**Введение**

На курсовое проектирование была поставлена задача, разработать программу на тему: Разработка информационно-познавательного приложения «Планета велосипедов».

Цель курсового проекта заключается в знакомстве массового пользователя с велосипедами и их устройством, трюками, которые можно выполнять на велосипеде и интересными маршрутами в городе Гродно.

Создаваемая программа будет рассчитана на любого рода пользователей. Применить данную программу смогут все люди, которые заинтересованы в этой теме.

Далее приведем краткое описание разделов пояснительной записки.

Первый раздел носит название “Анализ задачи”. В нем вы сможете ознакомиться с постановкой задачи, которая включает в себя: исследование предметной области поставленной задачи, определение ее организационно-экономической сущности. Также в этом разделе вы сможете узнать о том, как данная задача решается в настоящее время. Все входные и выходные данные тоже будут описаны в первом разделе. В подразделе “Инструменты разработки” будет рассмотрена среда, в которой создается данный курсовой проект. Здесь также будут установлены минимальные и оптимальные требования к аппаратным характеристикам, обеспечивающим правильное функционирование поставленной задачей.

В разделе “Проектирование задачи” будут рассмотрены основные аспекты разработки программного продукта. Здесь можно будет узнать об организации данных в контексте среды разработки. В данном разделе будет четко описан пользовательский интерфейс, составлены алгоритмы процесса обработки информации, описана разработка системы справочной информации.

“Реализация задачи” – это третий раздел пояснительной записки, в котором описываются все элементы и объекты, которые будут использованы при реализации данного приложения. В этом разделе будут четко описаны функции пользователя и их структура. Здесь можно будет найти таблицу, в которой будет представлена полная аннотация файлов, используемых в данном проекте.

Четвертый раздел – “Тестирование”. В нем будет описано полное и функциональное тестирование данной программы, т.е. будет оттестирован каждый пункт меню, каждая операция, которая выполняется приложением. Будут смоделированы все возможные действия пользователя при работе с программой, начиная от запуска до выхода.

В разделе “Применение” будет описано назначение, область применения, среда функционирования курсовой программы. Также в нем будет описано использование справочной системы.

“Заключение” будет содержать краткую формулировку задачи, результаты проделанной работы, описание использованных методов и средств, описание степени автоматизации процессов на различных этапах разработки.

В “Литературе” будет приведен список используемых при разработке источников.

В приложениях к пояснительной записке будет приведен листинг программы с необходимыми комментариями.

Схема работы системы будет представлена в графической части.

**1.** **Анализ задачи**

**1.1. Постановка задачи**

**Наименование задачи:** «Разработка информационно-познавательного приложения «Планета велосипедов»

**Цель разработки:** Знакомство пользователей с велосипедами и их устройством: подробный разбор элементов руля, трансмиссии, колёс, рамы и т.д. Разбор и обучение различным велосипедным трюкам: езда без рук, лестница, «банни хоп», вилли и т.д.

Также в программу будут включены интересные маршруты города Гродно, правила дорожного движения на велосипеде, советы по подбору велосипеда, мини-игра «Парные картинки с велосипедами». Закрепление полученных знаний будет проходить с помощью тестов: «Тест по устройству велосипеда», «Тест по ПДД», «Тест по видам велосипеда».

**Назначение:** данный программный продукт разрабатывается для людей любого возраста, желающим узнать информацию о велосипедах, либо закрепить уже имеющиеся знания.

**Периодичность использования:** зависит от нужд потребителя, может использоваться ежедневно.

**Источники и способы получения данных:** теоретические материалы об устройстве велосипедов

**Разрабатываемый программный продукт должен позволять выполнить следующие действия:**

- обучение пользователя о устройстве, видах, трюках велосипеда;

- прохождение тестов для закрепления знаний и доступу к игре;

- ознакомление пользователя с интересными маршрутами города Гродно;

**1.2 Инструменты разработки**

Для разработки данного проекта будет выбрана среда Delphi (Rad Studio 10.4) так как это одна из мощнейших сред для разработки приложений такого уровня. Delphi — это среда объектно-ориентированного программирования, относящаяся к классу RAD – (Rapid Application Development – «Средство быстрой разработки приложений»), основанная на языке Object Pascal. Используется для разработки и поддержки программного обеспечения.

Также для разработки программы необходимы:

- Visual Studio Code – текстовый редактор для создания html-страниц и css-стилей к ним

- Brave-браузер – нужен для нахождения текстовой и видеоинформации

- Word 2021 – нужен для написания пояснительной записки

- СУБД MS Access 2021 – создание базы данных пользователей

- Dr.Explain – создание справочной системы

- Smart Install Maker – создание файла установки приложения

При разработке данного программного продукта был использован компьютер со следующими характеристиками:

- Процессор: AMD Athlon Gold 3150U 2.4 GHz;

- ОЗУ: 8Gb;

- Память: SSD 256Gb;

- ОС – Windows 10 Pro.

**1.3 Требования к приложению**

На этапе исследования предметной области был установленный целый ряд требований, которые предъявляют к разрабатываемой задаче.

При моделировании форм следует учесть такие моменты:

- интуитивно понятный интерфейс

- большое количество полезной информации на каждой форме.

Так же следует обратить внимание на такие моменты:

- полноэкранные окна формы;

- визуально понятный и приятный глазу интерфейс;

- красивый пиксельный стиль.

Минимальные системные требования:

- 1.8 GHz процессор (или более быстрый);

- 1 GB RAM;

- от 450 MB доступного места на жестком диске.

Рекомендуемые:

- двухъядерный AMD Athlon или аналогичный процессор Intel (или более быстрый) ;

- 2 GB RAM ;

- от 500 MB доступного места на жестком диске.

Требования к интерфейсу: при разработке приложения должны быть использованы преимущественного гармонирующие с собой разноцветные оттенки. Основные разделы приложения должны быть доступны с первой страницы. Грамотный пользовательский интерфейс.

Следовательно, каждое окно должно иметь ясную визуальную иерархию своих элементов. Фрагменты текста должны располагаться на экране так, чтобы пользователю было просто и понятно принимать информацию.

Пользователь не должен испытывать какого-либо дискомфорта в плане восприятия информация, отображённой на экране. Объекты (рисунки и символы) не должны быть слишком мелкие.

Отказы программы вследствие некорректных действий пользователя при взаимодействии с программой через графический интерфейс не должны влиять на конечный результат.

**2 Проектирование задачи**

**2.1 Организация данных**

Проектирование задачи – это очень важный и ответственный этап в разработке любого приложения.

Важным является он вследствие того, что методы, по средствам которых пользователь управляет формами, построены на высокой степени специализации каждого из компонентов.

Основным средством хранения информации в приложении будут html-страницы с материалами, БД для хранения результатов прохождения тестов всех пользователей.

В проекте материалы по устройству велосипедов, трюки и виды, маршруты города Гродно будут храниться в виде html-страниц с подключением к ним фото- и видеоинформации. Тесты по ПДД, устройству велосипеда и видам велосипеда будут храниться в виде текстовых файлов. В базе данных будет храниться информация о пользователях и их прогрессе в приложении (рисунок 1):

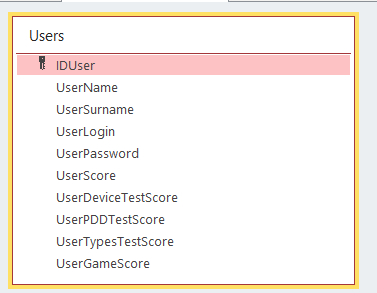


Рисунок 1 – База данных пользователя

Т.е. у нас будет одна таблица с пользователями, где будет храниться основная информации о них самих (ФИО, логин, пароль) и их прогрессе в приложении (общий счёт и счёт отдельно по каждому тесту). Счёт по игре «Парные картинки с велосипедами» будет вестись в отдельном числовом поле.

Система справочной информации будет представлена файлом справки, который будет содержать информацию о приложении и правила его использования.

Основными функциями приложения будут являться:

- изучение теоретического материала;

- прохождение тестов;

- игра «Парные картинки с велосипедами»;

- система накопления очков;

Воспользовавшись пунктом меню «Справка», можно будет узнать информацию о том, как пользоваться программой. Вкладка «О программе» будет расположена на главном меню главной формы.

**2.2 Процессы**

Согласно всем перечисленным требованиям и указаниям, которые были рассмотрены в разделе «Анализ задачи», было определено, чем конкретно должна заниматься разрабатываемая курсовая программа. Главной ее задачей будет являться: знакомство пользователя с миром велосипедов. Весь данный процесс будет заключаться в следующем.

Доступ к разделам будет осуществляться по кликам на компоненты TImage. Сами разделы будут представлены в виде html-страниц с css-стилями на новых формах. Для их отображения будет использоваться компонент TWebBrowser.

Плейлист фоновой музыки загружается с помощью компонента TMediaPlayer, сами пути к музыке расположены в компоненте TListBox, для управления звуком создана специальная кнопка (TImage), так же за музыку отвечает TTimer, который проверяет, закончилась ли музыка. Можно переключать музыки нажатием на кнопки-компоненты TImage.

Тесты будут сделаны с помощью компонентов TRadioGroup и TImage (кнопки), за каждый правильный ответ будут начисляться очки и вносится в базу данных пользователя. Будет организован последовательный доступ к тестам, т.е. пока пользователь не наберёт определённое количество очков по сумме предыдущих тестов, доступ к следующему тесту открыт не будет.

Мини-игра «Парные картинки» с велосипедами будет сделана с помощью склейки картинок в один bmp-файл. Потом эти картинки будут нарезаться и случайным образом располагаться на полях разной размерности на компоненте TPaintBox. Для окончания игры и начисления очков нужно будет открыть все пары картинок. Если пользователь ранее проходил уровень, то начисление очков происходить не будет. Для доступа к игре нужно будет набрать определённое количество очков в тестах.

Система меню для быстрой навигации по приложению будет расположена на главной форме в виде компонента TMainMenu, там будут находится вкладки «Общая информация» (будет раскрываться на подпункты «Устройство велосипеда», «Виды велосипедов», «Трюки на велосипеде», «Маршруты Гродно»). Далее будут идти вкладки «Тесты» и «Парные картинки» для доступа к соответствующим разделам. После них будут идти вкладки «Помощь» (будет раскрываться на «Справка» и «О программе») и «Выйти из аккаунта» Также будет предусмотрена кнопка для обнуления прогресса аккаунта.

**2.3 Описание внешнего пользовательского интерфейса**

Важным при выполнении курсового проекта является организация диалога между, пользователем и самой программой. Во многом это зависит от того, как программист разработает данную программу, какие компоненты будут использованы и какие методы будут автоматизированы. Во-первых, особое внимание следует уделить интерфейсу. Разработчик должен так организовать внешний вид своей программы, чтобы пользователь понял, что от него требуется.

Для организации эффективной работы пользователя нужно создать целостное приложение данной предметной области, в которой все компоненты приложения будут сгруппированы по функциональному назначению. При этом необходимо обеспечить удобный графический интерфейс пользователя. Таким образом ключевые кнопки графически выделены для визуализации интерактивных возможностей приложения (рисунок 2).



Рисунок 2 – Визуализация интерактивных возможностей приложения

Главное меню будет выполнено в пиксельном стиле, так как это простой и своеобразный стиль, который привлекает пользователей. На фоне будет играть музыка в стиле 8-bit, что также будет придавать атмосферности приложению. На рисунке 3 изображён пример пиксельной кнопки.



Рисунок 3 – Пример кнопки в приложении

Разделы будут выполнены с помощью html-страниц и css-стилей. Пример такой странички показан на рисунке 4.

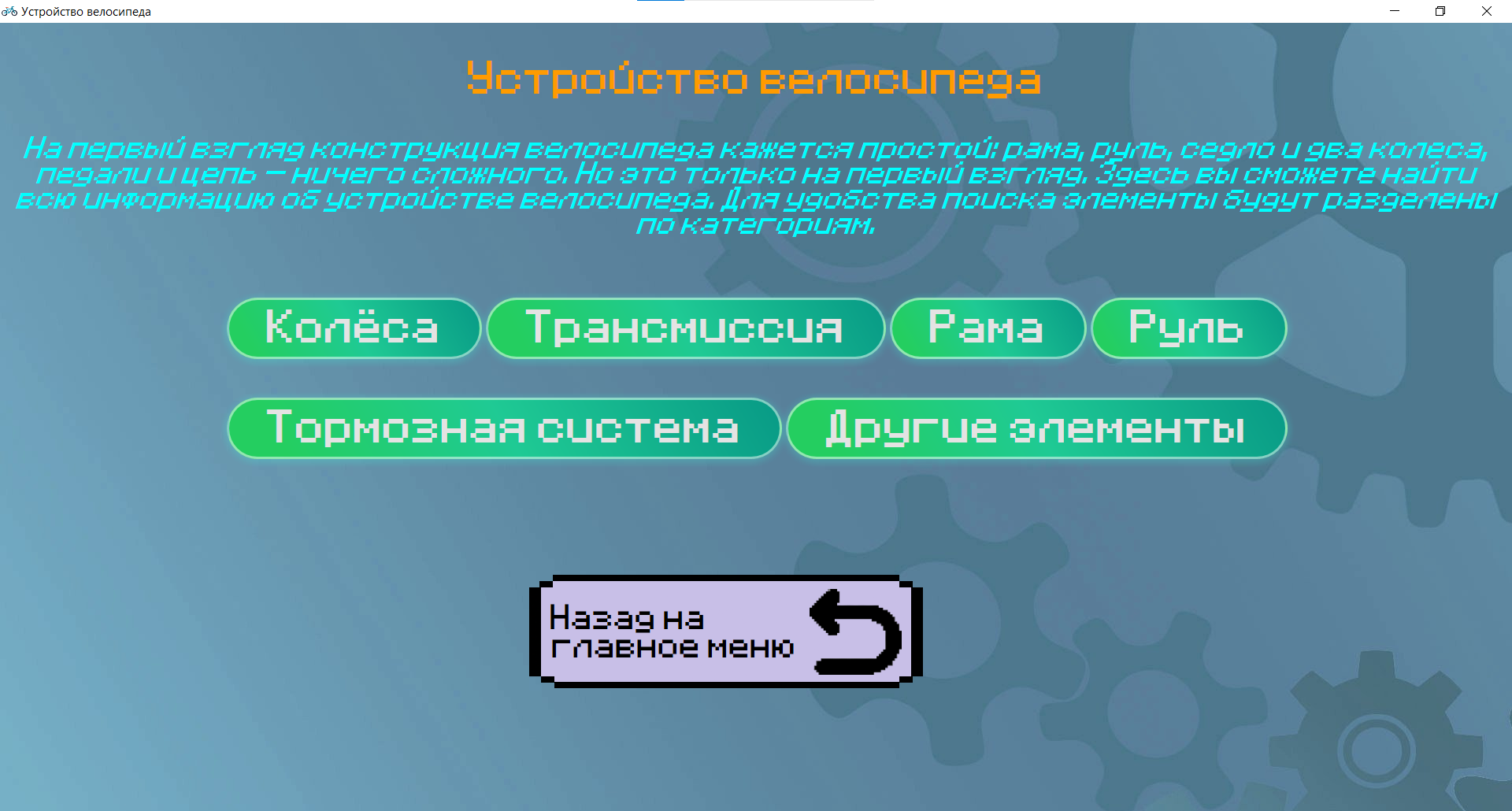


Рисунок 4 – Пример раздела теории

Подпункты страницы преимущественно будут выполнены в модальных окнах, пример показан на рисунке 5.

