МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

ОТЧЕТ ПО учебной вычислительной

(ознакомительной) практике

Алексейчик Валерии Сергеевны,

студентки 1 курса,

специальность «информатика»

Руководитель практики:

старший преподаватель

А. В. Сатолина

Минск, 2023

ОГЛАВЛЕНИЕ

[Введение 3](#_Toc145485728)

[Глава 1. Направление «.NET» 4](#_Toc145485729)

[1.1 Общая характеристика курса 4](#_Toc145485730)

[1.2 Теория 4](#_Toc145485731)

[1.3 Практика 4](#_Toc145485732)

[1.4 Дополнительный материал 8](#_Toc145485733)

[1.5 Проект «Louis Vuitton Date Code» 8](#_Toc145485734)

[Глава 2. Индивидуальный проекТ 10](#_Toc145485735)

[2.1 Функциональное назначение 10](#_Toc145485736)

[2.2 Структура проекта 10](#_Toc145485737)

[2.3 Интерфейс приложения 12](#_Toc145485738)

[Заключение 17](#_Toc145485739)

[Список использованных источников 18](#_Toc145485740)

Введение

Задачи практики:

* знакомство с порталом <https://wearecommunity.io> компании, возможностями обучения и повышения квалификации;
* изучение инструментов для совместной разработки проектов и контроля версий;
* ознакомление с разными направлениями профессиональной подготовки специалистов IT;
* подготовка и настройка программных сред и средств тестирования для прохождения тренинга по выбранному направлению;
* выполнение стандартных заданий тренинга;
* выполнение заданий тренинга повышенного уровня.

В ходе учебной практики на портале <https://wearecommunity.io/events/it-warm-up> были прослушаны лекции:

* Краткий обзор IT-профессий;
* SAP Development & Consulting;
* Data Analytics Engineering;
* Functional Testing;
* Robotic Process Automation;
* Performance Optimization;
* Cloud & DevOps.

При прохождении практики обязательным направлением было .NET Development.

Платформа .NET широко используется для создания множества различных приложений, таких как:

* облачные приложения,
* кроссплатформенные клиентские приложения,
* приложения для Windows.

Глава 1. Направление «.NET»

1.1 Общая характеристика курса

В основе этого курса лежит набор практических заданий, которые охватывают основные темы, связанные с программированием на языке C#. Курс предусматривает самостоятельное изучение теоретического материала, и задача по поиску решений полностью лежит на студенте. Курс разбит на две секции, и работа по каждой секции происходит в отдельном курсе Autocode:

* первая секция – .NET WarmUp (главы 1 – 10),
* вторая секция – .NET WarmUp (главы 11 – 18).

1.2 Теория

В ходе учебной практики были изучены следующиемодули:

* Integers – работа с целыми числами;
* Static Methods – знакомство со статическими методами;
* Bools & Logic Operators – работа с логическими операторами;
* Conditional Statements (if, if-else);
* Iteration Statements (while, for);
* Real Numbers – работа с числами с плавающей точкой;
* Exceptions – знакомство с исключительными ситуациями и их обработка;
* Formatting and Parsing Strings – задания на преобразование строк в другие типы данных и обратно;
* Working with Arrays – тренировка навыков по работе с массивами;
* Implementing Search Algorithms – освоение алгоритмов поиска;
* Implementing Sorting Algorithms – освоение алгоритмов сортировки.

1.3 Практика

В ходе учебной практики были решены задачи:

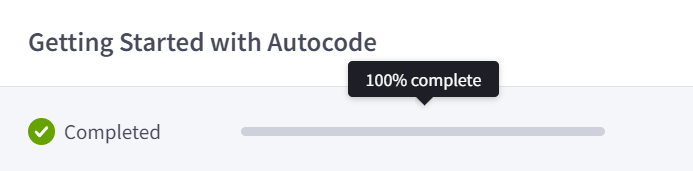
* Hello, World! (100/100);
* Fancy Calc (100/100);

Рисунок 1 – оценка 100% за прохождение главы I.

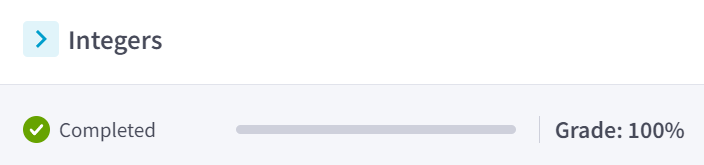
* Integers (100/100);

Рисунок 2 – оценка 100% за прохождение главы II.

