**Частное учреждение образования «Минский колледж предпринимательства»**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**ИГРА «EQUATION»**

**Пояснительная записка**

**МКП.3190607-01 81 01**

*technicaldocs.ru*

*Подпись и дата*

*Взам. инв. №*

*Инв. № дубл.*

*Инв. № подл.*

*Подпись и дата*

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  | Учащийся |
|  |  |  |
|  |  | Д.С. Брикез |
|  |  | «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2022 г. |
|  |  |  |
|  |  | Руководитель |
|  |  |  |
|  |  | И.В. Жилинский |
|  |  | «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2022 г. |
|  |  |  |
|  |  |  |
|  |  |  |

2022

АННОТАЦИЯ

Курсовой проект на тему «Игра Equation», состоит из программного продукта, пояснительной записки и графической части.

Игра предназначена для развлечения пользователя.

Программный продукт разработан в среде программирования Microsoft Visual Studio 2022 с использованием языка программирования C# и фреймворка WPF.

Пояснительная записка выполнена на 41 листе, содержит 4 раздела и одно приложение.

Графическая часть выполнена на двух листах, включает диаграмму компонентов и диаграмму вариантов использования.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 4

1. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 6

2. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 7

2.1. Постановка задачи 7

2.2. Описание организации данных 7

2.3. Описание программных средств 8

2.4. Проектирование интерфейса 9

2.5. Описание логической структуры программы 13

3. ТЕСТИРОВАНИЕ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ 15

4. РУКОВОДСТВО ПО ИСПОЛЬЗОВАНИЮ ПРОГРАММНОГО СРЕДСТВА 24

ЗАКЛЮЧЕНИЕ 31

ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ 32

ПЕРЕЧЕНЬ ПРИНЯТЫХ СОКРАЩЕНИЙ 33

ПРИЛОЖЕНИЕ 1 (ОБЯЗАТЕЛЬНОЕ) – ЛИСТИНГ ПРОГРАММЫ 34

ВВЕДЕНИЕ

Существует множество игр разного жанра.

Стратегическая игра – игра в которых игроку для победы необходимо применять стратегическое мышление. В популярных играх такого жанра игроку предлагается играть не за конкретного персонажа, а за их условные массы, к примеру, руководить строительством города или командовать целыми армиями в военных кампаниях.

Аркада – игра характеризующаяся коротким по времени, но интенсивным игровым процессом.

Головоломка – игра целью которой является решение логических задач, требующих от игрока задействования логики, стратегии и интуиции.

И множество других. Каждый день появляется десятки новых игр разного жанра для развлечения пользователей.

Главной целью курсового проектирования является создание компьютерной игры «Equation», которое призвано разнообразить и скрасить досуг пользователя. А также расширение знаний по дисциплине «Конструирование программ и языки программирования». Получение навыков проектирования и создания программного обеспечения.

Задачи выполнения курсового проекта:

* систематизация полученных знаний;
* углубление уровня и расширение объема профессионально значимых знаний, умений и навыков;
* формирование умений и навыков самостоятельной работы;
* овладение современными методами поиска, обработки и использования информации.

Пояснительная записка содержит следующие разделы:

* раздел «Назначение и область применения» описывает назначение программы, краткую характеристику области применения программы;
* раздел «Технические характеристики» описывает: постановку задачи, организацию данных, программные средства, проектирование интерфейса, логическую структуру программы;
* раздел «Тестирование и анализ полученных результатов» – описывается проверка работы программы на различных тестах и приводится анализ ожидаемых и полученных результатов, выводы по результатам тестирования;
* раздел «Руководство по использованию программного средства» – приведение последовательности действий пользователя, обеспечивающих загрузку, запуск, выполнение и завершение программы, приведение описание функций.

1. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

Основное назначение игры «Equation» развлечение пользователя и развитие математических способностей.

Область применения: сфера развлечений

Разработанная программа имеет хорошую оптимизацию, достаточно малую ресурсоемкость и дружественный интерфейс, что делает данную игру на порядок выше многих своих аналогов.

Аналогом разрабатываемой игры является игра «Math Riddles and Puzzles».

Плюсы разрабатываемой игры по сравнению с аналогом:

1. Присутствие таблицы рекордов.
2. Хорошая совместимость с операционной системой Windows.
3. Приятный и удобный дизайн.

Минусы разрабатываемой игры по сравнению с аналогом:

1. Игра совместима только с операционной системой Windows.
2. Небольшое количество уравнений.
3. Отсутствие масштабирование компонентов под разное разрешение

экрана.

Программное средство разработано для ОС Windows.

Минимальные условия и требования к аппаратному обеспечению:

* процессор «Intel(R) Core(TM) i5-1021OU CPU @ 1.60GHz»;
* оперативная память 8гб;
* операционная система Windows 10 Pro;
* видеоадаптеры Intel(R) UHD Graphics;
* свободное место на диске не менее 1000 мб;
* 64 - разрядная ОС.

Фиксированных максимальных требований нет. Подойдут сборки с минимальными требованиями и выше.

1. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ
   1. Постановка задачи

Разработать развлекательную игру «Equation» предназначенную для развлечения пользователя. Суть игры заключается в прохождении максимального количества раундов. Для пользователя реализовать следующие пункты меню:

1. Регистрация. При нажатии на кнопку «Registration» из главного меню реализовать переход на форму с регистрацией. На данной форме по нажатию на кнопку «Registration» реализовать возможность регистрации. Для регистрации необходимо ввести свое имя, пароль и повторно ввести пароль. пароль и после нажатия на кнопку «Registration» если аккаунта с таким именем не существует, то пользователь будет зарегистрирован.
2. Авторизация. При нажатии на кнопку «Autorisation» из главного меню реализовать переход на форму с авторизацией. На данной форме по нажатию на кнопку «Autorisation» реализовать возможность авторизации. Для авторизации необходимо ввести своего имя, пароль и после нажатия на кнопку «Autorisation» если аккаунт с таким именем существует и пароль совпадает, то пользователь будет авторизирован.
3. Tаблица рекордов. При нажатии на кнопку «Table recordes» из главного меню реализовать переход на форму с таблицей рекордов. На форме с таблицей рекордов реализовать функцию поиска введя имя игрока под заголовком «Name» и далее после нажатия на кнопку «Search» должен быть осуществлен поиск игрока. По нажатию на кнопку «Sort» реализовать функцию сортировки таблицы рекордов. По нажатию на кнопку «Update» реализовать функцию обновления таблицы рекордов.
4. Игра. При нажатии на кнопку «Play game» из главного меню реализовать переход на форму с игрой. На форме с игрой по нажатию на кнопку с уравнением реализовать функцию увеличение очков на результат уравнения. Реализовать функцию увеличения сложности после каждого выбранного уравнения, функцию отнятия очков в зависимости от сложности после каждого выбранного уравнения, функцию проигрыша если после отнятия очков у пользователя их осталось отрицательное количество.
   1. Описание организации данных

Данные, получаемые и изменяемые в результате работы приложения, сохраняются в конфигурационных файлах приложения.

Входные данные: имя пользователя, пароль, нажатия на правую кнопку мыши.

Выходные данные: таблица рекордов.

* 1. Описание программных средств

В данном проекте используется язык программирования C#, плюсами которого являются:

1. Поддержка подавляющего большинства продуктов Microsoft.
2. Бесплатность ряда инструментов для небольших компаний и некоторых индивидуальных разработчиков – Visual Studio, облако Azure, Windows Server, Parallels Desktop для Mac Pro и др.[1]
3. Типы данных имеют фиксированный размер (32-битный int и 64-битный long), что повышает «мобильность» языка и упрощает программирование, так как вы всегда знаете точно, с чем вы имеете дело. [2]
4. Автоматическая «сборка мусора». Это значит, что нам в большинстве случаев не придётся заботиться об освобождении памяти. Вышеупомянутая общеязыковая среда CLR сама вызовет сборщик мусора и очистит память.
5. Большое количество «синтаксического «сахара» - специальных конструкций, разработанных для понимания и написания кода. Они не имеют значения при компиляции.
6. Низкий порог вхождения. Синтаксис C# имеет много схожего с другими языками программирования, благодаря чему облегчается переход для программистов. Язык C# часто признают наиболее понятным и подходящим для новичков. [3]
7. С помощью Xamarin на C# можно писать программы и приложения для таких операционных систем, как iOS, Android, MacOS и Linux.[4]
8. Сегодня имеется немало вакантных мест на должность C# –программиста.

Однако, кроме положительных сторон, у C# есть и отрицательные стороны:

1. Приоритетная ориентированность на платформу Windows.
2. Язык бесплатен только для небольших фирм, индивидуальных программистов, стартапов и учащихся. Крупной компании покупка лицензионной версии этого языка обойдётся в круглую сумму. [5]

Также, для написания графиеского приложения был выбран особый фреймворк под названием WPF (Windows Presentation Foundation), который имеет такие отличительные черты, как:

1. Гибкая работа с Frontend и Backend частями приложения.
2. Простота обучения.
3. Оптимизированная работа работающего приложения.

К минусам WPF можно отнести:

1. Многие стандартные элементы других проектов могут попросту не запуститься из-за различности в типах проектов.
2. Большое количество потоков иногда нарушает работу приложения.
   1. Проектирование интерфейса

Приложение должно иметь 5 форм:

1. Главное меню.
2. Таблица рекордов.
3. Авторизация.
4. Игра.
5. Регистрация.

Страницы «Главное меню», «Регистрация», «Авторизация», «Игра», «Таблица рекордов» реализованы как формы, при нажатии на кнопку из Главного меню должен произойти переход на другую форму.

