Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Северо-Осетинский государственный университет

имени Коста Левановича Хетагурова»

**КУРСОВАЯ РАБОТА**

«Клавиатурный тренажёр»

Выполнил:

студент(ка) 1 курса направления

«Прикладная математика и информатика»

Гуссаов Георгий Асламбекович

Научный руководитель:

старший преподаватель кафедры

прикладной математики и информатики

Макаренко М.Д.

Владикавказ 2023

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ 3

ГЛАВА 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 4

* 1. Функционал 4
  2. Этапы реализации 4

ГЛАВА 2. ПРАКТИЧЕСКАЯ ЧАСТЬ 5

2.1 Инструкция пользователя 5

2.2 Инструкция системного администратора 8

2.3 Инструкция программиста 8

ЗАКЛЮЧЕНИЕ 13

СПИСОК ЛИТЕРАТУРЫ 14

**ВВЕДЕНИЕ**

Клавиатурный тренажер – это программа, предназначенная для обучения слепой печати и увеличения скорости печати на клавиатуре компьютера.

Сейчас такие тренажеры хоть и не особо востребованы, но актуальны, ведь всё понемногу переходит на компьютеры и было бы не лишним обладать искусством быстрой печати.

Клавиатурные тренажеры крайне разнообразны, где-то вы вводите необходимые символы, которые «падают» сверху экрана, где-то вводите причудливые сочетания букв, постепенно переходя на слова, но обычно клавиатурный тренажер представляет из себя текст и поле, на котором вы переписываете этот текст, после чего вы видите свой результат.

Постановка задачи: Разработать клавиатурный тренажер на языке программирования С#.

**Глава 1. Теоретическая часть**

Я решил разработать два режима. О первом я писал выше (переписывание текста), а второй представляет из себя появляющиеся на экране буквы, которые необходимо нажимать на клавиатуре.

**1.1 Функционал**

1. Выводить окно с выбором режима тренировки.
2. Первый режим (набор определенного текстового фрагмента).
3. Отображать на экране текст из фильма.
4. Отображать поле для ввода слов из текста.
5. При вводе слова стирать его и переходить к следующему.
6. Если в слове допущена ошибка предупредить об это пользователя.
7. Второй режим.
8. Пользователь должен нажимать необходимые клавиши на клавиатуре.
9. Вывод итогов (примерная скорость печати, количество ошибок) для обоих режимов.
10. Визуальная клавиатура на экране для обоих режимов.

**1.2 Этапы реализации**

1. Версия 0.1

* Выводить окно с выбором режима тренировки.
* Реализация первого режима.

1. Версия 0.2

* Реализация второго режима.

1. Версия 0.3

* Реализация вывода итогов для обоих режимов.
* Реализовать визуальную клавиатуру для обоих режимов.

**Глава 2. Практическая часть**

**2.1. Инструкция пользователя**

При запуске программы пользователь видит окно выбора режима:

Рисунок 1.1. Иллюстрация окна выбора режима.

Если пользователь выберет первый режим, то увидит нечто подобное:

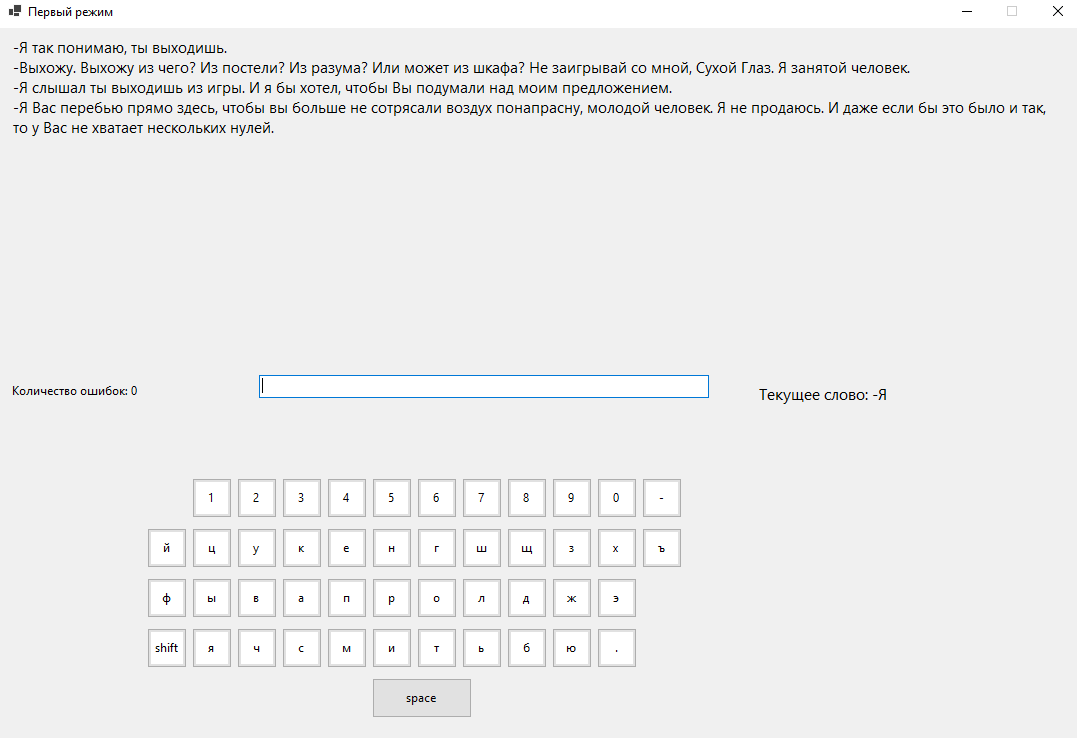


Рисунок 1.2. Внешний вид первого режима.

Далее все что ему требуется, это переписывать выданный ему текст пытаясь не промахиваться по клавишам, после чего он видит результат своих стараний:

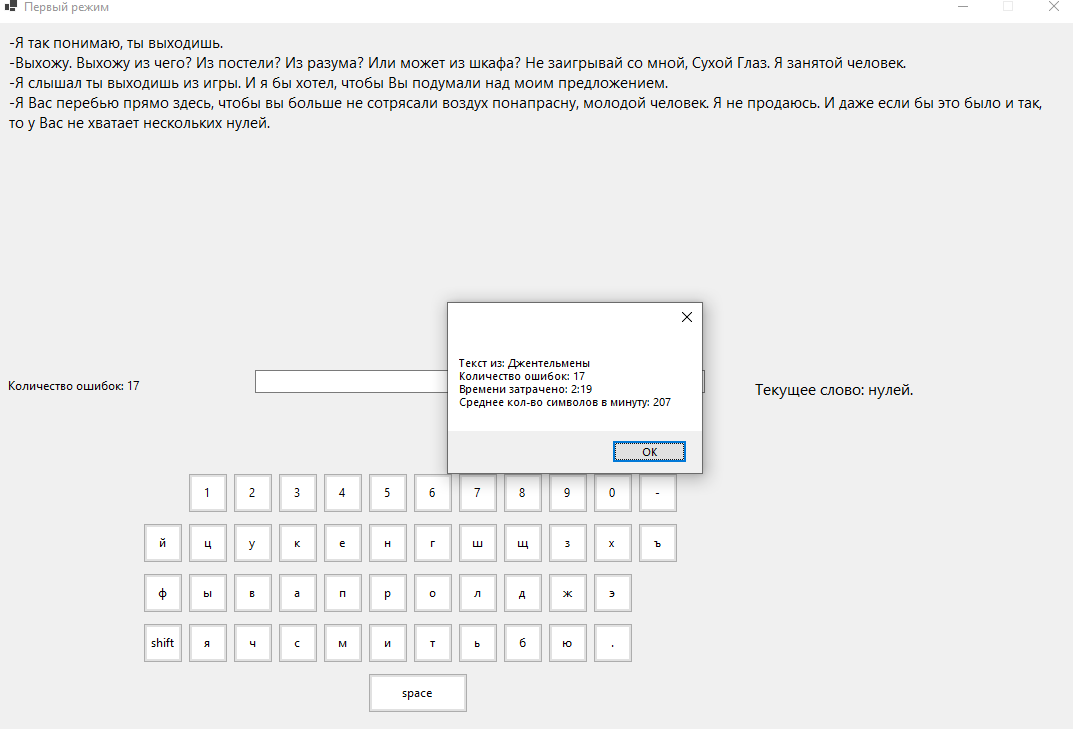


Рисунок 1.3. Иллюстрация вывода результатов первого режима.

Если же пользователь выберет второй режим, то перед ним появится пустое окно с надписью «Нажмите любую клавишу чтобы начать». Непосредственно после нажатия начнут появляться различные буквы теперь уже на английском

языке, а также визуальная клавиатура, подсказывающая где находится необходимая буква.