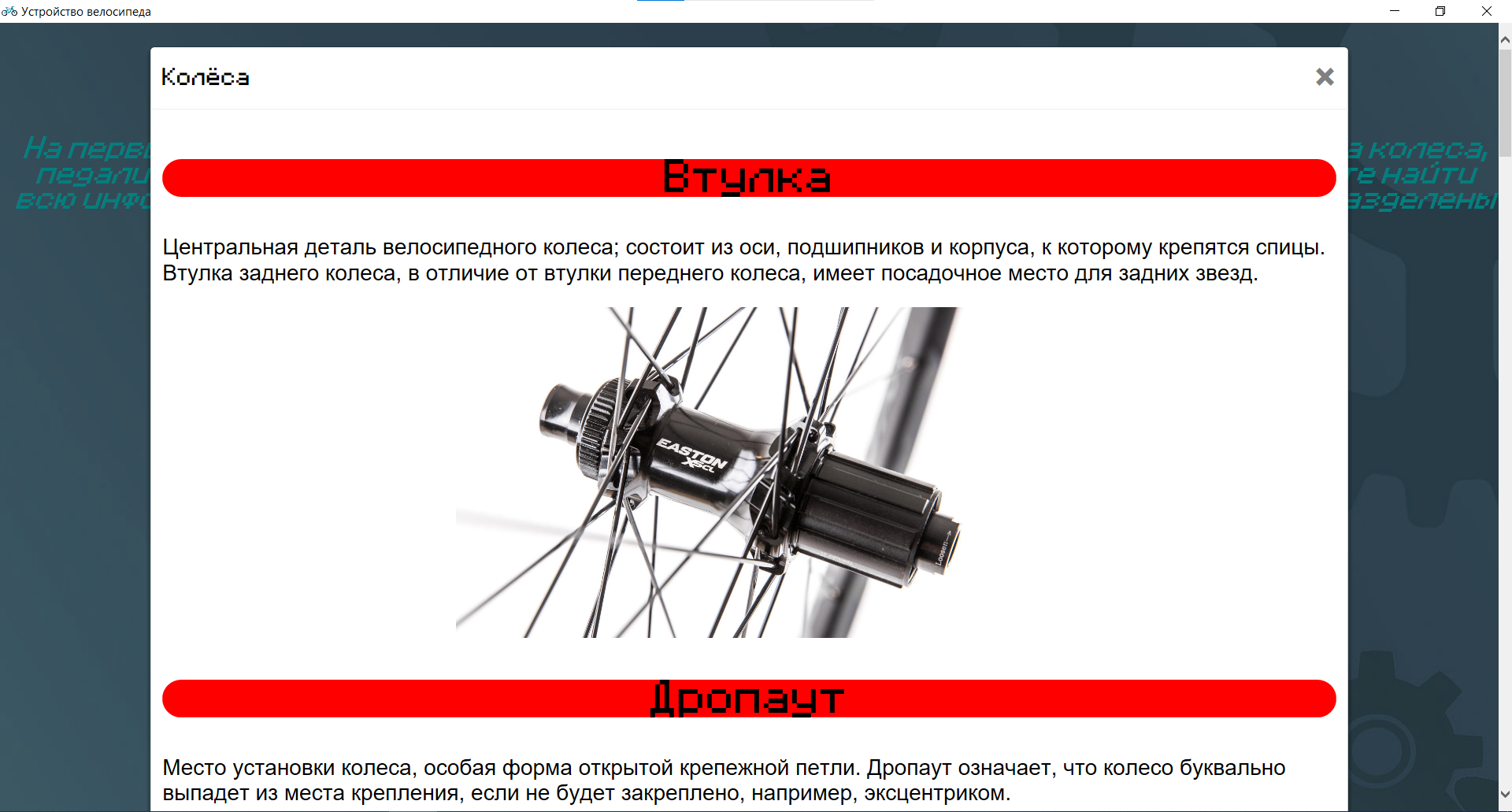


Рисунок 5 – Пример подпункта страницы

Таким образом, для успешной работы всего проекта реализован интуитивно понятный интерфейс и управление с приятной гаммой цветов и шрифтами.

Структура навигации по проекту представлена на рисунке 6.



Рисунок 6 – Структура навигации по проекту

**3 Реализация**

**3.1 Реализация проекта**

**3.1.1 Структура программы**

Данный курсовой проект содержит 10 модулей. Далее рассмотрим назначение каждого модуля:

1. Модуль LoadScr – отвечает за загрузочный экран приложения, после него появляется окно Регистрации/Авторизации (модуль RegistrationForm).

2. Модуль RegistrationForm - отвечает за вход и регистрацию пользователей, с этой формы осуществляется переход на главную форму.

3. Модуль Main – в этом модуле расположена главная форма программы, на ней находятся все кнопки для перехода на разделы теории, здесь осуществляется вывод очков авторизованного пользователя и под каким логином вошёл пользователь. Также тут расположены кнопка включения и выключения звука, выхода из программы и переход к тестам и игре «Парные картинки с велосипедами». В этой форме можно осуществить выход из аккаунта.

4. Модуль BicycleDevice – отвечает за раздел «Устройство велосипеда», тут выводятся категории с частями велосипеда, при выборе которых выводится информация о соответствующих частях велосипеда.

5. Модуль BicycleTypes – отвечает за раздел «Виды велосипедов», тут расположена информация о видах велосипедов, которые располагаются в разных категориях.

6. Модуль BicyclesTricks – отвечает за раздел «Трюки на велосипеде», тут расположена информация о трюках на горном велосипеде и на велосипеде типа BMX. К трюкам приложена видеоинформация.

7. Модуль HrodnaRoutes – отвечает за раздел «Маршруты Гродно», на котором расположена информация о пригородных и городских маршрутах с схемой поездки и достопримечательностями, которые встречаются на пути. Также тут расположены правила дорожного движения.

8. Модуль Test – отвечает за раздел «Тесты», в котором расположены три вида тестов («Тест по устройству велосипеда», «Тест по ПДД», «Тест по видам велосипеда») за которые начисляются очки. Если предыдущий тест не пройден на определённое количество баллов, то доступа к прохождению следующего не будет пока пользователь не наберёт нужно количество очков.

9. Модуль Game – отвечает за игру «Парные картинки с велосипедами», за прохождение программных уровней предусмотрено получение очков, при прохождении всех 4 уровней приложения будет достигнут максимум очков, которые можно получить в приложении.

10. Модуль AboutProgram – содержит информацию о программе;

**3.1.2 Структура и описание процедур и функций пользователя**

Описание разработанных процедур находятся в таблице 1.

Таблица 1 – Процедуры и функции

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 1 | procedure TLoadScreen.FormActivate(Sender: TObject); | LoadScr | LoadScreen | Анимация загрузочного экрана и установка ширины и высоты формы под размеры GIF-изображения |
| 2 | procedure TLoadScreen.LoadTimerTimer(Sender: TObject); | LoadScr | LoadTimer | Отключение обновления таймера |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 3 | procedure TLoadScreen.ProgressBarTimerTimer(Sender: TObject); | LoadScr | ProgressBarTimer | Работа полосы загрузки |
| 4 | procedure TReg.AuthButtonClick(Sender: TObject); | RegistrationForm | AuthButton | Авторизация пользователя |
| 5 | procedure TReg.AuthImgClick(Sender: TObject); | RegistrationForm | AuthImg | Включение полей авторизации |
| 6 | procedure TReg.FormCreate(Sender: TObject); | RegistrationForm | Reg | Загрузка картинок и форматирования текста для кнопок, установление расположения полей и кнопок на форме |
| 7 | procedure TReg.FormDestroy(Sender: TObject); | RegistrationForm | Reg | При завершении приложения удалить шрифты из системы пользователя |
| 8 | procedure TReg.N2Click(Sender: TObject); | RegistrationForm | MainMenuReg | Открытие справки |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 9 | procedure TReg.N3Click(Sender: TObject); | RegistartionForm | MainMenuReg | Открытие «О программе» |
| 10 | procedure TReg.RegButtonClick(Sender: TObject); | RegistrationForm | RegButton | Регистрация пользователя |
| 11 | procedure TReg.RegImgClick(Sender: TObject); | RegistrationForm | RegImg | Отображение полей регистрации |
| 12 | procedure TMainForm.ButtonDeviceClick(Sender: TObject); | Main | ButtonDevice | Отображение раздела «Устройство велосипеда» |
| 13 | procedure TMainForm.FormActivate(Sender: TObject); | Main | MainForm | Вывод логина авторизованного пользователя |
| 14 | procedure TMainForm.FormClose(Sender: TObject; var Action: TCloseAction); | Main | MainForm | Полное завершение программы |
| 15 | procedure TMainForm.FormCreate(Sender: TObject); | Main | MainForm | Загрузка шрифта, плейлиста, картинок и форматирование текста и формы |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 16 | procedure TMainForm.FormShow(Sender: TObject); | Main | MainForm | Обновление количества очков авторизованного пользователя |
| 17 | procedure TMainForm.GameButtonClick(Sender: TObject); | Main | GameButton | Открытие игры «Парные картинки с велосипедами» |
| 18 | procedure TMainForm.MusicTimerTimer(Sender: TObject); | Main | MusicTimer | Проверка окончания композиции и вывод названия играющей композиции |
| 19 | procedure TMainForm.AboutProgClick(Sender: TObject); | Main | MainFormMenu | Вывод окна «О программе» |
| 20 | procedure TMainForm.ResetProgressClick(Sender: TObject); | Main | MainFormMenu | Сброс прогресса пользователя |
| 21 | procedure TMainForm.ExitAccountClick(Sender: TObject); | Main | MainFormMenu | Выход из аккаунта пользователя |
| 22 | procedure TMainForm.TestsMenuClick(Sender: TObject); | Main | MainFormMenu | Открытие раздела «Тесты» |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 23 | procedure TMainForm.DeviceMenuClick(Sender: TObject); | Main | MainFormMenu | Открытие раздела «Устройство велосипеда» |
| 24 | procedure TMainForm.TypesMenuClick(Sender: TObject); | Main | MainFormMenu | Открытие раздела «Виды велосипедов» |
| 25 | procedure TMainForm.TricksMenuClick(Sender: TObject); | Main | MainFormMenu | Открытие раздела «Трюки на велосипеде» |
| 26 | procedure TMainForm.RoutesMenuClick(Sender: TObject); | Main | MainFormMenu | Открытие раздела «Маршруты Гродно» |
| 27 | procedure TMainForm.PicturesGamesMenuClick(Sender: TObject); | Main | MainFormMenu | Открытие игры «Парные картинки с велосипедами» |
| 28 | procedure TMainForm.HelpClick(Sender: TObject); | Main | MainFormMenu | Вызов справки |
| 29 | procedure TMainForm.NextMusicClick(Sender: TObject); | Main | NextMusic | Переключение трека вперёд |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 30 | procedure TMainForm.PrevMusicClick(Sender: TObject); | Main | PrevMusic | Переключение трека назад |
| 31 | procedure TMainForm.RouteButtonClick(Sender: TObject); | Main | RouteButton | Открытие раздела «Маршруты Гродно» |
| 32 | procedure TMainForm.ButtonSelectionClick(Sender: TObject); | Main | ButtonSelection | Открытие раздела «Виды велосипедов» |
| 33 | procedure TMainForm.TestsButtonClick(Sender: TObject); | Main | TestsButton | Открытие раздела «Тесты» |
| 34 | procedure TMainForm.ExitButtonClick(Sender: TObject); | Main | ExitButton | Выход из программы |
| 35 | procedure TMainForm.TitleColorTimer(Sender: TObject); | Main | TitleColor | Переливание цветом текста «Планета велосипедов» |
| 36 | procedure TMainForm.TrickButtonClick(Sender: TObject); | Main | TrickButton | Открытие раздела «Трюки на велосипеде» |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 37 | procedure TMainForm.VolumeClick(Sender: TObject); | Main | VolumeClick | Включение и выключение звука |
| 38 | procedure TDevice.FormActivate(Sender: TObject); | BicycleDevice | Device | Загрузка HTML-файла с разделом «Устройство велосипеда» |
| 39 | procedure TDevice.PageBeforeScriptExecute(ASender: TObject; const pDispWindow: IDispatch); | BicycleDevice | Device | Закрытие HTML-файла с разделом «Устройство велосипеда» и переход на главную форму |
| 40 | procedure TTypes.FormActivate(Sender: TObject); | BicyclesTypes | Types | Загрузка HTML-файла с разделом «Виды велосипедов» |
| 41 | procedure TTypes.Page2BeforeScriptExecute(ASender: TObject; const pDispWindow: IDispatch); | BicyclesTypes | Types | Закрытие HTML-файла с разделом «Виды велосипедов» и переход на главную форму |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 42 | procedure TTricks.FormActivate(Sender: TObject); | BicyclesTricks | Tricks | Загрузка HTML-файла с разделом «Трюки на велосипеде» |
| 43 | procedure TTricks.Page3BeforeScriptExecute(ASender: TObject; const pDispWindow: IDispatch); | BicyclesTrick | Tricks | Закрытие HTML-файла с разделом «Трюки на велосипеде» и переход на главную форму |
| 44 | procedure TRoutes.FormActivate(Sender: TObject); | HrodnaRoutes | Routes | Загрузка HTML-файла с разделом «Маршруты Гродно» |
| 45 | procedure TRoutes.Page4BeforeScriptExecute(ASender: TObject; const pDispWindow: IDispatch); | HrodnaRoutes | Routes | Закрытие HTML-файла с разделом «Маршруты Гродно» и переход на главную форму |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 46 | procedure TTest.FormClose(Sender: TObject; var Action: TCloseAction); | Tests | Test | Занесение полученных очков в БД и установка визуальных элементов на базовые места. Отображение главного меню |
| 47 | procedure TTest.FormCreate(Sender: TObject); | Tests | Test | Загрузка картинок для кнопок и размещение элементов управления тестом на форме |
| 48 | procedure TTest.HelpClick(Sender: TObject); | Tests | MainMenuTest | Вызов справки |
| 49 | procedure TTest.AboutProgrClick(Sender: TObject); | Tests | MainMenuTest | Вызов окна «О программе» |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 50 | procedure TTest.NextButtonClick(Sender: TObject); | Tests | NextButton | Переход к следующему вопроса теса. Проверка правильности ответа. Итоговое начисление очков. |
| 51 | procedure TTest.OnMainFormClick(Sender: TObject); | Tests | OnMainForm | Выход на главную форму |
| 52 | procedure TTest.TestsVariantClick(Sender: TObject); | Tests | TestsVariantClick | Выбор теста и изменение расположения элементов управления в зависимости от выбранного теста. |
| 53 | procedure TTest.TimerTimer(Sender: TObject); | Tests | Timer | Ограничение теста по времени |
| 54 | procedure Kletka(col,row: integer); | Game | — | Рисование клетки поля игры «Парные картинки с велосипедами» |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 55 | procedure ShowPole; | Game | — | Отрисовка поля игры |
| 56 | Procedure NewGame; | Game | — | Начало новой игры |
| 57 | procedure TPicturesGame.FormClose(Sender: TObject; var Action: TCloseAction); | Game | PicturesGame | Обновление очков и установка элементов управления в начальное положение при закрытии формы |
| 58 | procedure TPicturesGame.FormCreate(Sender: TObject); | Game | PicturesGame | Загрузка изображений для кнопок, установка расположения элементов управления на форме |
| 59 | procedure TPicturesGame.LevelOneButtonClick(Sender: TObject); | Game | LevelOneButton | Запуск первого уровня игры (поле 2х2) |
| 60 | procedure TPicturesGame.LevelTwoButtonClick(Sender: TObject); | Game | LevelTwoButton | Запуск второго уровня игры (поле 4х4) |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 61 | procedure TPicturesGame.HelpClick(Sender: TObject); | Game | MainMenuGame | Вызов справки |
| 62 | procedure TPicturesGame.AboutProgrClick(Sender: TObject); | Game | MainMenuGame | Вызов окна «О программе» |
| 63 | procedure TPicturesGame.LevelThreeButtonClick(Sender: TObject); | Game | LevelThreeButton | Запуск третьего уровня игры (поле 6х6) |
| 64 | procedure TPicturesGame.LevelFourButtonClick(Sender: TObject); | Game | LevelFourButton | Запуск четвёртого уровня игры (поле 8х8) |
| 65 | procedure TPicturesGame.BackToMainMenuClick(Sender: TObject); | Game | BackToMainMenu | Возврат на главное меню |
| 66 | procedure TPicturesGame.FieldMouseDown(Sender: TObject; Button: TMouseButton; | Game | Field | Обработка нажатия на клетку поля |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя процедуры (функции) | В каком модуле находится | За каким компонентом закреплена | Назначение |
| 67 | procedure TPicturesGame.FieldPaint(Sender: TObject); | Game | Field | Отрисовка игрового поля |
| 68 | procedure TPicturesGame.UpdateTimer(Sender: TObject); | Game | Update | Стирание одинаковых картинок с поля |
| 69 | procedure TPicturesGame.TimerTimer(Sender: TObject); | Game | Timer | Ограничение по времени для уровней |
| 70 | procedure TPicturesGame.TitleColorTimer(Sender: TObject); | Game | TitleColor | Переливание цветом заголовка «Парные картинки с велосипедами» |
| 71 | procedure TAProgramm.FormCreate(Sender: TObject); | AboutProgram | AProgram | Вывод информации о программе |

