**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
федеральное государственное автономное образовательное   
учреждение высшего образования   
«Санкт-Петербургский политехнический университет Петра Великого»  
ИНСТИТУТ СРЕДНЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

|  |  |
| --- | --- |
|  | **Допустить к защите** Заместитель директора  по УМР \_\_\_\_\_\_\_\_\_\_\_\_ Е.Г. Конакина\_  (Подпись) (ФИО)  «\_\_\_»\_\_\_\_\_\_\_\_\_\_ 2024 г. |
|  |

**курсовой проект**Тема: «ИГРА С ГРАФИЧЕСКИМ ИНТЕРФЕЙСОМ РУССКОЕ ЛОТО»

специальность 09.02.07 группа 42919/7

Студент (ка) Косовский С.А.  
 (подпись) (ФИО)  
Преподаватель Девятко Н.С.

(подпись) (ФИО)

Санкт-Петербург  
2024

Содержание

[ВВЕДЕНИЕ 3](#_Toc168995872)

[1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ РАЗРАБОТКИ 4](#_Toc168995873)

[1.1. Описание предметной области 4](#_Toc168995874)

[1.2. Анализ методов решения 6](#_Toc168995875)

[1.3. Обзор средств программирования 7](#_Toc168995876)

[1.4. Описание языка C# 8](#_Toc168995877)

[2. ПРАКТИЧЕСКАЯ ЧАСТЬ 11](#_Toc168995878)

[2.1. Постановка задачи 11](#_Toc168995879)

[2.1.1. Основания для разработки 11](#_Toc168995880)

[2.1.2. Назначение программы 11](#_Toc168995881)

[2.2. Проектирование приложения 11](#_Toc168995882)

[2.2.1. Диаграмма прецедентов 11](#_Toc168995883)

[2.2.2. Диаграмма последовательностей 12](#_Toc168995884)

[2.2.3. Диаграмма классов 14](#_Toc168995885)

[2.2.4. Диаграмма активности 14](#_Toc168995886)

[2.3. Текст программы 16](#_Toc168995887)

[2.4. Описание программы 16](#_Toc168995888)

[2.4.1. Общие сведения 16](#_Toc168995889)

[2.4.2. Функциональное назначение 16](#_Toc168995890)

[2.4.3. Описание логической структуры системы 16](#_Toc168995891)

[2.4.4. Используемые технические и программные средства 18](#_Toc168995892)

[2.4.5. Вызов и загрузка 19](#_Toc168995893)

[2.5. Руководство оператора 19](#_Toc168995894)

[2.5.1. Назначение программы 19](#_Toc168995895)

[2.5.2. Выполнение программы и сообщения оператору 19](#_Toc168995896)

[2.6. Программа и методика испытаний 20](#_Toc168995897)

[2.6.1. Объект испытаний 20](#_Toc168995898)

[2.6.2. Цель испытаний 20](#_Toc168995899)

[2.6.3. Требования к программе 20](#_Toc168995900)

[2.6.4. Требования к программной документации 22](#_Toc168995901)

[2.6.5. Средства и порядок испытаний 22](#_Toc168995902)

[2.6.6. Методы испытаний 23](#_Toc168995903)

[2.7. Протокол испытаний 25](#_Toc168995904)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 28](#_Toc168995905)

[ПРИЛОЖЕНИЕ А 2](#_Toc168995906)

[ПРИЛОЖЕНИЕ Б 10](#_Toc168995907)

# ВВЕДЕНИЕ

В современном мире развлечения играют важную роль в жизни каждого человека. Игры сопровождают нас на отдыхе, в кругу семьи и друзей, создавая атмосферу веселья и общения. Развитие цифровых технологий значительно изменило подход к созданию и проведению игр. Компьютеры и различные программные приложения активно используются для создания увлекательных и интерактивных игровых процессов. Они открывают новые возможности для развлечений и социального взаимодействия.

Программные имитаторы классических игр, таких как лото, становятся всё более популярными. Они позволяют пользователям, не имеющим возможности участвовать в традиционных играх, наслаждаться процессом в цифровом формате. Такие приложения также используются в образовательных учреждениях для обучения и проведения развлекательных мероприятий, делая процесс игры более доступным и интерактивным.

Разработанное в данном проекте приложение «Русское лото» предназначено для широкого круга пользователей. Оно может быть использовано как для семейных игр, так и для организованных игровых мероприятий. Приложение также может служить инструментом для развлечения и досуга, позволяя пользователям наслаждаться процессом игры в любое время и в любом месте, обеспечивая удобный и увлекательный игровой опыт.

# 1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ РАЗРАБОТКИ

## 1.1. Описание предметной области

Приложение «Русское лото» представляет собой цифровую версию классической игры в лото, предназначенную для предоставления пользователям удобного и увлекательного игрового процесса. Виртуальная версия игры использует современные технологии для имитации традиционного игрового опыта, предлагая ряд преимуществ и возможностей, которые трудно достичь в классическом варианте.

Компьютерные игры являются важной частью современной культуры и развлечений. Они могут быть основаны на разнообразных источниках, таких как фильмы, книги, исторические события или полностью оригинальные концепции. В США компьютерные игры были официально признаны отдельным видом искусства с 2011 года, что подчеркивает их значимость и влияние на культуру.

Компьютерные игры могут быть классифицированы по нескольким признакам:

Развлечения и времяпрепровождение: Основная цель приложения «Русское лото» — предоставить пользователям возможность насладиться игрой в лото, предлагая развлекательный опыт, который можно использовать для отдыха и досуга. Игровой процесс включает в себя генерацию случайных чисел, управление игровыми карточками и учет результатов, что создает увлекательный и динамичный игровой опыт.

**Социальные взаимодействия: Игровое приложение может поддерживать социальные взаимодействия между участниками, что делает его подходящим для совместного времяпрепровождения. Пользователи могут играть вместе, делиться результатами и обсуждать игровые стратегии, что способствует укреплению социальных связей и созданию сообщества вокруг игры.**

**Технологии и инструменты: Разработка приложения «Русское лото» включает использование современных технологий для создания графического интерфейса и управления игровым процессом. Использование C# и Windows Forms позволяет создать удобный и интуитивно понятный интерфейс, а MS SQL Server обеспечивает надежное управление данными, такими как результаты розыгрышей и информация о пользователях.**

**Графический интерфейс: Интерфейс приложения разрабатывается с акцентом на удобство и визуальную привлекательность. Пользователи могут легко взаимодействовать с игровыми элементами, такими как карточки и числа, а также настраивать параметры игры и внешний вид интерфейса в соответствии с их предпочтениями.**

Примеры и категории приложений:

* Образовательные приложения – эти приложения предназначены для обучения и развития навыков. В контексте игры в лото, образовательные элементы могут включать в себя инструкции по правилам игры, стратегии выигрыша и развивающие задания для улучшения понимания игрового процесса.
* Развлекательные приложения: Приложения, которые обеспечивают развлечение и удовольствие. Приложение «Русское лото» попадает в эту категорию, предлагая пользователям легкий и увлекательный способ провести время.
* Социальные и казуальные игры: Приложения, которые ориентированы на социальное взаимодействие и простоту освоения. «Русское лото» предлагает казуальный игровой процесс, подходящий для широкой аудитории, и способствует социализации пользователей через совместное участие в игре.
* Генераторы случайных чисел: Важный аспект приложения, который обеспечивает случайную генерацию чисел для розыгрыша. Это ключевой элемент, который поддерживает честность и непредсказуемость игры, что является важной частью игрового опыта.

В целом, приложение «Русское лото» сочетает элементы развлечения, образования и социального взаимодействия, предлагая пользователям увлекательный и доступный способ играть в лото в цифровом формате. Технологическое решение и функциональные возможности приложения обеспечивают высокое качество игрового опыта, соответствующее современным требованиям и ожиданиям пользователей.

## 1.2. Анализ методов решения

Приложение «Русское лото» представляет собой цифровую версию классической настольной игры в лото, выполненную с графическим интерфейсом. Виртуальная версия игры предоставляет пользователям возможность наслаждаться игрой в удобном и современном формате, интегрируя элементы классического лото с современными технологиями для улучшения игрового опыта. В этой категории игр пользователи взаимодействуют с игровым процессом через визуальные и интерактивные элементы на экране. Приложение создано для обеспечения увлекательного и удобного игрового процесса, имитируя классическое лото с элементами автоматизации и настройки, характерными для современных цифровых игр.