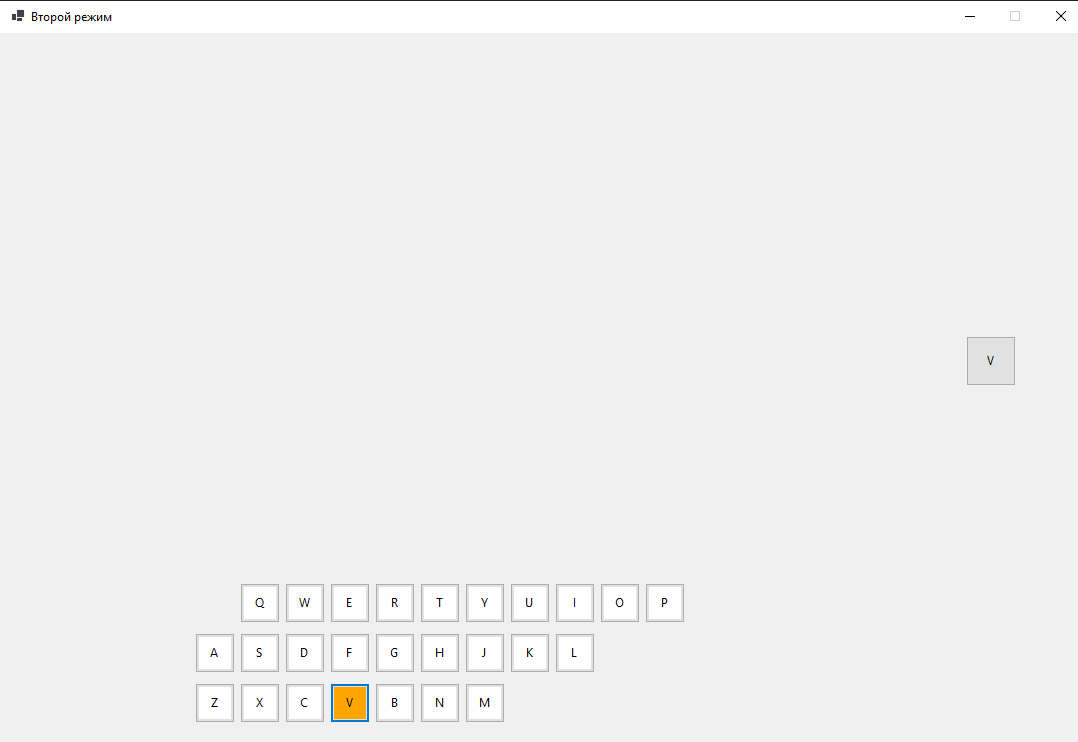


Рисунок 1.4. Внешний вид второго режима.

По истечению 1 минуты пользователь также видит свой результат:

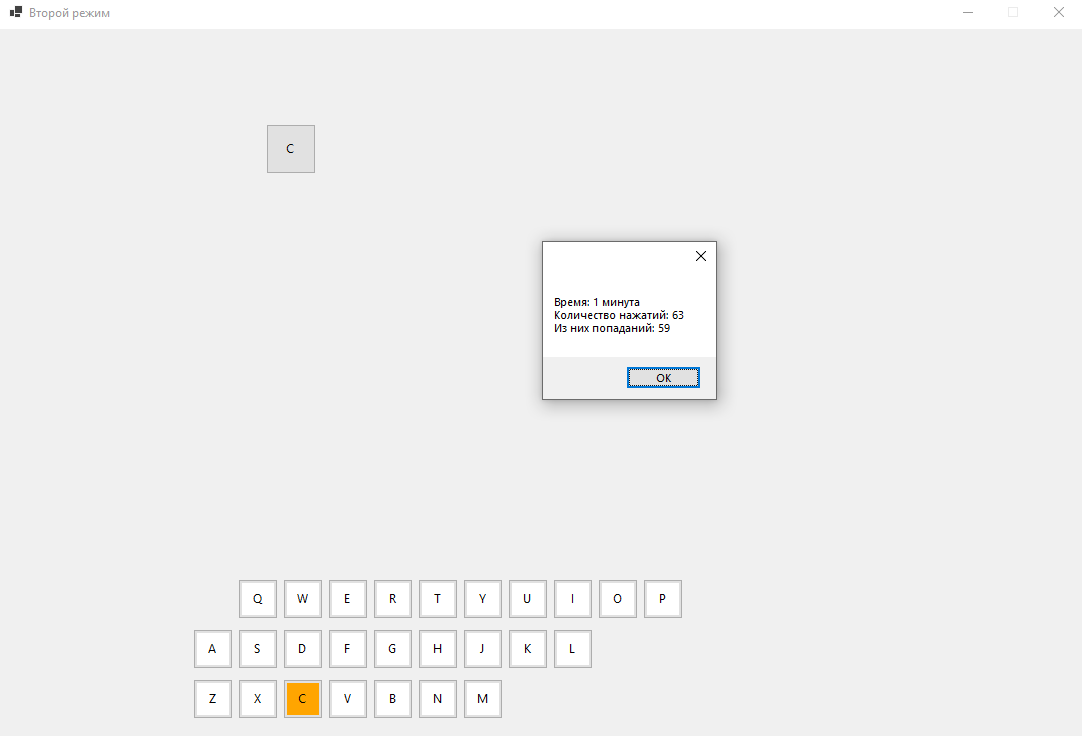


Рисунок 1.5. Иллюстрация вывода результатов второго режима.

**2.2. Инструкция системного администратора**

Для запуска приложения требуется операционная система Windows 7 (или выше) и платформа .NET Framework 4.6.2 (или выше).