**3.1.3 Описание использованных компонентов**

Описание использованных для разработки приложения компонентов приводится в таблице 2.

Таблица 2 – Использованные компоненты

|  |  |  |  |
| --- | --- | --- | --- |
| № | Компонент | На какой форме расположен | Назначение |
| 1 | TImage | LoadScreen, Reg, MainForm, Test, PicturesGame, AProgramm | Используется для кнопок в пиксельном стиле, загрузочных и фоновых экранов |
| 2 | TTimer | LoadScreen, MainForm, Test, PicturesGame | Используется для управления музыкой, продолжительностью загрузочного экрана, ограничения по времени игр |
| 3 | TProgressBar | LoadScreen | Используется для отображения полосы загрузки |
| 4 | TLabel | LoadScreen, Reg, MainForm, Test, PicturesGame, AProgramm | Используется для вывода стилизованной текстовой информации |
| 5 | TEdit | Reg | Используется для заполнения полей авторизации и регистрации |

Продолжение таблицы 2

|  |  |  |  |
| --- | --- | --- | --- |
| № | Компонент | На какой форме расположен | Назначение |
| 6 | TMainMenu | Reg, MainForm, Test, PictureGame | Используется для отображения меню для быстрого доступа к главным разделам страницы |
| 7 | TADOConnection | Reg | Используется для подключения базы данных пользователей |
| 8 | TADOTable | Reg | Используется для связи с одной таблицей из БД, подключённой с помощью компонента TADOConnection |
| 9 | TMediaPlayer | MainForm | Используется для проигрывания фоновой музыки |
| 10 | TListBox | MainForm | Используется для создания музыкального плейлиста |
| 11 | TWebBrowser | Device, Types, Tricks, Routes | Используется для открытия HTML-страниц |
| 12 | TRadioGroup | Test | Используется для организации тестов |
| 13 | TPaintBox | Game | Используется для отрисовки поля в игре «Парные картинки с велосипедами» |

**3.2 Спецификация программы**

Точное название проекта и его состав приводится в Таблице 3.

Таблица 3 – Спецификация программы

|  |  |  |
| --- | --- | --- |
| № | Имя файла | Назначение |
| 1 | Velo\_Land.exe | Исполняемый файл проекта |
| 2 | Velo\_Land.dproj | Содержит настройки проекта и опции компилятора |
| 3 | Velo\_Land.dpr | Файл проекта, который связывает все файлы, из которых состоит приложение |
| 4 | AboutProgram.pas | Файл программного модуля для формы AProgram |
| 5 | BicycleDevice.pas | Файл программного модуля для формы Device |
| 6 | BicyclesTricks.pas | Файл программного модуля для формы Tricks |
| 7 | BicyclesTypes.pas | Файл программного модуля для формы Types |
| 8 | Game.pas | Файл программного модуля для формы PicturesGames |
| 9 | HrodnaRoutes.pas | Файл программного модуля для формы Routes |
| 10 | LoadScr.pas | Файл программного модуля для формы LoadScreen |
| 11 | Main.pas | Файл программного модуля для формы MainForm |
| 12 | RegistrationForm.pas | Файл программного модуля для формы Reg |

Продолжение таблицы 3

|  |  |  |
| --- | --- | --- |
| № | Имя файла | Назначение |
| 13 | Tests.pas | Файл программного модуля для формы Test |
| 14 | Setup.exe | Установочный файл |
| 15 | VeloLandHelp.chm | Файл справки, содержащий помощь по работе с программой |
| 16 | AboutProgram.dfm | Форма, содержащая информацию о программе |
| 17 | BicycleDevice.dfm | Форма, содержащая информацию об устройстве велосипеда |
| 18 | BicycleTricks.dfm | Форма, содержащая информацию о трюках на велосипеде |
| 19 | BicycleTypes.dfm | Форма, содержащая информацию о видах велосипедов |
| 20 | Game.dfm | Форма, содержащая игру «Парные картинки с велосипедами» |
| 21 | HrodnaRoutes.dfm | Форма с маршрутами города Гродно и правилами дорожного движения для велосипедистов |
| 22 | LoadScr.dfm | Форма с загрузочным экраном приложения |
| 23 | Main.dfm | Форма, содержащая главное меню приложения. Из неё можно перейти на любую форму программы. |

Продолжение таблицы 3

|  |  |  |
| --- | --- | --- |
| № | Имя файла | Назначение |
| 24 | RegistrtaionForm.dfm | Форма авторизации и регистрации |
| 25 | Tests.dfm | Форма, содержащая тесты |
| 26 | \*.html | Веб-страницы для разделов «Устройство велосипеда», «Виды велосипедов», «Трюки на велосипеде», «Маршруты Гродно» |
| 27 | \*.jpg,\*.png | Файлы изображения для стилизации кнопок. Изображения для теста, игры «Парные картинки с велосипедами» |
| 28 | \*.mp3 | Файлы для проигрывания фоновой музыки |
| 29 | minecraft.ttf | Файл шрифта, который используется в программе |
| 30 | VeloLandDatabase.mdb | База данных пользователей приложения |

**4. Тестирование**

При разработке данной программы многие возникающие ошибки и недоработки были исправлены на этапе реализации проекта. После завершения этапа написания программы было проведено тщательное функциональное тестирование. Функциональное тестирование должно гарантировать работу всех элементов программы в автономном режиме.

Отчёт о результатах тестирования представлен в таблице 4.

Таблица 4 – Отчёт о результатах тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № теста | Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| 1 | Проверка кнопки «Авторизация» и «Регистрация" на форме регистрации и авторизации | Появление полей авторизации или регистрации | Появление полей авторизации или регистрации | Выполнено |
| 2 | Проверка кнопки «Зарегистрироваться» на форме регистрации и авторизации при вводе данных с ошибками | Появление уведомления об ошибке | Появление уведомления об ошибке | Выполнено |
| 3 | Проверка кнопки «Далее» на форме регистрации и авторизации при неверно введённых данных | Появление уведомления о том, что логин или пароль введён неверно | Появление уведомления о том, что логин или пароль введён неверно | Выполнено |

Продолжение таблицы 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № теста | Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| 4 | Открытие пункта меню «Справка» из любой формы | Вызов справки | Вызов справки | Выполнено |
| 5 | Открытие пункта меню «О программе» из любой формы | Открытие окна «О программе» | Открытие окна «О программе» | Выполнено |
| 6 | Кнопка «Зарегистрироваться!» на форме авторизации и регистрации при правильно введённых данных | Появление уведомления о том, что пользователь успешно зарегистрирован | Появление уведомления о том, что пользователь успешно зарегистрирован | Выполнено |
| 7 | Кнопка «Далее» на форме авторизации и регистрации при правильно введённых данных | Переход пользователя на главное меню программы | Переход пользователя на главное меню программы | Выполнено |
| 8 | Проверка отображения накопленных очков пользователя на главной форме | Отображение корректного числа очков авторизованного пользователя | Отображение корректного числа очков авторизованного пользователя | Выполнено |

Продолжение таблицы 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № теста | Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| 9 | Проверка работоспособности пунктов меню «Общая информация» | При нажатии на пункт меню открывается соответствующий раздел и информацией | При нажатии на пункт меню открывается соответствующий раздел и информацией | Выполнено |
| 10 | Проверка кнопок на главном меню с основной информацией («Устройство велосипеда», «Виды велосипедов», «Трюки на велосипеде», «Маршруты Гродно») | Открытие HTML-страницы с выбранным разделом | Открытие HTML-страницы с выбранным разделом | Выполнено |
| 11 | Проверка работоспособности переключения музыки вперёд и назад | Переключение музыки на следующую или предыдущую из плейлиста | Переключение музыки на следующую или предыдущую из плейлиста | Выполнено |

Продолжение таблицы 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № теста | Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| 12 | Проверка кнопки включения/выключения звука | При нажатии происходит включение/выключение звука | При нажатии происходит включение/выключение звука | Выполнено |
| 13 | Проверка кнопки «Тесты» на главном меню и верхнем меню | Открытие раздела «Тесты» | Открытие раздела «Тесты» | Выполнено |
| 14 | Проверка кнопки «Игра «Парные картинки с велосипедами» на главном меню и верхнем меню при достаточном количестве очков | Открытие окна с игрой «Парные картинки с велосипедами» | Открытие окна с игрой «Парные картинки с велосипедами» | Выполнено |
| 15 | Проверка кнопки «Выход» на главном меню и с помощью крестика | Выход из программы | Выход из программы | Выполнено |

Продолжение таблицы 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № теста | Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| 16 | Проверка радиокнопок с темами тестов в окне с тестами | Запуск теста с выбранной темой | Запуск теста с выбранной | Выполнено |
| 17 | Проверка таймера времени на формах тестов и игры «Парные картинки с велосипедами» | По истечении времени программа уведомит пользователя об окончании времени | По истечении времени программа уведомит пользователя об окончании времени | Выполнено |
| 18 | Проверка начисления очков за тесты и игру «Парные картинки с велосипедами» | После прохождения теста или уровня игры пользователю будут отображены заработанные им очки с занесением их в БД | После прохождения теста или уровня игры пользователю будут отображены заработанные им очки с занесением их в БД | Выполнено |

Продолжение таблицы 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № теста | Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| 19 | Проверка преждевременного выхода из теста или игры «Парные картинки с велосипедами» на главное меню | При преждевременном выходе из теста очки пользователю не начисляются | При преждевременном выходе из теста очки пользователю не начисляются | Выполнено |
| 20 | Проверка кнопки «Назад на главное меню на всех формах (кроме регистрационной и главной) | Выход на главное меню | Выход на главное меню | Выполнено |
| 21 | Проверка кнопок выбора уровней в игре «Парные картинки с велосипедами» | Открывается выбранный уровень | Открывается выбранный уровень | Выполнено |

Продолжение таблицы 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № теста | Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| 22 | Проверка кнопок выбора уровней в игре «Парные картинки с велосипедами» если предыдущий уровень не пройден | Уведомление от программы о том, что нужно пройти предыдущий уровень | Уведомление от программы о том, что нужно пройти предыдущий уровень | Выполнено |
| 23 | Переходы на главную форму с обновлением очков | При возврате на главную форму происходит обновление очков пользователя | При возврате на главную форму происходит обновление очков пользователя | Выполнено |
| 24 | Обнуление аккаунта | Полный сброс очков пользователя | Полный сброс очков пользователя | Выполнено |
| 25 | Выход из аккаунта | Пользователь может сменить аккаунт и войти под другим логином | Пользователь может сменить аккаунт и войти под другим логином | Выполнено |

Продолжение таблицы 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № теста | Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| 26 | Проверка сохранения всего прогресса в БД | После повторного входа в приложение все накопленные очки пользователя сохраняются | После повторного входа в приложение все накопленные очки пользователя сохраняются | Выполнено |

Вывод: все тесты были успешно выполнены, все обнаруженные ошибки были исправлены. Весь функционал, что изначально был составлен, реализован.