1. Ручной способ: В этом режиме пользователи играют в лото, активно управляя игровым процессом с помощью компьютерной мыши или клавиатуры. Они могут вручную выбирать числа для своих игровых карточек и отмечать их по мере выпадения в розыгрыше. Этот метод обеспечивает традиционный игровой опыт, позволяя пользователям полностью контролировать процесс игры. Вручную управление числами и карточками предоставляет возможность более глубокого взаимодействия с игрой и её элементами. Этот способ требует активного участия и внимания со стороны пользователя, что может быть интересно для тех, кто предпочитает традиционный формат игры.
2. Автоматизированный способ: В этом режиме приложение предлагает пользователям автоматизированные функции для упрощения игрового процесса. Например, приложение может автоматически генерировать случайные числа для розыгрыша, заполнять карточки и отслеживать результаты игры. Автоматизированный способ делает процесс игры более удобным и эффективным. Пользователи могут настроить параметры игры, такие как количество карточек, формат розыгрыша и другие параметры, через интуитивно понятный интерфейс. Этот метод снижает необходимость ручного управления и упрощает процесс игры, позволяя пользователям сосредоточиться на развлекательных аспектах.
3. Автоматический способ: В этом варианте приложение обеспечивает полностью автоматизированный игровой процесс. Пользователи могут включить режим, при котором приложение самостоятельно управляет всеми аспектами игры, включая генерацию чисел, заполнение карточек и подсчет результатов. Автоматический способ подходит для пользователей, которые предпочитают минимальное участие в процессе игры. Приложение может загружать предустановленные игровые сценарии или использовать алгоритмы для автоматического выполнения всех операций.

Для достижения целей проекта, направленных на предоставление пользователю удобного и увлекательного способа играть в лото, наиболее подходящим является **автоматизированный способ.** Этот метод сочетает в себе преимущества ручного и автоматического управления, обеспечивая интуитивно понятный интерфейс и упрощенный игровой процесс.

## 1.3. Обзор средств программирования

C# является одним из самых мощных и современных языков программирования, широко используемым для разработки приложений под платформу Windows. Язык C# интегрирован с .NET Framework, который предоставляет разработчикам доступ к богатому набору библиотек и инструментов. Это значительно упрощает процесс создания сложных приложений, таких как «Русское лото».

Одним из ключевых преимуществ C# является его интеграция с Visual Studio, одной из наиболее удобных и функциональных сред разработки (IDE). Visual Studio предлагает расширенные возможности для кодирования, отладки и тестирования, что ускоряет процесс разработки и упрощает отладку приложений. C# также поддерживает объектно-ориентированное программирование (ООП), что способствует созданию структурированного, масштабируемого и легко поддерживаемого кода.

Для создания графического интерфейса приложения «Русское лото» используется Windows Forms (WinForms). WinForms представляет собой мощный инструмент для разработки настольных приложений, предоставляющий возможности для создания интерактивных и визуально привлекательных пользовательских интерфейсов. Этот фреймворк поддерживает привязку данных, что позволяет динамически отображать информацию на экране, и предлагает множество элементов управления для создания гибких и пользовательски настроенных интерфейсов. WinForms также упрощает работу с формами, кнопками, панелями и другими визуальными компонентами, что делает процесс разработки удобным и быстрым.

В качестве системы управления базами данных для проекта «Русское лото» выбран MS SQL Server. MS SQL Server — это мощная реляционная система управления базами данных, которая обеспечивает высокую надежность, масштабируемость и производительность. Он предоставляет богатые возможности для работы с данными, включая поддержку транзакций, сложные запросы и надежное управление данными. MS SQL Server идеально подходит для хранения и управления данными игровых сессий, карточками лото и результатами игр, обеспечивая эффективное и надежное хранение информации.

Интеграция MS SQL Server с C# осуществляется через ADO.NET, что упрощает взаимодействие с базой данных. ADO.NET предоставляет набор классов для работы с данными и выполнения SQL-запросов, что позволяет разработчикам эффективно извлекать, обновлять и манипулировать данными в базе данных. Это делает работу с базой данных удобной и эффективной, обеспечивая быстрый доступ к данным и их обработку.

Использование C# вместе с Windows Forms и MS SQL Server обеспечивает создание мощного, удобного и функционального приложения для игры в «Русское лото». Эти технологии позволяют создать современный и интерактивный пользовательский интерфейс, а также надежную систему хранения и управления данными. Применение этих инструментов в проекте обеспечивает простоту разработки, высокую производительность и отличные возможности для расширения и модификации приложения в будущем.

Таким образом, для разработки данного курсового проекта был выбран язык программирования C#, среда разработки Visual Studio и система управления базами данных MS SQL Server. Эти инструменты обеспечивают оптимальные условия для создания качественного и функционального приложения «Русское лото».

## 1.4. Описание языка C#

С# («Си Шарп») – один из наиболее быстро растущих, востребованных и при этом «удобных» языков программирования. Это модификация фундаментального языка С от компании Microsoft, призванная создать наиболее универсальное средство для разработки программного обеспечения для большого количества устройств и операционных систем.

С# – это объектно-ориентированный язык программирования. Он был создан в период с 1998 по 2002 год командой инженеров Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота.

Язык входит в семью С-подобных языков. Синтаксис приближен к Java и C++. Его особенности:

* статистическая типизация,
* поддерживается полиморфизм,
* поддерживается перегрузка операторов,
* доступна делегация, атрибуты, события, обобщенные типы и анонимные функции.

**Ключевые слова C#:**

|  |  |  |  |
| --- | --- | --- | --- |
| abstract | extern | null | struct |
| as | false | object | switch |
| base | finally | operator | this |
| bool | fixed | out | throw |
| break | float | override | true |
| byte | for | params | try |
| case | foreach | private | typeof |
| catch | goto | protected | uint |
| const | if | public | ulong |
| continue | implicit | readonly | unchecked |
| decimal | in | ref | unsafe |
| default | int | return | ushort |
| delegate | interface | sbyte | using |
| do | internal | sealed | virtual |
| double | is | short | void |
| else | lock | sizeof | while |
| enum | long | stackalloc |  |
| event | namespace | static |  |
| explicit | new | string |  |

**Контекстные ключевые слова C#:**

|  |  |  |  |
| --- | --- | --- | --- |
| add | equals | join | set |
| ascending | from | let | value |
| async | get | on | var |
| await | global | orderby | where |
| by | group | partial | yield |
| descending | in | remove |  |
| dynamic | into | select |  |

**Основные алгоритмические конструкции C#:**

|  |  |
| --- | --- |
| Оператор присваивания | string hello = "hello " + "world"; //результат равен "hello world"  int x1 = 2 + 4; // результат равен 6  int x2 = 10 - 6; //результат равен 4  int x3 = 10 \* 6; //результат равен 60  double x4 = 10.0 / 4.0; //результат равен 2.5  double x5 = 10.0 % 4.0; //результат равен 2  int y1 = 5;  int z1 = ++y1; // z1=6; y1=6  int y2 = 5;  int z2 = y2++; // z2=5; y2=6  int y3 = 5;  int z3 = --y3; // z3=4; y3=4  int y4 = 5;  int z4 = y4--; // z4=5; y4=4 |
| Условный оператор | *if (условие) {(действие) } else {(альтернатива)} ;*  int num1 = 8;  int num2 = 6;  if(num1 > num2)  {      Console.WriteLine("Число {0} больше числа {1}", num1, num2);  }  else  {      Console.WriteLine("Число {0} меньше числа {1}", num1, num2);  } |
| Оператор множественного ветвления | *Конструкция switch/case аналогична конструкции if/else, так как позволяет обработать сразу несколько условий:*  Console.WriteLine("Нажмите Y или N");  string selection = Console.ReadLine();  switch (selection)  {      case "Y":          Console.WriteLine("Вы нажали букву Y");          break;      case "N":          Console.WriteLine("Вы нажали букву N");          break;      default:          Console.WriteLine("Вы нажали неизвестную букву");          break;  }  *После ключевого слова switch в скобках идет сравниваемое выражение. Значение этого выражения последовательно сравнивается со значениями, помещенными после оператора сase. И если совпадение будет найдено, то будет выполняться определенный блок сase.*  *В конце блока сase ставится оператор break, чтобы избежать выполнения других блоков.*  *Если мы хотим также обработать ситуацию, когда совпадения не будет найдено, то можно добавить блок default,.* |
| Арифметический цикл  **(**применяется, когда известно количество повторений цикла) | for *([инициализация счетчика]; [условие]; [изменение счетчика])*  for (int i = 0; i < 10; i++)  {      Console.WriteLine("Квадрат числа {0} равен {1}", i, i \* i);  } |
| Цикл с предусловием  **(**применяется, когда неизвестно количество повторений цикла) | *while (условие) { тело цикла };*  Этот цикл будет выполняться до тех пор, пока истинно *условие* (логическое выражение, возвращающее значение типа **Boolean**). При этом если это выражение сразу равно **false**, *тело цикла* не будет выполнено ни разу. Нужно очень внимательно следить за написанием *условия* и контролем завершения цикла, так как в результате ошибки цикл **while** будет повторяться бесконечное количество раз, что приведёт к "зацикливанию" и "зависанию" программы. |
| Цикл с постусловием  **(**применяется, когда неизвестно количество повторений цикла) | *do { тело цикла } while (условие);*  Повторения сначала выполняет *тело цикла*, а затем уже проверяет выполнение *условия*. Таким образом, этот вариант цикла гарантирует, что *тело цикла* будет выполнено по крайней мере один раз. И будет выполняться до тех пор, пока *условие* не станет истинным (**true**). |

# 2. ПРАКТИЧЕСКАЯ ЧАСТЬ

## 2.1. Постановка задачи

### 2.1.1. Основания для разработки

Разработка ведётся на основании задания к курсовому проекту по профессиональному модулю ПМ.01 «Разработка программных модулей программного обеспечения для компьютерных систем» МДК 01.01 «Разработка программных модулей» и утверждена Институтом среднего профессионального образования.