**2.3. Инструкция программиста**

Сперва распишу работу первого режима. Вначале осуществляется выбор текста. Всего их 10, через тип данных Random мы получаем 1 из них. Также тут этот текст записывается в список (List<string>) для дальнейшей работы с ним:

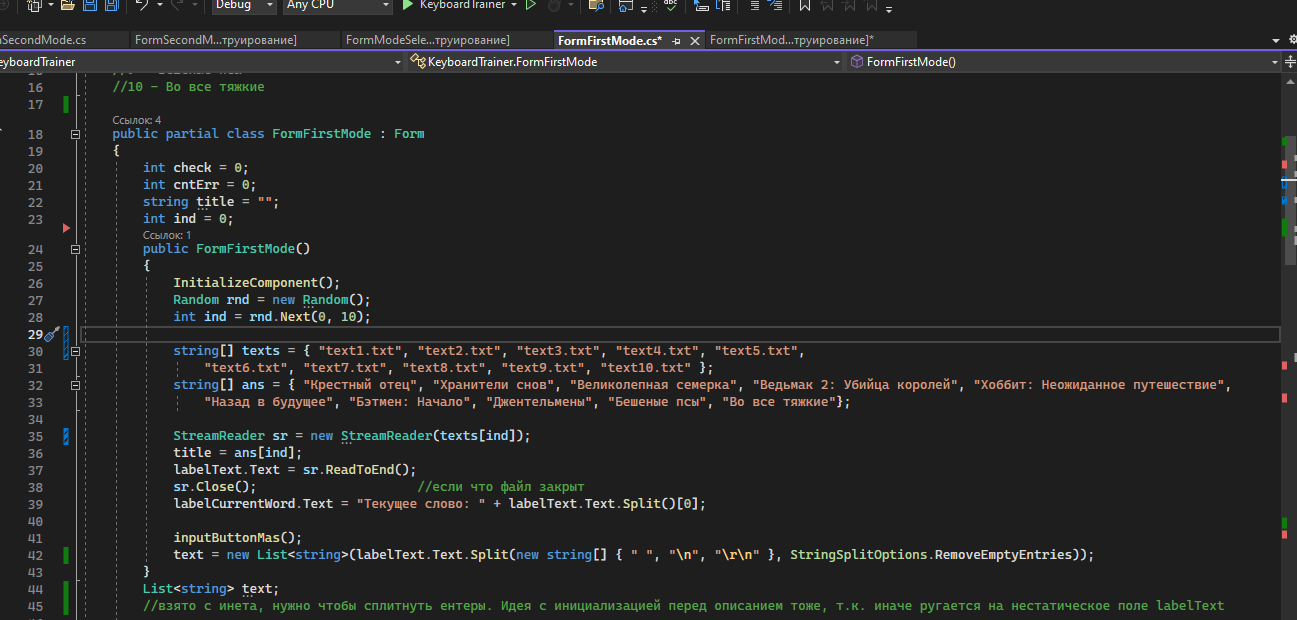


Рисунок 2.1. Выбор текста.