На рисунке 2.1. показана форма «Главное меню».

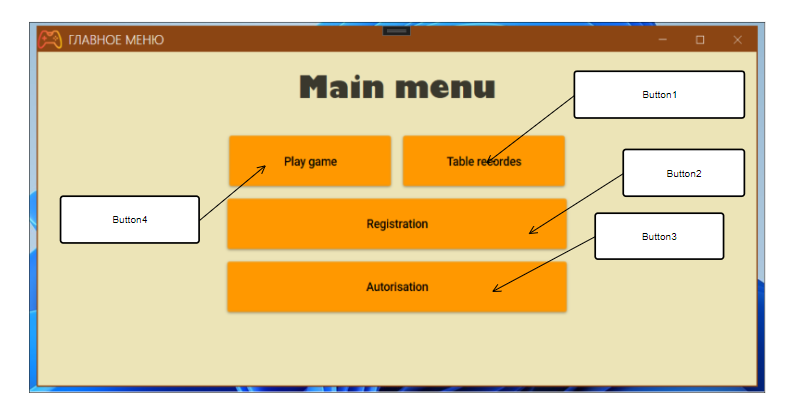


Рис. 2.1. Форма «Главное меню»

Таблица 2.1. – Компоненты формы «Главное меню»

|  |  |
| --- | --- |
| Наименование компонента | Назначение компонента |
| Button1 | Кнопка для перехода к форме с таблицей рекордов. |
| Button2 | Кнопка для перехода к форме с регистрацией. |
| Button3 | Кнопка для перехода к форме с авторизацией. |
| Button4 | Кнопка для перехода к форме с игрой. |

На рисунке 2.2 показана форма «Таблица рекордов».



Рис. 2.2. Форма «Таблица рекордов»

Таблица 2.2 – Компоненты формы «Таблица рекордов»

|  |  |
| --- | --- |
| Наименование компонента | Назначение компонента |
| TableRecordes | Таблица отображающая таблицу рекордов. |
| TextBox1 | Поле для ввода имени искомого человека. |
| Button1 | Кнопка для сортировки таблицы рекордов. |
| Button2 | Кнопка для поиска пользователя по введённому имени. |
| Button3 | Кнопка для обновления таблицы рекордов. |

На рисунке 2.3 показана форма «Авторизация».

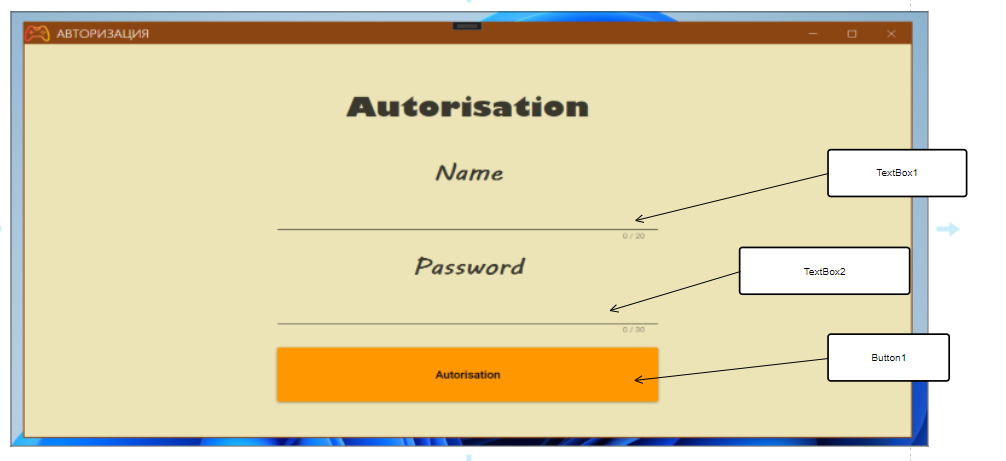


Рис. 2.3. Форма «Авторизация»

Таблица 2.3 – Компоненты формы «Авторизация»

|  |  |
| --- | --- |
| Наименование компонента | Назначение компонента |
| TextBox1 | Поле для ввода имени. |
| Button1 | Кнопка при нажатии на которую происходит авторизация. |
| TextBox2 | Поле для ввода пароля. |

На рисунке 2.4 показана форма «Игра».

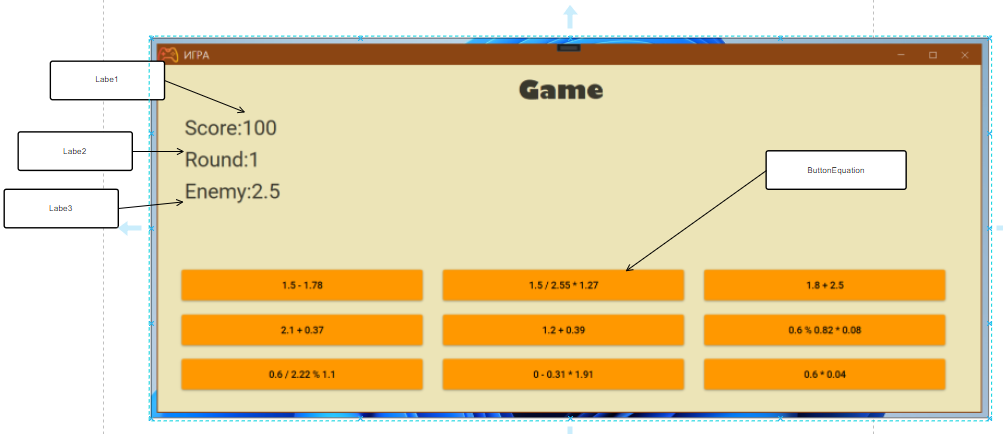


Рис. 2.4. Форма «Игра»

Таблица 2.4 – Компоненты формы «Игра»

|  |  |
| --- | --- |
| Наименование компонента | Назначение компонента |
| Button1 | Кнопка при нажатии на которую высчитываются очки. |
| Label1 | Поле для отображения количества очков |
| Label2 | Поле для отображения количества прошедших раундов. |
| Label3 | Поле для отображения количества отнимаемых очков в этом ходу. |

На рисунке 2.5 показана форма «Регистрация».



Рис. 2.5. Форма «Регистрация»

Таблица 2.5 – Компоненты формы «Регистрация»

|  |  |
| --- | --- |
| Наименование компонента | Назначение компонента |
| TextBox1 | Поле для ввода имени. |
| Button1 | Кнопка при нажатии на которую происходит регистрация. |
| TextBox2 | Поле для ввода пароля. |
| TextBox3 | Поле для повторного ввода пароля. |

* 1. Описание логической структуры программы
* public void Registration() – функция добавление аккаунта в базу данных;
* public void Autorisation() – функция для авторизации пользователя;
* public void Update() – функция для обновления таблицы рекордов пользователя;
* public void Sort() – функция для сортировки таблицы рекордов пользователя;
* public void Search() – функция для поиска аккаунта из базы данных пользователя;
* public void LoadData() – функция загрузки сохраненных данных;
* public void SaveData() – функция сохранения данных;
* public void CheckPlaylists() – функция для проверки значений плейлистов;
* public void AddTime(TimeSpan ts, TimeSpan UpdateTime) – функция добавления проигранного времени к статистике;
* public void Timer\_Tick(object sender, object e) – функция при срабатывании таймера;
* public async Task<List<StorageFile>> GetMusicFilesAsync(StorageFolder folder) – функция получения файлов;
* public async void LoadMusicFiles() – функция для прогрузки всех файлов;
* private void PopupTimer\_Tick(object sender, object e) – функция для отображения уведомления;
* private async void Page\_Loaded(object sender, RoutedEventArgs e) – функция при загрузке страницы;
* public void Calculate() – функция для подсчета очков пользователя;
* public void AddRecord() – функция для добавления рекорда пользователя в таблицу рекордов;
* public static void InManeShowOtherForm(Window This, Type A) – функция для перехода на другую форму из главной формы;
* public static void ShowOtherForm(Window This, Type A) – функция для перехода на другую форму не из основной формы;
* public static void Exit() – функция выхода из приложения.

1. ТЕСТИРОВАНИЕ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ

Метод «черного ящика» основан на принципе «вход-выход». Программе даются данные на вход с целью получения определенных данных на выходе или какого-либо несоответствия. При этом методе тестирования совершенно не важна логика программы.

В методе «белого ящика» тестовые данные получают путем анализа логики программы.

Процесс тестирования производиться с помощью проверки в нормальных условиях, на основе данных, которые характерны для реальных условий функционирования системы.

В проекте используется метод «черного ящика». Программа была протестирована в следующих тестах:

1. Тест №1: Проверка перехода на форму регистрации по клику на кнопку «Registration».
2. Тест №2: Проверка перехода на форму авторизации по клику на кнопку «Autorisation».
3. Тест №3: Проверка перехода на форму таблицы рекордов по клику на кнопку «Table recordes».
4. Тест №4: Проверка перехода на форму с игрой по клику на кнопку «Play game».
5. Тест №5: Проверка перехода на форму с главным меню по нажатию на клавишу “ESC”.
6. Тест №6: Проверка закрытия приложения из главной формы по нажатию на клавишу “ESC”.
7. Тест №7: Проверка работы функции сортировки на форме с таблицей рекордов.
8. Тест №8: Проверка работы функции обновления на форме с таблицей рекордов.
9. Тест №9: Проверка работы функции поиска на форме с таблицей рекордов.

Тест №1. Если в главном меню (рис 3.1) нажать на кнопку «Registration». То вы должны перейди на форму с регистрацией (рис 3.2).