**5 Руководство пользователя**

**5.1 Общие сведения о программном продукте**

Разработанная программа является информационно-познавательным приложением «Планета велосипедов», предназначенная для ознакомления пользователя с важными аспектами использования велосипеда и может быть использована любым родом пользователей.

Возможностями программы являются обучение пользователя устройству велосипеда, изучение трюков, маршрутов города Гродно и правил дорожного движения для велосипедистов, закрепление полученных знаний проходит с помощью тестов на разные темы, развлекательной частью приложения является игра «Парные картинки с велосипедами».

Периодичность использования у приложения любая. Пользователь может пользоваться приложением в любое время.

Областью применения приложения является тема велосипедов.

Быстродействие любой программы зависит от характеристик выбранного персонального компьютера: рабочей частоты процессора, объема оперативной памяти и т.д. Несмотря на все реализованные задачи и идеи, она легко запускается на любых машинах.

Тестирование проводилось на разных классах ЭВМ, но даже на системе следующего типа работало комфортно:

- процессор Pentium 1000 Mhz;

- объем ОЗУ 512 Mb;

- видеоподсистема: 1024\*768 точек с глубиной цвета 16 Bit.

- операционная система WINDOWS XP sp2.

**5.2 Инсталляция**

Для того чтобы установить программу необходимо запустить файл Setup.exe. Появится окно установки приложения «Планета велосипедов» (Рисунок 7).

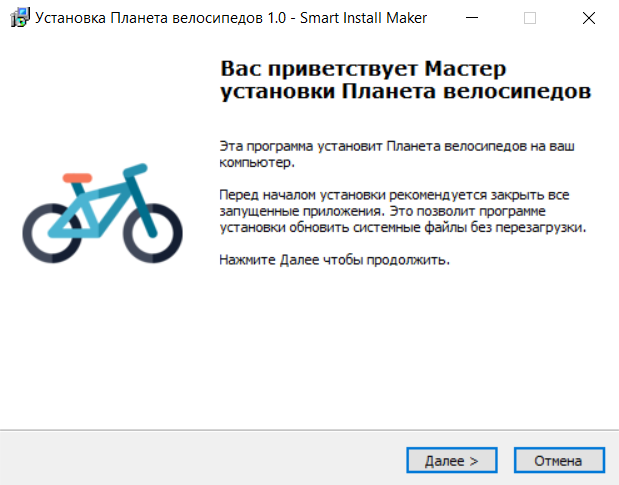


Рисунок 7 – Окно установки приложения «Планета велосипедов»

После нажатия на кнопку Далее. Появится окно, где нужно выбрать путь к папке, куда установится приложение (Рисунок 8). По умолчанию папка C:\Program Files (x86)\Планета велосипедов.

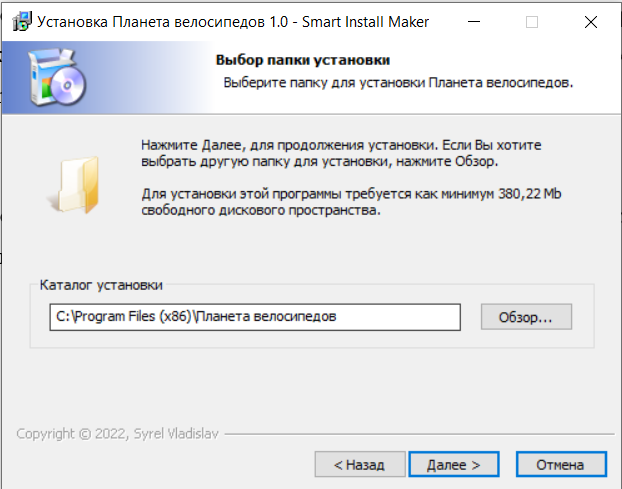


Рисунок 8 – Выбор пути установки приложения

После того, как вы указали путь, нажмите Далее. Появится окно, где нужно выбрать дополнительные опции при установке (Рисунок 9).



Рисунок 9 – Выбор дополнительных опций при установке приложения

После нажатия на кнопку Далее проверьте всю информацию и, если всё указано верно, нажмите Установить (Рисунок 10).

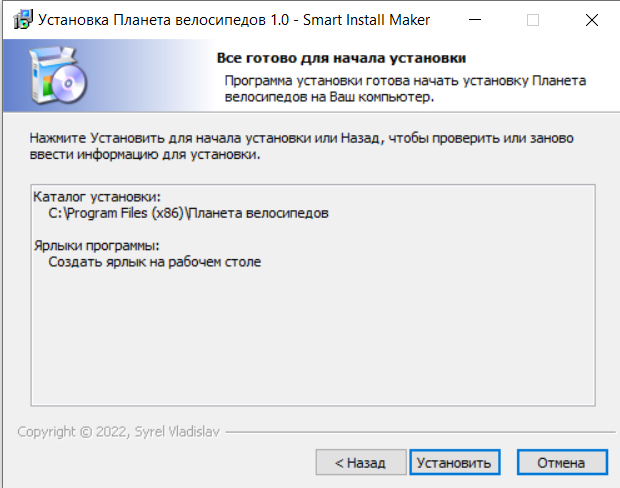


Рисунок 10 – Проверка указанных данных при установке приложения

Далее начнётся сам процесс установки (Рисунок 11).

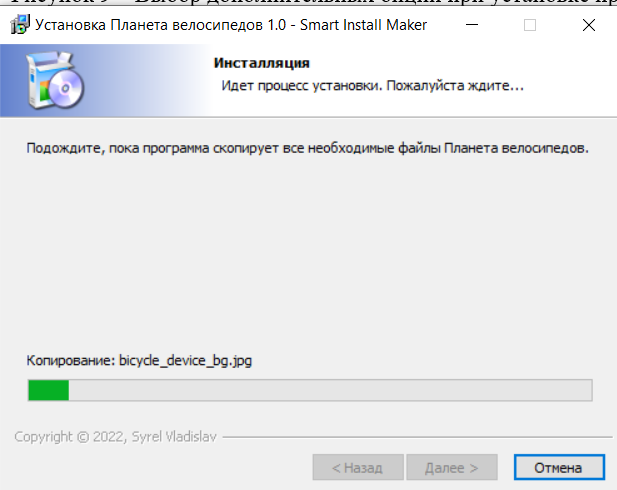


Рисунок 11 – Процесс установки приложения

После завершения процесса установки появится окно с сообщением о том, что приложение успешно установлено. Пользователю будет предложено посетить социальную сеть разработчика и сразу запустить приложение (Рисунок 12).

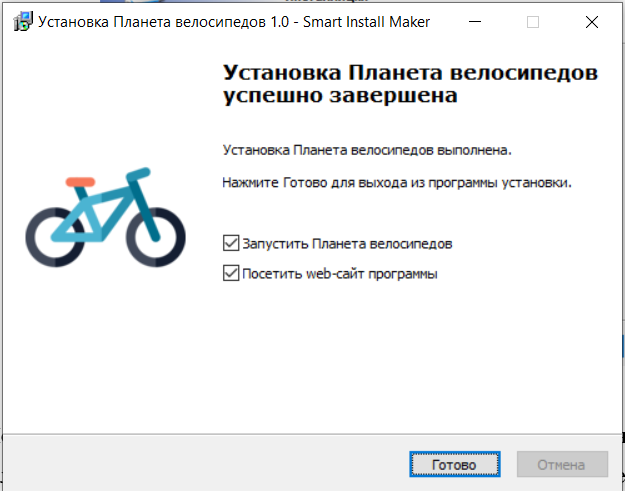


Рисунок 12 – Завершение установки

На этом инсталляция успешно завершена.

**5.3 Выполнение программы**

**5.3.1 Запуск программы**

Данную программу можно запустить следующими способами:

Первым из них является запуск с ярлыка на рабочем столе. Необходимо дважды кликнуть кнопкой мыши на ярлык под названием «Планета велосипедов» (Рисунок 13).

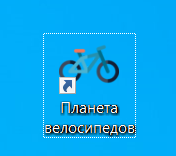


Рисунок 13 – Ярлык программы «Планета велосипедов»

Вторым способом является запуск через меню Пуск. Необходимо нажать на меню Пуск и ввести в поисковик «Планета велосипедов» (Рисунок 14).

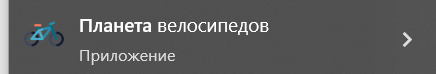


Рисунок 14 – Поиск приложения в меню Пуск

Третьим способ является запуск приложения из каталога, в который оно было установлено (По умолчанию C:\Program Files (x86)\Планета велосипедов) (Рисунок 15).



Рисунок 15 – Приложение в директории установки

**5.3.2 Инструкция по работе с программой**

После запуска приложения на экране появится заставка (Рисунок 16).

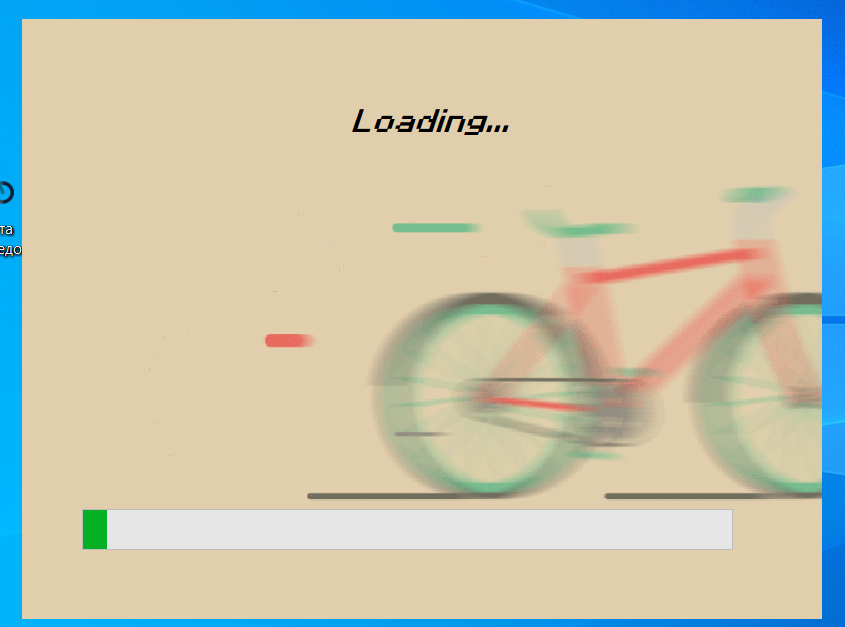


Рисунок 16 – Заставка приложения

После заставки открывается форма регистрации и авторизации (Рисунок 17).

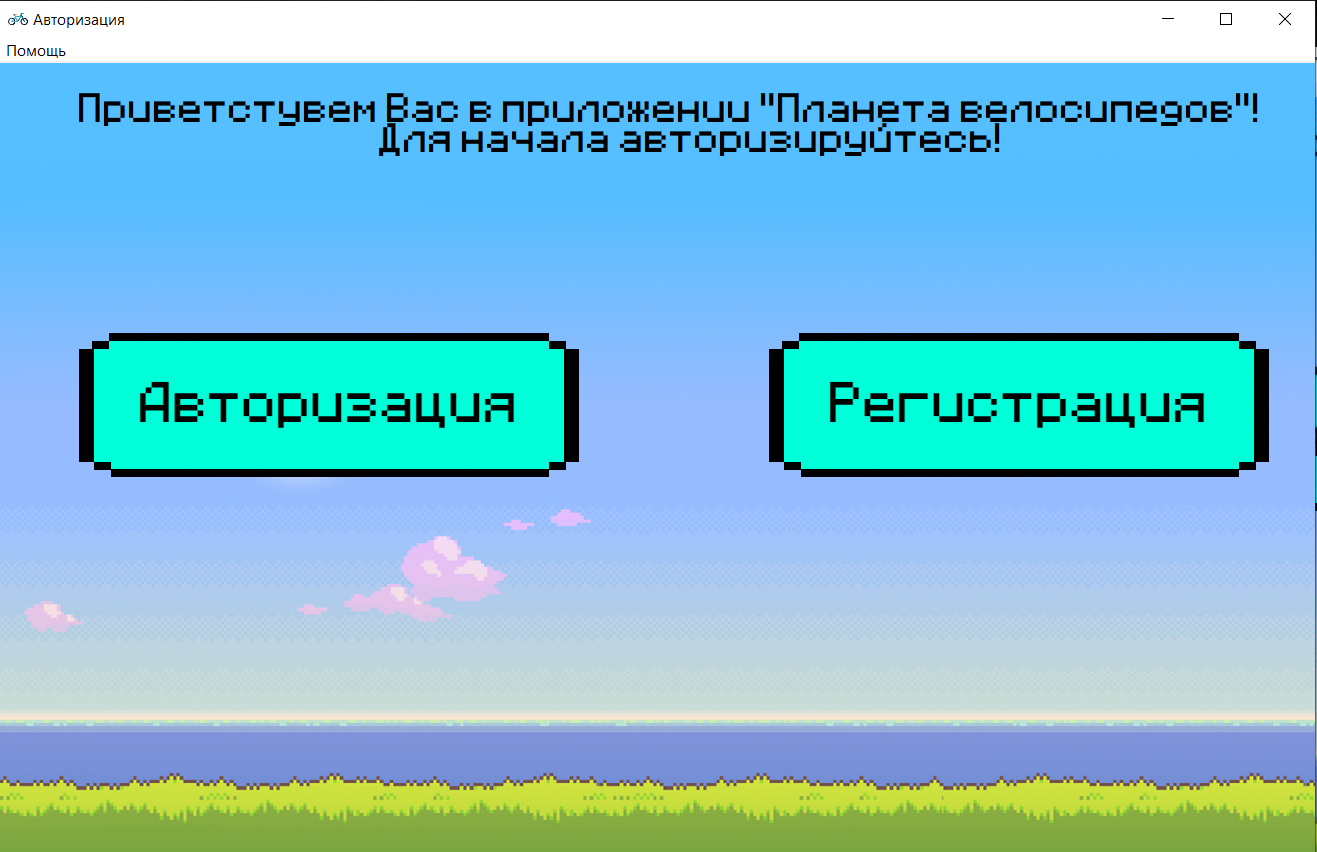


Рисунок 17 – Форма регистрации и авторизации

Для начала работы пользователь должен зарегистрироваться. При нажатии кнопки «Регистрация» пользователю отображаются поля регистрации (Рисунок 18).