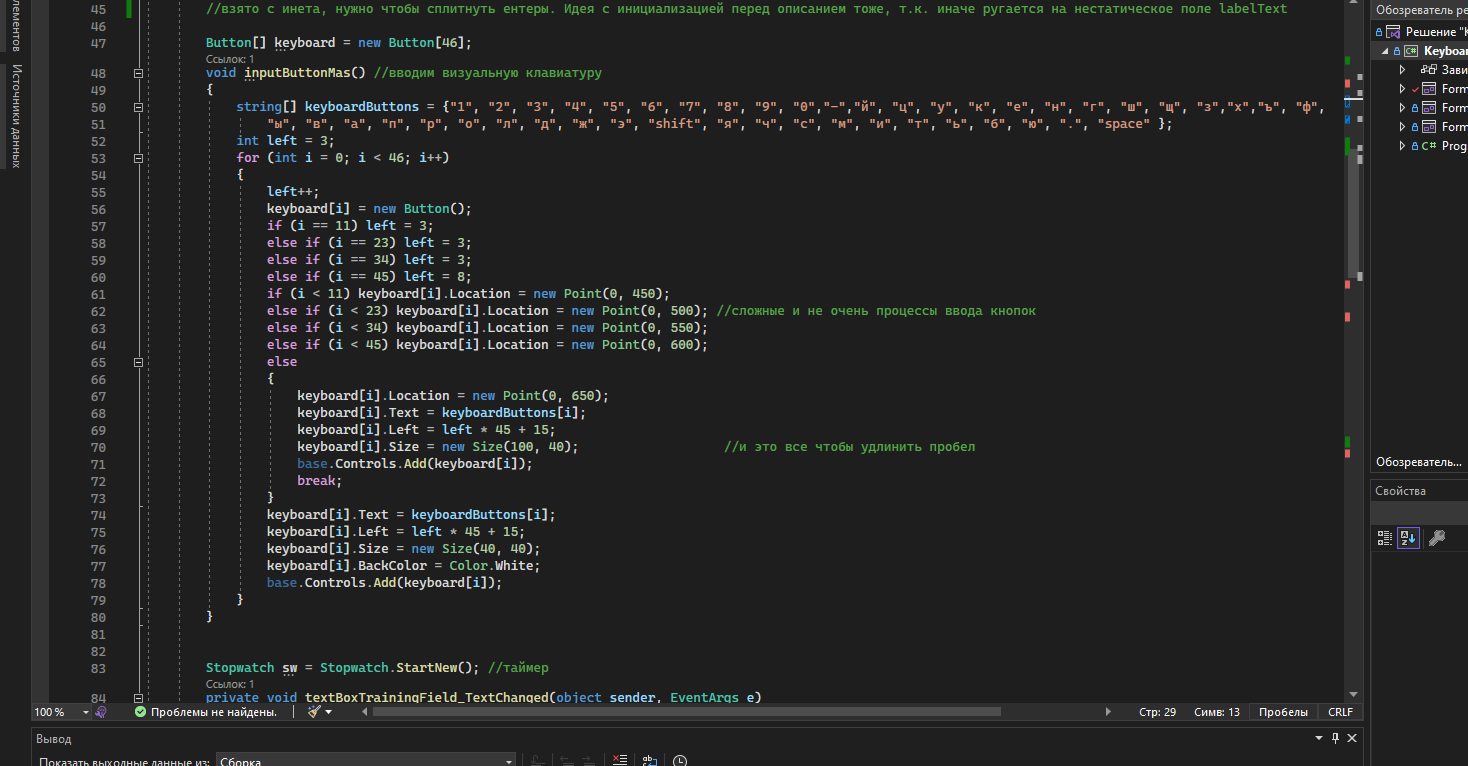
Далее работаю с функцией inputBattonMas() которая уже есть на картинке выше. В ней в основном работа с визуальной частью, как и куда расположить клавиши, чтобы было похоже на реальную клавиатуру.

Рисунок 2.2. Ввод визуальной клавиатуры.

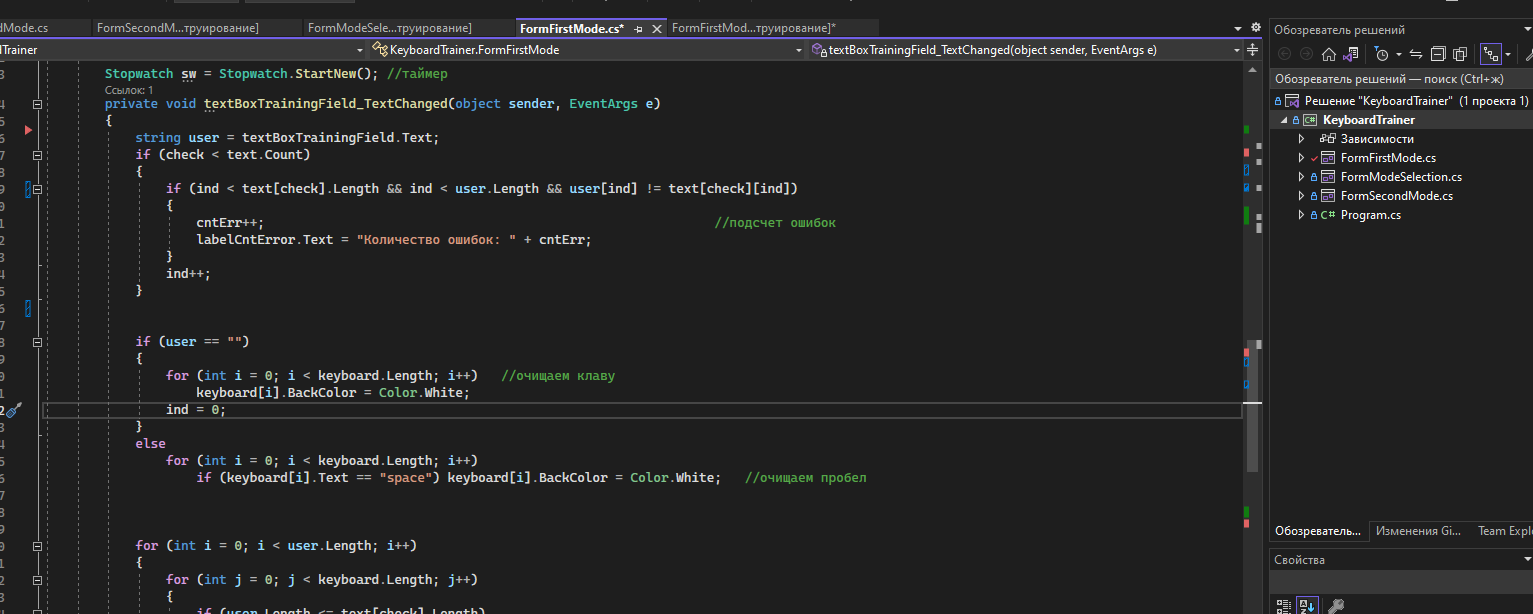
Далее идет самая сложная часть. Начинается работа с событием, которое срабатывает каждый раз при смене текста в поле ввода:

Рисунок 2.3. Подсчет ошибок и очистка клавиатуры после введенного слова

На картинке выше показаны: запуск таймера, вложенное условие, которое считает ошибки и записывает их в переменную cntErr и еще одно условие, стирающее все закрашенные клавиши.