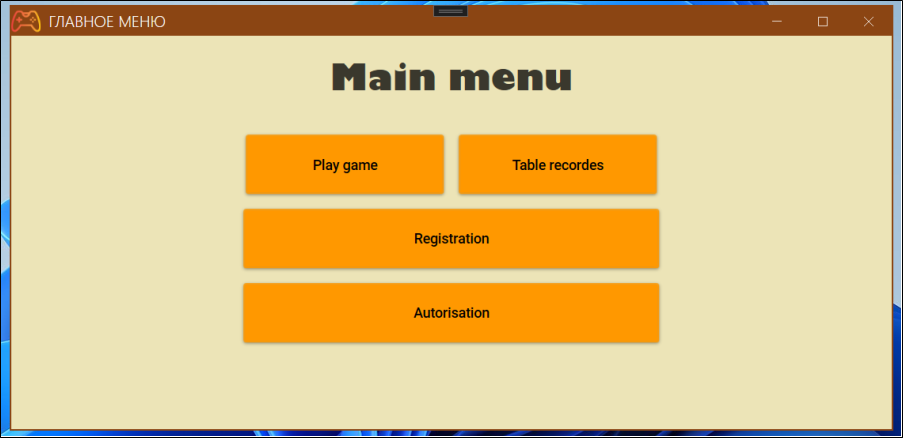


Рис. 3.1. Главное меню

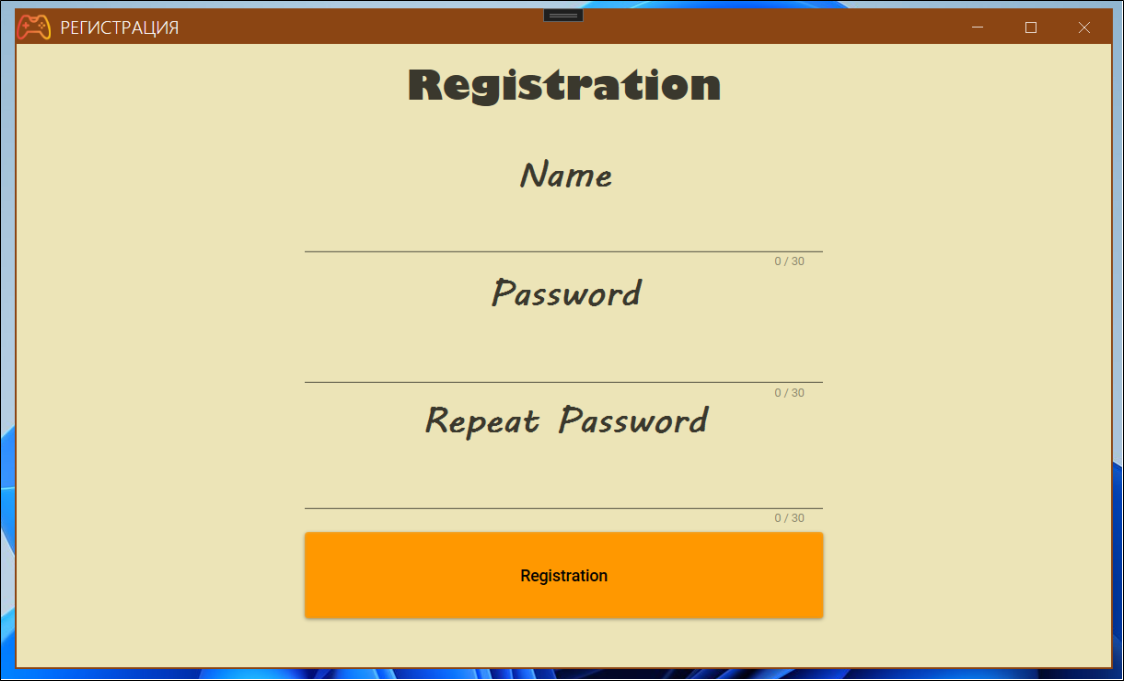


Рис. 3.2. Форма с регистрацией

Тест №2. Если в главном меню (рис 3.3) нажать на кнопку «Autorisation». То вы должны перейди на форму с авторизацией (рис 3.4).

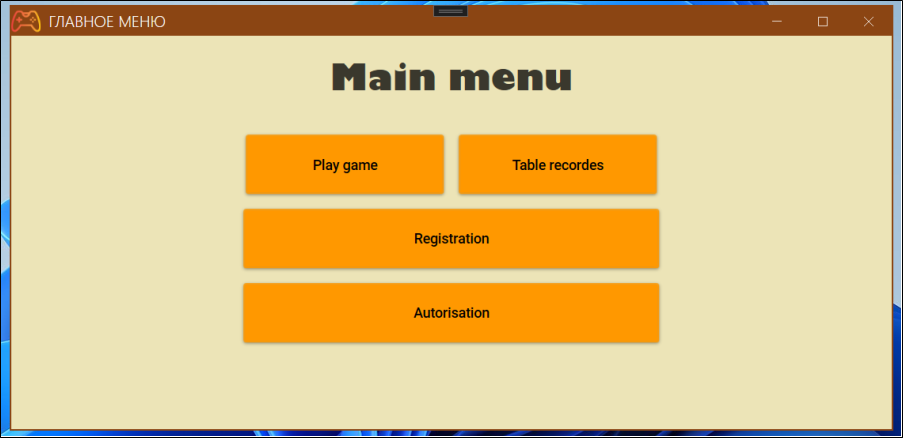


Рис. 3.3. Главное меню

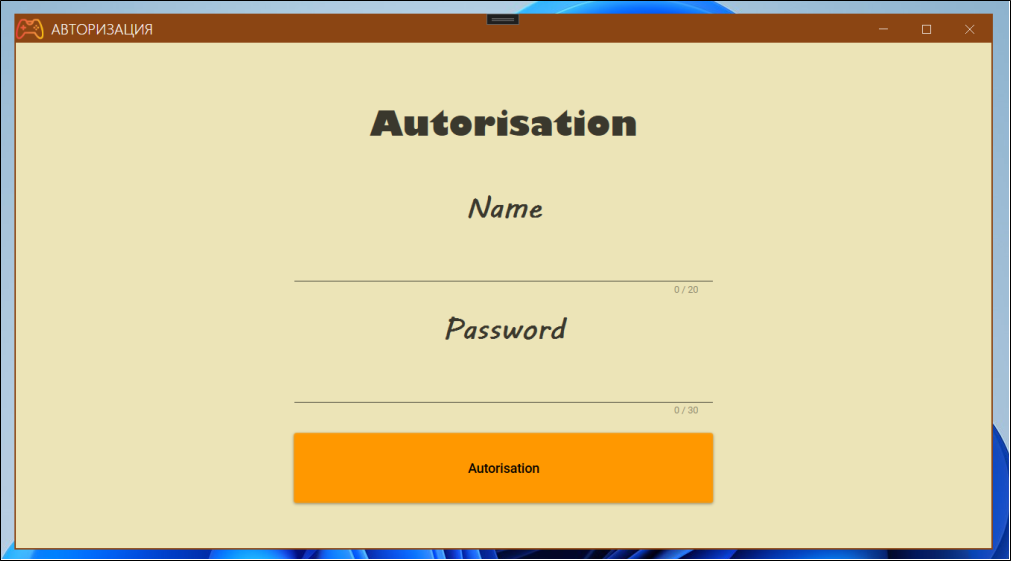


Рис. 3.4. Форма с авторизацией

Тест №3. Если в главном меню (рис 3.5) нажать на кнопку «Table recordes». То вы должны перейди на форму с таблицей рекордов (рис 3.6).

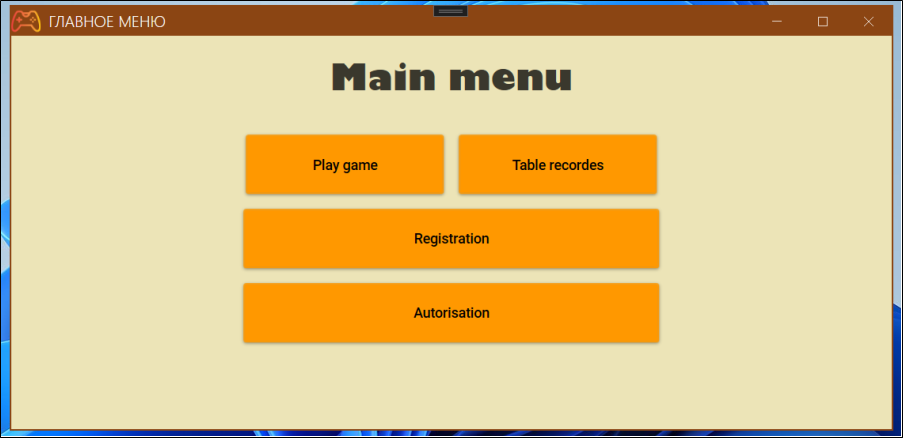


Рис. 3.5. Главное меню



Рис. 3.6. Форма с таблицей рекордов

Тест №4. Если в главном меню (рис 3.7) нажать на кнопку «Play game». То вы должны перейди на форму с игрой (рис 3.8).