### 2.1.2. Назначение программы

Приложение «Русское лото» — это программа с графическим интерфейсом, предназначенная для проведения игры в лото в цифровом формате. Пользователи могут участвовать в игровом процессе, управлять игровыми сессиями и отслеживать результаты, используя удобный и интуитивно понятный интерфейс.

## 2.2. Проектирование приложения

На этапе проектирования были разработаны диаграмма прецедентов, диаграмма последовательностей, диаграмма классов и диаграмма активности.

### 2.2.1. Диаграмма прецедентов

Диаграммы прецедентов частично описывает use case – прецедент использования проектируемой системы, давая частичное описание частичного применения системы с точки зрения условного внешнего обозревателя. При этом описание фокусируется на том, что должна делать система по отношению к своему внешнему окружению (периферии), а не то на том, как она эта делает, то есть диаграмма прецедентов есть частичная спецификация (Рисунок 1).

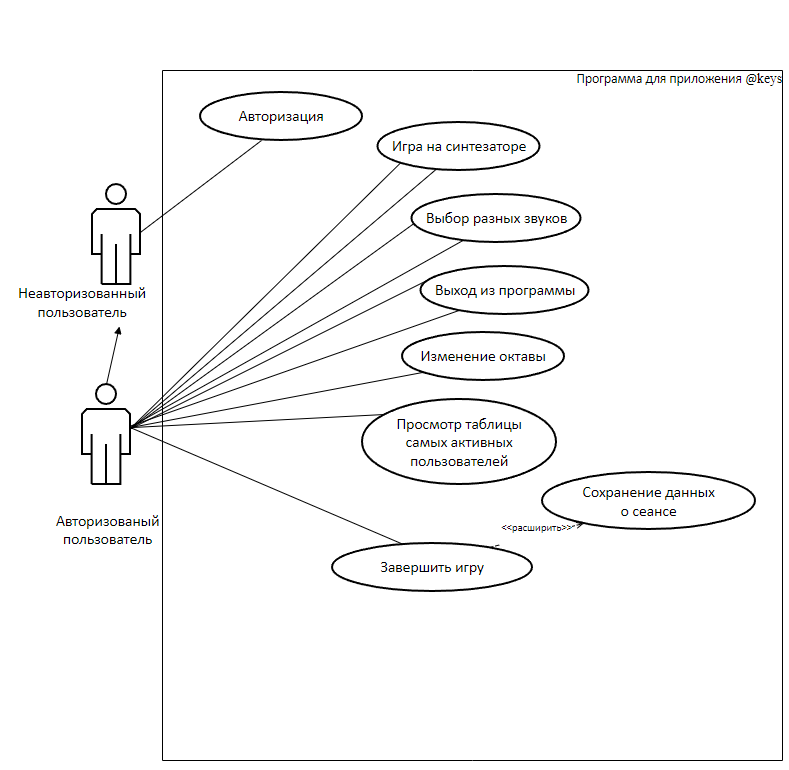


Рисунок 1 Диаграмма прецедентов

### 2.2.2. Диаграмма последовательностей

Диаграмма последовательности отражает поток событий, происходящих в рамках варианта использования.

Все действующие лица показаны в верхней части диаграммы. Стрелки соответствуют сообщениям, передаваемым между действующим лицом и объектом или между объектами для выполнения требуемых функций. На диаграмме последовательности объект изображается в виде прямоугольника, от которого вниз проведена пунктирная вертикальная линия. Эта линия называется линией жизни (lifeline) объекта. Она представляет собой фрагмент жизненного цикла объекта в процессе взаимодействия.

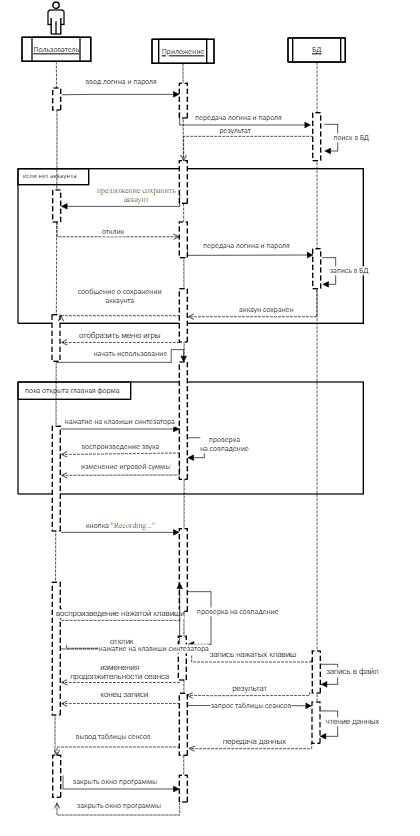


Рисунок 2 Диаграмма последовательностей

### 2.2.3. Диаграмма классов

На рисунке 3 показана диаграмма классов проекта. Программа содержит 5 основных классов: Form1-класс формы игры, класс LoginForm.cs содержит методы работы с базой данных, в которой хранится информация об игроках. Этот класс использует модуль MasterForm во время авторизации и при отображении таблицы продолжительности сеансов пользования. Класс SigninForm.cs необходимо для регистрации пользователя в приложении. Класс Settings.cs помогает настроить внешний вид приложения.

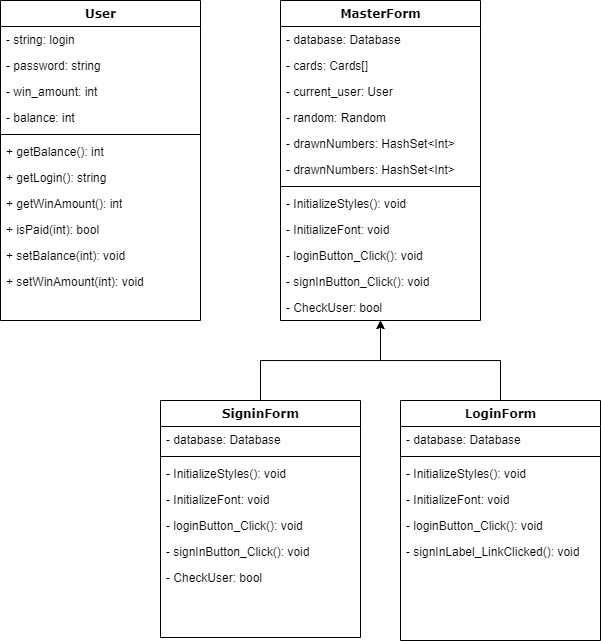


Рисунок 3 Диаграмма классов проекта

### 2.2.4. Диаграмма активности

На рисунке 4 показана диаграмма активности (деятельности) во время сеанса игры.

Диаграмма активности – это UML-диаграмма, на которой показаны действия, состояния. Под деятельностью понимается спецификация исполняемого поведения в виде координированного последовательного и параллельного выполнения подчинённых элементов – вложенных видов деятельности и отдельных действий action, соединённых между собой потоками, которые идут от выходов одного узла ко входам другого.

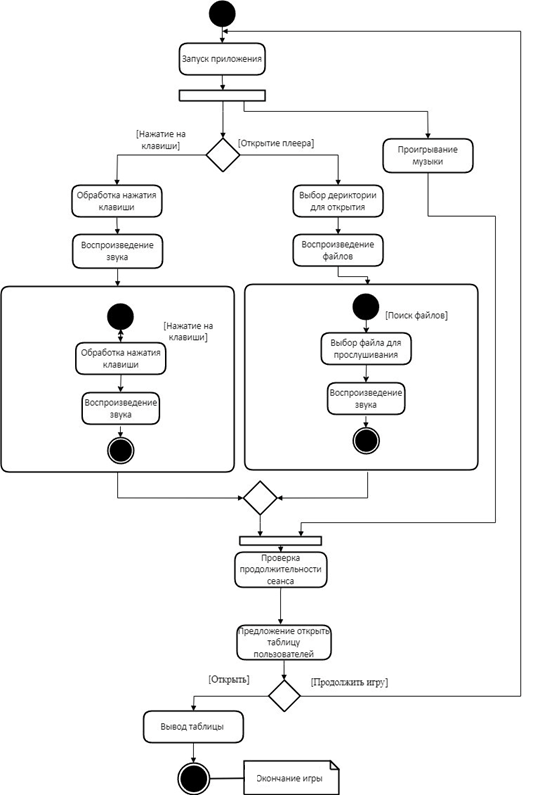


Рисунок 4 Диаграмма активности сеанса игры

## 2.3. Текст программы

Текст программы в соответствии с ГОСТ 19.101-77 (СТ СЭВ 1626-79) и ГОСТ 19.401-79 (СТ СЭВ 3746-82) представляет собой запись программы на исходном языке программирования с необходимыми комментариями. Текст программы представляет собой документ, выполненный машинным способом, и приведен в приложении В.

## 2.4. Описание программы

### 2.4.1. Общие сведения

Приложение «Русское лото» — это интерактивная игра с графическим интерфейсом, созданная для проведения игры в классическое лото в цифровом формате. Пользователь может участвовать в игре, управлять игровыми сессиями и отслеживать результаты, а также настраивать приложение под свои предпочтения.