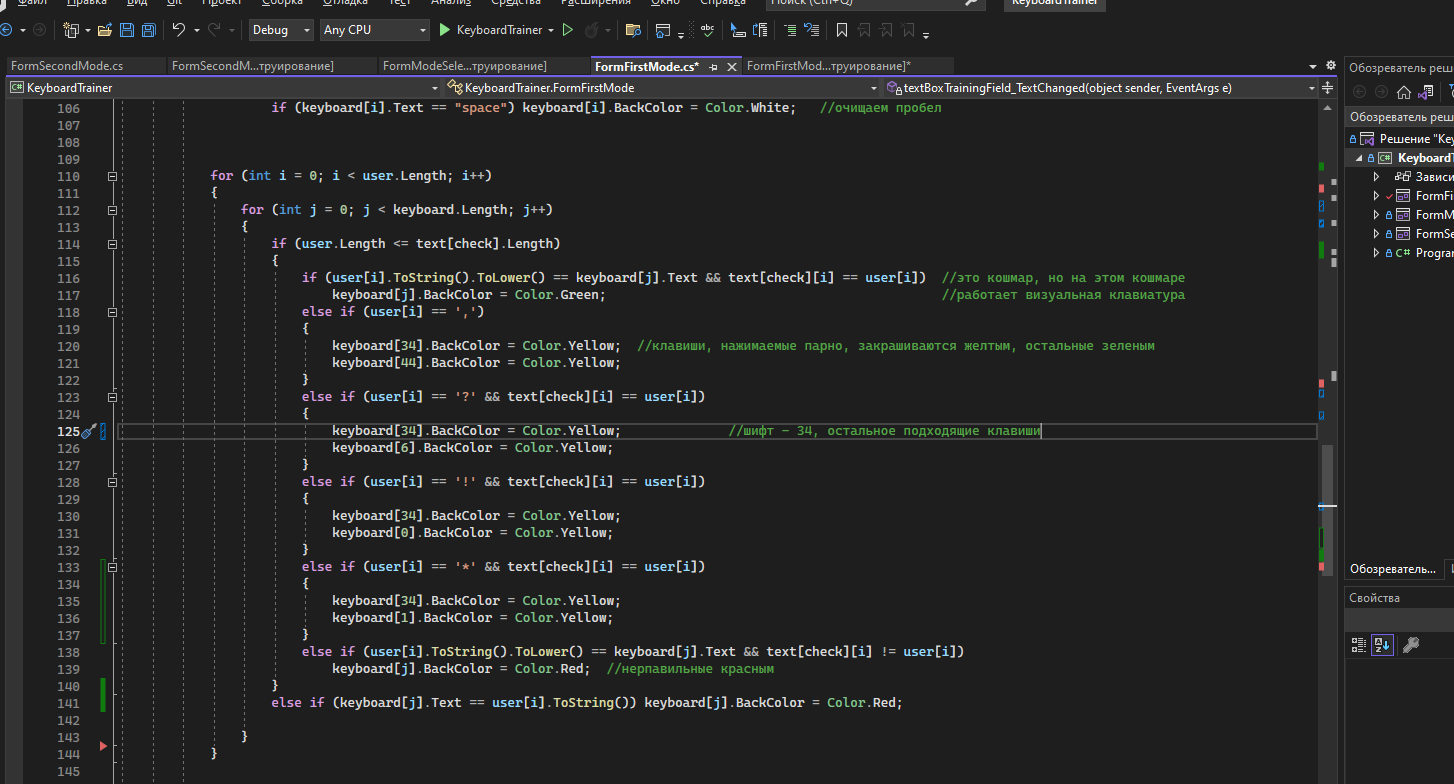
Далее работа с теми самыми клавишами:

Рисунок 2.4. Работа с визуальной клавиатурой.

Тут через вложенный цикл и условие(без условия индекс выходит за границы списка) если буква введенная пользователям (переменная user это строка введенная пользователем) совпадает с буквой в списке text то она закрашивается в зеленый, а в особых случаях(нажимаемые с парно) в желтый, иначе же в красный.