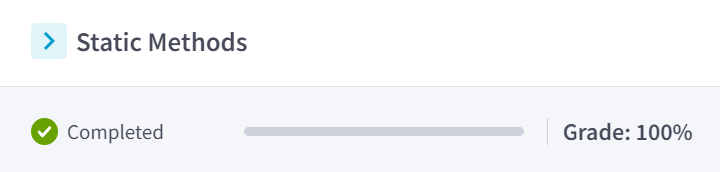
* Static Methods (100/100);
* Compiler Errors (100/100);

Рисунок 3 – оценка 100% за прохождение главы III.

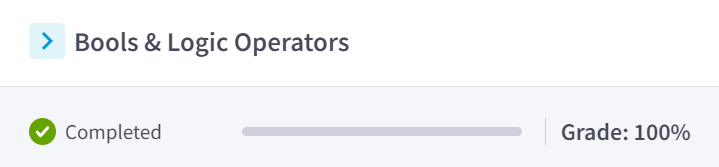
* Bools (100/100);

Рисунок 4 – оценка 100% за прохождение главы IV.

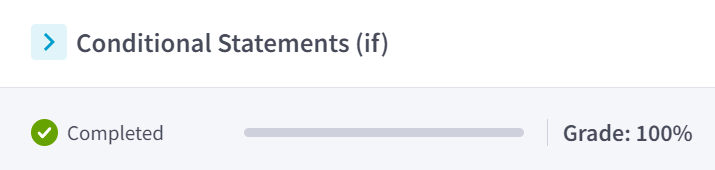
* If Statements (100/100);

Рисунок 5 – оценка 100% за прохождение главы V.

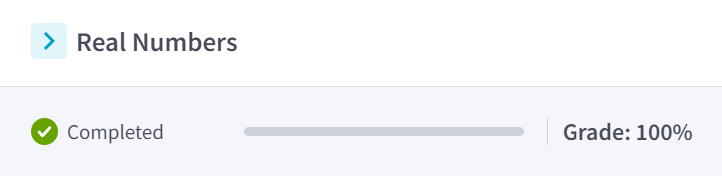
* Real numbers (100/100);
* Numeric Conversions (100/100);

Рисунок 6 – оценка 100% за прохождение главы VI.

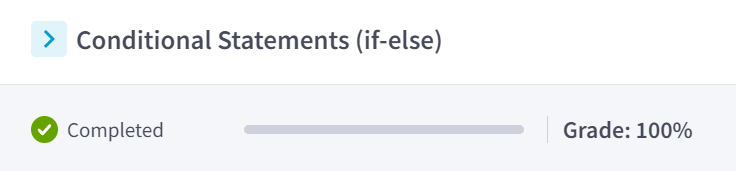
* If-Else statements (100/100);
* Leap Year (100/100);
* Darts Game (100/100);

Рисунок 7 – оценка 100% за прохождение главы VII.

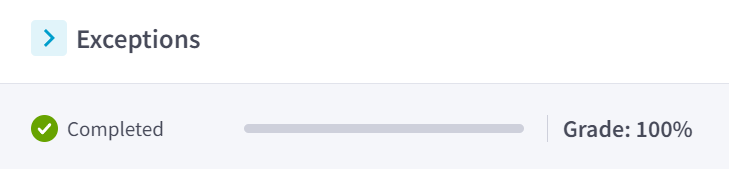
* Exceptions (100/100);

Рисунок 8 – оценка 100% за прохождение главы VIII.

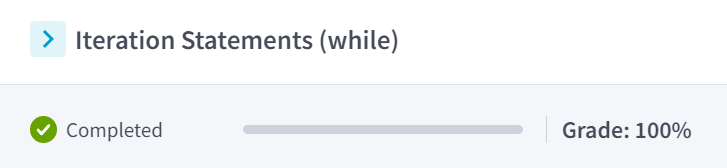
* While Statements (100/100);
* While Practice (100/100);

Рисунок 9 – оценка 100% за прохождение главы IX.

