**Содержание**

[**1. Исследование предметной области** 1](#_Toc89349320)

[**1.1 Описание программы** 2](#_Toc89349321)

[**1.2 Основные функции и подфункции** 2](#_Toc89349322)

[**1.3 Распределение ролей в команде** 2](#_Toc89349323)

[1.3.1 Мясников Сергей Николаевич 2](#_Toc89349324)

[**2. Использование система контроля версий** 8](#_Toc89349325)

[**3. Тестирование информационной системы** 12](#_Toc89349326)

[**3.1 Тестируемые функции** 12](#_Toc89349327)

[**3.2 Наборы тестов для тестируемых функций** 12](#_Toc89349328)

[**3.3 Пример фрагмента кода с использованием Юнит-тестов** 13](#_Toc89349329)

[**3.4 Пример сценария тестирования** 16](#_Toc89349330)

[**4. Кодирование** 17](#_Toc89349331)

[**4.1 Правила форматирования кода** 17](#_Toc89349332)

[**4.2 Оценка цикломатической сложности** 18](#_Toc89349333)

[**4.3 метрика Холстеда** 18](#_Toc89349334)

[**5. Результат работы программы** 19](#_Toc89349335)

# **1. Исследование предметной области**

***Цель работы*** – необходимо проанализировать предметную область, а именно описание программы, участника, определения функций и подфункций, макет программы.

## **1.1 Описание программы**

Кроссворд – обучающая программа предназначена для внутреннего тестирования студентов «Волгоградского технического колледжа». Графическое приложение содержит уровни с разными нагрузками и вариантами. Для помощи в прохождении кроссворда студентам предоставлено окно с подсказками, которыми можно пользоваться в случае необходимости. Так же для понимания пользования приложения есть «справка», в которой предоставлена информация, как правильно пользоваться программой. Данная программа является «интеллектуальной» игрой, которую используют на сегодняшний день в целях развлечения и проверки эрудиции. Программа будет запускаться на современных операционных системах (Windows 7/10).

## **1.2 Основные функции и подфункции**

* *Вход в систему* – для начала тестирования приложение предоставляет студенту возможность ввести свои пользовательские данные (ФИО), после чего перейти в главное меню.
* *Выбор уровня* – для начала прохождения приложение дает возможность студенту выбрать уровень кроссворда.
* *Прохождение уровня* – непосредственно само прохождение, которое должен пройти студент за неограниченное время.
  + *Проверка на ввод латиницы и цифр* – подфункция, которая проверяет введенные символы на кириллицу.
  + *Право на ошибку* – подфункция, дающее возможность решить кроссворд еще раз.
* *Просмотр подсказок*– программа дает возможность студенту посмотреть подсказку по нужному ему слову.
  + *Выбор подсказки* – подфункция, которая дает возможность выбрать нужную цифру по вертикали или горизонтали, для получения подсказки по данному слову.
* *Просмотр справки* – окно для помощи студентам при прохождении кроссворда, где выводится правила игры.

## **1.3 Распределение ролей в команде**

### 1.3.1 Мясников Сергей Николаевич

Студент ПКС 1-19

**Тим-Лидер**, разработчик функционала программы (разработка всего функционала программы, разработка дизайна, создание структуры базы данных, создание макеты, создание классов и методов, работа с репозиторием и т.д)

**1.4 Макет программы**

Ниже будет представлен макет «программы Кроссворд».

1. Макет Входа

По центру окна находится поле для ввода *никнейма* – это сделано для того, чтобы студент мог ввести своё имя, которое будет отображаться в главном меню.

Снизу по центру находится кнопка “Начать” – это сделано для того, чтобы студент после ввода *никнейма* мог перейти в главное меню.

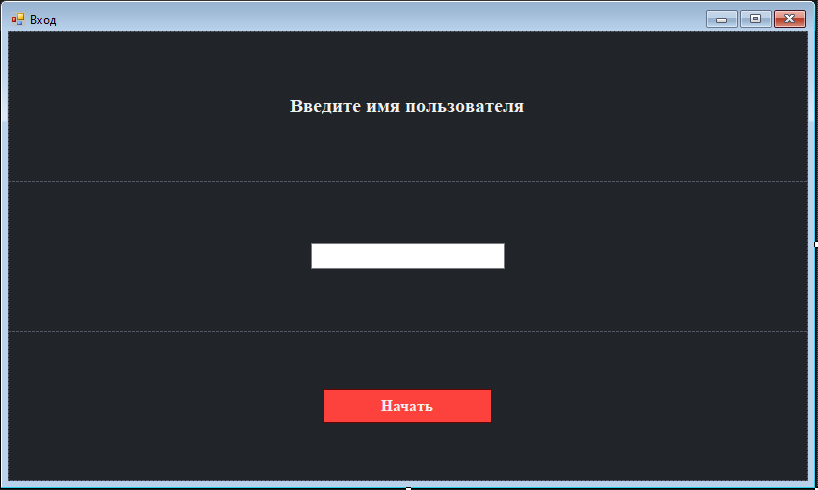
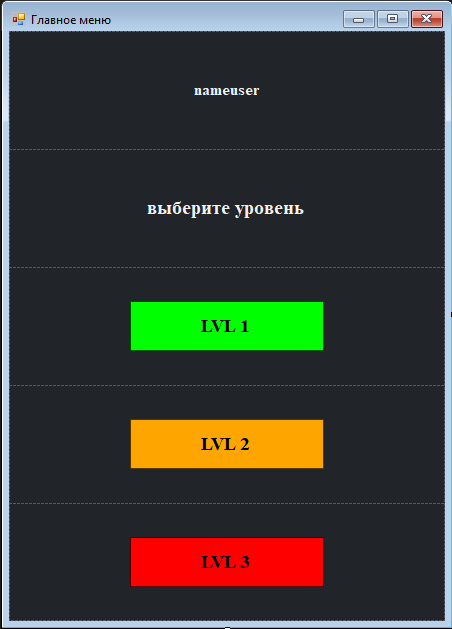