Почему берем text[check] станет понятно после следующего скрина, надеюсь:

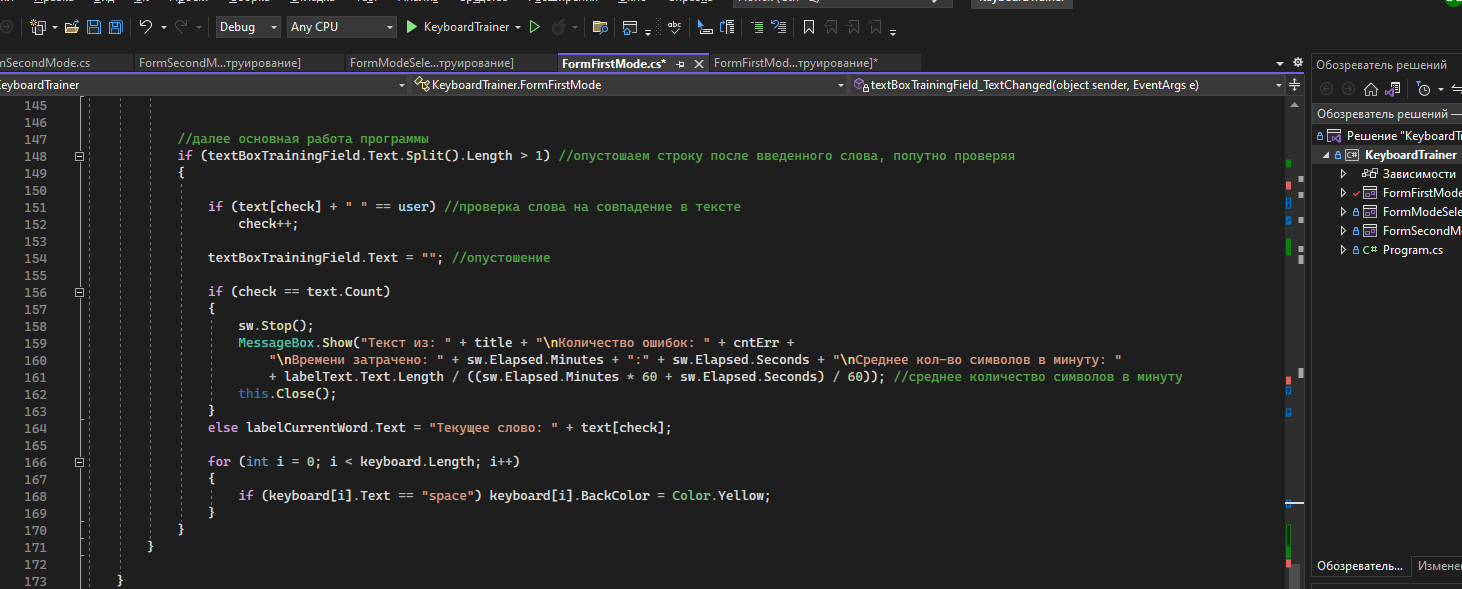


Рисунок 2.5. Главная часть кода первого режима.

Здесь при совпадении слова введенного пользователем и слова в тексте увеличивается check и соответственно начинает сравниваться со следующим словом. А когда check совпадет с длинной списка text это будет означать что пользователь ввел все слова и ему покажется результат в виде MessageBox.