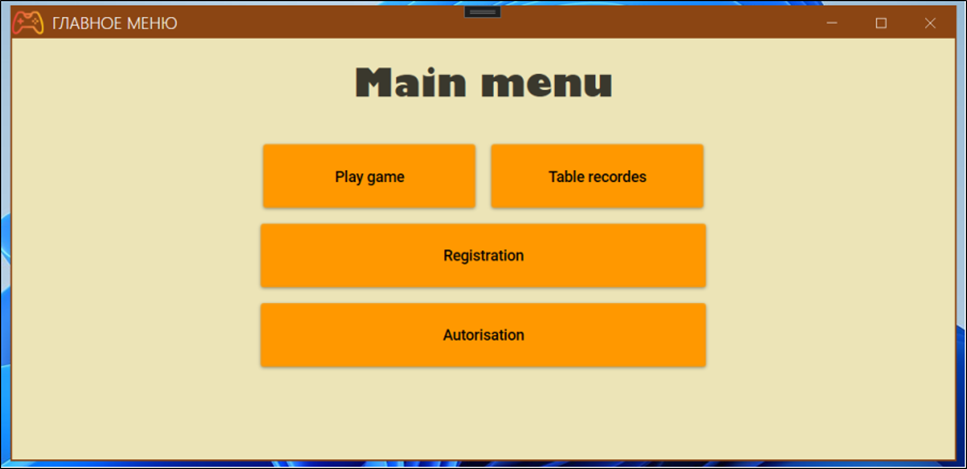


Рис. 3.7. Главное меню

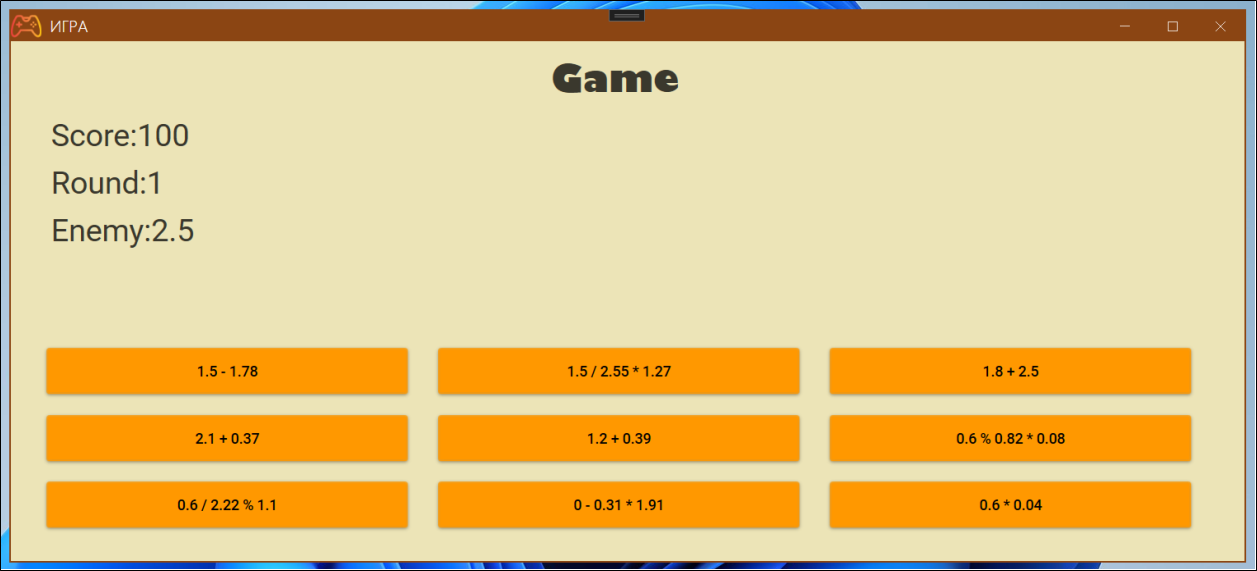


Рис. 3.8. форма с игрой

Тест №5. Если на форме регистрации (рис 3.9) нажать на клавишу «ESC». То вы должны перейди в главное меню (рис 3.10).

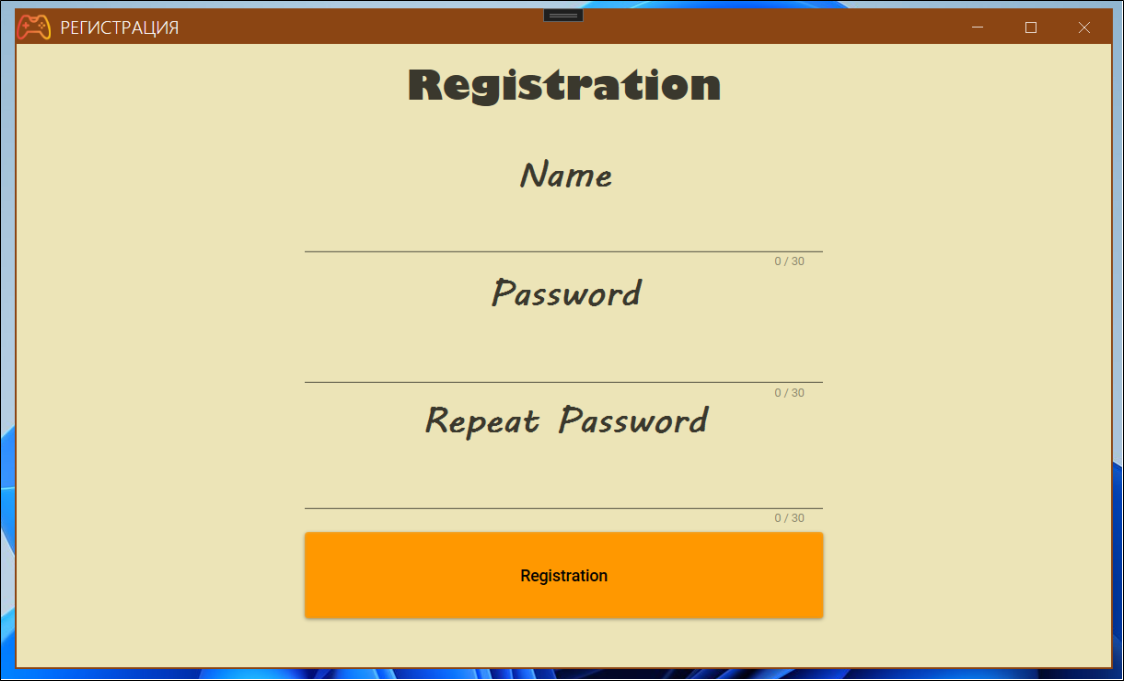


Рис. 3.9. Форма с регистрацией

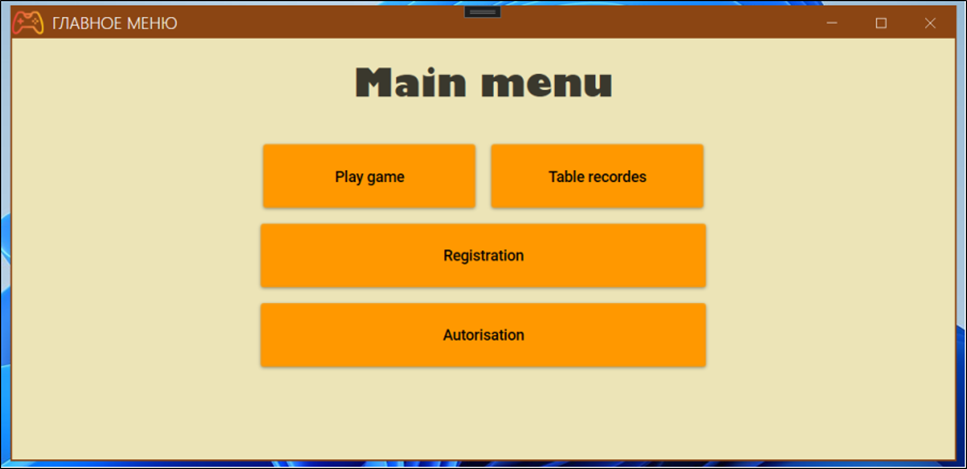


Рис. 3.10. Главное меню

Тест №6. Если в главном меню (рис 3.11) нажать на клавишу «ESC». То игра должна закрыться (рис 3.12).

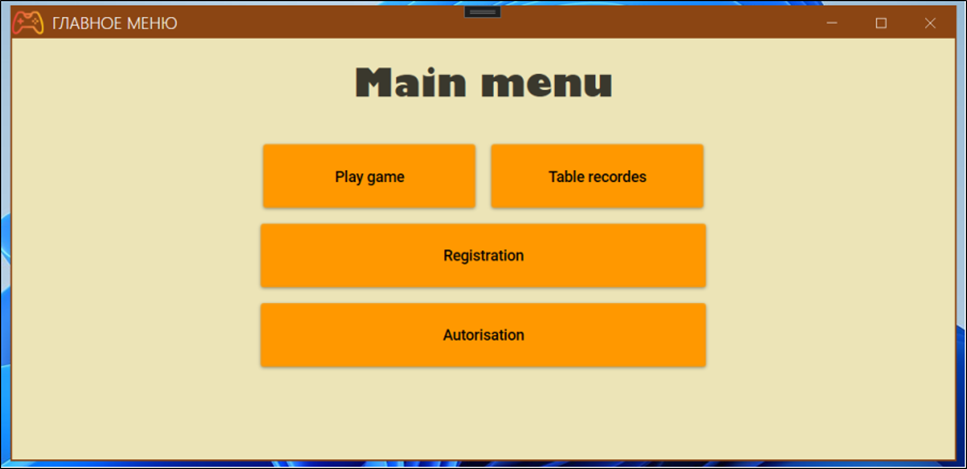


Рис. 3.11. Главное меню

Тест №7. Если на форме с таблицей рекордов (рис 3.12) кликнуть на кнопку «Sort». То таблица рекордов должна быть отсортирована по убыванию (рис 3.13).