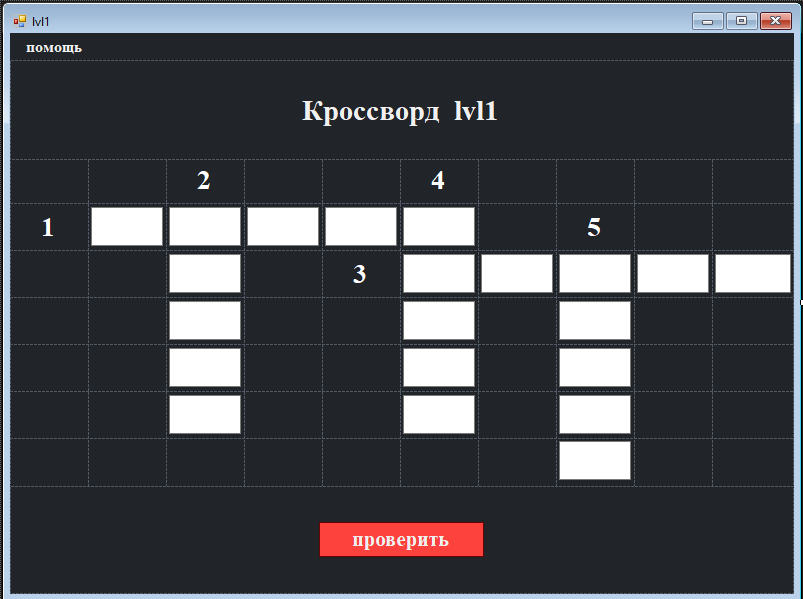
Рисунок 1 – Вход программы “кроссворд”

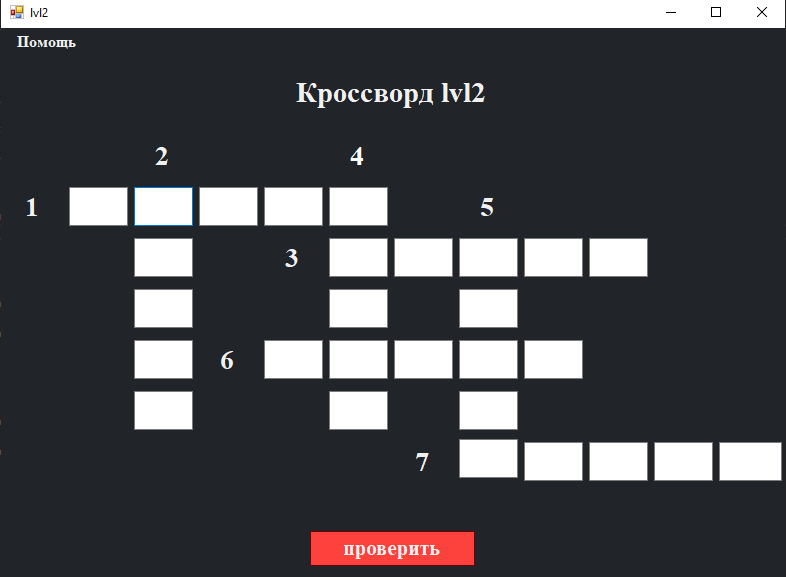
2. Макеты программы после нажатия на кнопку «Начать»

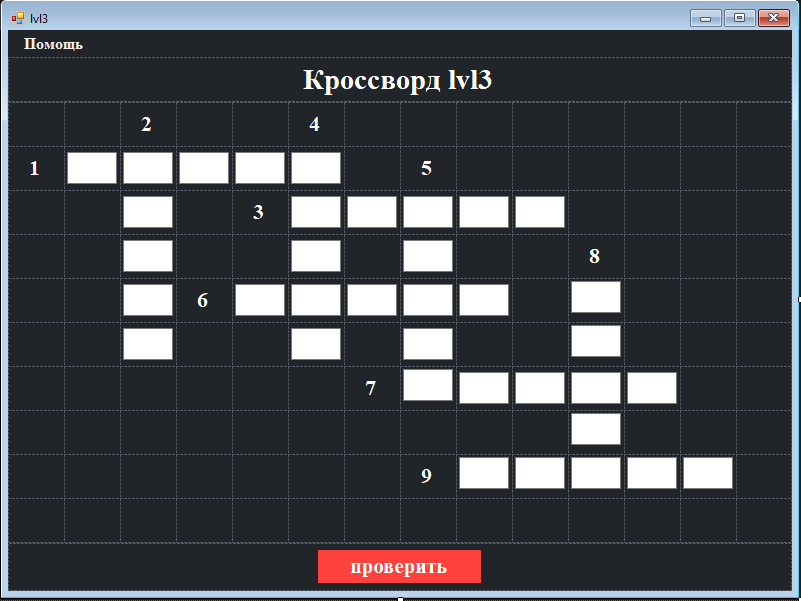
Снизу по центру представлены 3 кнопки “lvl1”, “lvl2”, “lvl3” – это сделано для того, чтобы пользователь при нажатии на кнопку мог перейти на прохождение кроссворда. С каждым уровнем кроссворд будет увеличиваться в размерах полей и колличестве слов.



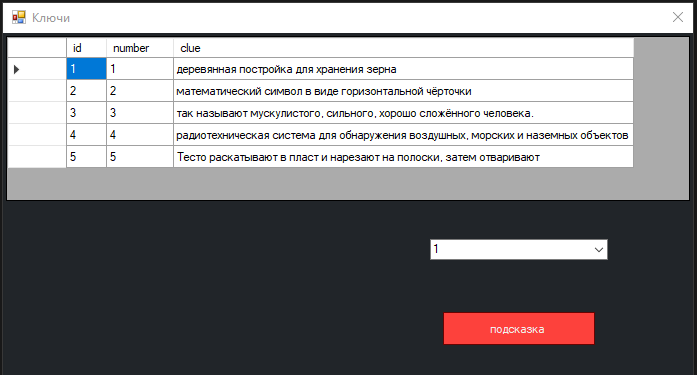
3. Макет уровней







4. Макет окна подсказок



Вывод: предложенный нами интерфейс программы удовлетворяет фактору качества разработки ИС, как удобство пользования потому что он выполнен в минималистическом стиле и поэтому такой вид интерфейса не будет нагружать студента лишней информацией и «простой вид» интерфейса позволяет легче его понять.

# **2. Использование система контроля версий**

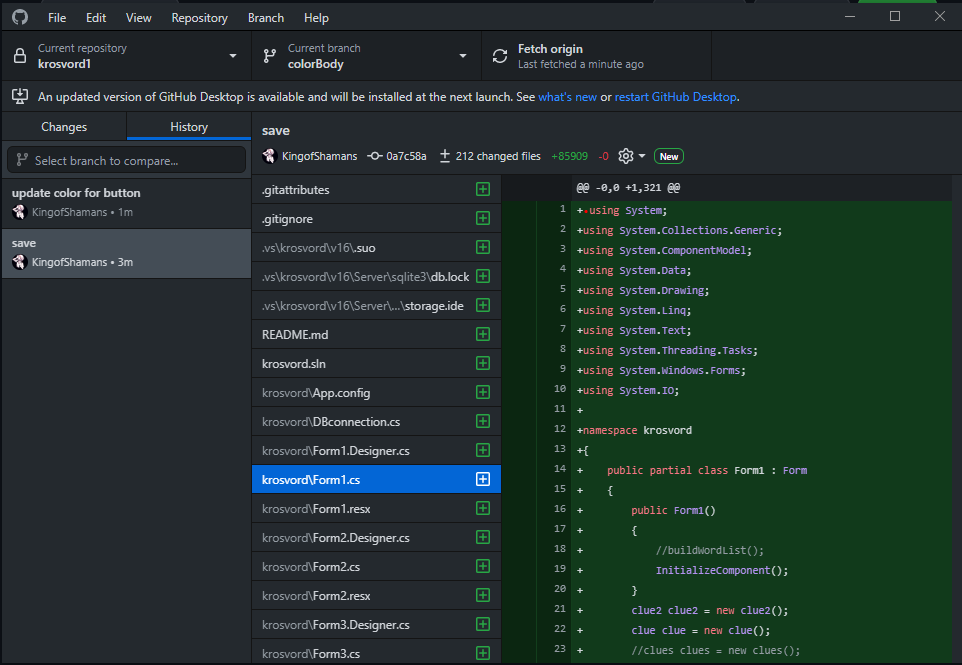
***Цель работы*** – работа и использование системы контроля версий, привести пример коммита, дерево ведения и ветки проекта.