Рисунок 18 – Поля регистрации

Если пользователь допускает ошибки при заполнении полей, то программа будет уведомлять его об этом (Рисунок 19).

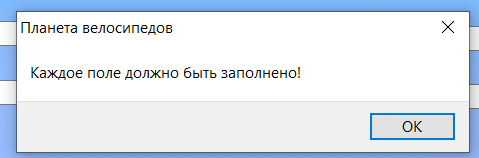


Рисунок 19 – Пример уведомления о некорректном заполнении полей

После успешной регистрации пользователь получит соответствующее уведомление и ему будет предложено авторизоваться (Рисунок 20).

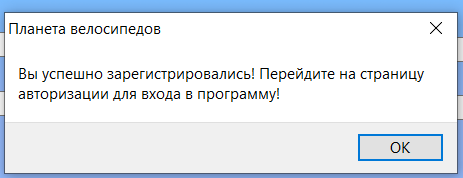


Рисунок 20 – Уведомление об успешной регистрации

Далее пользователю нужно авторизоваться, для этого нужно нажать на кнопку «Авторизация». После этого появятся поля авторизации (Рисунок 21).



Рисунок 21 – Поля авторизации

При вводе неверного логина или пароля приложение уведомит пользователя об этом (Рисунок 22).

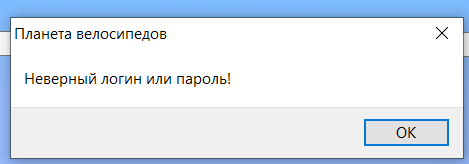


Рисунок 22 – Уведомление о неверном логине или пароле

При успешной авторизации откроется главное меню программы (Рисунок 23).



Рисунок 23 – Главное меню программы

Для доступа к разделам общая информации (разделы «Устройство велосипеда», «Виды велосипедов», «Трюки на велосипеде», «Маршруты Гродно») представлены кнопки, которые изображены на Рисунке 24.

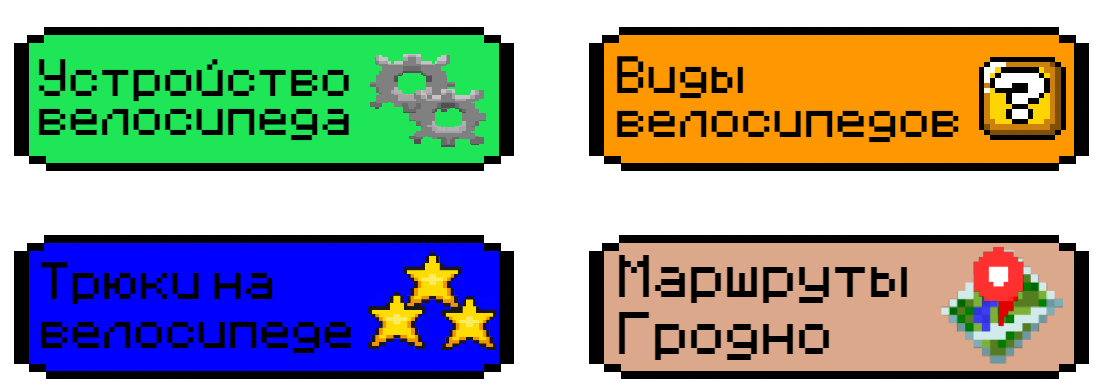


Рисунок 24 – Кнопки для доступа к разделам с общей информацией

Также доступ к этим разделам можно получить с помощью верхнего меню (Рисунок 25).

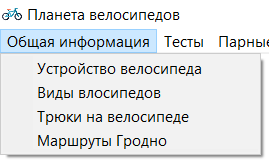


Рисунок 25 – Доступ к разделам с общей информацией с помощью верхнего меню

На главном меню предусмотрены кнопки для управления музыкой. Это кнопка включения/выключения музыки (Рисунок 26) и кнопки переключения музыки вперёд и назад (Рисунок 27).



Рисунок 26 – Кнопка включения и выключения звука

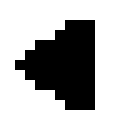
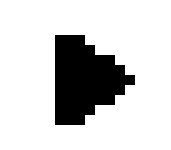
 

Рисунок 27 – Кнопки переключения музыки

Нажав на кнопку «Тесты» пользователь попадёт на окно с тестами (Рисунок 28).



Рисунок 28 – Главное окно раздела «Тесты

Пользователь может выбрать тему теста на верхней панели. При нажатии на одну из них запустится тест с соответствующей темой (Рисунок 29).

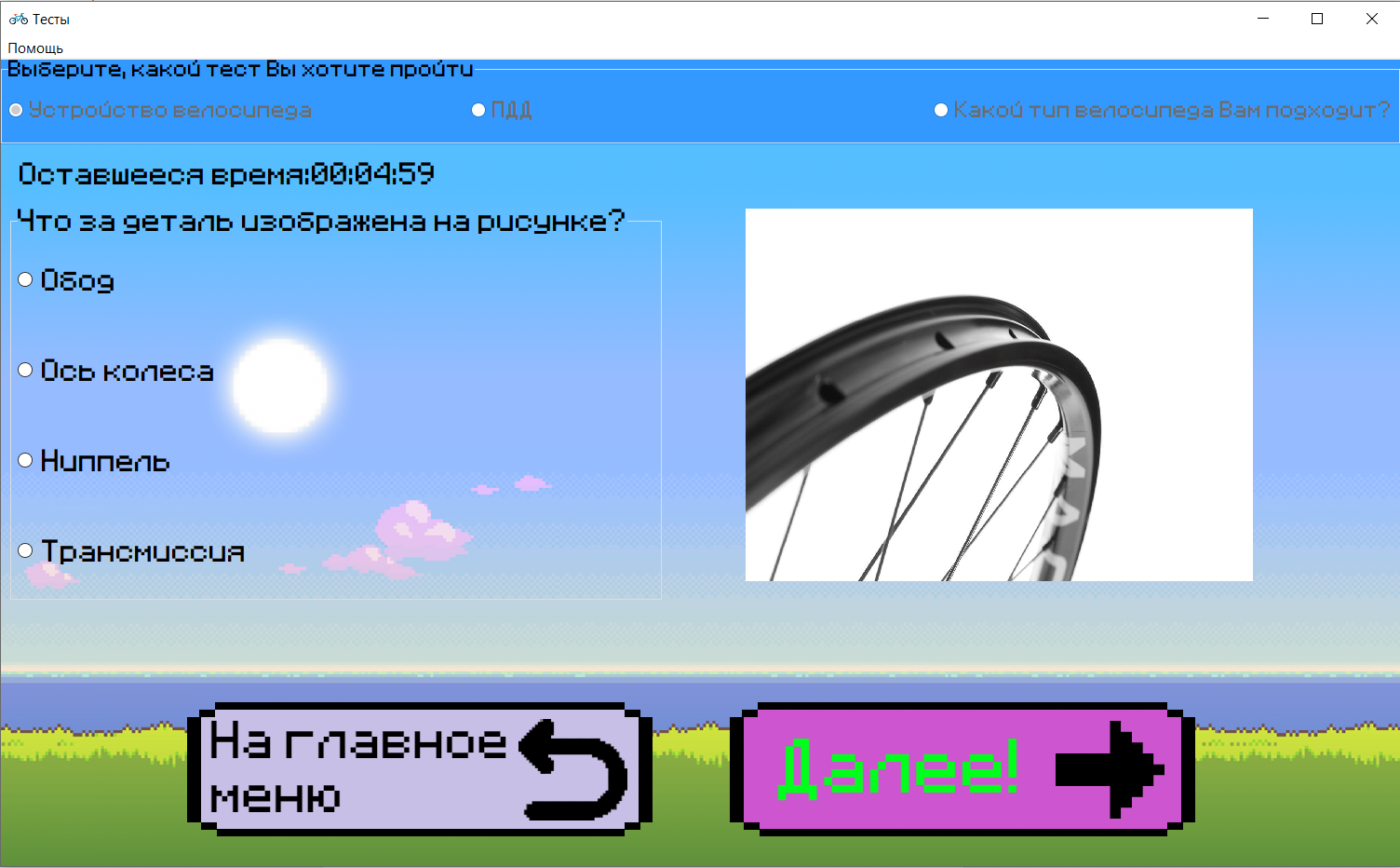


Рисунок 29 – Тест в процессе выполнения

Доступ к тестам является последовательным, то есть пока пользователь не наберёт определённое количество очков на предыдущих тестах, к другому доступ будет закрыт (Рисунок 30).

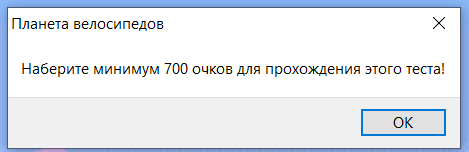


Рисунок 30 – Уведомление о недостаточном количестве очков

Все тесты ограничены по времени. Если время истекло программа оповестит пользователя об этом и начислит то количество очков, которое он успел получить. Если пользователь успел ответить на все вопросы до конца таймера, то он получите количество очков в зависимости от правильных ответов (Рисунок 31).

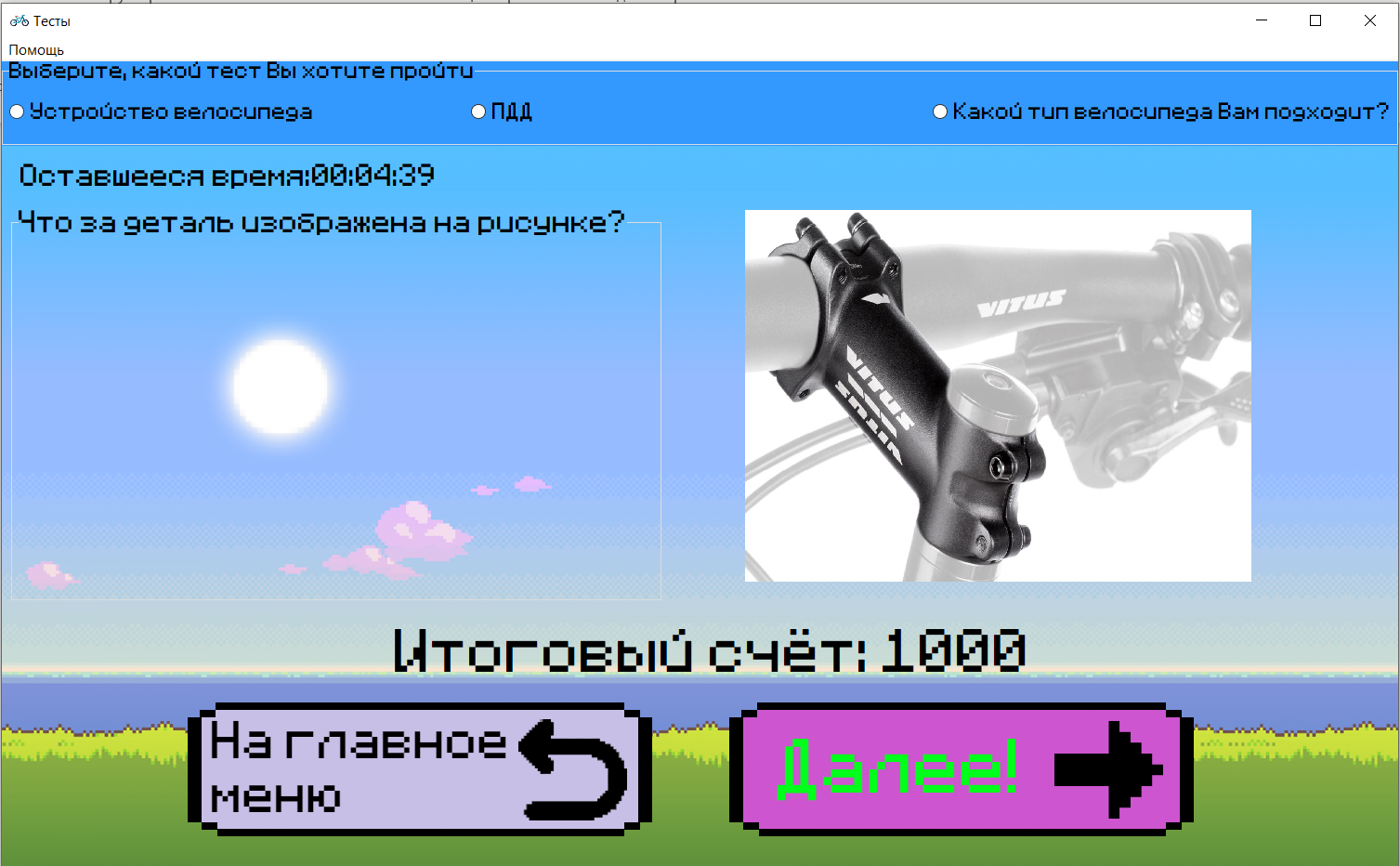


Рисунок 31 – Начисление очков

При нажатии на кнопку «Игра «Парные картинки с велосипедами» пользователь, если у него достаточное количество очков, попадает на главную страницу игры (Рисунок 32).



Рисунок 32 – Главная страница игры «Парные картинки с велосипедами»

**Правила игры:** формируется поле (2х2, 4х4, 6х6, 8х8) (Рисунок 33) в зависимости от уровня. Квадраты заполняются картинками, и одна картинка встречается на поле 2 раза. Задача пользователя: найти все пары картинок, чтобы на поле не осталось ни одной. Если пользователь открывает две картинки, и они не одинаковые, то они снова переворачиваются и становятся невидимыми. В зависимости от уровня сложности даётся разное время на прохождение, если пользователь не успел найти все пары или вышел посреди уровня, то программа не начисляет ему баллы и не засчитывает прохождение уровня (Рисунок 34)**.** Пример успешного прохождения уровня представлен на рисунке 35.