### 2.4.2. Функциональное назначение

Основное назначение приложения «Русское лото» — предоставление возможности пользователям участвовать в игре в классическое лото в цифровом формате. Приложение предназначено для развлечения и весёлого времяпрепровождения, создавая условия для увлекательной игры в лото в любое время и в любом месте

### 2.4.3. Описание логической структуры системы

Программа содержит 5 основных модулей:

Form1.cs — модуль формы игры, модуль LoginForm.cs — содержит методы работы с базой данных пользователей. Этот класс использует модуль Form1 во время авторизации и при отображении таблицы рекордов. Для регистрации был создал модуль SigninForm.cs, который позволит создать нового пользователя. Модуль SettingsForm.cs помогает настроить графический интерфейс приложения. Так же был добавлен модуль MediaForm.cs с миди-плеером для анализа созданных работ,

Программа использует функции следующих библиотек среды С#:

System, System.Collections.Generic, System.Drawning System.Wave, System.Drawing System.IO, System.Windows.Forms, System.NAudio.CoreAudioApi, System,Diagnostics;

Исполняемый файл программы создан средствами среды C#, имеет имя   
@keys.exe и размер 91 105 Кбайт.

#### 2.4.3.1. Описание методов класса LoginForm.cs

Класс LoginForm предназначен для организации работы с базой данных.

Модуль содержит 2 поля password, и login, и соответствующие им свойства для чтения и установки значений полей, и 2 метода:

public void signInLinkLabel\_LinkClicked() метод для открытия формы регистрации;

public void loginButton\_Click() метод для открытия главной формы после успешной авторизации(Рис. В1);

#### 2.4.3.2. Описание методов класса SigninForm.cs

Класс SinginForm предназначен для регистрирования пользователей в базе данных. Модуль содержит 3 поля: password, login, confirm\_password. Пользователь вводит логин, пароль и подтверждает пароль. После ввода данных, пользователя успешно вносится в базу данных и имеет доступ к своему аккаунту.

#### 2.4.3.3. Описание методов класса Form1.cs

Класс формы Form1.cs предназначен для организации игры, данная форма является главной.

Модуль содержит 10 методов:

public void StartRecording() предназначен для начала записи звуков в файл.

public void StopRecording() предназначен для окончания записи звуков в файл.

private void PlaySound(UnmanagedMemoryStream soundsStream) предназначен для считывания неуправляемых блоков кода из управляемого кода и воспроизведение в звук;

private void OnApplySettings(Color whiteNotes, Color blackNotes…) — метод для обработки события изменения настроек интерфейса пользователем и их сохранение.

private void Start\_Click(object sender, EventArgs e) - метод нажатия на кнопку Start запускает таймер 1 (для прокрутки картинок);

private void timer\_Tick(object sender, EventArgs e) - метод работы таймера, необходимо для подсчёта продолжительности сеанса пользователя и дальнейшего подсчёта самый активных пользователей.

private void Form1\_Load(object sender, EventArgs e) - метод обработки события загрузки формы служит для инициализации таймера, а так же настройки приложения под текущего пользователя.

private void Form1\_FormClosing(object sender, EventArgs e) - метод обработки события закрытия главной формы, открытие таблицы продолжительностей сеанса.

private void whiteKeys\_MouseUp(object sender, EventArgs e) - метод для обработки события нажатия на клавиши пианино, их окраска в другой цвет, характерный для зажатой клавиши.

private void octaveComboBox\_SelectedIndexChanged(object sender, EventArgs e) - метод для обработки события изменения текущей октавы.

### 2.4.4. Используемые технические и программные средства

Для нормального функционирования данной информационной системы необходим компьютер, клавиатура, мышь и следующие технические средства:

* процессор Intel или другой совместимый;
* объем свободной оперативной памяти ~500 Кб;
* объем необходимой памяти на жестком диске ~3Мб;
* стандартный VGA-монитор или совместимый;
* стандартная клавиатура;
* манипулятор «мышь»;
* дополнительно: звуковая карты, колонки или наушники.

### 2.4.5. Вызов и загрузка

Программа может быть загружена как с диска, так и с жесткого диска. В последнем случае требуется предварительно переписать программу с диска на жесткий диск.

Исполняемым файлом программы является файл loto.exe. Для его запуска необходимо дважды щелкнуть по исполняемому файлу левой кнопкой мышки.

## 2.5. Руководство оператора

### 2.5.1. Назначение программы

Основная идея «Русское лото» - предоставить пользователям возможность наслаждаться классической игрой в лото в удобном цифровом формате. Приложение воспроизводит все аспекты традиционной игры, предлагая игрокам виртуальные карточки и генератор случайных чисел для розыгрыша. Пользователи могут выбирать карточки, отслеживать выпавшие числа и отмечать их на своих виртуальных карточках.

### 2.5.2. Выполнение программы и сообщения оператору

Для запуска программы дважды щелкните левой кнопкой мыши по исполняемому файлу @keys.exe.

Процесс работы, этапы работы приложения показаны в виде иллюстраций в приложении Б.

При запуске работы приложения загружается форма авторизации, на которой появляется панель для ввода логина и пароля. Главная форма скрываются до окончания авторизации. При вводе действующего логина и пароля происходит авторизация, панель авторизации скрывается, и главная форма со всеми её компонентами становится доступной. Пользователь может воспользоваться настройками интерфейса нажав на кнопку “interface”. Так же пользователь может записывать свою игру и загружать её по удобному для него пути. При авторизации пользователя начинается подсчёт продолжительности сеанса его нахождения в приложении. При выходе из приложения пользователю предлагается просмотр таблицы самых активных участников, либо можно сразу выйти из приложения.

Так же на главной форме есть Label с логином пользователя, на который в случае нажатия появится окно с информацией о том, сколько времени пользователь провёл в приложении.

## 2.6. Программа и методика испытаний

### 2.6.1. Объект испытаний

Объектом испытаний является игровая программа loto.exe. В этой программе пользователи могут участвовать в классической игре в лото, используя виртуальные карточки и генератор случайных чисел. Пользователи могут выбирать карточки, отслеживать розыгрыш чисел и отмечать выигрышные числа на своих виртуальных карточках. Программа также предоставляет возможность настройки параметров игры, таких как количество карточек, формат розыгрыша и оформление интерфейса, что позволяет адаптировать игру под предпочтения каждого пользователя. В дополнение, пользователи могут изменять графическое оформление приложения, включая выбор цветовой схемы и оформление элементов интерфейса. Программа также включает функции для записи и сохранения игровых результатов, что позволяет пользователям возвращаться к предыдущим играм и анализировать их результаты.

### 2.6.2. Цель испытаний

Целью испытаний является проверка соответствия программы требованиям Технического Задания.

### 2.6.3. Требования к программе

В процессе испытаний подлежат проверке следующие требования к программе:

#### 2.6.3.1. Требования к функциональным характеристикам

**1 версия:**

* Приложение должно обеспечивать генерацию случайных чисел в диапазоне от 1 до 90 для розыгрыша. Пользователь должен иметь возможность видеть выпавшие числа в реальном времени.
* Пользователи могут получать и просматривать свои карточки лото, которые автоматически заполняются уникальными наборами чисел.
* Должна быть реализована функция автоматического подсчета и отображения выигрышных комбинаций на карточках после каждого розыгрыша.
* Пользователи могут осуществлять настройку приложения из предварительно подготовленных тем для изменения дизайна, включая цвет фона, цвета панели инструментов и другие элементы интерфейса.

**2 версия:**

* Приложение должно поддерживать импорт и экспорт данных о текущей игре, таких как карточки и результаты розыгрыша, в простом формате (например, текстовый файл), чтобы пользователи могли делиться данными с другими игроками.
* Должна быть предусмотрена возможность сохранения настроек игры, чтобы пользователи могли быстро возобновить игру с последней сохранённой конфигурации.

#### 2.6.3.2. Требования к информационной и программной совместимости

Для полноценного функционирования данной системы необходимо наличие операционной системы выше Microsoft Windows 7 или совместимой. Язык интерфейса - русский.

#### 2.6.3.3. Требования к маркировке и упаковке

Программа должна поставляться в виде проекта, исполняемого (еxе) файла, установщика и документации.

#### 2.6.3.4. Требования к транспортировке и хранению

Программа распространяется в электронном виде. Требования к транспортировке и хранению не предъявляются.

### 2.6.4. Требования к программной документации

На испытания должны быть представлены следующие программные документы:

* техническое задание;
* текст программы;
* описание программы;
* руководство оператора;
* описание языка.

### 2.6.5. Средства и порядок испытаний

Для проведения испытаний необходимы следующие технические средства:

* процессор Intel или другой совместимый;
* объем свободной оперативной памяти ~500 Кб;
* объем необходимой памяти на жестком диске ~3Мб;
* стандартный VGA-монитор или совместимый;
* стандартная клавиатура;
* манипулятор «мышь»;
* дополнительно: звуковая карты, колонки или наушники;

Для проведения испытаний необходимы следующие программные средства:

Операционная система Windows 7.