Любая программа должна реализовываться с использованием распределенной системы контроля версий и наша не исключение. Для нашего приложения мы используем бесплатный и крупнейший веб-сервис GitHub (<https://github.com>).

Логотип GitHub

1. Пример коммита программы:



**Описание коммита**

**Название коммита**

**Ветка**

**Какие файлы были изменены**

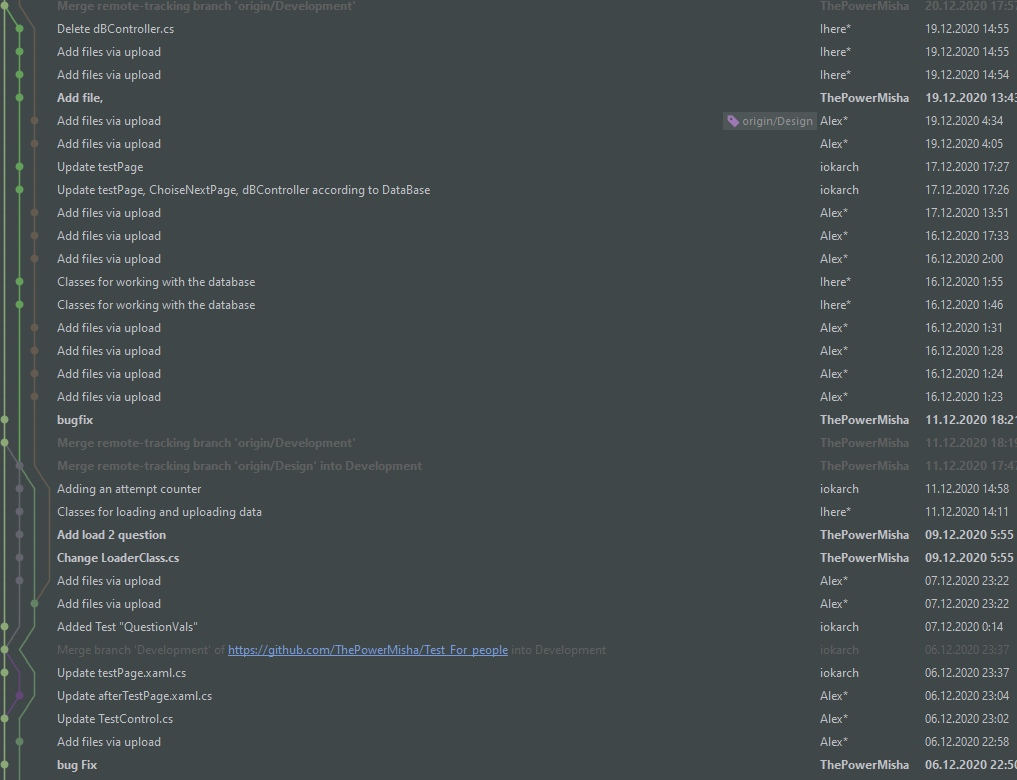
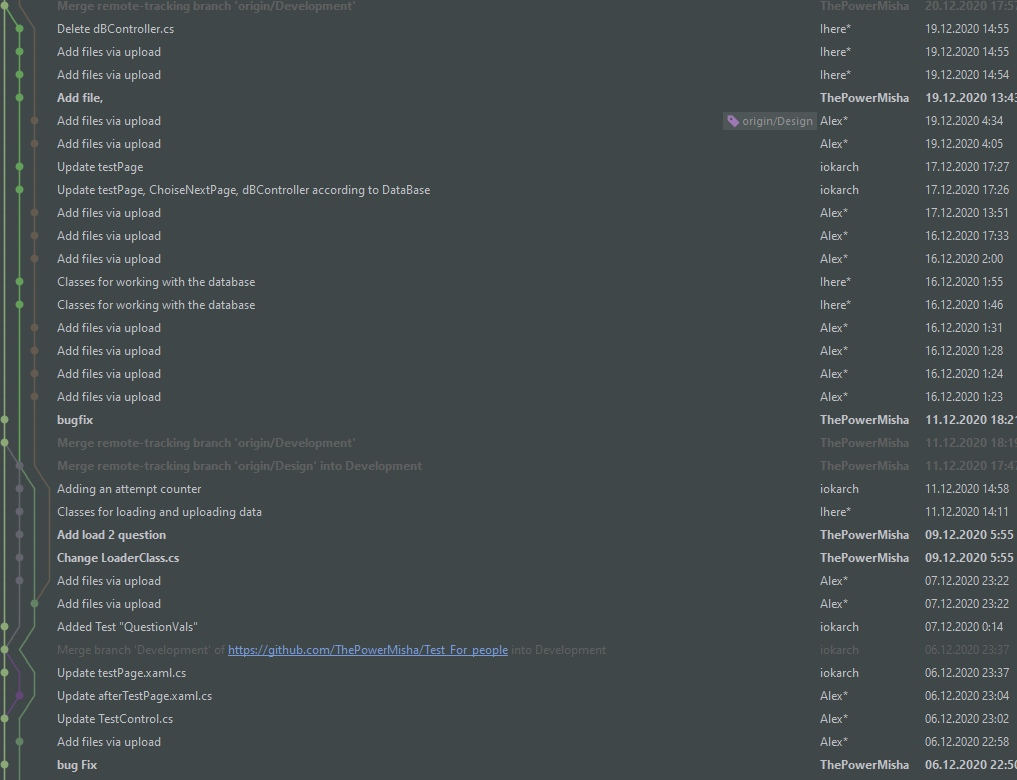
**Репозиторий**

**Коммит**

Ниже представлено дерево веток проекта

* Ветка main – основная ветка, в которую сливаются ветки
* Ветка Development – ветка, в которой реализуются основные функции программы
* Ветка Design – ветка, в которой реализуется графический интерфейс