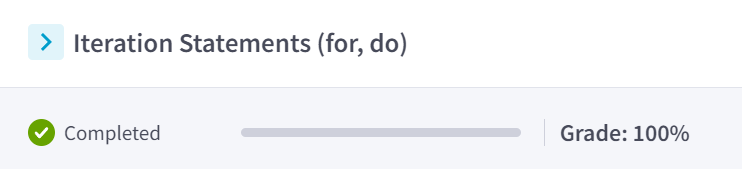
* For Statements (100/100);
* Population (100/100);

Рисунок 10 – оценка 100% за прохождение главы X.

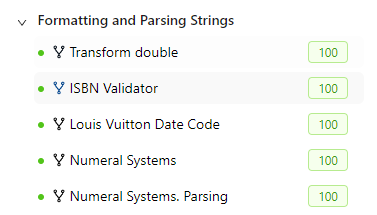
* Transform Double (100/100);
* ISBN Validator (100/100);
* Louis Vuitton Date Code (100/100);
* Numeral Systems (100/100);
* Numeral Systems. Parsing (100/100);

Рисунок 11 – оценка 100% за прохождение главы XI.

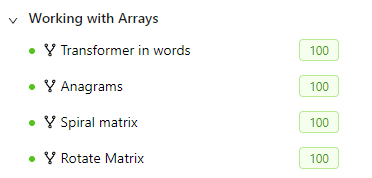
* Transformer In Words (100/100);
* Anagrams (100/100);
* Spiral Matrix (100/100);
* Rotate Matrix (100/100);

Рисунок 12 – оценка 100% за прохождение главы XII.

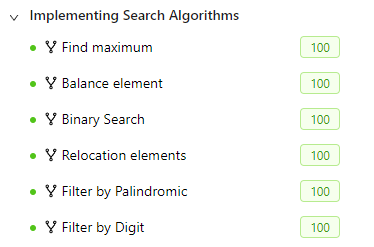
* Find Maximum (100/100);
* Balance Element (100/100);
* Binary Search (100/100);
* Relocation Elements (100/100);
* Filter by Palindromic (100/100);
* Filter by Digit (100/100);

Рисунок 13 – оценка 100% за прохождение главы XIII.

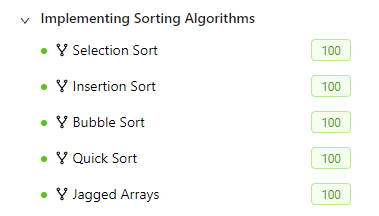
* Selection Sort (100/100);
* Insertion Sort (100/100);
* Bubble Sort (100/100);
* Quick Sort (100/100);
* Jagged Arrays (100/100).

Рисунок 14 – оценка 100% за прохождение главы XIV.

1.4 Дополнительный материал

Курс «Version Control with Git» был пройден полностью:

Рисунок 15 – оценка 100% за прохождение курса по Git.

1.5 Проект «Louis Vuitton Date Code»

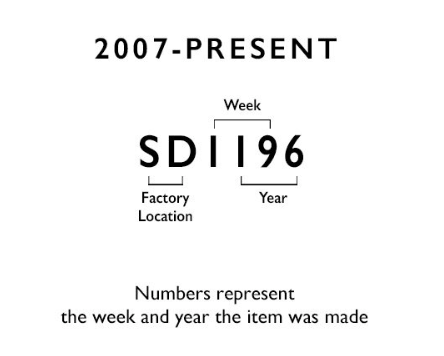
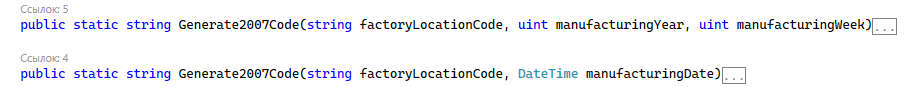
Во время прохождения курса было предложено следующее задание: реализовать алгоритмы генерации и парсинга кода сумки бренда «Louis Vuitton». После изучения предложенного дополнительного материала выяснилось, что в зависимости от десятилетия сумки кодировались разными способами. Так, разберём самый современный и совершенный метод кодирования сумок, который используется с 2007 года по настоящее время.

Рисунок 16 – демонстрация генерации кода сумки в период 2007-наст. время.

Рисунок 17 – метод *Generate2007Code*.