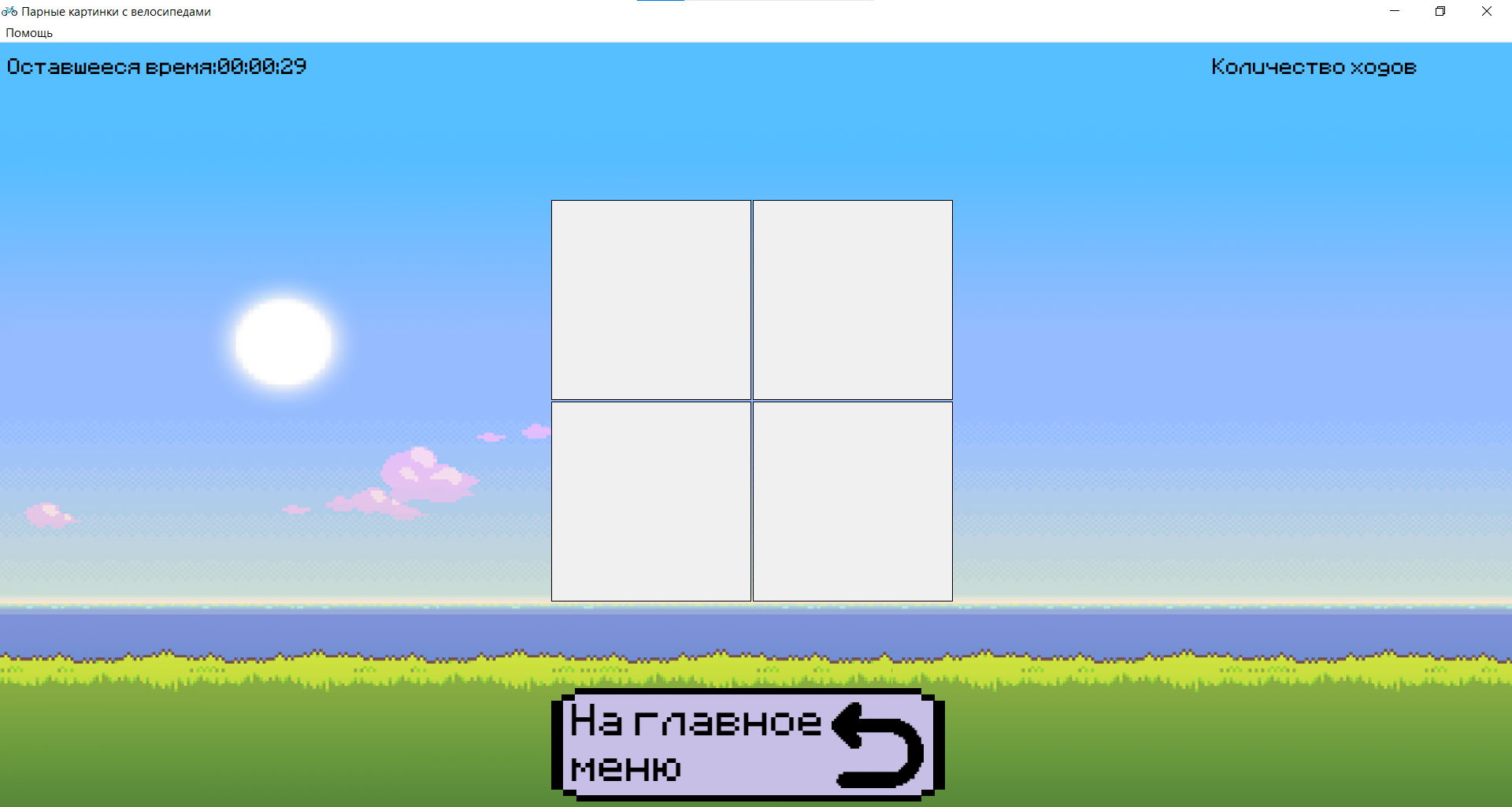


Рисунок 33 – Пример игрового поля

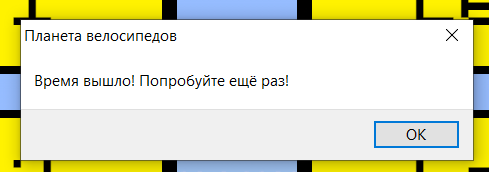


Рисунок 34 – Уведомление об окончании времени

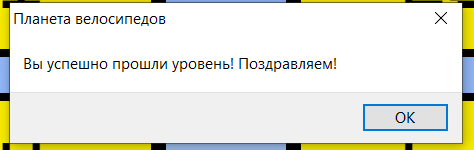


Рисунок 35 – Уведомление об успешном прохождении уровня

В данной игре также присутствует последовательное продвижение по уровням. Поэтому если предыдущий уровень не пройден, то доступ к следующему уровню будет закрыт.

Приложение имеет также функцию обнуления прогресса, т.е. у пользователя снимаются все очки. Обнулить аккаунт можно нажатием соответствующей кнопки на верхнем меню (Рисунок 36). Перед обнулением программа попросит подтвердить, что вы уверены в обнулении аккаунта.



Рисунок 36 – Кнопка обнуления аккаунта

Также можно выйти из своего аккаунта. Для этого нужно нажать соответствующую кнопку в верхнем меню главного меню (Рисунок 37). Перед этим действием программа попросит подтверждения пользователя.



Рисунок 37 – Кнопка выхода из аккаунта

**5.3.3 Завершение работы программы**

Для того, чтобы завершить работу программы пользователь должен нажать на кнопку выхода на главном меню (Рисунок 38).



Рисунок 38 – Кнопка выхода из программы на главном меню

Либо можно нажать на крестик в панели управления окном на главном меню (Рисунок 39). Перед выходом программа попросит подтверждение действия.



Рисунок 39 – Выход из программы с помощью крестика

**5.4. Использование системы справочной информации**

Справочную систему в приложении можно вызвать либо нажатием в верхней панели меню пунктов «Помощь» - «Справка» (Рисунок 40), либо нажать горячую клавишу F1. Окно справки представлено на рисунке 41.

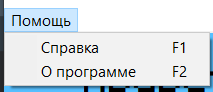


Рисунок 40 – Вызов справки их верхнего меню на главном меню

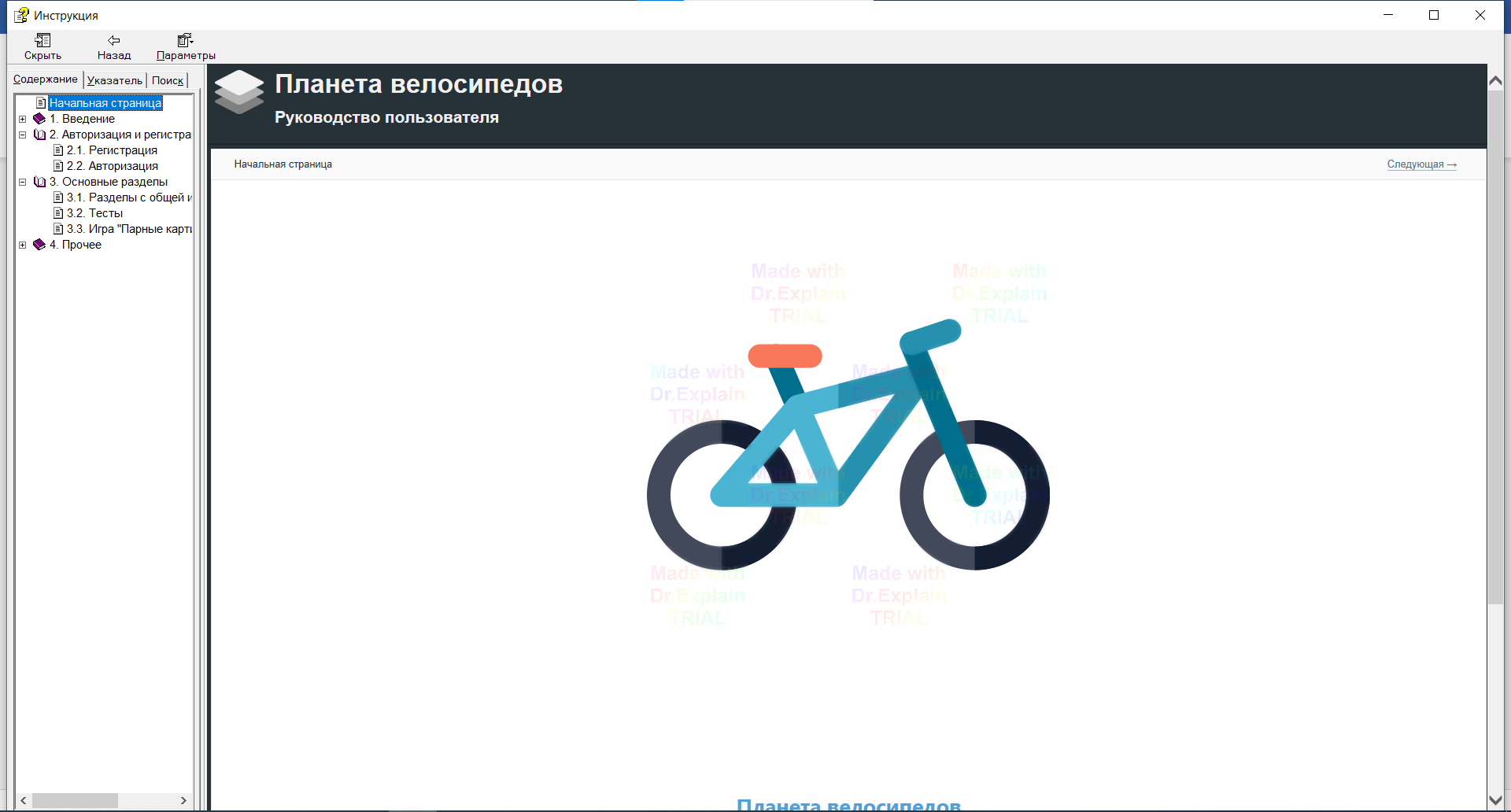


Рисунок 41 – Окно справки

**Заключение**

Задачей данного курсового проекта являлась разработка информационно-познавательного приложения «Планета велосипедов».

Поставленная задача была выполнена в среде Delphi. В ходе реализации поставленной задачи был укреплён пройденный курс программирования в данной Delphi, а также получено много дополнительной информации о её возможностях.

Следует также учесть, что в поставленной задаче был реализован простой интерфейс, который позволяет использовать приложение пользователю, не обладающему дополнительными знаниями ЭВМ.

После тщательного тестирования приложения были выявлены некоторые недоработки, которые были исправлены на стадии проектирования, и полностью исключены на стадии тестирования программы.

Программу можно модифицировать, добавив рейтинговую таблицу и переписав работу с базой данных на язык SQL-запросов.

В целом при реализации программы, были выполнены все условия, перечисленные в предыдущих разделах пояснительной записки. Таким образом, можно сказать, что программа была реализована успешна.

**Список использованных источников**

1. ВелоГродно [Электронный ресурс]. – Электронные данные. – Режим доступа: https://velogrodno.by/tourism/tri-velomarshruta-v-okrestnostyah-grodno-ot-moguchih-dubrav-k-oreshnikovoj-sone-i-muzeyu-drevnostej.html

2. Belarus.travel [Электронный ресурс]. – Электронные данные. – Режим доступа: https://ru.belarus.travel/news/velodorozhki-grodno-gde-pokatatsia-v-gorode

3. Сириус [Электронный ресурс]. – Электронные данные. – Режим доступа: http://velosirius.ru/velodetali

4. VeloFans [Электронный ресурс]. – Электронные данные. – Режим доступа: https://velofans.ru/sovety/kakie-byvayut-tryuki-velosipedah-kak-sdelat

5. VELOMOZG [Электронный ресурс]. – Электронные данные. – Режим доступа: https://velomozg.com/tipy-i-vidy-velosipedov-polnyj-spisok

6. YouTube [Видео ресурс]. – Электронные данные. – Режим доступа: https://www.youtube.com/

7. Уроки Delphi. [Видео ресурс]. – Режим доступа: https://www.youtube.com/watch?v=y6WUUXWMk1E. – Дата доступа 18.06.2021.

8. Иллюстрированный самоучитель по Delphi [Электронный ресурс]. – Режим доступа: samoychiteli.ru/document27136.html. – Дата доступа 22.06.2021.

9. Электронная книга по Delphi [Электронный ресурс]. – Режим доступа: https://lp.embarcadero.com/DelphiBookInRussia. – Дата доступа 24.06.2021

**Приложение А**

**Листинг программы**

unit LoadScr;// модуль загрузочного экрана

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.Imaging.GIFImg,

Vcl.StdCtrls, Vcl.ComCtrls;

type

TLoadScreen = class(TForm)

LoadTimer: TTimer;

LoadImage: TImage;

Text: TLabel;

ProgressBar1: TProgressBar;

ProgressBarTimer: TTimer;

procedure LoadTimerTimer(Sender: TObject);

procedure FormActivate(Sender: TObject);

procedure ProgressBarTimerTimer(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

LoadScreen: TLoadScreen;

i:integer;

implementation

{$R \*.dfm}

procedure TLoadScreen.FormActivate(Sender: TObject); //задание GIF-анимации

begin

(LoadImage.Picture.Graphic as TGIFImage).Animate := True;//анимировать GIF

(LoadImage.Picture.Graphic as TGIFImage).AnimationSpeed := 120;//скорость анимации

LoadScreen.Height := LoadImage.Height;

LoadScreen.Width := LoadImage.Width;

i := 0;//переменная для полосы загрузки

end;

procedure TLoadScreen.LoadTimerTimer(Sender: TObject);//выключение таймера

begin

LoadTimer.Enabled := False;

end;

procedure TLoadScreen.ProgressBarTimerTimer(Sender: TObject);//полоса загрузки

begin

Inc(i);

ProgressBar1.Position:=ProgressBar1.Position+1;

if i=100 then ProgressBarTimer.Enabled := False;

end;

end.

unit RegistrationForm;//модуль регистрации

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.StdCtrls, Data.DB,

Data.Win.ADODB, Vcl.Mask, Vcl.DBCtrls, ShellAPI, Vcl.Menus;

type

TReg = class(TForm)

RegBg: TImage;

Title: TLabel;

NameLabel: TLabel;

SurnameLabel: TLabel;

VeloLandDatabase: TADOConnection;

RegButton: TImage;

Name: TEdit;

Surname: TEdit;

VeloLandUsers: TADOTable;

AuthImg: TImage;

RegImg: TImage;

Title2: TLabel;

Pass: TEdit;

ConfrimPass: TEdit;

PassLabel: TLabel;

ConfrimPassLabel: TLabel;