Рисунок 2 – Дерево веток проекта



main

Development

Design

Название коммитов

Автор коммита

Дата коммита

3. Пример веток

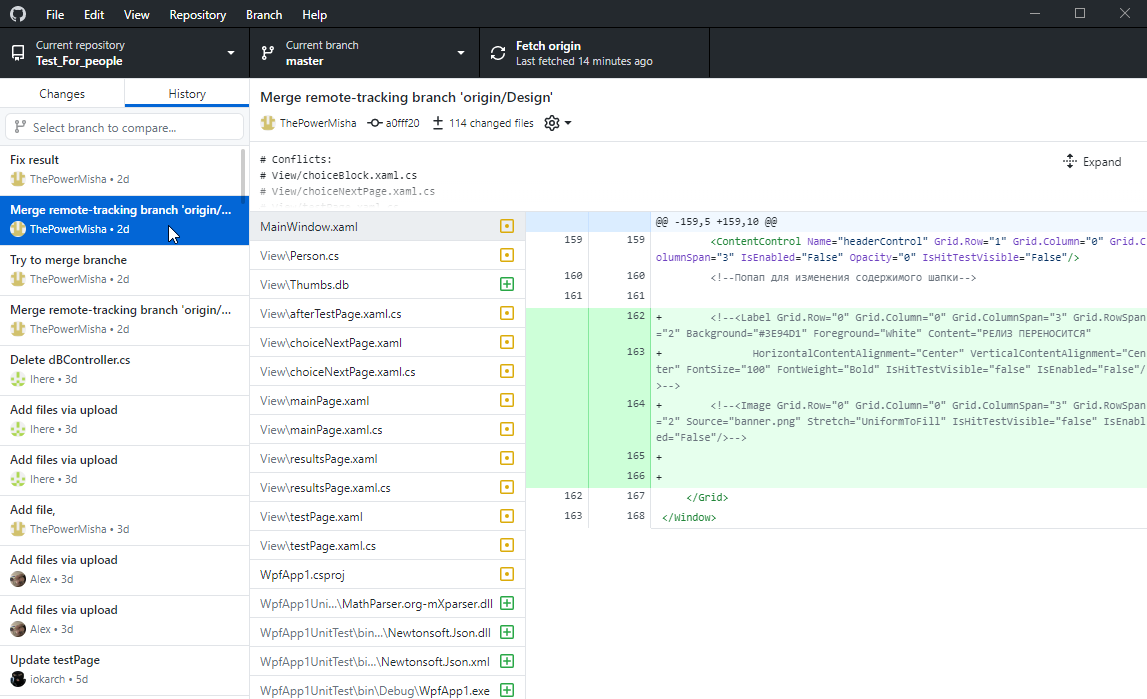
 3.1 Ветка ***master –*** основная ветка, в которую сливаются ветки

Рисунок 3 – Ветка master

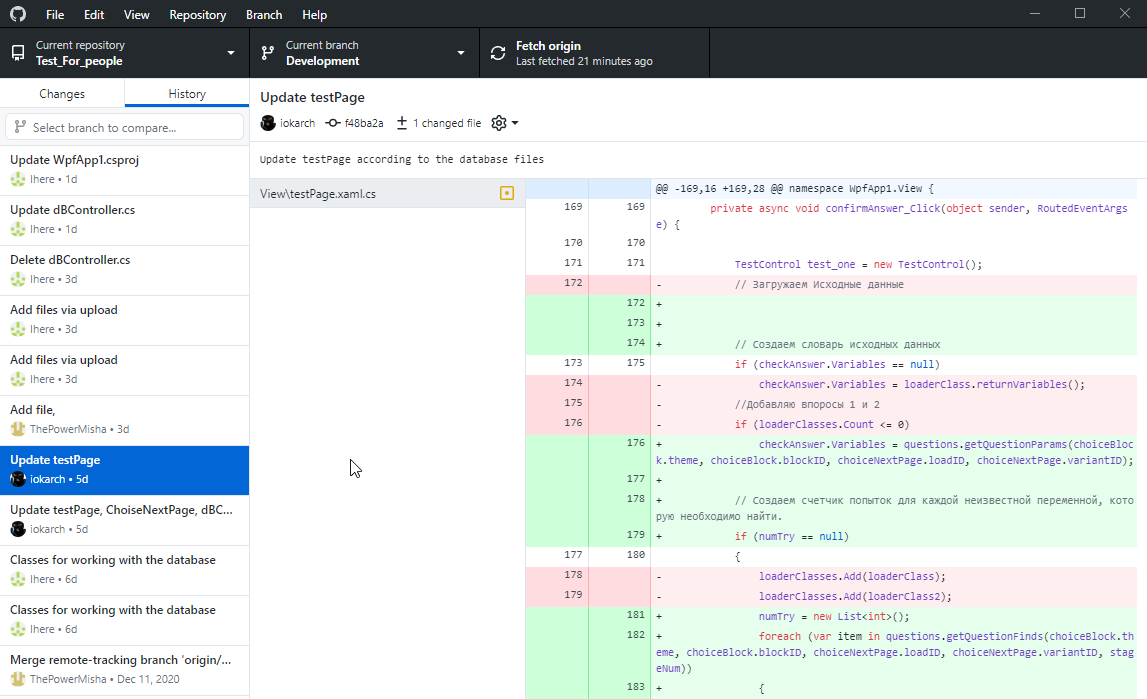
 3.2 Ветка Development – ветка, в которой реализуются основные функции программы