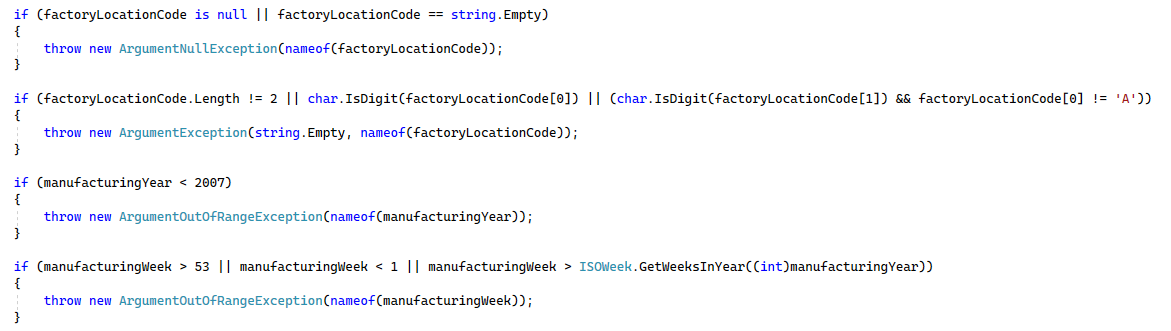
Рассмотрим первый перегруженный метод *Generate2007Code*. Для начала требуется проверить входные данные на правильность и в противном случае выбросить исключение:

Рисунок 18 – выбрасывание исключений, если входная строка *factoryLocationCode* – null, или эта строка пустая, или не удовлетворяет своему виду; выбрасывание исключений, если входные целочисленные значения *manufactoringYear* и *manufactoringWeek* не соответствуют допустимому диапазону значений.

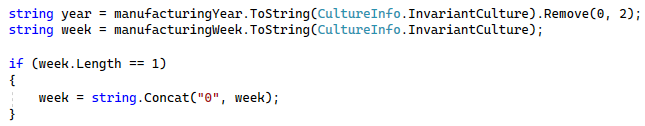
Далее инициализируем строки *year* и *week*, которые являются составными частями конечной строки-результата – кода сумки. Поскольку код есть строка длины 6, при необходимости корректируем значение *week*:

Рисунок 19 – инициализация переменных.

Итак, генерируем код сумки по частям:

Рисунок 20 – составление строки *code* и её возвращение.

Рассмотрим теперь второй перегруженный метод *Generate2007Code*. Сперва инициализируем целочисленные значение *year* и *week*:

Рисунок 21 – инициализация переменных.

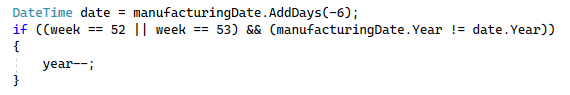
Изучив особенности генерации кода в данный период, стало ясно, что есть загвоздка в правилах составления кода сумки, если она была изготовлена в новогоднюю неделю. Так, в ситуации, когда новый календарный год уже наступил, но 1 января – день, попадающий на последнюю неделю прошлого года, код сумки составляется таким образом, будто бы она была сделана ещё в прошлом году. Обработаем этот случай:

Рисунок 22 – корректировка значения *year* при необходимости.

Все данные получены, воспользуемся первым методом, чтобы сгенерировать код сумки:

Рисунок 23 – генерация кода сумки.

Глава 2. Индивидуальный проекТ

В данной главе представлен разработанный проект «Wordle».

2.1 Функциональное назначение

Цель данного проекта – создание аналога игры в слова «Wordle», разработанной британским программистом Джошем Уордлом.

Суть игры такова: случайным образом выбирается слово из пяти букв, которое игрок должен угадать за шесть попыток. После каждого предположения буквы отмечаются жёлтым, белым или серым цветом; жёлтый означает, что буква правильная и находится в правильном месте, белый означает, что буква есть в ответе, но на другой позиции, а серый означает, что буквы в загаданном слове нет вообще.

2.2 Структура проекта

Проект создан с помощью WindowsForms – фреймворка для разработки кроссплатформенных оконных приложений на языке программирования C#. Этот фреймворк является частью Microsoft .NET Framework. В качестве среды разработки была использована Visual Studio 2022.