Рис. 3.12. Форма с таблицей рекордов



Рис. 3.13. Форма с отсортированной таблицей рекордов

Тест №8. Если на форме с таблицей рекордов кликнуть на кнопку «Update». То таблица рекордов должна быть обновлена (рис 3.14).



Рис. 3.14. Форма с обновленной таблицей рекордов

Тест №9. Если на форме с таблицей рекордов (рис 3.15) ввести имя искомого пользователя и кликнуть на кнопку «Search». То будет отображено сообщение о том, что пользователь найден или же что пользователь не найден (рис 3.16).



Рис. 3.15. Форма с таблицей рекордов

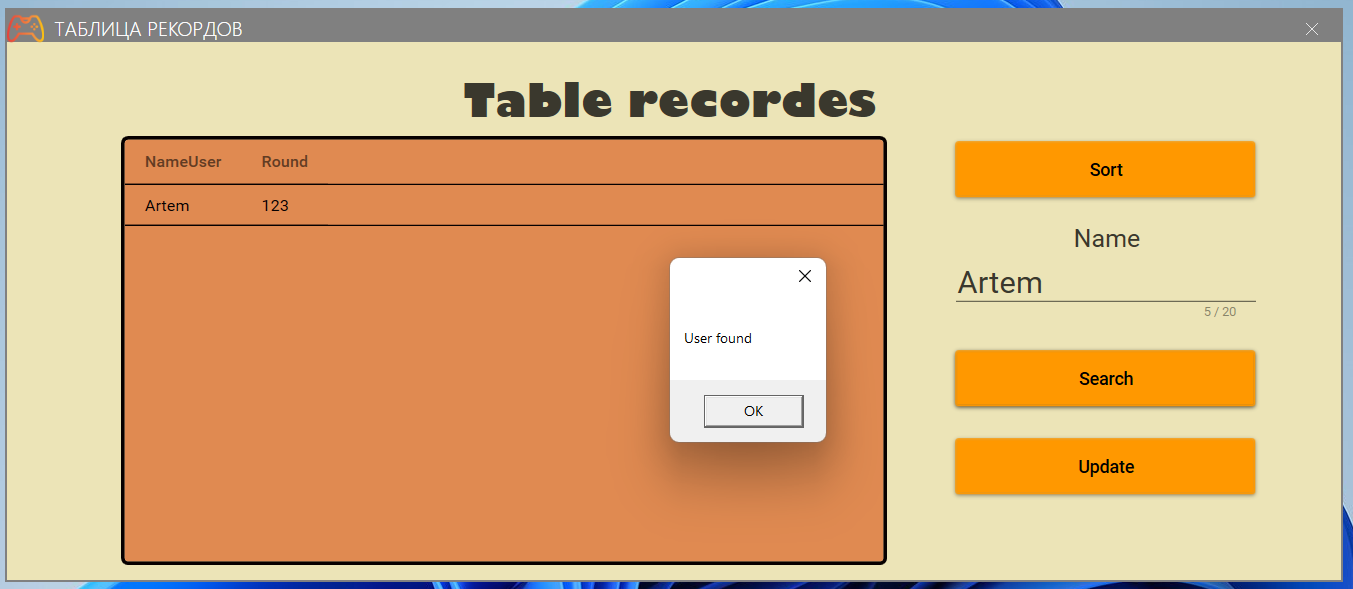


Рис. 3.16. Форма с сообщением

1. РУКОВОДСТВО ПО ИСПОЛЬЗОВАНИЮ ПРОГРАММНОГО СРЕДСТВА

При запуске программы появляется форма с главным меню (рис. 4.1). На форме главного меню пользователь может перейти в формы «Регистрация», «Авторизация», «Таблица рекордов», «Игра».

Для перехода на нужную пользователю форму ему нужно нажать на соответствующую кнопку.

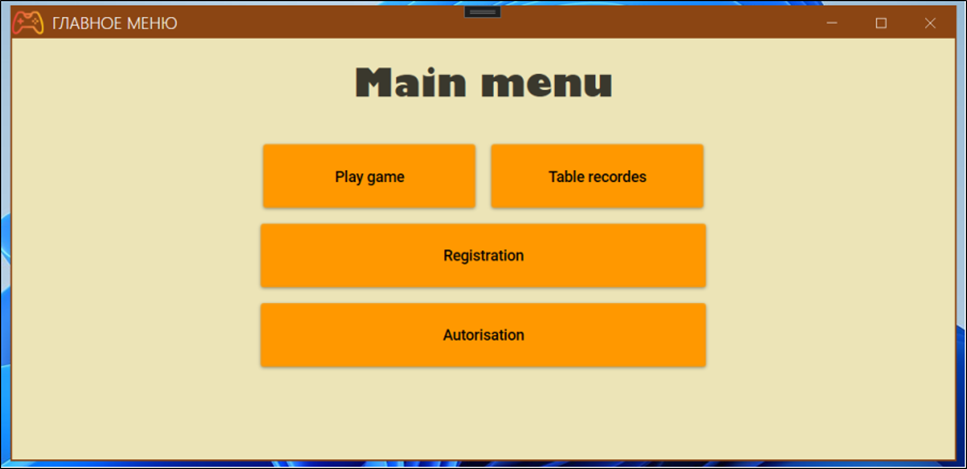


Рис. 4.1. Главное меню

Для регистрации пользователю нужно кликнуть по кнопке «Registration» из главного после этого пользователя перенесет на форму с регистрацией.

На форме регистрации (рис. 4.2) пользователь должен ввести требуемые входные данные (рис. 4.3) и нажать на кнопку «Registration» чтобы зарегистрироваться (рис. 4.4).

Чтобы вернуться в главное меню пользователю нужно нажать на клавишу «ESC».

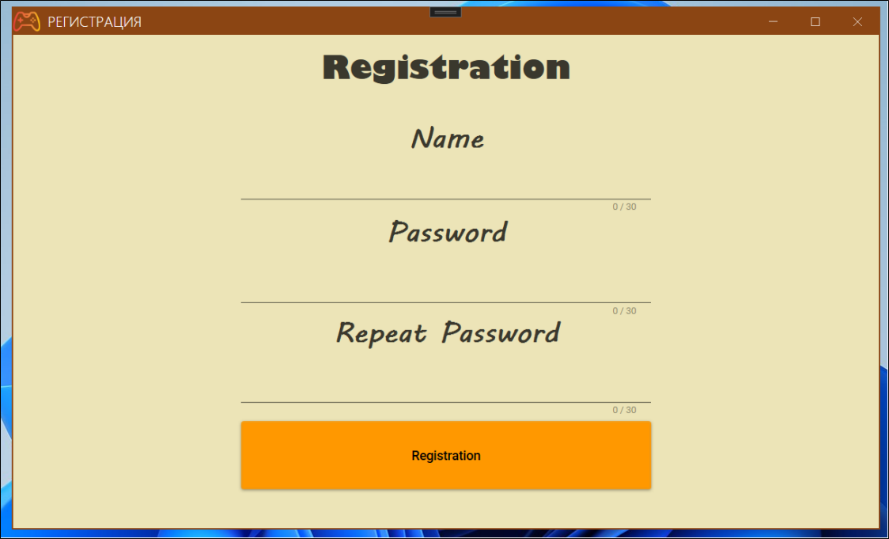


Рис. 4.2. Форма с регистрацией

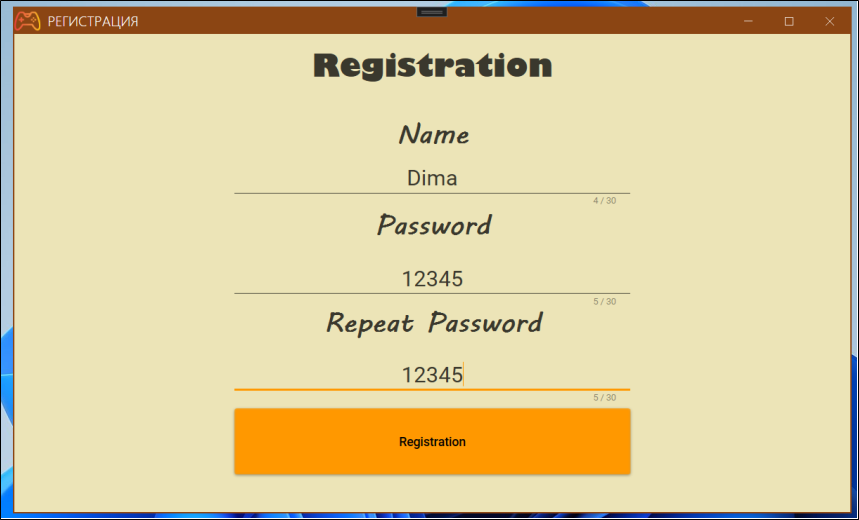


Рис. 4.3. Форма с регистрацией и заполненными полями

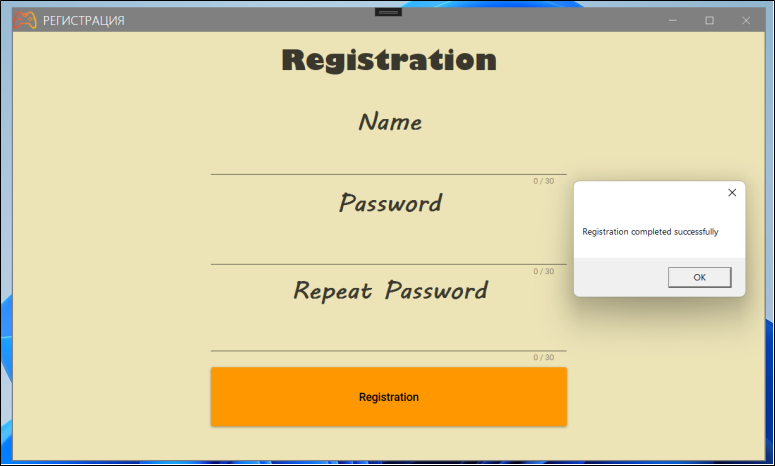


Рис. 4.4. Форма с регистрацией и сообщением

Для авторизации пользователю нужно кликнуть на кнопку «Autorisation» из главного меню после этого пользователя перенесет на форму с авторизацией.