Рисунок 4 - ветка development

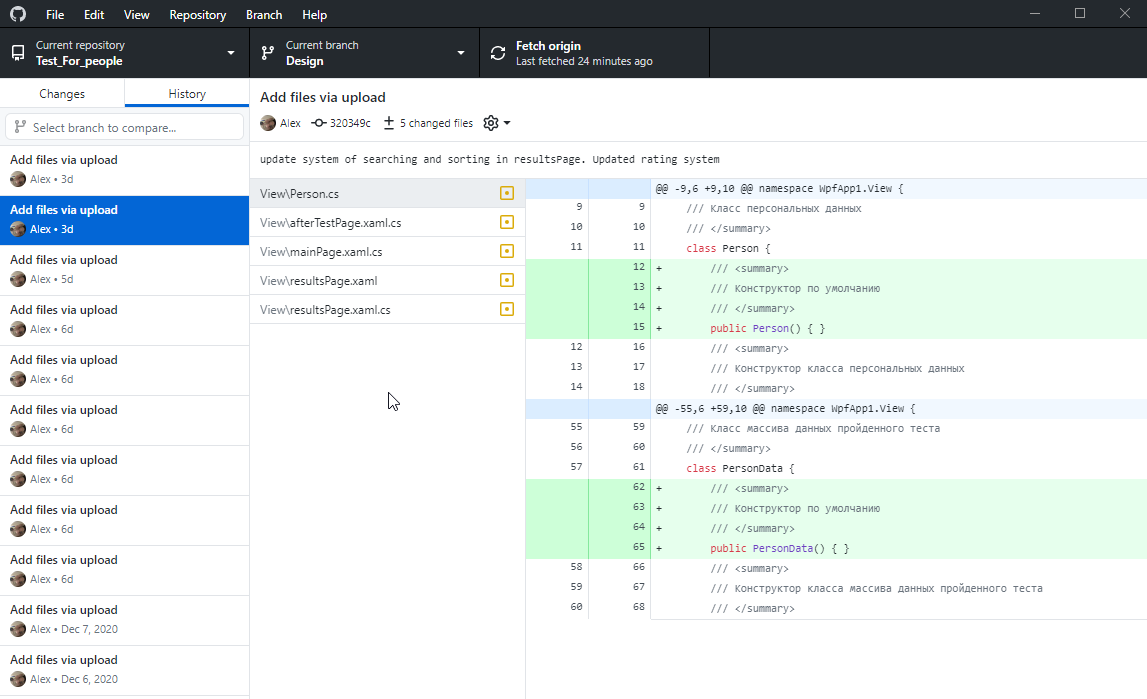
3.3 Ветка Design – ветка, в которой реализуется графический интерфейс

Рисунок 5 – Ветка design

# **3. Тестирование информационной системы**

***Цель работы*** – Протестировать программный код с помощью технологий автоматизированного и ручного тестирования.

## **3.1 Тестируемые функции**

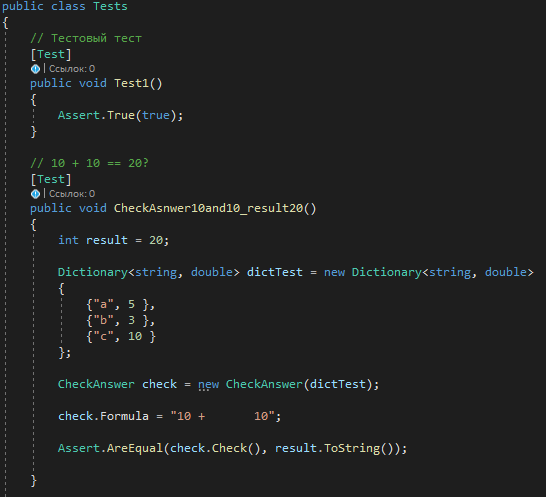
* Checkwords – check
* Checkwords – SyntaxСheck
* Checkwords – BracketCheck
* dBController – getcluesText
* dBController – getcluesFinds

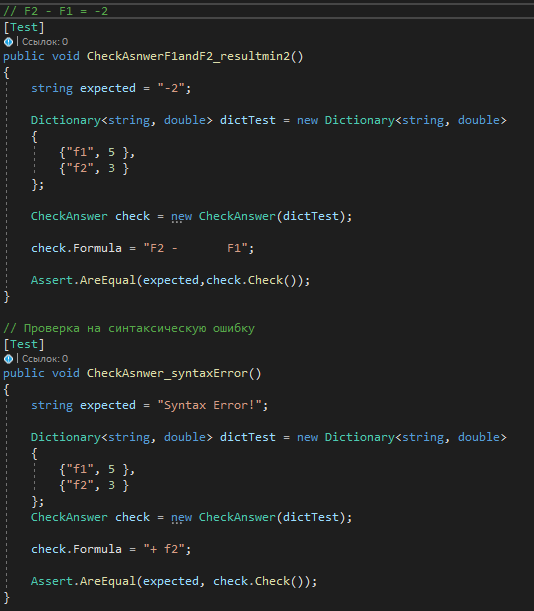
## **3.2 Наборы тестов для тестируемых функций**

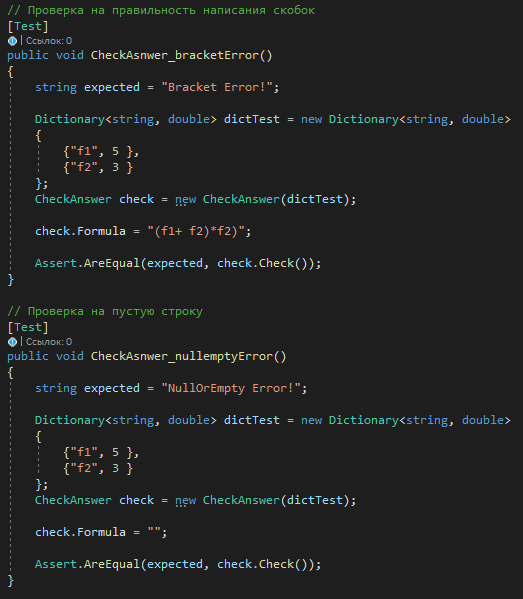
* check – Проверка на расчет введенной функции
* check – Проверка на кириллицу
* SyntaxСheck – Проверка на неправильно введённое слово
* getcluesText – Получить текст из БД
* getcluesFinds – Получить список списков подсказок

## **3.3 Пример фрагмента кода с использованием Юнит-тестов**

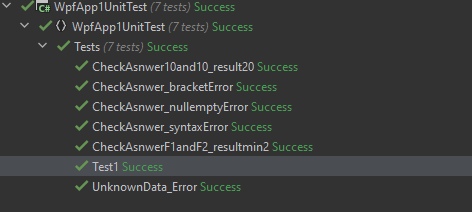
Для Юнит-теста мы взяли проверку функцию check класса Checkwords





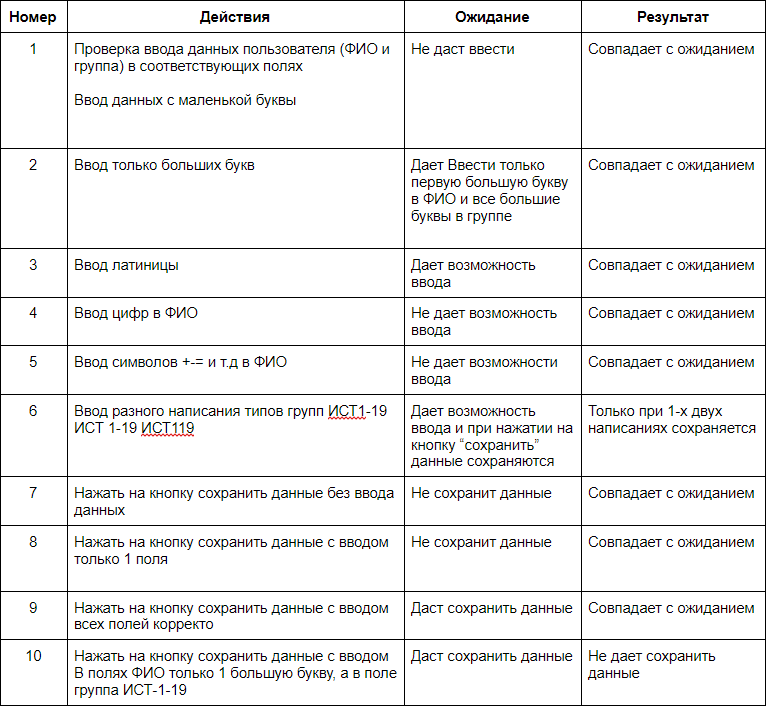


Результат работы Юнит-теста:



## **3.4 Пример сценария тестирования**

Для сценария тестирования мы взяли *ввод пользовательских данных* в главном меню приложения.