Испытания проводятся в следующем порядке:

1) проверяется наличие и комплектность программной документации (п.2.6.4);

2) проверяется соответствие требованиям к маркировке и упаковке (п.2.6.3.3);

3) проверяется соответствие требованиям к функциональным характеристикам (п.2.6.3.1);

4) проверяется соответствие требованиям к информационной и программной совместимости (п.2.6.3.2).

### 2.6.6. Методы испытаний

#### 2.6.6.1. Для проверки способности программы осуществлять авторизацию пользователя по введённому логину и паролю, необходимо:

* запустить программу;
* ввести действующий логин и пароль в поля авторизации;
* нажать кнопку «Войти»;
* убедиться, что форма авторизации скрылась, а форма с игрой стала доступна. (авторизация произошла).

#### 2.6.6.2. Для проверки способности программы обеспечивать добавление новых пользователей (регистрацию), необходимо:

* запустить программу;
* нажать на ссылку «Зарегистрироваться»;
* убедиться, что появилось новое окно с вводом данных «Регистрация»;
* после ввода данных нажать «Зарегистрироваться»;
* нажать кнопку «Войти» и убедиться в открытии главной формы (авторизация произошла).

#### 2.6.6.3. Для проверки способности программы воспроизводить несколько клавиш одновременно необходимо:

* запустить программу;
* ввести действующий логин и пароль в поля авторизации;
* нажать кнопку «log in»;
* убедиться, что клавиши воспроизводятся одновременно, используя клавиатуру.

#### 2.6.6.4. Для проверки способности программы обеспечивать запись мелодии, которую играет пользователь:

* запустить программу;
* ввести действующий логин и пароль в поля авторизации;
* нажать кнопку «log in»;
* нажать на кнопку «start»;
* нажать на кнопку «stop»;

#### 2.6.6.5. Для проверки способности программы обеспечивать открытие формы “@settings” необходимо:

* запустить программу;
* ввести действующий логин и пароль в поля авторизации;
* нажать кнопку «log in»;
* нажать кнопку «interface»;
* убедиться, изменении цветов необходимых компонентов и нажатии кнопки “apply”, изменения происходят на главной форме.

#### 2.6.6.6. Для проверки способности программы завершать свою работу в случае закрытия окна авторизации вручную, необходимо:

* запустить программу;
* ввести действующий логин и пароль в поля авторизации;
* нажать кнопку «log in»;
* убедиться, что при закрытии окна с игрой, работа приложения завершается.

2.6.6.7. Для проверки способности программы сохранять продолжительность сеанса пользователя и обеспечивать просмотр таблицы аккаунтов различных пользователей, необходимо:

* запустить программу;
* ввести действующий логин и пароль в поля авторизации;
* нажать кнопку «log in»;
* убедиться, что при продолжительном сеансе нахождения в приложении, таймер увеличивается;
* выйти из приложения нажав на кнопку в menuStrip.
* убедиться в измененном результате таблицы при закрытии главной формы.

## 2.7. Протокол испытаний

Результаты испытаний программы представлены в таблице 1, рисунки приведены в приложении Б.

**Результаты испытаний программы**

*Таблица 1 – Протокол испытаний*

| **№** | **Проверяемые требования** | **Сообщения программы и вводимые значения** | **Ожидаемые результаты** | **Результат** |
| --- | --- | --- | --- | --- |
| 1 | Способность программы осуществлять авторизацию пользователя по введённому логину и паролю | * запустить программу; * ввести действующий логин и пароль в поля авторизации; * нажать кнопку «log in». | Скрываются поля для авторизации и становится главная форма (авторизация произошла). | Пройден  Рис. Б1 |
| 2 | Способность программы обеспечивать добавление новых пользователей (регистрацию) | * запустить программу; * нажать на ссылу “haven’t account yet?” * ввести логин, пароль и подтвердить пароль. * нажать sign in | Скрываются поля для авторизации и становиться доступным форма «Sign in» (добавление нового пользователя произошло) | Пройден  Рис. Б2 |
| 3 | Способность программы изменять графический интерфейс пользователем. | * запустить программу; * ввести действующий логин и пароль в поля авторизации; * нажать кнопку «log in»; * нажать кнопку «interface». | После вносимых изменений и нажатия кнопки “apply”, все изменения должны отобразится в главной форме | Пройден  Рис. Б3-Б4 |
| 4 | Способность программы обеспечивать запись игры пользователя и вести подсчёт продолжительности сеанса | * запустить программу; * ввести действующий логин и пароль в поля авторизации; * нажать кнопку «log in»; * нажать кнопку «start»; * нажать кнопку «stop»; * убедиться в правильности звучания клавиш | После входа на главную форму, сразу начинается подсчёт продолжительности сеанса работы в приложении пользователем. После записи пользователем своей игры, она должна отобразиться там же, где выбрал пользователь. | Пройден  Рис. Б4 |
| 5 | Способность программы завершать свою работу в случае закрытия игрового окна вручную, или нажатии на кнопку “Exit” | * запустить программу; * ввести действующий логин и пароль в поля авторизации; * нажать кнопку «Exit»; | При нажатии на кнопку “Exit” программа должна завершать сеанс игры и выводить таблицу рекордов; при закрытии окна с игрой работа приложения завершается. | Пройден  Рис. Б5 |
| 6 | Способность программы сохранять продолжительность сеанса работы пользователя и обеспечивать просмотр таблицы аккаунтов различных пользователей | * запустить программу; * ввести действующий логин и пароль в поля авторизации; * нажать кнопку «log in» * убедиться, что при продолжительном сеансе игры, таймер в верхнем правом углу ведёт подсчёт времени работы пользователя в приложении. | В базе данных SQlite должны сохраняться данные: логин, пароль, а также количество времени, проведенного пользователем в приложении. | Пройден  Рис. Б6 |

ЗАКЛЮЧЕНИЕ

Разработанное приложение «Русское лото» полностью соответствует всем требованиям технического задания, что подтверждается протоколом испытаний. Это приложение предназначено для использования как в развлекательных целях, так и для организации игровых мероприятий. Пользователи могут участвовать в игре в лото, управлять игровыми сессиями, изменять графический интерфейс программы и сохранять данные о текущих играх. Приложение предлагает различные настройки для адаптации игрового процесса под предпочтения пользователей и обеспечивает удобное и увлекательное времяпрепровождение в формате цифрового лото.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Г. Дженнифер, С. Эндрю, Язык программирования C#, СПб:-ПИТЕР, 2022 (Дата обращения: 01.09.2024)
2. Марк Прайс C# 10 и .NET 6. Современная кросс-платформенная разработка, СПб:-БХВ, 2019 (Дата обращения: 05.09.2024)
3. Евдокимов П., C# на примерах. Практика, практика и только практика, СПб:-БХВ, 2022
4. Документация по C# [Электронный ресурс] — URL: <https://learn.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp> (дата обращения: 10.09.2024)
5. Стандарты Единой Системы Программной Документации:

ГОСТ 19.105-78 Общие требования к программным документам

ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом

ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению

ГОСТ 19.301-78 Программа и методика испытаний. Требования к содержанию и оформлению

ГОСТ 19.401-78 Текст программы. Требования к содержанию и оформлению

ГОСТ 19.402-78 Описание программы. Требования к содержанию и оформлению

ГОСТ 19.505-79 Руководство оператора. Требования к содержанию и оформлению

ГОСТ 19.701-90 Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения

ПРИЛОЖЕНИЕ А

Министерство ОБРАЗОВАНИЯ И науки Российской Федерации  
Федеральное автономное образовательное учреждение высшего профессионального образования  
«Санкт-Петербургский государственный политехнический университет Петра Великого»  
(ФГАОУ ВО «СПбПУ)  
**Институт среднего профессионального образования**

УТВЕРЖДАЮ  
Председатель ПЦК   
\_\_\_\_\_\_\_\_\_\_\_ Андреев В.А.  
\_\_ .\_\_. 2024

ИГРА С ГРАФИЧЕСКИМ ИНТЕРФЕЙСОМ «Русское лото»  
**Техническое задание**  
Листов 8

ПРОВЕРИЛ  
Преподаватель   
\_\_\_\_\_\_\_\_\_\_\_ Девятко Н.С.  
22.01.2024

ВЫПОЛНИЛ  
Студент группы 42919/7   
\_\_\_\_\_\_\_\_\_\_\_ Косовский С.А  
22.01.2024