На форме авторизации (рис. 4.5) пользователь должен ввести требуемые входные данные (рис. 4.6) и нажать на кнопку «Autorisation» чтобы зарегистрироваться (рис. 4.7).

Чтобы вернуться в главное меню пользователю нужно нажать на клавишу «ESC».

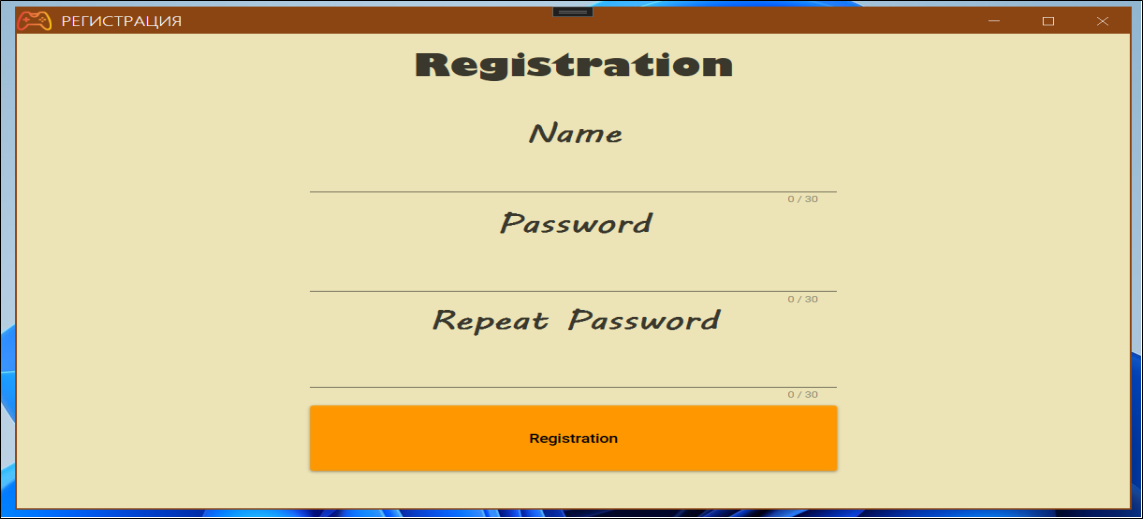


Рис. 4.5. Форма с авторизацией

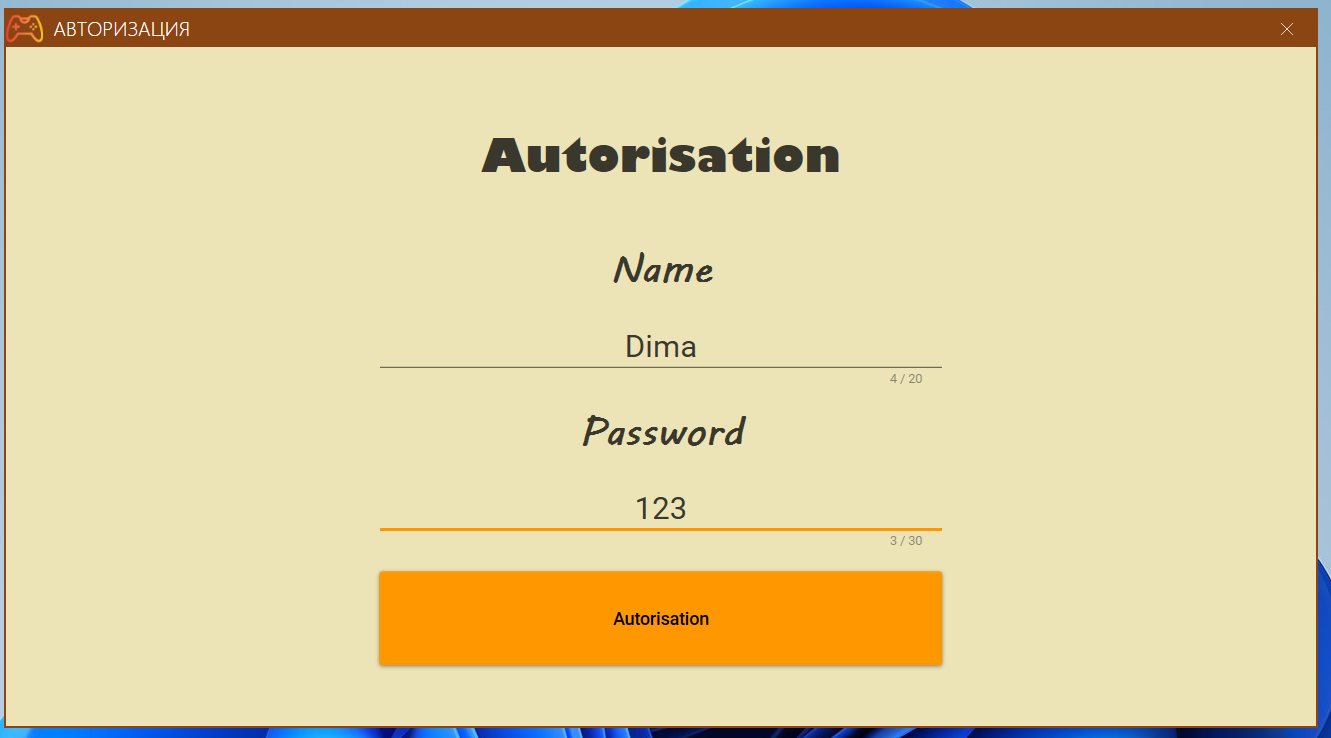


Рис. 4.6. Форма с авторизацией и заполненными полями

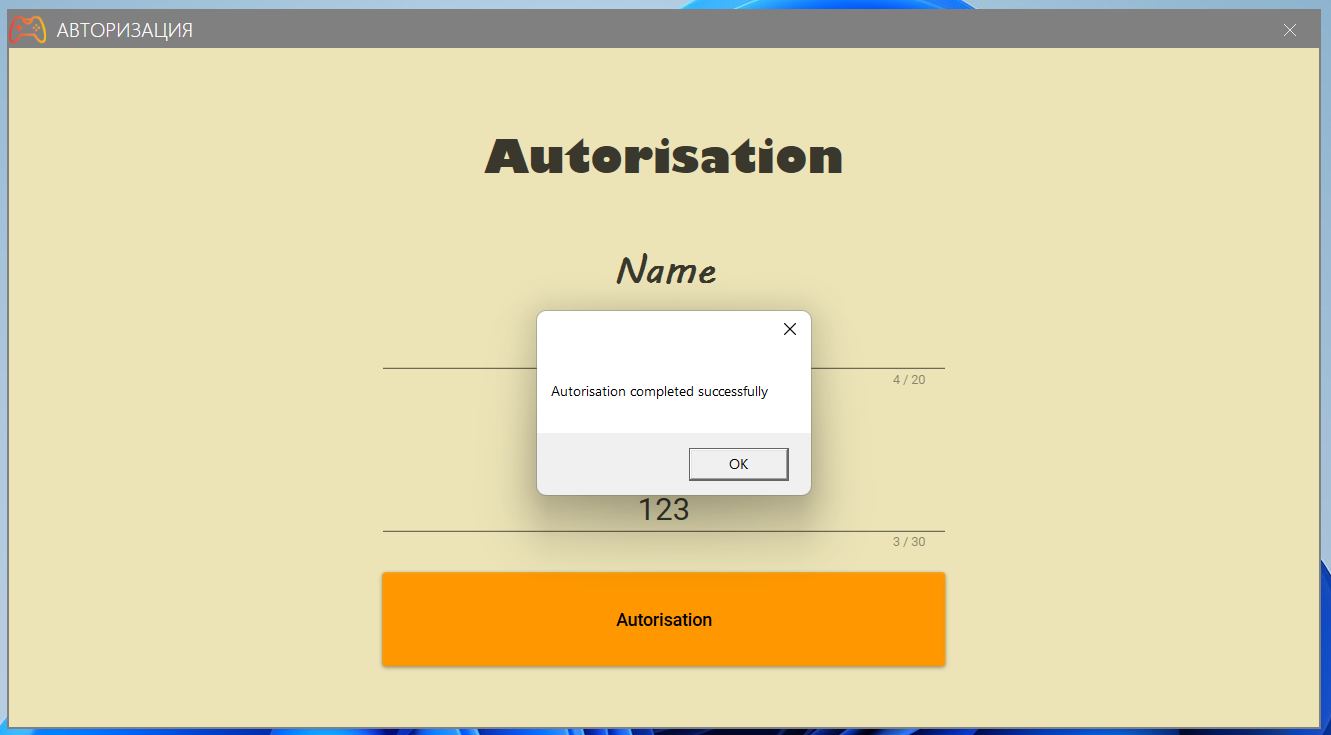


Рис. 4.7. Форма с авторизацией и сообщением

Для просмотра таблицы рекордов пользователю нужно кликнуть на кнопку «Table recordes» из главного меню после этого пользователя перенесет на форму с таблицей рекордов (рис. 4.8).