# **4. Кодирование**

***Цель работы*** – соблюдать кодирование согласно правилам форматирования кода. Выполнить оценку цикломатической сложности и использование метрики Холстеда.

## **4.1 Правила форматирования кода**

* Нотация кода – camelCase.
* Фигурные скобки на той-же строке что и функция
* Табуляция в 4 пробела
* Между разными функциями интервал 1 строка
* Стиль написания комментариев – над функциями, объединение общих элементов
* Краткое описание всех открытых функциях и входящих в них переменных через *///summary*

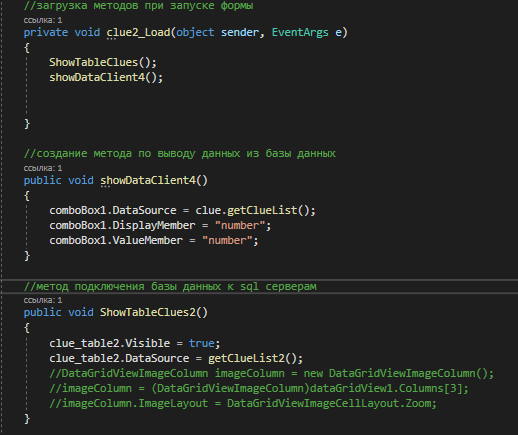


Рисунок 6 – Пример 1

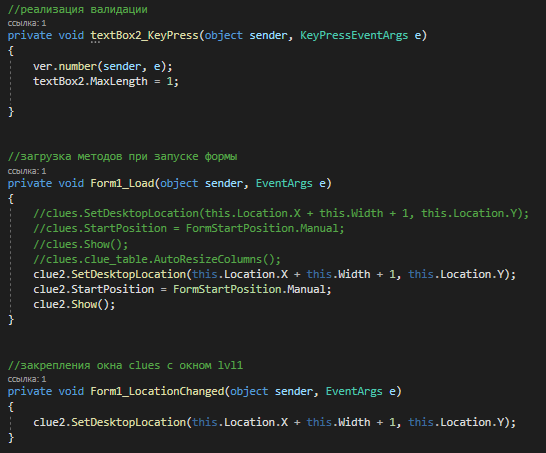
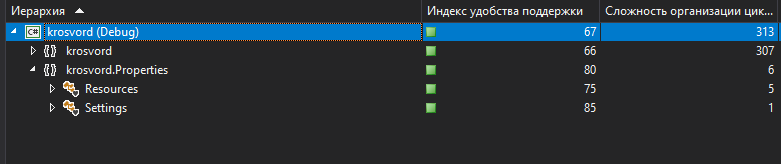
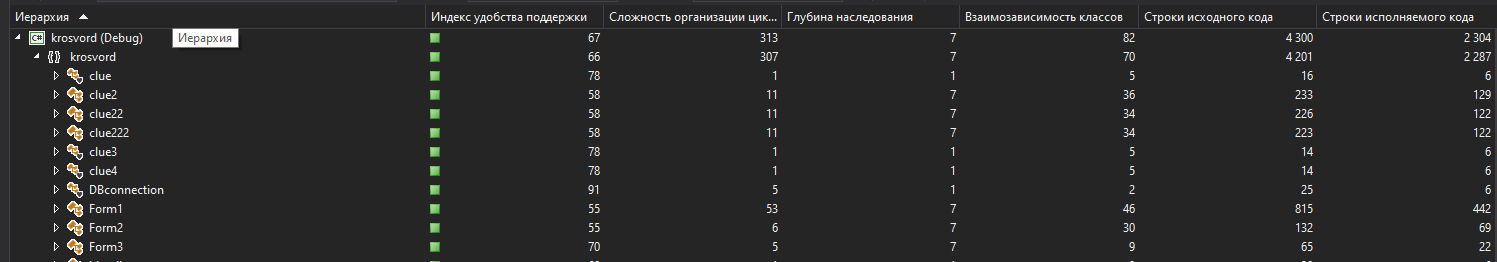


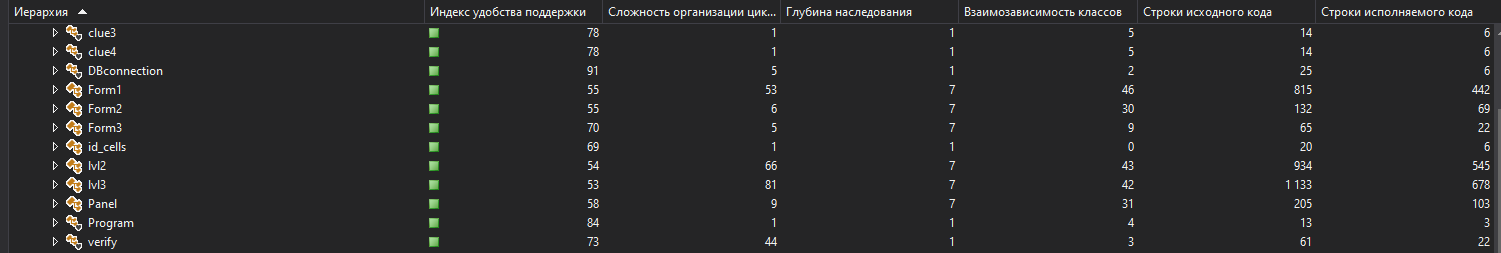
Рисунок 7 – Пример 2

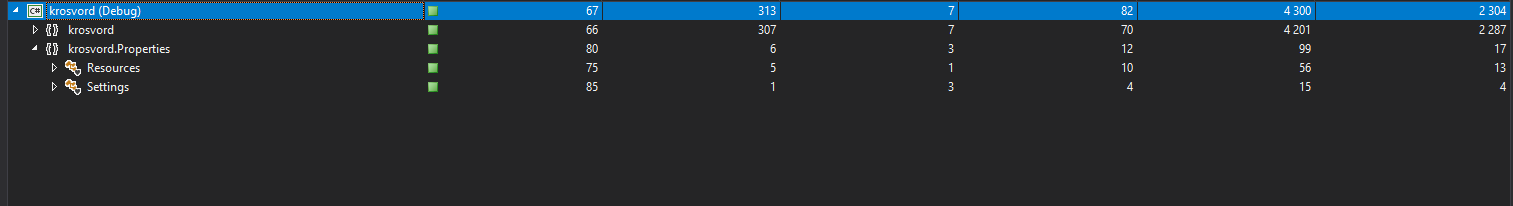
## **4.2 Оценка цикломатической сложности**