2024

1. ВВЕДЕНИЕ
   1. Полное наименование программной разработки: «ИГРА С ГРАФИЧЕСКИМ ИНТЕРФЕЙСОМ «Русское лото».
   2. Приложение с интуитивно понятным графическим интерфейсом, предназначенное для имитации звуков пианино. Пользователь может создавать музыкальные композиции, нажимая на виртуальные клавиши пианино на экране. Цель программы – предоставить пользователю возможность экспериментировать с мелодиями и создавать собственные музыкальные произведения, аналогично игре на настоящем пианино. Управление осуществляется с помощью мыши или клавиатуры, позволяя гибко и удобно взаимодействовать с приложением.
   3. Программа «Русское лото» разделена на теоретическую и практическую части. Теоретическая часть включает детальное описание предметной области задачи, анализ методов создания и управления игровым процессом, а также обзор применяемых средств программирования, таких как C# с использованием Windows Forms. Практическая часть охватывает проектирование программного продукта, создание программных модулей для реализации игрового процесса «Русское лото» на языке программирования C#, с использованием интегрированной среды разработки Visual Studio и фреймворка Windows Forms. В этой части будет рассмотрено создание алгоритмов генерации и отображения чисел, управление игровыми сессиями, а также взаимодействие с пользовательским интерфейсом, обеспечивая удобный и интуитивно понятный дизайн с помощью Windows Forms.
2. ОСНОВАНИЕ ДЛЯ РАЗРАБОТКИ
   1. Разработка ведётся на основании задания к курсовому проекту по профессиональному модулю ПМ.01 «Разработка программных модулей программного обеспечения для компьютерных систем» МДК 01.01 «Разработка программных модулей» и утверждена Институтом среднего профессионального образования.
3. НАЗНАЧЕНИЕ РАЗРАБОТКИ
   1. Этот программный продукт — это интерактивное виртуальное приложение для игры в «Русское лото». Он предоставляет пользователям возможность участвовать в классической игре в лото через графический интерфейс, позволяя легко и удобно управлять игровыми сессиями. Приложение включает в себя генерацию чисел, ведение учета выпавших номеров и контроль за заполнением карточек, создавая увлекательный и увлекательный игровой процесс. Наше приложение «Русское лото» обладает привлекательным дизайном и интуитивно понятным интерфейсом, что делает процесс игры комфортным и
   2. Этот игровой программный продукт разработан для универсального использования и призван доставить удовольствие пользователям любого возраста. Он не требует финансовых вложений или использования платежных систем для игры, делая его доступным для всех без исключения. Программа предназначена исключительно для развлекательных целей, обеспечивая при этом широкий спектр интерактивного игрового опыта.

4. ТРЕБОВАНИЯ К РАЗРАБОТКЕ

4.1. Требования к функциональным характеристикам:

**1 версия:**

* Приложение должно обеспечивать генерацию случайных чисел в диапазоне от 1 до 90 для розыгрыша. Пользователь должен иметь возможность видеть выпавшие числа в реальном времени.
* Пользователи могут получать и просматривать свои карточки лото, которые автоматически заполняются уникальными наборами чисел.
* Должна быть реализована функция автоматического подсчета и отображения выигрышных комбинаций на карточках после каждого розыгрыша.
* Пользователи могут осуществлять настройку приложения из предварительно подготовленных тем для изменения дизайна, включая цвет фона, цвета панели инструментов и другие элементы интерфейса.

**2 версия:**

* Приложение должно поддерживать импорт и экспорт данных о текущей игре, таких как карточки и результаты розыгрыша, в простом формате (например, текстовый файл), чтобы пользователи могли делиться данными с другими игроками.
* Должна быть предусмотрена возможность сохранения настроек игры, чтобы пользователи могли быстро возобновить игру с последней сохранённой конфигурации.

4.2. Требования к надежности:

* использование лицензированного программного обеспечения;
* проверка программы на наличие вирусов;
* организация бесперебойного питания.

4.3. Требования к составу и параметрам технических средств

Для нормального функционирования данной информационной системы необходим компьютер, клавиатура, мышь и следующие технические средства:

- процессор Intel или другой совместимый;

- объем свободной оперативной памяти ~500 Кб;

- объем необходимой памяти на жестком диске ~20Мб;

- стандартный VGA-монитор или совместимый;

- стандартная клавиатура;

- манипулятор «мышь»;

- встроенная звуковая карта;

- наушники, колонки.

4.4. Требования к информационной и программной совместимости

Для полноценного функционирования данной системы необходимо наличие операционной системы выше Microsoft Windows 10 или совместимой. Язык интерфейса – русский.

4.5. Требования к маркировке и упаковке

Программа должна поставляться в виде проекта, исполняемого (еxе) файла, установщика и документации.

4.6. Требования к транспортировке и хранению

Программа распространяется в электронном виде. Требования к транспортировке и хранению не предъявляются.

4.7. Специальные требования

Теоретическая часть включает описание предметной области задачи, анализ методов её решения, обзор и обоснование выбора средств программирования. Практическая часть включает подробное описание проектирования программного продукта, разработку и реализацию на языке программирования C# с использованием среды программирования Visual Studio программных модулей программного продукта.

5. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

5.1. Предварительный состав программной документации:

- «Техническое задание»;

- разрабатываемые программные модули должны быть самодокументированы, т.е. тексты программ должны содержать все необходимые комментарии;

- разрабатываемое программное обеспечение должно включать справочную систему.

5.2. Перечень материалов пояснительной записки



Рисунок 1 – Перечень материалов пояснительной записки

6. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

Технико-экономические показатели не рассчитываются.

7. СТАДИИ И ЭТАПЫ РАЗРАБОТКИ

Таблица 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание стадии** | **Содержание этапа** | **Срок 2024 г.** | | **Форма  отчетности** |
| **начало** | **конец** |
| Техническое задание | Составление технического задания | - | - | Техническое задание |
| Эскизный проект | Проектирование программы | - | - | UML-диаграммы |
| Рабочий проект | Разработка спецификаций | - | - | Спецификации отдельных компонентов |
| Составление программы | - | - | Программная документация |
| Приёмо-сдаточные испытания | - | - | Протокол испытаний (п. 2.7 пояснительной записки) |
| Приёмка | Защита курсового проекта | - | - | Оценка за курсовой проект |

8. ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ

8.1. Порядок контроля

Контроль выполнения должен осуществляться руководителем курсового проекта (преподавателем) в соответствие с п.7.

8.2. Порядок приемки

Приемка должна осуществляться с участием руководителя после проведения приемо-сдаточных испытаний. В результате защиты курсового проекта должна быть выставлена оценка за курсовой проект.