Чтобы вернуться в главное меню пользователю нужно нажать на клавишу «ESC».



Рис. 3.8. Форма с таблицей рекордов

Для сортировки таблицы рекордов пользователю нужно кликнуть на кнопку «Sort» после этого таблица от сортируется по убыванию (рис. 4.9).



Рис. 3.9. Форма с отсортированной таблицей рекордов

Если пользователь хочет обновить таблицу рекордов ему нужно кликнуть на кнопку «Update» после этого таблица обновиться. (рис. 4.10).



Рис. 3.9. Форма с обновленной таблицей рекордов

Для того чтобы начать играть пользователю нужно кликнуть на кнопку «Play game» из главного меню после этого пользователя перенесет на форму с игрой (рис. 4.11).

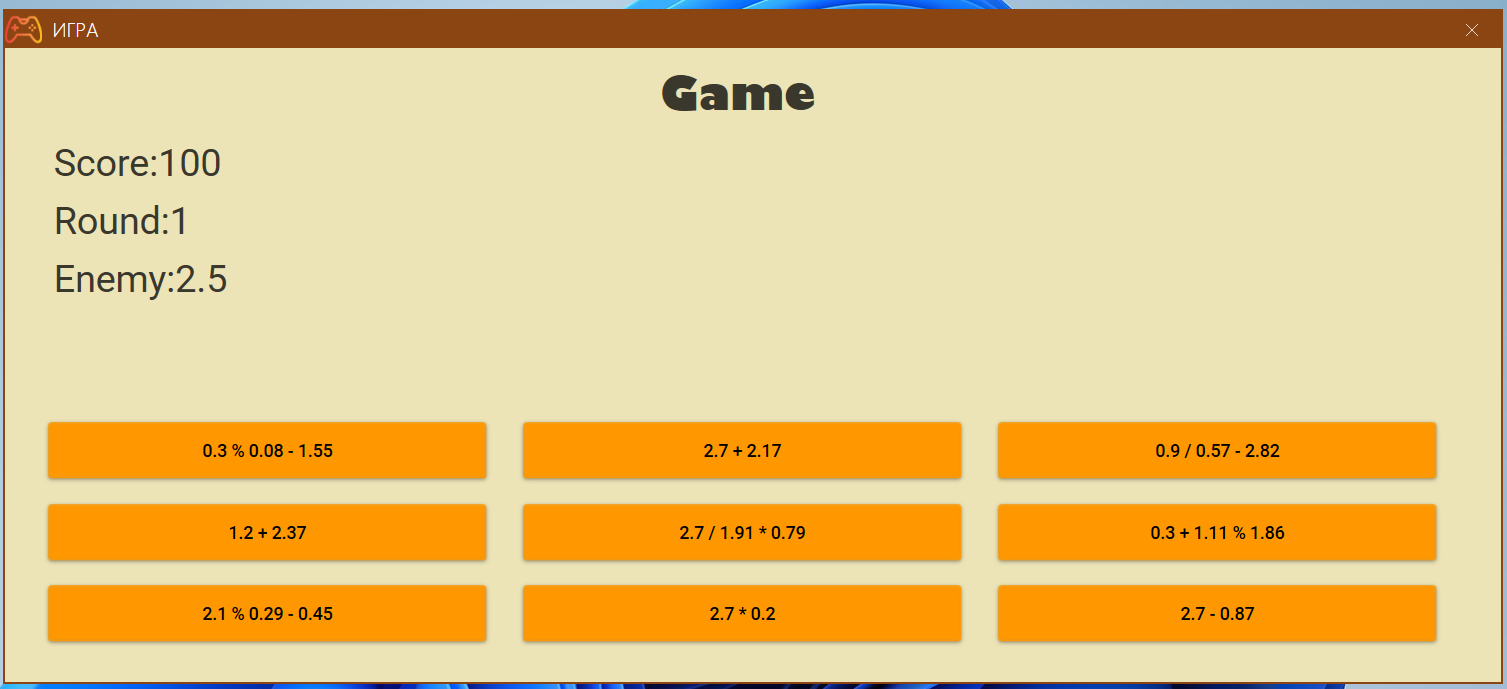


Рис. 4.11. Форма с игрой

Основная задача пользователя продержаться наибольшее количество раундов для этого пользователю необходимо просмотреть все уравнения и выбирать самые выгодные ему, то есть те уравнение которые добавит ему максимально количество очков. После нажатие на кнопку с уравнением будет происходить пересчет очков. В самом начале игры у пользователя 100 очков и отнимается 2.5 очка в конце раунда, но в последующих раундах количество отнимаемых очков будет расти (рис. 4.12).



Рис. 4.12. Форма с игрой со вторым раундом

Если пользователь проиграл и был авторизирован, то отобразиться сообщении о том побил ли пользователь свой рекорд или же нет (рис. 4.13).

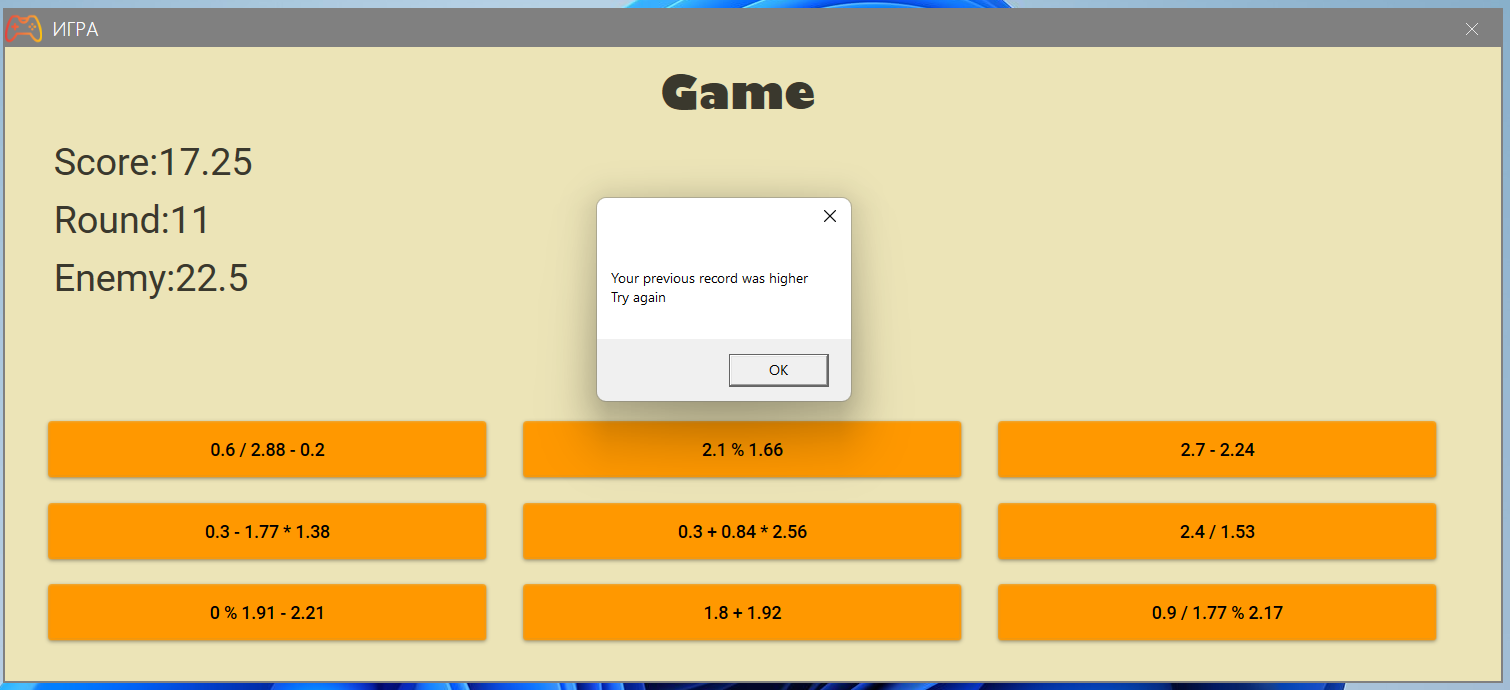


Рис. 4.13. Форма с игрой и сообщением

ЗАКЛЮЧЕНИЕ

В ходе курсового проекта было разработана игра «Equation». В игре были реализованы возможности регистрации, авторизации, просмотра таблицы рекордов ее сортировки и поиска в ней, а также играть в игру, ставить рекорды и повышать свой рекорд.

В ходе курсового проектирования был использован язык программирования C#.

Игра была протестирована. Тестирование прошло успешно на всех его этапах.

ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ

1. Cyberforum [Электронный ресурс]. Режим доступа: <https://www.cyberforum.ru/wpf-silverlight/>. Дата доступа: 10.06.2022.
2. Документация Microsoft [Электронный ресурс]. Режим доступа: https://docs.microsoft.com/ru-ru/windows/uwp/. Дата доступа: 10.06.2022.
3. Metanit [Электронный ресурс]. Режим доступа: <https://metanit.com/sharp/uwp/>. Дата доступа: 10.06.2022.
4. Хабр [Электронный ресурс]. Режим доступа: https://habr.com/ru/search/?q=uwp. Дата доступа: 10.06.2022.
5. Stackoverflow [Электронный ресурс]. Режим доступа: https://stackoverflow.com/search?q=uwp. Дата доступа: 10.06.2022.
6. Github [Электронный ресурс]. Режим доступа: <https://github.com/search?q=uwp>. Дата доступа: 10.06.2022.
7. ][AKEP [Электронный ресурс]. Режим доступа: https://xakep.ru/2015/11/04/uwp-windows-10/. Дата доступа: 10.06.2022.
8. Medium.com [Электронный ресурс]. Режим доступа: https://medium.com/southworks/developing-desktop-apps-for-windows-xaml-islands-vs-winui-vs-uwp-12dd0281e37c. Дата доступа: 10.06.2022.

