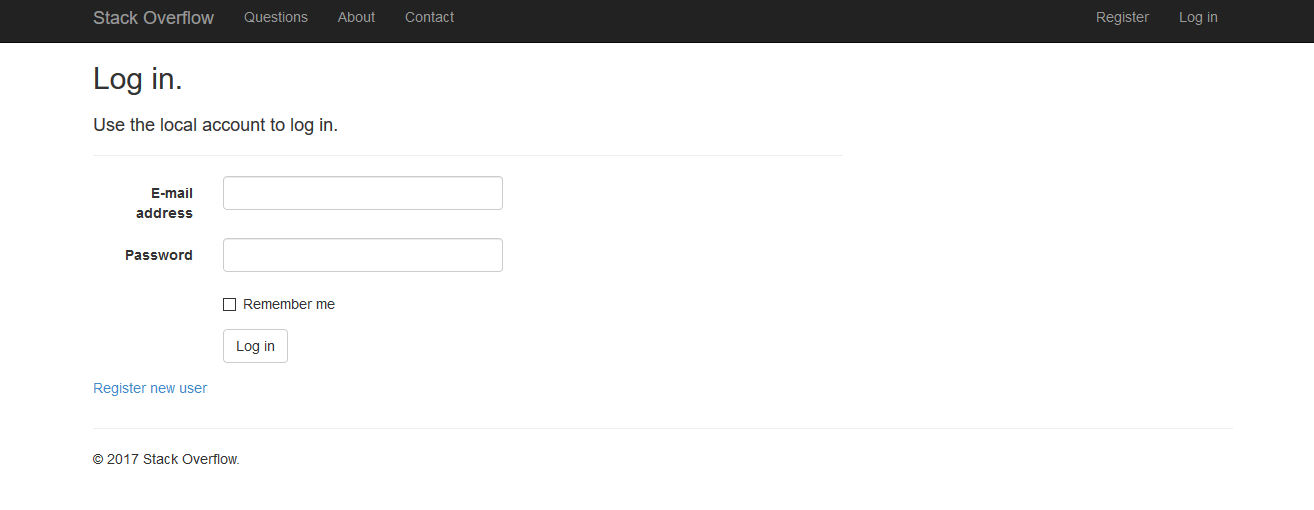
Авторизированный пользователь имеет больше возможностей на сайте в сравнении с неавторизированным.

**Итоговый результат:**

Было создано веб-приложение представляющее собой систему вопросов и ответов о программировании.



33 Stack overflow главная страница



34 Stack overflow страница входа

Список используемой литературы

1. В.В. Мержевич В.В. HTML и CSS на примерах. / В. В. Мержевич. – СПб: BHV-Санкт-Петербург, 2005.
2. Т. А. Павловская. С#. Программирование на языке высокого уровня: Учебник для вузов. / Т. А. Павловская. – СПб: Питер, 2012.
3. Ю. А. Щупак. Win32 API. Эффективная разработка приложений / Ю. А. Щупак. – СПб: Питер, 2007.
4. Т. А. Павловская. C++. Программирование на языке высокого уровня: Учебник для вузов. /Т. А. Павловская. – СПб: Питер, 2012.
5. <https://msdn.microsoft.com> – Электронный ресурс
6. <https://stackoverflow.com> – Электронный ресурс

Приложение