ПРИЛОЖЕНИЕ Б

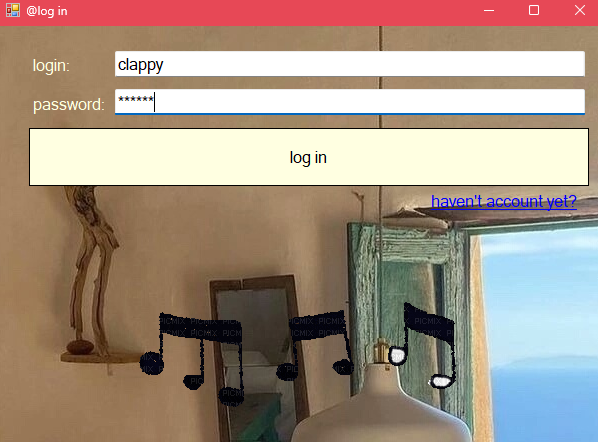


Рисунок Б1 Авторизация

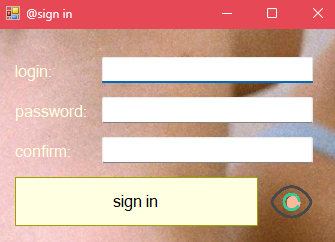


Рисунок Б2 Регистрация нового пользователя

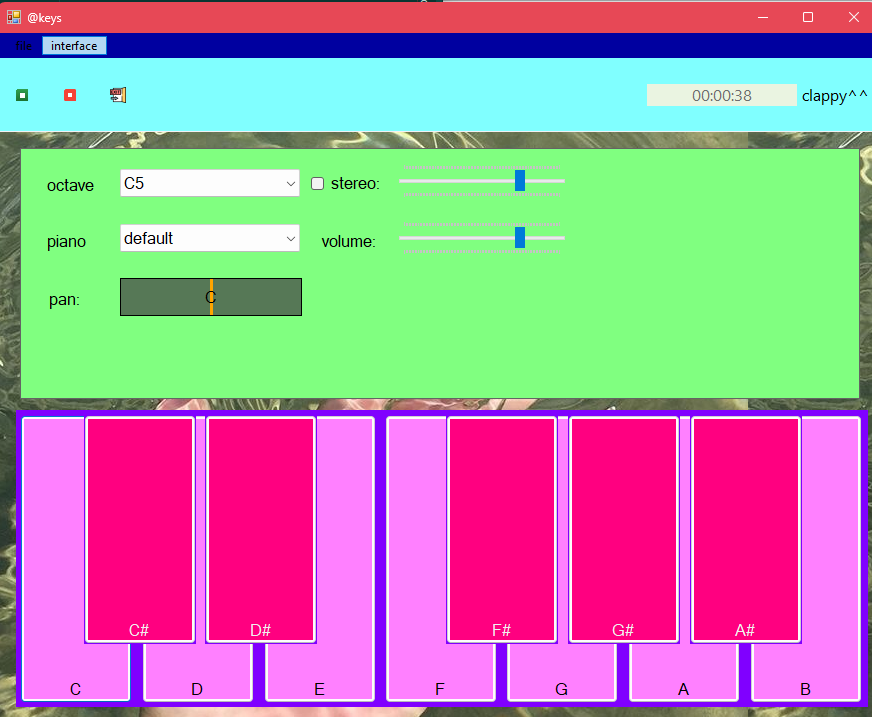


Рисунок Б3 Изменение интерфейса



Рисунок Б4 Форма настроек

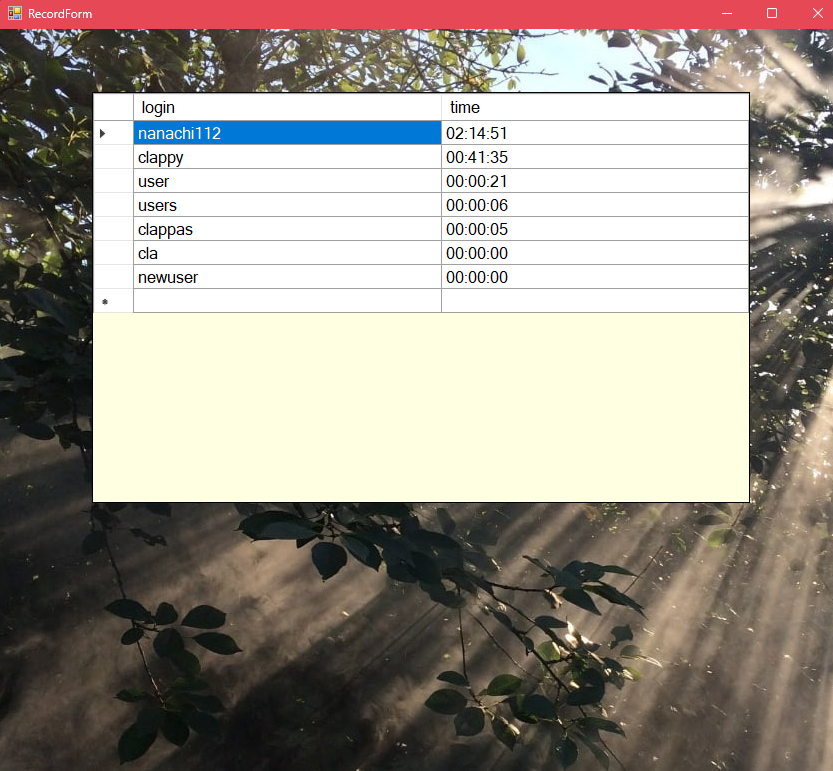


Рисунок Б5 Окончание работы приложение, просмотр таблицы продолжительности сеанса

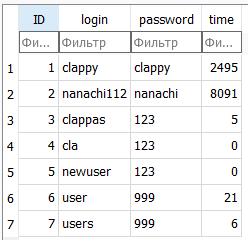


Рисунок Б6 Изменения в базе данных.

ПРИЛОЖЕНИЕ В

Модуль Form1.cs

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Windows.Forms;

namespace RussianLoto

{

public partial class MainForm : Form

{

private Database database = new Database();

private Card[] cards = new Card[] { new Card() };

private User current\_player;

private Random random = new Random();

private HashSet<Int32> drawnNumbers = new HashSet<Int32>();

private static int current\_barrel\_number = 0;

private Difficulty difficulty;

private int win\_bonus\_pay;

private List<Barrel> barrels = new List<Barrel>();

public MainForm(User current\_user)

{

InitializeComponent();

InitializeFont();

InitializeDifficulty();

InitializeColors();

InitializePlayer(current\_user);

InitializeBarrels();

InitializeCards();

InitializeDataGridView();

InitializeOtherComponents();

}

// load

private void MainForm\_Load(object sender, EventArgs e) { }

// initializations

private void InitializeDifficulty() { this.difficulty = Difficulty.Easy; }

private void InitializeCards() { cards[0] = new Card(); }

private void InitializeOtherComponents()

{

balanceToolStripLabel.Text = ("Баланс: " + current\_player.getBalance() + " €").Trim().ToString();

nameToolStripLabel.Text = ("Логин: " + current\_player.getLogin()).Trim().ToString();

}

private void InitializeBarrels()

{

for (Int32 i = 1; i <= 90; ++i)

{

barrels.Add(new Barrel(i));

}

ShakeTheBagOfBarrels();

}

private void ShakeTheBagOfBarrels()

{

this.barrels = barrels.OrderBy(a => random.Next()).ToList();

}

private void InitializeTimer()

{

switch(difficulty)

{

case Difficulty.Easy:

timer.Interval = 5000;

win\_bonus\_pay = 300;

break;

case Difficulty.Medium:

timer.Interval = 4000;

win\_bonus\_pay = 400;

break;

case Difficulty.Hard:

timer.Interval = 3500;

win\_bonus\_pay = 500;

break;

}

timer.Tick += new EventHandler(timer\_Tick);

timer.Start();

}

private void InitializeDataGridView()

{

cardsFlowLayoutPanel.Controls.Clear();

for (int i = 0; i < cards.Length; ++i)

{

DataGridView dataGridView = new DataGridView();

dataGridViewSettings(dataGridView);

fillDataGridView(dataGridView, cards[i]);

cardsFlowLayoutPanel.Controls.Add(dataGridView);

}

}

private void InitializePlayer(User current\_user) { this.current\_player = current\_user; }

private void InitializeColors()

{

}

private void InitializeFont() { Font = new Font("Roboto", 12, FontStyle.Regular); }

// datagridview settings

private void fillDataGridView(DataGridView dataGridView, Card card)

{

for (int row = 0; row < 3; row++)

{

for (int col = 0; col < 9; col++)

{

if (card.numbers[row, col] == 0)

{

dataGridView.Rows[row].Cells[col].Value = "";

}

else

{

dataGridView.Rows[row].Cells[col].Value = card.numbers[row, col];

}

}

}

}

private void dataGridViewSettings(DataGridView dataGridView)

{

dataGridView.CellClick += DataGridView\_CellClick;

dataGridView.ColumnCount = 9;

dataGridView.RowCount = 3;

dataGridView.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

dataGridView.AutoSizeRowsMode = DataGridViewAutoSizeRowsMode.AllCells;

dataGridView.RowHeadersVisible = false;

dataGridView.ColumnHeadersVisible = false;

dataGridView.ReadOnly = true;

dataGridView.AllowUserToResizeColumns

= dataGridView.AllowUserToResizeRows = false;

dataGridView.Width = 315;

}

private int full\_row\_marked\_award = 0;

private void DataGridView\_CellClick(object sender, DataGridViewCellEventArgs e)

{

try

{

DataGridView dataGridView = (DataGridView)sender;

int rowIndex = e.RowIndex,

columnIndex = e.ColumnIndex,

cardIndex = cardsFlowLayoutPanel.Controls.IndexOf(dataGridView);

if (dataGridView.Rows[rowIndex].Cells[columnIndex].Value != null &&

(Int32)dataGridView.Rows[rowIndex].Cells[columnIndex].Value != current\_barrel\_number)

{

dataGridView.Rows[rowIndex].Cells[columnIndex].Style.BackColor = Color.LightGreen;

dataGridView.Rows[rowIndex].Cells[columnIndex].Style.ForeColor = Color.DarkGreen;

cards[cardIndex].marked[rowIndex, columnIndex] = true;

if (cards[cardIndex].IsRowComplete(rowIndex))

{

switch(rowIndex)

{

case 0:

full\_row\_marked\_award = 300;

break;

case 1:

full\_row\_marked\_award = 600;

break;

case 2:

full\_row\_marked\_award = 1200;

break;

}

MessageBox.Show($"Строка {rowIndex + 1} на карточке {cardIndex + 1} заполнена!", "Поздравляем!", MessageBoxButtons.OK, MessageBoxIcon.Information);

this.current\_player.setBalance(this.current\_player.getBalance() + full\_row\_marked\_award + win\_bonus\_pay);

this.current\_player.setWinAmount(this.current\_player.getWinAmount() + full\_row\_marked\_award + win\_bonus\_pay);

}

}

}

catch (Exception)

{

MessageBox.Show("Это не правильное число!", "=(", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

// click events

private void nextRoundButton\_Click(object sender, EventArgs e)

{

if (!this.current\_player.isPaid(cards\_count))

{

MessageBox.Show("К сожалению у вас не хватает средств для покупки билета!", "=(", MessageBoxButtons.OK, MessageBoxIcon.Information);

return;

}

else

{

this.current\_player.setBalance(this.current\_player.getBalance() - 50 \* cards\_count);

InitializeOtherComponents();

gameSettingsPanel.Enabled = false;

InitializeTimer();

Play();

}

}

private void Play()

{

int drawnNumber;

do

{

drawnNumber = random.Next(1, 91);

} while (drawnNumbers.Contains(drawnNumber));

drawnNumbers.Add(drawnNumber);

drawnNumberLabel.Text = "Последние бочки -> " + string.Join(", ", drawnNumbers.Reverse());

current\_barrel\_number = drawnNumber;

}

private void TheEnd()

{

MessageBox.Show("Игра закончена!", "Игра", MessageBoxButtons.OK, MessageBoxIcon.Information);

gameSettingsPanel.Enabled = true;

timer.Stop();

CalculateResults();

}

private void CalculateResults()

{

}

// tick event

private void timer\_Tick(object sender, EventArgs e)

{

if (drawnNumbers.Count < 90)

Play();

else TheEnd();