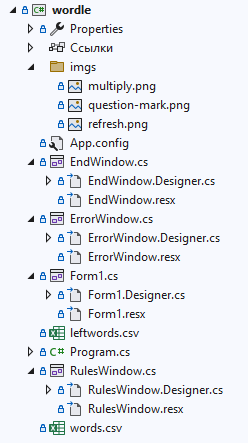
Далее представлена файловая структура проекта:

Рисунок 24 – файловая структура проекта.

Ниже приведено назначение каждой структурной единицы проекта:

* Form1.cs – класс формы, где реализованы все методы, связывающие алгоритм игры и её интерфейс;
* RulesWindow.cs – класс формы, отвечающей за визуализацию правил игры;
* ErrorWindow.cs – класс формы, которая появляется, если игрок вводит несуществующее слово;
* EndWindow.cs – класс формы, которая появляется, когда заканчивается игра, и сообщает, выиграл или проиграл игрок;
* imgs – папка, в которой находятся картинки для интерфейса;
* leftwords.csv – текстовый файл, содержащий слова, которые необходимо отгадать;
* words.csv – текстовый файл, содержащий все или почти все 5-буквенные английские слова.

2.3 Интерфейс приложения

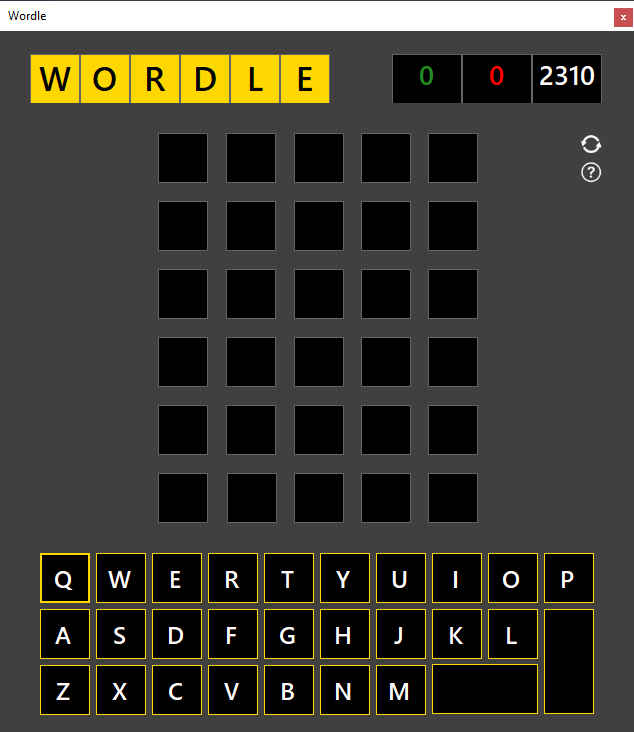
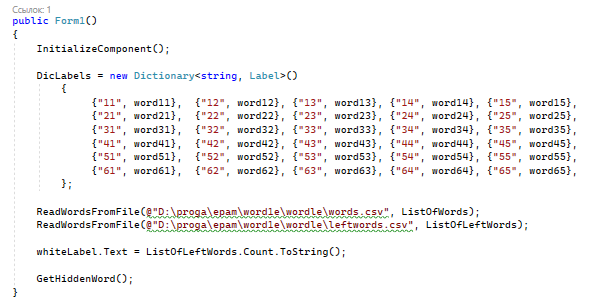
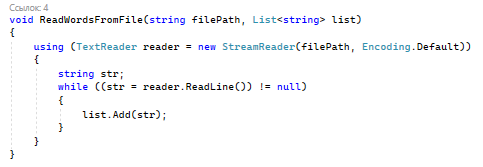
На следующем рисунке изображено главное окно приложения. В левом верхнем углу можно увидеть 6 элементов *Label*, которые изображают название игры. В правом верхнем углу расположены 3 элемента *Label*, отвечающие за статистику текущей игры. Чуть ниже находятся 2 кликабельных элемента *PictureBox*. В центре окна размещается элемент *Panel*, на котором упорядоченным образом расставлены 30 (6 по 5) элементов *Label* – ячейки для букв. В самом низу расположен элемент *Panel*, которая представляет собой виртуальную клавиатуру английской раскладки: всего 28 элементов *Button*.

Рисунок 25 – главное окно приложения.

Рассмотрим процесс игры. Игрой загадывается слово из предложенного списка слов:

Рисунок 26 – конструктор формы *Form1*, где, помимо всего прочего, присваиваются значения полям класса *ListOfWords*, *ListOfLeftWords* и *HiddenWord*.