## **4.3 метрика Холстеда**







# **5. Результат работы программы**

***Цель работы*** – показать работоспособность программы в целом или отдельных функций.

5.1. Пример работы «Вход»

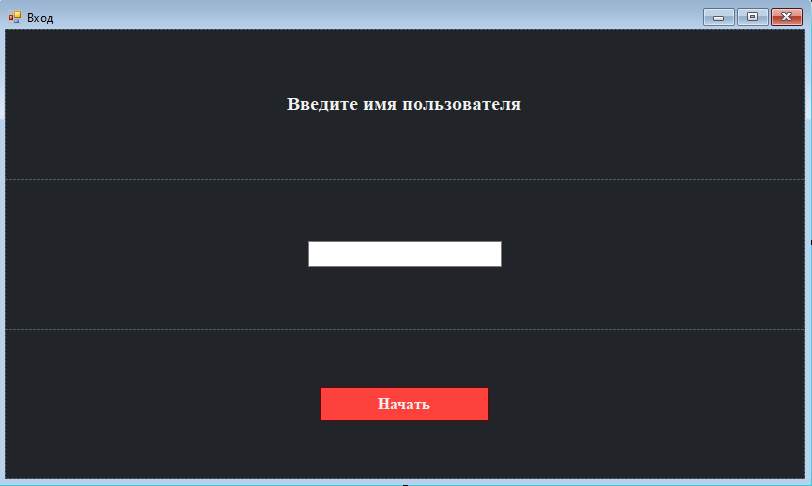


Рисунок 8 – данные не введены

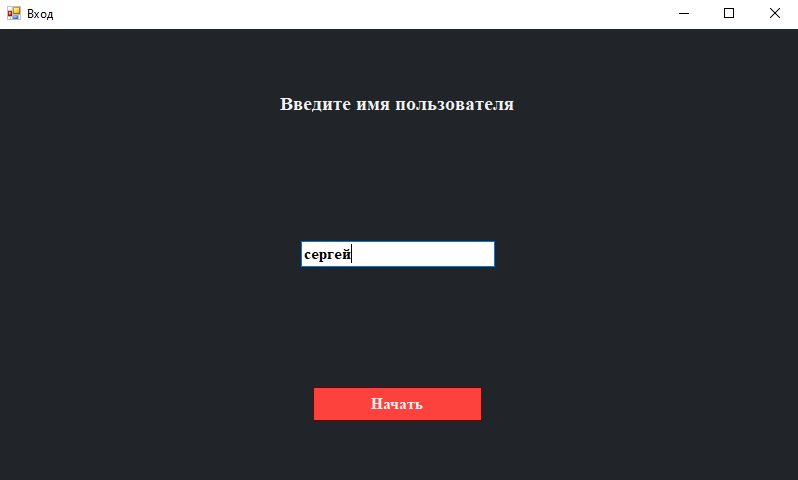


Рисунок 9 – заполнили данные

5.2 Пример работы выбора части/положения/варианта



Рисунок 10 – Главное меню уровни не выбраны

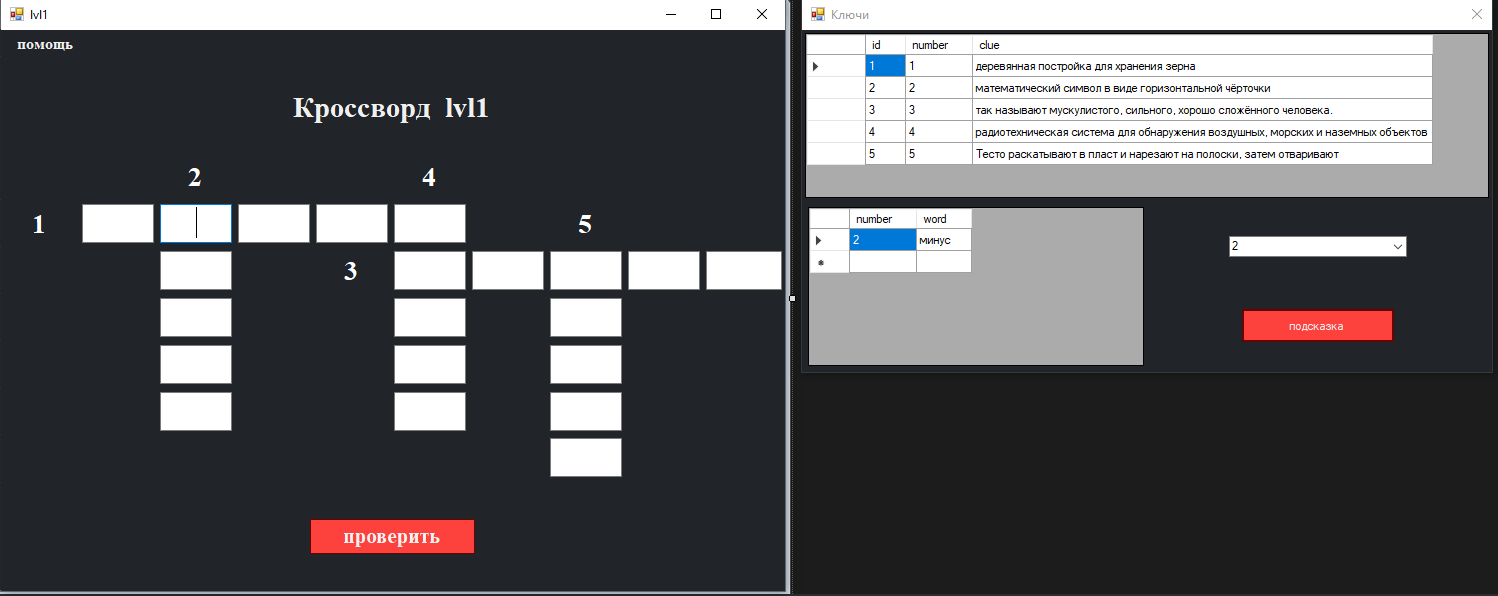


Рисунок 11 – Запуск одного из уровней

5.3 Пример тестирования

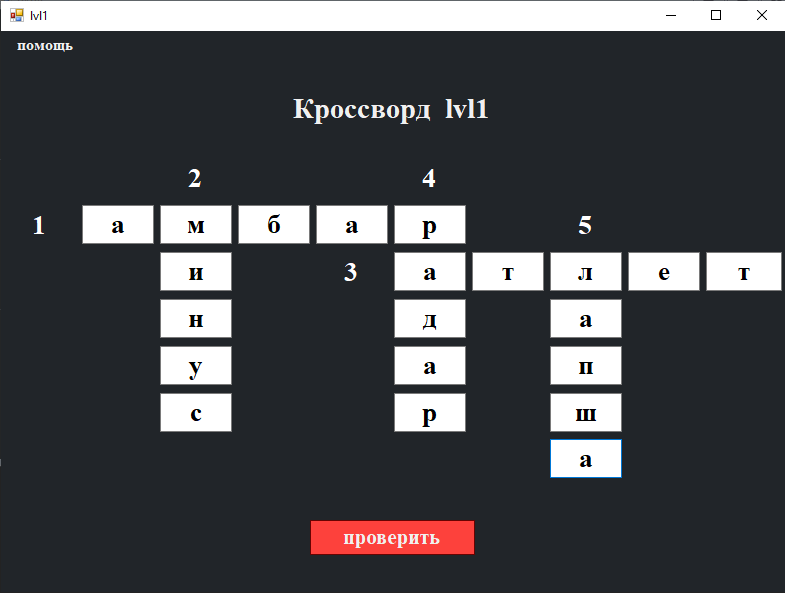


Рисунок 12 – окно тестирования уровня(ввод слов)

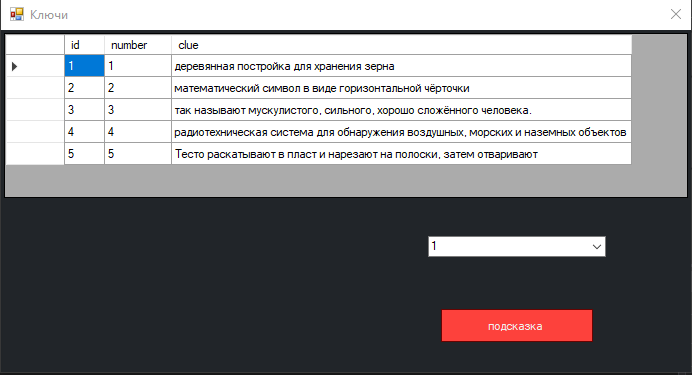


Рисунок 13 – окно тестирования “ключи”

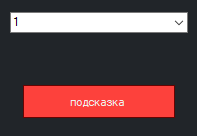