LoginLabel: TLabel;

Login: TEdit;

AuthButton: TImage;

MainMenuReg: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

procedure FormCreate(Sender: TObject);

procedure RegButtonClick(Sender: TObject);

procedure RegImgClick(Sender: TObject);

procedure AuthImgClick(Sender: TObject);

procedure AuthButtonClick(Sender: TObject);

procedure FormDestroy(Sender: TObject);

procedure N2Click(Sender: TObject);

procedure N3Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Reg: TReg;

implementation

{$R \*.dfm}

uses Main, AboutProgram;

procedure TReg.AuthButtonClick(Sender: TObject); //авторизация

begin

VeloLandUsers.Active := True;

if VeloLandUsers.Locate('UserLogin', Login.Text, []) then //проверка существования логина

if VeloLandUsers.FieldByName('UserPassword').AsString = Pass.Text then //проверка пароля к логину

MainForm.ShowModal

else

ShowMessage('Неверный логин или пароль!')

else

ShowMessage('Неверный логин или пароль!');

end;

procedure TReg.AuthImgClick(Sender: TObject); //отображение полей авторизации

begin

AuthImg.Visible := False;

Login.Visible := True;

Pass.Visible := True;

LoginLabel.Visible := True;

PassLabel.Visible := True;

Name.Visible := False;

Surname.Visible := False;

ConfrimPass.Visible :=False;

ConfrimPassLabel.Visible := False;

NameLabel.Visible := False;

SurnameLabel.Visible := False;

LoginLabel.Top := 285;

Login.Top := 285;

RegImg.Top := 502;

RegImg.Left := 100;

PassLabel.Left := 650;

RegButton.Visible := False;

AuthButton.Visible := True;

RegImg.Visible := True;

end;

procedure TReg.FormCreate(Sender: TObject); //загрузка картинок, подключение БД, расположение элементов упралвения

begin

Application.Title := 'Планета велосипедов';

VeloLandDatabase.ConnectionString := 'Provider=Microsoft.Jet.OLEDB.4.0; '+

'Persist Security Info=False; Data Source='+

ExtractFilePath(Application.ExeName)+'VeloLandDatabase.mdb';

VeloLandUsers.Connection := VeloLandDatabase;

VeloLandUsers.TableName := 'Users';

AddFontResource('resources\fonts\minecraft.ttf');

SendMessage(HWND\_BROADCAST, WM\_FONTCHANGE, 0, 0) ;

RegButton.Picture.LoadFromFile('resources\buttons\regcomplete.png');

AuthButton.Picture.LoadFromFile('resources\buttons\authcomplete.png');

AuthButton.AutoSize := True;

AuthButton.Visible := False;

Reg.Position := poDesktopCenter;

RegBg.Picture.LoadFromFile('resources\screens\_and\_bg\device\_test\_bg.png');

RegBg.AutoSize := True;

AuthImg.Picture.LoadFromFile('resources\buttons\auth.png');

AuthImg.AutoSize := True;

RegImg.Picture.LoadFromFile('resources\buttons\reg.png');

RegImg.AutoSize := True;

Name.Visible := False;

Surname.Visible := False;

Pass.Visible := False;

ConfrimPass.Visible :=False;

PassLabel.Visible := False;

ConfrimPassLabel.Visible := False;

LoginLabel.Visible := False;

Login.Visible := False;

NameLabel.Visible := False;

SurnameLabel.Visible := False;

RegButton.Visible := False;

LoginLabel.Caption := 'Логин:';

LoginLabel.Font.Name := 'Minecraft Rus Regular';

LoginLabel.Font.Size := 16;

NameLabel.Caption := 'Имя:';

NameLabel.Font.Name := 'Minecraft Rus Regular';

NameLabel.Font.Size := 16;

SurnameLabel.Caption := 'Фамилия:';

SurnameLabel.Font.Name := 'Minecraft Rus Regular';

SurnameLabel.Font.Size := 16;

PassLabel.Caption := 'Пароль:';

PassLabel.Font.Name := 'Minecraft Rus Regular';

PassLabel.Font.Size := 16;

ConfrimPassLabel.Caption := 'Подтвердите пароль:';

ConfrimPassLabel.Font.Name := 'Minecraft Rus Regular';

ConfrimPassLabel.Font.Size := 12;

Title.Caption := 'Приветстувем Вас в приложении "Планета велосипедов"!';

Title.Font.Name := 'Minecraft Rus Regular';

Title.Font.Size := 20;

Title2.Caption := 'Для начала авторизируйтесь!';

Title2.Font.Name := 'Minecraft Rus Regular';

Title2.Font.Size := 20;

end;

procedure TReg.FormDestroy(Sender: TObject); //удаление шрифта из системы пользователя при закрытии

begin

RemoveFontResource('resources\fonts\minecraft.ttf') ;

SendMessage(HWND\_BROADCAST, WM\_FONTCHANGE, 0, 0) ;

end;

procedure TReg.N2Click(Sender: TObject); //вызов справки

begin

ShellExecute(0, PChar('Open'),PChar('VeloLandHelp.chm'),nil,nil,SW\_SHOW);

end;

procedure TReg.N3Click(Sender: TObject); //окно "О программе"

begin

AProgramm.ShowModal;

end;

procedure TReg.RegButtonClick(Sender: TObject); //регистрация пользователя

begin

VeloLandUsers.Active := True;

if (Login.Text = '') or (Name.Text = '') or (Surname.Text = '') or (Pass.Text = '') or (ConfrimPass.Text = '') then //проверка на корректность ввода

ShowMessage('Каждое поле должно быть заполнено!')

else if (Length(Pass.Text) < 4) then

ShowMessage('Длина пароля не может быть меньше 4 символов!')

else if VeloLandUsers.Locate('UserLogin', Login.Text, []) then

ShowMessage('Данный пользователь уже присутствует в системе! Перейдите на страницу авторизации!')

else

begin //добавление нового пользователя в БД

VeloLandUsers.Insert;

VeloLandUsers.FieldByName('UserName').AsString := Name.Text;

VeloLandUsers.FieldByName('UserSurname').AsString := Surname.Text;

VeloLandUsers.FieldByName('UserLogin').AsString := Login.Text;

VeloLandUsers.FieldByName('UserPassword').AsString := Pass.Text;

VeloLandUsers.FieldByName('UserScore').AsInteger := 0;

VeloLandUsers.FieldByName('UserDeviceTestScore').AsInteger := 0;

VeloLandUsers.FieldByName('UserPDDTestScore').AsInteger := 0;

VeloLandUsers.FieldByName('UserTypesTestScore').AsInteger := 0;

VeloLandUsers.FieldByName('UserGameScore').AsInteger := 0;

VeloLandUsers.post;

ShowMessage('Вы успешно зарегистрировались! Перейдите на страницу авторизации для входа в программу!');

end;

Pass.Text := '';

ConfrimPass.Text := '';

end;

procedure TReg.RegImgClick(Sender: TObject); //отображение полей регистрации

begin

RegImg.Visible := False;

AuthImg.Top := 502;

AuthImg.Visible := True;

Login.Top := 220;

LoginLabel.Top := 220;

Name.Visible := True;

Surname.Visible := True;

Pass.Visible := True;

ConfrimPass.Visible := True;

Login.Visible := True;

LoginLabel.Visible := True;

NameLabel.Visible := True;

SurnameLabel.Visible := True;

PassLabel.Visible := True;

ConfrimPassLabel.Visible := True;

RegButton.Visible := True;

AuthButton.Visible := False;

end;

end.

unit Main; //модуль главного меню

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Imaging.GIFImg, Vcl.ExtCtrls,

Vcl.Imaging.jpeg, Vcl.MPlayer, Vcl.OleCtrls, Vcl.StdCtrls,

Vcl.Buttons, Vcl.Imaging.pngimage, Vcl.Menus, ShellAPI, IOUtils;

type

TMainForm = class(TForm)

MainFormbg: TImage;

Title: TLabel;

TitleColor: TTimer;

ButtonDevice: TImage;

Volume: TImage;

Musicbg: TMediaPlayer;

ExitButton: TImage;

ButtonSelection: TImage;

TrickButton: TImage;

RouteButton: TImage;

TestsButton: TImage;

MusicTimer: TTimer;

MusicList: TListBox;

WelcomeLabel: TLabel;

Score: TLabel;

GameButton: TImage;

MainFormMenu: TMainMenu;

N1: TMenuItem;

TestsMenu: TMenuItem;

DeviceMenu: TMenuItem;

TypesMenu: TMenuItem;

TricksMenu: TMenuItem;

RoutesMenu: TMenuItem;

PicturesGamesMenu: TMenuItem;

N8: TMenuItem;

Help: TMenuItem;

AboutProg: TMenuItem;

ExitAccount: TMenuItem;

ResetProgress: TMenuItem;

MusicSwap: TLabel;

NextMusic: TImage;

PrevMusic: TImage;

procedure TitleColorTimer(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure VolumeClick(Sender: TObject);

procedure ExitButtonClick(Sender: TObject);

procedure ButtonDeviceClick(Sender: TObject);

procedure MusicTimerTimer(Sender: TObject);

procedure ButtonSelectionClick(Sender: TObject);

procedure TrickButtonClick(Sender: TObject);

procedure RouteButtonClick(Sender: TObject);

procedure FormActivate(Sender: TObject);

procedure TestsButtonClick(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure GameButtonClick(Sender: TObject);

procedure DeviceMenuClick(Sender: TObject);

procedure TypesMenuClick(Sender: TObject);

procedure TricksMenuClick(Sender: TObject);

procedure RoutesMenuClick(Sender: TObject);

procedure TestsMenuClick(Sender: TObject);

procedure PicturesGamesMenuClick(Sender: TObject);

procedure ExitAccountClick(Sender: TObject);

procedure HelpClick(Sender: TObject);

procedure AboutProgClick(Sender: TObject);

procedure ResetProgressClick(Sender: TObject);

procedure NextMusicClick(Sender: TObject);

procedure PrevMusicClick(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

private

var

{ Private declarations }

public

{ Public declarations }

end;

var

MainForm: TForm;

i, music\_index: integer;

implementation

{$R \*.dfm}

uses BicycleDevice, BicyclesTypes, BicyclesTricks, HrodnaRoutes,

RegistrationForm, Tests, Game, AboutProgram;

procedure TMainForm.ButtonDeviceClick(Sender: TObject); // кнопка для отображения раздела "Устройство велосипеда"

begin

Device.ShowModal;

end;

procedure TMainForm.FormActivate(Sender: TObject); // отображение логина авторизованного пользователя

begin

WelcomeLabel.Caption :='Вы авторизовались как ' + Reg.VeloLandUsers.FieldByName('UserLogin').AsString + '!';

end;

procedure TMainForm.FormClose(Sender: TObject; var Action: TCloseAction); // заркыть полностью приложении при закрытии главного меню

begin

Application.Terminate;

end;

procedure TMainForm.FormCreate(Sender: TObject); // загрузка картинок, музыки, установка расположения элементов управления

begin

Application.Title := 'Планета велосипедов';

MainForm.WindowState := wsMaximized;

TitleColor.Enabled := True;

MusicList.Items.Add('resources/music/8-Bit Misfits - Get Lucky.mp3');

MusicList.Items.Add('resources/music/Similar But Differet - Blinding Lights.mp3');

MusicList.Items.Add('resources/music/Dynamite - Hobismorning.mp3');

MusicList.Items.Add('resources/music/8-Bit Misfits - Shake It Off.mp3');

MusicList.Items.Add('resources/music/8-Bit Misfits - Watermelon Sugar.mp3');

Musicbg.FileName := MusicList.Items[0];

music\_index := 0;

Musicbg.Open;

Musicbg.Play;

ButtonDevice.Picture.LoadFromFile('resources\buttons\button\_device.png');

ButtonSelection.Picture.LoadFromFile('resources\buttons\button\_selection.png');

TrickButton.Picture.LoadFromFile('resources\buttons\button\_trick.png');

RouteButton.Picture.LoadFromFile('resources\buttons\route\_button.png');

Volume.Picture.LoadFromFile('resources\buttons\volume\_on.png');

ExitButton.Picture.LoadFromFile('resources\buttons\exit.png');

TestsButton.Picture.LoadFromFile('resources\buttons\bicycletest\_button.png');

GameButton.Picture.LoadFromFile('resources\buttons\game\_button.png');

WelcomeLabel.Font.Size := 23;

Score.Font.Size := 16;

WelcomeLabel.Font.Name := 'Minecraft Rus Regular';

Score.Font.Name := 'Minecraft Rus Regular';

MusicSwap.Font.Name := 'Minecraft Rus Regular';

MusicSwap.Font.Size := 20;

NextMusic.Picture.LoadFromFile('resources/buttons/nextmusic.png');

NextMusic.AutoSize := True;

PrevMusic.Picture.LoadFromFile('resources/buttons/prevmusic.png');

PrevMusic.AutoSize := True;

end;

procedure TMainForm.FormShow(Sender: TObject); // отобразить счёт пользователя

begin

Score.Caption := 'Текущий счёт:' + #13#10 + Reg.VeloLandUsers.FieldByName('UserScore').AsString;

end;

procedure TMainForm.GameButtonClick(Sender: TObject); // кнопка для перехода на игру "Парные картинки с велосипедами"

begin

if Reg.VeloLandUsers.FieldByName('UserScore').AsInteger < 2700 then

ShowMessage('Недостаточно очков для открытия игры! Наберите минимум 2700 очков!')

else

begin

PicturesGame.Show;

MainForm.Hide;

end;

end;

procedure TMainForm.MusicTimerTimer(Sender: TObject); // проверка закончилась ли музыка и отображение текущей композиции

begin

if Musicbg.Mode = mpStopped then

begin

if music\_index = MusicList.Count-1 then

music\_index := -1;

Musicbg.FileName := MusicList.Items[music\_index+1];

music\_index := music\_index + 1;

Musicbg.Open;

Musicbg.Play;

end;

MusicSwap.Caption := 'Сейчас играет: ' + StringReplace(Copy(MusicList.Items[music\_index], 17), '.mp3', '', [rfReplaceAll, rfIgnoreCase]);

end;

procedure TMainForm.AboutProgClick(Sender: TObject); // отображение окна "О программе"

begin

AProgramm.ShowModal;

end;

procedure TMainForm.ResetProgressClick(Sender: TObject); // сбросить прогресс пользователя

var buttonSelected: integer;

begin

buttonSelected := MessageDlg('Вы уверены, что хотите сбросить прогресс? Для этого нужно будет перезайти в аккаунт',mtConfirmation, mbOKCancel, 0);

if buttonSelected = mrOK then

begin

Reg.VeloLandUsers.Edit;