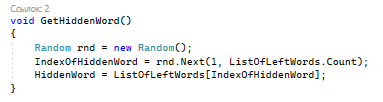
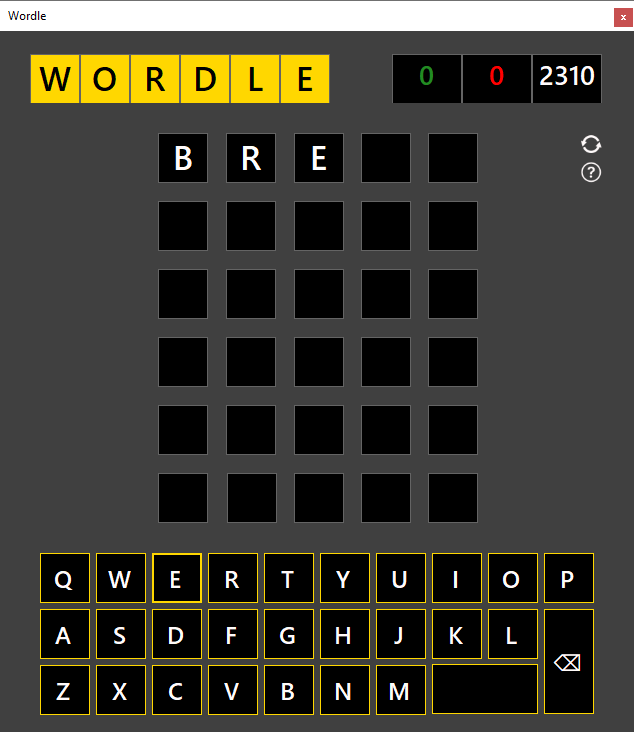
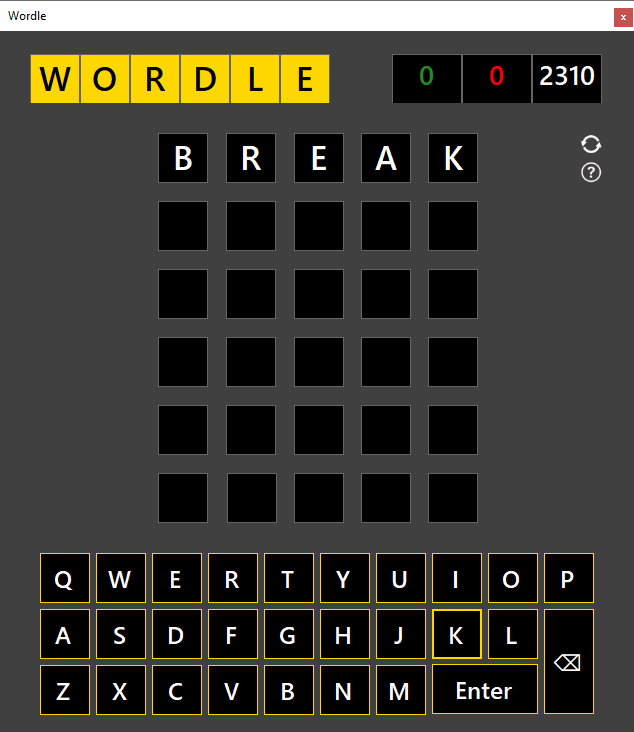
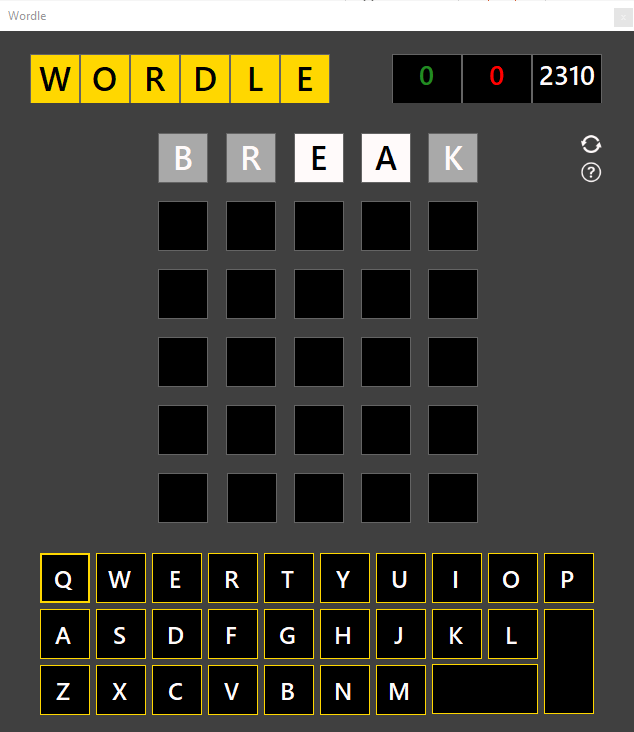
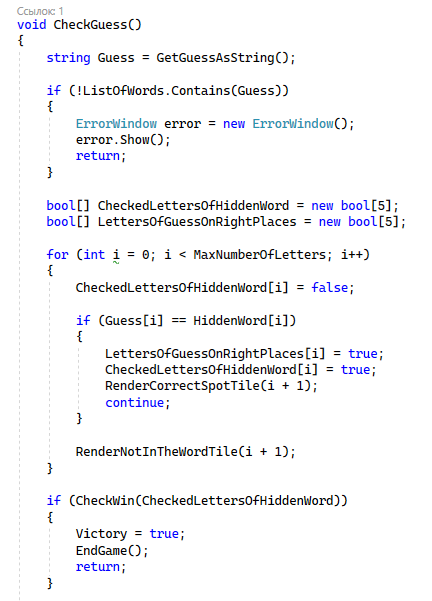
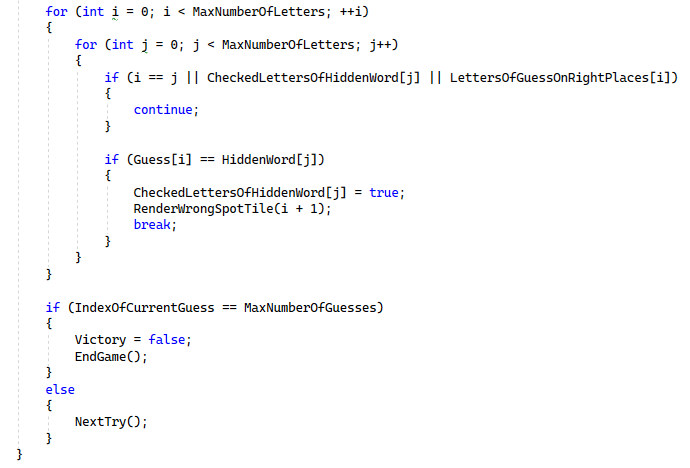
Рисунок 27 – метод *ReadFromFile*, который считывает строки из файла в список.