ПЕРЕЧЕНЬ ПРИНЯТЫХ СОКРАЩЕНИЙ

ПО – программное обеспечение

ПП – программный продукт

ПРИЛОЖЕНИЕ 1 (ОБЯЗАТЕЛЬНОЕ) – ЛИСТИНГ ПРОГРАММЫ

Компонент инициализации приложения:

using MahApps.Metro.Controls;

using System.Windows;

using System.Windows.Input;

using WPF.Classes;

namespace WPF

{

public partial class PRegistration : MetroWindow

{

public PRegistration()

{

InitializeComponent();

}

private bool EqualPasswords(string password,string repeatPassowrd)

{

if (password != repeatPassowrd)

{

FPassword.Clear();

FReplPassword.Clear();

MessageBox.Show("Password not equal");

}

return password == repeatPassowrd;

}

private bool CheckVoidFields(string nameUser,string password,string repitPassword)

{

if (nameUser == "" || password == "" || repitPassword == "")

{

MessageBox.Show("Feilds is void");

}

return nameUser == "" || password == "" || repitPassword == "";

}

private void Registration\_Click(object sender, RoutedEventArgs e)

{

string nameUser = FName.Text;

string password = FPassword.Text;

string repeatPassword = FReplPassword.Text;

if(!EqualPasswords(password, repeatPassword))

return;

if(CheckVoidFields(nameUser, password, repeatPassword))

return;

if (!TUsers.SearchUser(nameUser))

{

TUsers.Add(nameUser, password);

ClearFeild();

MessageBox.Show("Registration completed successfully");

}

else

{

MessageBox.Show("Accaunt with name is exists");

FName.Clear();

}

}

private void ClearFeild()

{

FName.Clear();

FPassword.Clear();

FReplPassword.Clear();

}

private void Window\_KeyDown(object sender, KeyEventArgs e)

{

if (Key.Escape == e.Key)

{

WindowMethods.CheckExit = false;

WindowMethods.ShowMain(this);

}

}

private void Window\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

if (WindowMethods.CheckExit) WindowMethods.Exit();

}

}

}

using MahApps.Metro.Controls;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using WPF.Classes;

using WPF.Database;

namespace WPF

{

/// <summary>

/// Interaction logic for PRecordes.xaml

/// </summary>

public partial class PRecordes : MetroWindow

{

public PRecordes()

{

InitializeComponent();

DataGridRecordes.ItemsSource = TRecordes.GetTable();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

DataGridRecordes.ItemsSource = TRecordes.Sort();

}

private void Window\_KeyDown(object sender, KeyEventArgs e)

{

if (Key.Escape == e.Key)

{

WindowMethods.CheckExit = false;

WindowMethods.ShowMain(this);

}

}

private void Window\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

if (WindowMethods.CheckExit) WindowMethods.Exit();

}

private void BUpdate\_Click(object sender, RoutedEventArgs e)

{

DataGridRecordes.ItemsSource = TRecordes.GetTable();

}

private bool CheckFeildIsVoid(string enterName)

{

if (enterName == "")

{

MessageBox.Show("Field is void");

}

return enterName == "";

}

private void BSearch\_Click(object sender, RoutedEventArgs e)

{

string enterName = FName.Text;

if (CheckFeildIsVoid(enterName))

return;

Record? record = new Record();

if (TRecordes.Search(enterName, out record))

{

DataGridRecordes.ItemsSource = new List<Record?>() { record };

MessageBox.Show("User found");

}

else

{

MessageBox.Show("User not found");

}

}

}

}

using MahApps.Metro.Controls;

using System;

using System.Windows;

using System.Windows.Input;

namespace WPF

{

public partial class PPlayGame : MetroWindow

{

public PPlayGame()

{

InitializeComponent();

ButtonEquation.GetEvent(UpdateLabels);

}

public void UpdateLabels()

{

LEnemy.Content = "Enemy:" + ButtonEquation.GetEnemy();

LRound.Content = "Round:" + ButtonEquation.GetRound();

LScore.Content = "Score:" + ButtonEquation.GetScore();

}

private void Window\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

if (WindowMethods.CheckExit) WindowMethods.Exit();

}

private void WPlay\_KeyDown(object sender, KeyEventArgs e)

{

if (Key.Escape == e.Key)

{

WindowMethods.CheckExit = false;

WindowMethods.ShowMain(this);

}

}

private void WPlay\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e)

{

ButtonEquation.Update();

UpdateLabels();

}

}

}

using MahApps.Metro.Controls;

using System.Windows;

namespace WPF

{

public partial class MainWindow : MetroWindow

{

public MainWindow()

{

InitializeComponent();

}

private void BRecordes\_Click(object sender, RoutedEventArgs e)

{

WindowMethods.InManeShowOtherForm(this, typeof(PRecordes));

}

private void BAutorisation\_Click(object sender, RoutedEventArgs e)

{

WindowMethods.InManeShowOtherForm(this, typeof(PAuvtorisation));

}

private void BPaly\_Click(object sender, RoutedEventArgs e)

{

WindowMethods.InManeShowOtherForm(this, typeof(PPlayGame));

}

private void BRegistration\_Click\_1(object sender, RoutedEventArgs e)

{

WindowMethods.InManeShowOtherForm(this, typeof(PRegistration));

}

private void Menu\_KeyDown(object sender, System.Windows.Input.KeyEventArgs e)

{

WindowMethods.Exit();

}

private void Menu\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

WindowMethods.Exit();

}

}

}

using MahApps.Metro.Controls;

using System.Windows;

using System.Windows.Input;

using WPF.Classes;

namespace WPF

{

public partial class PAuvtorisation : MetroWindow

{

public PAuvtorisation()

{

InitializeComponent();

}

private void Window\_KeyDown(object sender, KeyEventArgs e)

{

if (Key.Escape == e.Key)

{

WindowMethods.CheckExit = false;

WindowMethods.ShowMain(this);

}

}

private void ClearFeilds()

{

FName.Clear();

FPassword.Clear();

}

private void Window\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

if (WindowMethods.CheckExit) WindowMethods.Exit();

}

private bool CheckFilds(string name, string password)

{

if (name == "" || password == "")

{

MessageBox.Show("Feilds is void");

}

return name == "" || password == "";

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

string name = FName.Text;

string password = FPassword.Text;

if (CheckFilds(name,password))

return;

if (TUsers.CheckPassword(name, password))

{

ActiveUser.SetNewUser(name);

MessageBox.Show("Autorisation completed successfully");

ClearFeilds();

}

else

{

MessageBox.Show("Autorisation faild\nWorng name or password");

ClearFeilds();

ActiveUser.Disactive();

}

}

}

}

using System;

using System.Collections.Generic;

using System.Windows;

using System.Windows.Controls;

using WPF.Classes;

using WPF.Tasking;

namespace WPF

{

/// <summary>

/// Interaction logic for ButtonEquation.xaml

/// </summary>

public partial class ButtonEquation : UserControl

{

private Equation \_equation;

private static Game game;

private static List<ButtonEquation> \_buttonEquations = new();

private delegate void EventHandlerClick();

private static event EventHandlerClick? UpdateInfoOnLabel;

private static bool IsClick = true;

public static void Update(double score = 100,double enemy = 10)

{

game = new Game(score, enemy);

UpdateAllButtonEquation();

UpdateInfoOnLabel?.Invoke();

IsClick = true;

}

public static void GetEvent(Action action)

{

UpdateInfoOnLabel += () => action();

}

public ButtonEquation()

{

InitializeComponent();

\_equation = new Equation();

GameButton.Content = \_equation.GetEqution();

\_buttonEquations.Add(this);

game = new Game(100, 10);

}

private static void UpdateAllButtonEquation()

{

for (int i = 0; i < \_buttonEquations.Count; i++)

{

\_buttonEquations[i].\_equation = new Equation();

\_buttonEquations[i].GameButton.Content = \_buttonEquations[i].\_equation.GetEqution();

}

}

public static int GetRound() => game.GetRound();

public static double GetEnemy() => game.GetEnemy();

public static double GetScore() => game.GetScore();

private void Block() => IsClick = false;

private void CheckRecord()

{

int lastRound = game.GetRound();

if (!ActiveUser.CheckActive())

{

MessageBox.Show($"You not been autorisation\nYour high score {lastRound} but your high score not be add or update");

return;

}

if (TRecordes.Search(ActiveUser.GetName()))

{

if (ActiveUser.UpdateRecord(lastRound))

{

MessageBox.Show($"Your high score {lastRound} and he has been updated");

}

else

{

MessageBox.Show($"Your previous record was higher\nTry again");

}

}

else

{

MessageBox.Show($"Your high score {lastRound} and it been added to the table recordes");

ActiveUser.AddRecord(lastRound);

}

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

if (IsClick)

{

double unswer = \_equation.GetUnswer();

game.NextRound(unswer);

if (game.CheckLose())

{

Block();

CheckRecord();

UpdateInfoOnLabel?.Invoke();

IsClick = false;

return;

}

UpdateInfoOnLabel?.Invoke();

UpdateAllButtonEquation();

}

}

}

}