Рисунок 14 – окно тестирования “ключи” выбор номера слова для подсказки и нажатие кнопки

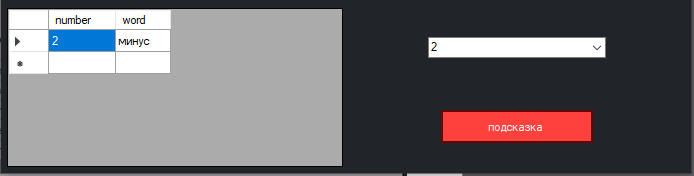
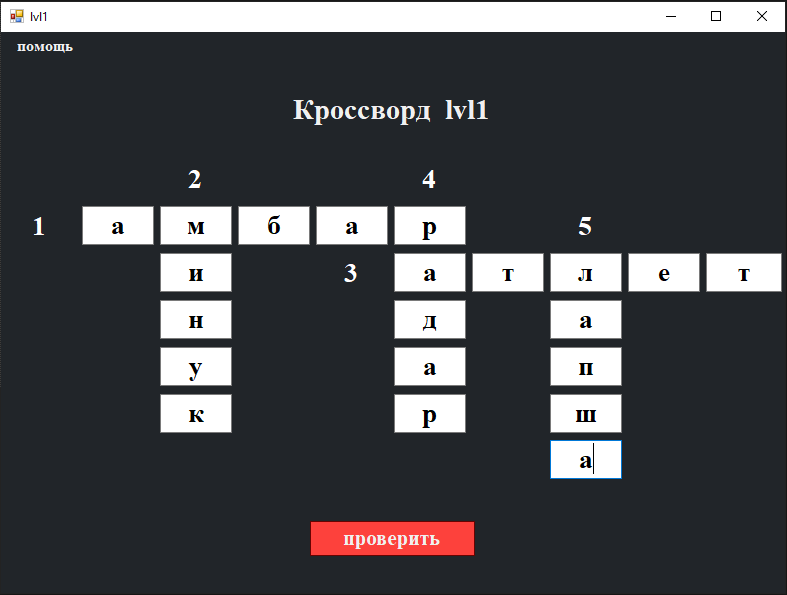


Рисунок 15 – окно тестирования “ключи” вывод подсказки слева



Нажатие кнопки

Рисунок 16 – окно тестирования уровня, нажатие на кнопку “проверить”

5.4 Пример работы кнопки “проверить”

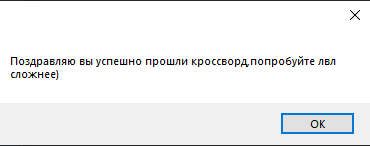


Рисунок 17 – окно сообщения после удачного прохождения

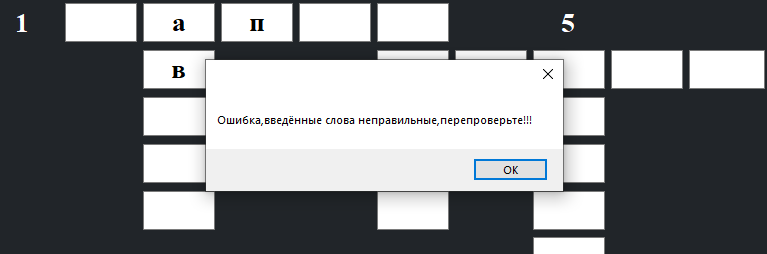


Рисунок 18 – окно сообщения после неудачного прохождения

5.5 Пример работы кнопки помощь



Рисунок 19 – кнопка помощь на верхней панели

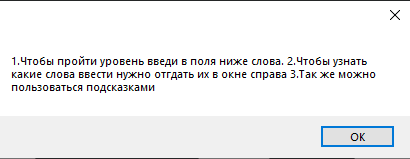


Рисунок 20 – окно сообщения после нажатия на “помощь”

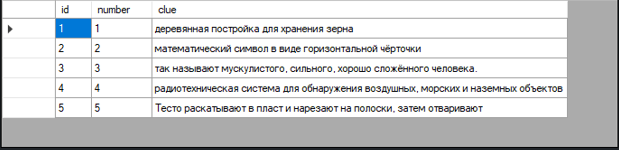


Рисунок 21 – строение базы данных “clues”