Рисунок 28 – метод *GetHiddenWord*, который присваивает значение полю *HiddenWord*.

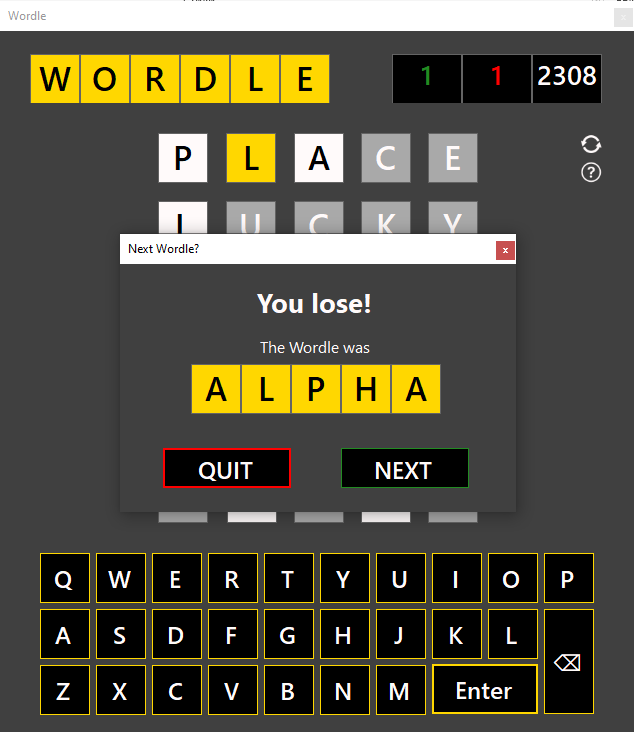
Введём слово-попытку и проверим его:

Рисунки 29, 30 – ввод слова; кнопка *Backspace* стала активна, как только был введён первый символ; кнопка *Enter* активировалась, когда были введены все буквы.