}

private static int cards\_count = 0;

private void allChanges\_RadioButton(object sender, EventArgs e)

{

RadioButton current\_radiobutton = (RadioButton)sender;

switch (current\_radiobutton.Name)

{

case "oneCardRadioButton":

cards\_count = 1;

break;

case "twoCardRadioButton":

cards\_count = 2;

break;

case "threeCardRadioButton":

cards\_count = 3;

break;

case "fourthCardRadioButton":

cards\_count = 4;

break;

}

Array.Resize(ref cards, cards\_count);

for (int i = 0; i < cards\_count; ++i)

{

cards[i] = new Card();

}

InitializeDataGridView();

}

private void changeDifficulty\_RadioButtons(object sender, EventArgs e)

{

RadioButton current\_radiobutton = (RadioButton)sender;

switch(current\_radiobutton.Name)

{

case "easyRadioButton":

this.difficulty = Difficulty.Easy;

MessageBox.Show("Вы выбрали легкую сложность", "Игра", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

break;

case "mediumRadioButton":

this.difficulty = Difficulty.Medium;

MessageBox.Show("Вы выбрали среднюю сложность", "Игра", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

break;

case "hardRadioButton":

this.difficulty = Difficulty.Hard;

MessageBox.Show("Вы выбрали тяжёлую сложность", "Игра", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

break;

}

}

private void appartmentButton\_Click(object sender, EventArgs e)

{

}

private void MainForm\_FormClosing(object sender, FormClosingEventArgs e)

{

int userId = GetUserIdByLogin(current\_player.getLogin());

if (userId != -1) // Если удалось получить ID

{

string change\_query = $"UPDATE Players SET balance = {this.current\_player.getBalance()}, win\_amount = {this.current\_player.getWinAmount()} WHERE ID = {userId}";

using (SqlCommand command = new SqlCommand(change\_query, database.get\_connection()))

{

try

{

database.open\_connection();

command.ExecuteNonQuery(); // Выполняем запрос на обновление

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка при обновлении данных", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

database.close\_connection();

}

}

}

else

{

MessageBox.Show("Ошибка: ID пользователя не найден!", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private int GetUserIdByLogin(string login)

{

int userId = -1;

string query = "SELECT ID FROM Players WHERE login = @login";

using (SqlCommand command = new SqlCommand(query, database.get\_connection()))

{

command.Parameters.AddWithValue("@login", login);

try

{

database.open\_connection();

userId = (int)command.ExecuteScalar(); // Получаем ID

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка при получении ID", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

database.close\_connection();

}

}

return userId;

}

private void exitToolStripButton\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Вы уверены что хотите выйти? Процесс будет не сохранен!", "Игра", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == DialogResult.Yes) this.Close();

}

private void informationToolStripMenuItem\_Click(object sender, EventArgs e)

{

MessageBox.Show("hello");

}

}

}

Модуль LoginForm.cs

using using System;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Windows.Forms;

namespace RussianLoto

{

public partial class LoginForm : Form

{

private Database database = new Database();

public LoginForm()

{

InitializeComponent();

InitializeFont();

InitializeStyles();

StartPosition = FormStartPosition.CenterScreen;

}

private void InitializeStyles()

{

Font = new Font("Roboto", 14, FontStyle.Regular);

BackColor = Color.LightBlue;

loginTextBox.Font = new Font("Roboto", 12);

loginTextBox.BackColor = Color.White;

loginTextBox.ForeColor = Color.DarkBlue;

loginTextBox.BorderStyle = BorderStyle.FixedSingle;

passwordTextBox.BackColor = Color.White;

passwordTextBox.ForeColor = Color.DarkBlue;

passwordTextBox.BorderStyle = BorderStyle.FixedSingle;

passwordTextBox.UseSystemPasswordChar = true;

// Стили для кнопки входа

loginButton.Font = new Font("Roboto", 14, FontStyle.Bold);

loginButton.BackColor = Color.DarkBlue;

loginButton.ForeColor = Color.White;

loginButton.FlatStyle = FlatStyle.Flat;

signInLabel.Font = new Font("Roboto", 10, FontStyle.Underline);

signInLabel.ForeColor = Color.DarkBlue;

signInLabel.LinkColor = Color.Blue;

signInLabel.ActiveLinkColor = Color.Red;

signInLabel.LinkBehavior = LinkBehavior.HoverUnderline;

}

private void InitializeFont() { Font = new Font("Roboto", 14, FontStyle.Regular); }

private void loginButton\_Click(object sender, EventArgs e)

{

try

{

SqlDataAdapter data\_adapter = new SqlDataAdapter();

DataTable data\_table = new DataTable();

string query\_string = $"SELECT ID, [login], [password], win\_amount, balance FROM Players WHERE [login] = '{loginTextBox.Text}' AND [password] = '{passwordTextBox.Text}'";

SqlCommand sql\_command = new SqlCommand(query\_string, database.get\_connection());

data\_adapter.SelectCommand = sql\_command;

data\_adapter.Fill(data\_table);

if (data\_table.Rows.Count == 1)

{

var user = new User(data\_table.Rows[0].ItemArray[0].ToString(), data\_table.Rows[0].ItemArray[1].ToString(), Convert.ToInt32(data\_table.Rows[0].ItemArray[2].ToString()), Convert.ToInt32(data\_table.Rows[0].ItemArray[3].ToString()));

MessageBox.Show("Вы успешно авторизовались!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);

MainForm main\_form = new MainForm(user);

this.Hide();

main\_form.ShowDialog();

this.Show();

}

else MessageBox.Show("Вы не авторизовались!", "Авторазицаия", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

catch (Exception)

{

MessageBox.Show("Что-то пошло не так! Повторите попытку.", "Авторазицаия", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void signInLabel\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

SigninForm signinForm = new SigninForm();

this.Hide();

signinForm.ShowDialog();

this.Show();

}

}

}

Модуль SigninForm.cs

using System;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Windows.Forms;

namespace RussianLoto

{

public partial class SigninForm : Form

{

private Database database = new Database();

public SigninForm()

{

InitializeComponent();

InitializeFont();

InitializeColors();

InitializeStyles();

InitializeComponents();

}

private void InitializeStyles()

{

StartPosition = FormStartPosition.CenterScreen;

BackColor = Color.LightCyan;

loginTextBox.BackColor = Color.White;

loginTextBox.ForeColor = Color.DarkBlue;

loginTextBox.BorderStyle = BorderStyle.FixedSingle;

passwordTextBox.BackColor = Color.White;

passwordTextBox.ForeColor = Color.DarkBlue;

passwordTextBox.BorderStyle = BorderStyle.FixedSingle;

passwordTextBox.UseSystemPasswordChar = true;

signInButton.BackColor = Color.DarkBlue;

signInButton.ForeColor = Color.White;

signInButton.FlatStyle = FlatStyle.Flat;

}

private void SigninForm\_Load(object sender, EventArgs e) { passwordTextBox.UseSystemPasswordChar = true; }

private void InitializeComponents() { StartPosition = FormStartPosition.CenterScreen; }

private void InitializeFont() { Font = new Font("Roboto", 12, FontStyle.Regular); }

private void InitializeColors() { BackColor = Color.LightCyan; }

private void signInButton\_Click(object sender, EventArgs e)

{

try

{

if (CheckUser()) return;

string query\_string = $"INSERT INTO Players([login], [password], win\_amount, balance) VALUES('{loginTextBox.Text}', '{passwordTextBox.Text}', 250, 250)";

SqlCommand sql\_command = new SqlCommand(query\_string, database.get\_connection());

database.open\_connection();

try

{

if (Convert.ToBoolean(sql\_command.ExecuteNonQuery() == 1))

{

MessageBox.Show("Вы успешно зарегистрированы!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);

LoginForm login\_form = new LoginForm();

this.Hide();

login\_form.ShowDialog();

}

else MessageBox.Show("Вы не зарегистрованы! Повторите попытку.", "Регистрация", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception ex) { MessageBox.Show(ex.Message, "Регистрация", MessageBoxButtons.OK, MessageBoxIcon.Error); }

database.close\_connection();

}

catch (Exception)

{

MessageBox.Show("Что-то пошло не так! Повторите попытку.", "Регистрация", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private bool CheckUser()

{

SqlDataAdapter sql\_adapter = new SqlDataAdapter();

DataTable data\_table = new DataTable();

string query\_string = $"SELECT ID, [login], [password] FROM Players WHERE [login] = '{loginTextBox.Text.Trim().ToString()}' and [password] = '{passwordTextBox.Text.Trim().ToString()}'";

SqlCommand sql\_command = new SqlCommand(query\_string, database.get\_connection());

sql\_adapter.SelectCommand = sql\_command;

sql\_adapter.Fill(data\_table);

if (data\_table.Rows.Count > 0)

{

MessageBox.Show("Пользователь уже существует!", "Регистрация", MessageBoxButtons.OK, MessageBoxIcon.Information);

return true;

}

else return false;

}

private void SigninForm\_FormClosed(object sender, FormClosedEventArgs e)

{

LoginForm login\_form = new LoginForm();

this.Hide();

login\_form.ShowDialog();

}

}

}

Модуль SettingsForm.cs