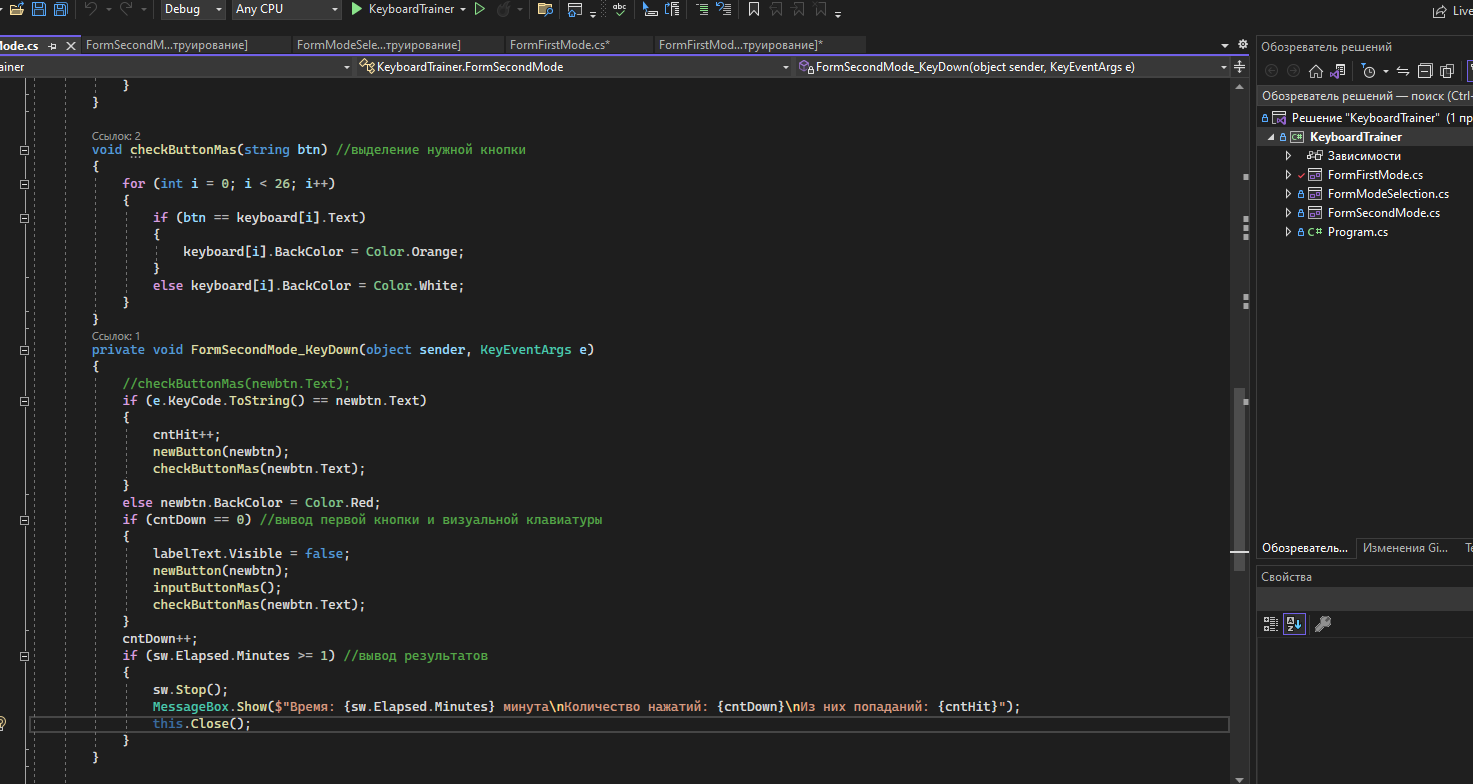
Со **вторым режимом** все куда проще. Визуальную клавиатуру я ввел таким же образом как и в первом режиме, только на английскую раскладку. А основная часть происходит в событии формы, которая срабатывает при нажатии пользователем на клавиши.

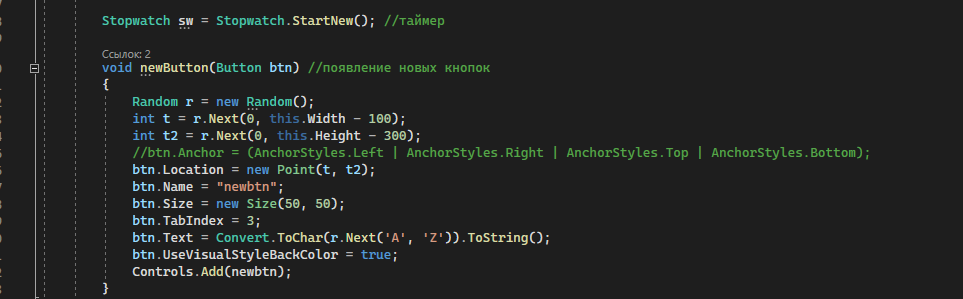
Рисунок 2.6. Главная часть кода второго режима.

Рисунок 2.7. Функция создания кнопки.

На рисунке 2.6. В условии if (cntDown == 0) программа стирает надпись «нажмите на любую клавишу» и в произвольном месте ставит новую кнопку при помощи функции на рисунке 2.7., а далее на каждом следующем нажатии по клавише программа проверяет правильно ли пользователь нажал. Если да, то опять выполняется функция из рисунка 2.7., то есть опять появляется новая кнопка, если же пользователь нажал неправильную клавишу то кнопка закрашивается красным.

По истечению минуты, которая так же как и в первом режиме считается через stopwatch программа показывает пользователю результат.

**ЗАКЛЮЧЕНИЕ**

Программа, конечно, далеко не идеальна. Не решена проблема с размером окна (при увеличении пользователям окна в первом режиме исчезает клавиатура), не проработан достаточно хорошо интерфейс. Но в целом приложение рабочее.

За время работы с кодом я узнал много нового. Например, научился работать с таймером, улучшил навыки работы с файлами, научился заводить массив кнопок и многое другое.

**СПИСОК ЛИТЕРАТУРЫ**

## 1. Троелсен и Джепикс. Язык программирования C# и платформы .NET и .NET Core

##### **2.** Стивен С. Скиена. Алгоритмы. Руководство по разработке

3. Чарльз Петцольд. Код. Тайный язык информатики

4. Полное руководство по языку программирования С# и платформе .NET. Получено из metanit.com: <https://metanit.com/sharp/tutorial/>

5. Официальная документация по С# [Документация по C# Microsoft Docs](https://docs.microsoft.com/ru-ru/dotnet/csharp/)