Рисунки 31, 32 – была нажата кнопка *Enter*, буквы в введённом слове подсветились соответствующим образом; показан процесс «угадывания».

На следующих рисунках представлена реализация ключевого для игры метода *CheckGuess*, который вызывается по нажатию кнопки *Enter*. Он проверяет, насколько совпадают введённое слово и загаданное слово, вызывает методы окрашивания ячеек в соответствии с позицией букв, а также проверяет, закончена ли игра:

Рисунки 33, 34 – метод *CheckGuess*.

Если игрок угадал слово или, наоборот, попыток больше не осталось, появляется соответствующее окно:

Рисунки 35, 36 – окно после победы и окно после поражения.

Чтобы продолжить игру, необходимо нажать на кнопку *NEXT* на окне в конце игры. Чтобы выйти из игры, нужно нажать кнопку закрытия на главном окне приложения, кнопку закрытия на окне в конце игры или на кнопку *QUIT* на этом же окне.

Также на рисунках 35, 36 можно увидеть, как изменилась статистика текущей игры. В первом случае игра была выиграна, поэтому число побед в ячейке с зеленым текстом увеличилось на единицу, во втором случае игра была проиграна, и в ячейке с красным текстом число также стало на единицу больше. В обоих случаях в ячейке с белом текстом число уменьшилось на один, эта ячейка демонстрирует, сколько слов из списка ещё осталось угадать, причём те слова, что были загаданы ранее, в последующие разы загадываться не будут.

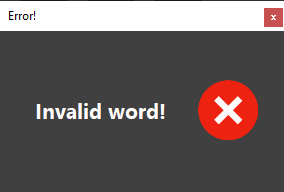
Далее пойдёт речь ещё о некоторых функциях игры. Если игрок захочет проверить несуществующее слово, игра обязательно ему помешает:

Рисунок 37 – окно ошибки, которое сообщает, что введённого слова не существует.

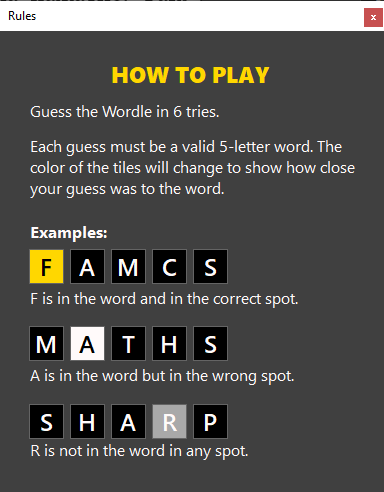
Как говорилось ранее, два элемента *PictureBox*, расположенные чуть ниже области статистики, среагируют на нажатие. Так, нажав на картинку-кнопку рестарта, игрок начнёт новую игру и сбросит всю статистику. Если же игрок нажмёт на картинку-кнопку вопросительного знака, то появится окно, в котором можно посмотреть правила игры:

Рисунок 38 – окно с правилами.

Заключение

В ходе учебной практики мною были изучены основы языка программирования C# и фреймворка .NET.

Были пройдены первая и часть второй секции курса по .NET разработке. Все задания выполнены на максимальный балл.

В результате прохождения курсов я приобрела фундаментальные знания, касающиеся разработки веб-сайтов и приложений на платформе .NET. Полученные знания, хоть и не являются полными, открывают возможность дальнейшего беспрепятственного углубленного изучения данных направлений.

Проект «Wordle» является полностью рабочим, однако в дальнейшем можно реализовать следующий дополнительный функционал:

* музыкальное сопровождение;
* ввод с клавиатуры;
* подробная статистика.

Список использованных источников

1. Официальная документация по C# от Microsoft – Режим доступа: <https://docs.microsoft.com/en-us/dotnet/csharp/>;
2. Metanit – C# – Режим доступа: <https://metanit.com/sharp/>;
3. CodeProject – Режим доступа: https://www.codeproject.com/;
4. Wordle – Wikipedia – Режим доступа: https://en.wikipedia.org/wiki/Wordle.