Reg.VeloLandUsers.FieldByName('UserScore').AsInteger := 0;

Reg.VeloLandUsers.FieldByName('UserDeviceTestScore').AsInteger := 0;

Reg.VeloLandUsers.FieldByName('UserPDDTestScore').AsInteger := 0;

Reg.VeloLandUsers.FieldByName('UserTypesTestScore').AsInteger := 0;

Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger := 0;

Reg.VeloLandUsers.Post;

MainForm.Hide;

end;

end;

procedure TMainForm.ExitAccountClick(Sender: TObject); // выход из аккаунта

var buttonSelected: integer;

begin

buttonSelected := MessageDlg('Вы уверены, что хотите выйти из аккаунта?',mtConfirmation, mbOKCancel, 0);

if buttonSelected = mrOK then

begin

MainForm.Close;

end

end;

procedure TMainForm.TestsMenuClick(Sender: TObject); // отображение раздела "Тесты"

begin

Test.Show;

MainForm.Hide;

end;

procedure TMainForm.DeviceMenuClick(Sender: TObject); // отображение раздела "Устройство велосипеда"

begin

Device.ShowModal;

end;

procedure TMainForm.TypesMenuClick(Sender: TObject); // отображение раздела "Типы велосипедов"

begin

Types.ShowModal;

end;

procedure TMainForm.TricksMenuClick(Sender: TObject); // отображение раздела "Трюки на велосипеде"

begin

Tricks.ShowModal;

end;

procedure TMainForm.RoutesMenuClick(Sender: TObject); // отображение раздела "Маршруты Гродно"

begin

Routes.ShowModal

end;

procedure TMainForm.PicturesGamesMenuClick(Sender: TObject); // игры "Парные картинки с велосипедами

begin

if Reg.VeloLandUsers.FieldByName('UserScore').AsInteger < 2700 then

ShowMessage('Недостаточно очков для открытия игры! Наберите минимум 2700 очков!')

else

begin

PicturesGame.Show;

MainForm.Hide;

end;

end;

procedure TMainForm.HelpClick(Sender: TObject); // вызов справки

begin

ShellExecute(0, PChar('Open'),PChar('VeloLandHelp.chm'),nil,nil,SW\_SHOW);

end;

procedure TMainForm.NextMusicClick(Sender: TObject); //переключение музыки вперёд

begin

if music\_index = MusicList.Count-1 then

music\_index := 0

else

music\_index := music\_index + 1;

Musicbg.FileName := MusicList.Items[music\_index];

Musicbg.Open;

Musicbg.Play;

end;

procedure TMainForm.PrevMusicClick(Sender: TObject); // переключение музыки назад

begin

if music\_index = 0 then

music\_index := 4

else

music\_index := music\_index - 1;

Musicbg.FileName := MusicList.Items[music\_index];

Musicbg.Open;

Musicbg.Play;

end;

procedure TMainForm.RouteButtonClick(Sender: TObject); // кнопка для отображения раздела "Маршруты Гродно"

begin

Routes.ShowModal;

end;

procedure TMainForm.ButtonSelectionClick(Sender: TObject); // кнопка отображения раздела "Виды велосипедов"

begin

Types.ShowModal;

end;

procedure TMainForm.TestsButtonClick(Sender: TObject); // кнопка отображения раздела "Тесты"

begin

Test.Show;

MainForm.Hide;

end;

procedure TMainForm.ExitButtonClick(Sender: TObject); // кнопка выхода из приложения

var buttonSelected: integer;

begin

buttonSelected := MessageDlg('Вы уверены, что хотите выйти?',mtConfirmation, mbOKCancel, 0);

if buttonSelected = mrOK then

begin

Application.Terminate;

end;

end;

procedure TMainForm.TitleColorTimer(Sender: TObject); // переливание текста "Планета велосипедов"

begin

Title.Font.Color := rgb(random(255),random(255), random(255));

end;

procedure TMainForm.TrickButtonClick(Sender: TObject);

begin

Tricks.ShowModal;

end;

procedure TMainForm.VolumeClick(Sender: TObject); // кнопка включения и выключения музыки

begin

if Musicbg.Mode = mpPlaying then

begin

Musicbg.Pause;

Volume.Picture.LoadFromFile('resources\buttons\volume\_off.png');

end

else if Musicbg.Mode = mpPaused then

begin

Musicbg.Play;

Volume.Picture.LoadFromFile('resources\buttons\volume\_on.png');

end;

end;

end.

unit BicycleDevice; // модуль раздела "Устройство велосипеда"

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.OleCtrls, SHDocVw,

Vcl.StdCtrls, Vcl.Imaging.pngimage;

type

TDevice = class(TForm)

Page: TWebBrowser;

procedure FormActivate(Sender: TObject);

procedure PageBeforeScriptExecute(ASender: TObject;

const pDispWindow: IDispatch);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Device: TDevice;

implementation

{$R \*.dfm}

uses Main;

procedure TDevice.FormActivate(Sender: TObject); //открытие HTML-страницы с разделом "Устройство велосипеда"

begin

Page.Navigate('file://' + ExtractFilePath(paramstr(0)) +'resources/html\_pages/bicycle\_device/bicycle\_device.html');

end;

procedure TDevice.PageBeforeScriptExecute(ASender: TObject;

const pDispWindow: IDispatch); // закртие формы

begin

Device.Close;

end;

end.

unit BicyclesTypes; // модуль раздела "Виды велосипедов"

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.OleCtrls, SHDocVw;

type

TTypes = class(TForm)

Page2: TWebBrowser;

procedure FormActivate(Sender: TObject);

procedure Page2BeforeScriptExecute(ASender: TObject;

const pDispWindow: IDispatch);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Types: TTypes;

implementation

{$R \*.dfm}

procedure TTypes.FormActivate(Sender: TObject); // открытие HTML-страницы с разделом "Виды велосипедов"

begin

Page2.Navigate('file://' + ExtractFilePath(paramstr(0)) +'resources/html\_pages/types\_of\_bicycles/types.html');

end;

procedure TTypes.Page2BeforeScriptExecute(ASender: TObject;

const pDispWindow: IDispatch); // закрытие формы

begin

Types.Close;

end;

end.

unit BicyclesTricks; // модуль раздела "Трюки на велосипеде"

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.OleCtrls, SHDocVw;

type

TTricks = class(TForm)

Page3: TWebBrowser;

procedure FormActivate(Sender: TObject);

procedure Page3BeforeScriptExecute(ASender: TObject;

const pDispWindow: IDispatch);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Tricks: TTricks;

implementation

{$R \*.dfm}

procedure TTricks.FormActivate(Sender: TObject); // открытие HTML-страницы с разделом "Трюки на велосипеде"

begin

Page3.Navigate('file://' + ExtractFilePath(paramstr(0)) + 'resources/html\_pages/tricks\_bicycle/tricks\_bicycles.html');

end;

procedure TTricks.Page3BeforeScriptExecute(ASender: TObject;

const pDispWindow: IDispatch); // закрытие формы

begin

Tricks.Close;

end;

end.

unit HrodnaRoutes; // модуль раздела "Маршруты Гродно"

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.OleCtrls, SHDocVw;

type

TRoutes = class(TForm)

Page4: TWebBrowser;

procedure FormActivate(Sender: TObject);

procedure Page4BeforeScriptExecute(ASender: TObject;

const pDispWindow: IDispatch);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Routes: TRoutes;

implementation

{$R \*.dfm}

procedure TRoutes.FormActivate(Sender: TObject); // открытие HTML-страницы с разделом "Маршруты Гродно"

begin

Page4.Navigate('file://' + ExtractFilePath(paramstr(0)) +'resources/html\_pages/Hrodna\_routes/Hrodna\_routes.html');

end;

procedure TRoutes.Page4BeforeScriptExecute(ASender: TObject;

const pDispWindow: IDispatch); // закрытие формы

begin

Routes.Close;

end;

end.

unit Tests; // модуль раздела "Тесты"

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.StdCtrls,

System.ImageList, Vcl.ImgList, Vcl.Menus, ShellAPI;

type

TTest = class(TForm)

DeviceTBg: TImage;

TestsVariant: TRadioGroup;

NextButton: TImage;

Questions: TRadioGroup;

Score: TLabel;

PictureDevice: TImage;

OnMainForm: TImage;

Timer: TTimer;

Time: TLabel;

MainMenuTest: TMainMenu;

N1: TMenuItem;

Help: TMenuItem;

AboutProgr: TMenuItem;

procedure FormCreate(Sender: TObject);

procedure TestsVariantClick(Sender: TObject);

procedure NextButtonClick(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure OnMainFormClick(Sender: TObject);

procedure TimerTimer(Sender: TObject);

procedure HelpClick(Sender: TObject);

procedure AboutProgrClick(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Test: TTest;

f: text;

s: string;

Nvern, ball: integer;

img\_index:integer;

i:integer;

implementation

{$R \*.dfm}

uses RegistrationForm, Main, AboutProgram;

procedure TTest.FormClose(Sender: TObject; var Action: TCloseAction); // запись прогресса в БД при выходе из формы

begin

Reg.VeloLandUsers.Edit;

Reg.VeloLandUsers.FieldByName('UserScore').AsInteger := Reg.VeloLandUsers.FieldByName('UserDeviceTestScore').AsInteger

+ Reg.VeloLandUsers.FieldByName('UserPDDTestScore').AsInteger

+ Reg.VeloLandUsers.FieldByName('UserTypesTestScore').AsInteger

+ Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger;

Reg.VeloLandUsers.Post;

TestsVariant.ItemIndex := -1;

TestsVariant.Enabled := True;

Timer.Enabled := False;

Time.Visible := False;

Questions.Items.Clear;

Questions.Caption := 'Вопросы';

Questions.Visible := False;

PictureDevice.Visible := False;

MainForm.Show;

end;

procedure TTest.FormCreate(Sender: TObject); // базовое расположение элементов управления

begin

Application.Title := 'Планета велосипедов';

Test.Position := poDesktopCenter;

DeviceTBg.Picture.LoadFromFile('resources\screens\_and\_bg\device\_test\_bg.png');

NextButton.Picture.LoadFromFile('resources\buttons\authcomplete.png');

OnMainForm.Picture.LoadFromFile('resources\buttons\return\_button.png');

OnMainForm.AutoSize := True;

NextButton.AutoSize := True;

NextButton.Enabled := False;

DeviceTBg.AutoSize := True;

Timer.Enabled := False;

TestsVariant.Font.Name := 'Minecraft Rus Regular';

Score.Font.Name := 'Minecraft Rus Regular';

Questions.Font.Name := 'Minecraft Rus Regular';

Questions.Font.Size := 16;

Score.Font.Size := 32;

Time.Font.Name := 'Minecraft Rus Regular';

Time.Font.Size := 16;

Time.Visible := False;

Score.Font.Color := clBlack;

Score.Visible := False;

Questions.Visible := False;

TestsVariant.Font.Size := 12;

end;

procedure TTest.HelpClick(Sender: TObject); // вызов справки

begin

ShellExecute(0, PChar('Open'),PChar('VeloLandHelp.chm'),nil,nil,SW\_SHOW);

end;

procedure TTest.AboutProgrClick(Sender: TObject); // вызов окна "О программе"

begin

AProgramm.ShowModal;

end;

procedure TTest.NextButtonClick(Sender: TObject); // переход к следующему вопросу

begin

if (Questions.ItemIndex>-1) and (not Eof(f)) then begin

if Questions.ItemIndex = Nvern-1 then ball:=ball+100; //Если выбранный вариант соответствует

Questions.Items.Clear; //номеру верного ответа то балл прибавляется

Repeat //и очищается поле для следующего вопроса

if (s[1]='-') then begin

delete(s,1,1);

Questions.Caption:=s;

end

else if s[1]='\*' then begin

delete(s,1,1);

Nvern:=StrToInt(s);

end

else Questions.Items.Add(s);

readln(f,s);

Score.Caption:=s;

until (s[1]='-') or (Eof(f));

if TestsVariant.ItemIndex = 0 then

begin

img\_index := img\_index + 1;

PictureDevice.Picture.LoadFromFile('resources\tests\device\_test\' + IntToStr(img\_index) + '.jpg') // загрузка картинок для впоросов

end

else if TestsVariant.ItemIndex = 2 then

begin

img\_index := img\_index + 1;

PictureDevice.Picture.LoadFromFile('resources\tests\types\_test\' + IntToStr(img\_index) + '.jpg') // загрузка картинок для вопросов

end;

end

//Если конец файла достигнут, значит вопросы закончились

Else if Eof(f) then begin

delete(s,1,1);

Nvern:=StrToInt(s);

if Questions.ItemIndex = Nvern-1 then ball:=ball+100;

Score.Visible := True;

Score.Caption:='Итоговый счёт: ' +IntToStr(ball); //Вывод количества баллов

Reg.VeloLandUsers.Edit;

if TestsVariant.ItemIndex = 0 then

if ball > Reg.VeloLandUsers.FieldByName('UserDeviceTestScore').AsInteger then // проверка был ли пройден ранее тест

Reg.VeloLandUsers.FieldByName('UserDeviceTestScore').AsInteger := ball;

if TestsVariant.ItemIndex = 1 then

if ball > Reg.VeloLandUsers.FieldByName('UserPDDTestScore').AsInteger then

Reg.VeloLandUsers.FieldByName('UserPDDTestScore').AsInteger := ball;

if TestsVariant.ItemIndex = 2 then

if ball > Reg.VeloLandUsers.FieldByName('UserTypesTestScore').AsInteger then

Reg.VeloLandUsers.FieldByName('UserTypesTestScore').AsInteger := ball;

Reg.VeloLandUsers.Post;

CloseFile(f);

NextButton.Enabled:=False; //кнопка становится недоступной в завершении проекта

TestsVariant.Enabled := True;

TestsVariant.ItemIndex := -1;

Questions.Items.Clear;

Timer.Enabled := False;

end;

end;

procedure TTest.OnMainFormClick(Sender: TObject);

begin

Test.Close;

end;

procedure TTest.TestsVariantClick(Sender: TObject);

begin

case TestsVariant.ItemIndex of //В зависимости от выбранного варианта переменная f

0: begin

AssignFile( f ,'resources\tests\device\_test\device\_test.txt');//связывается с разными файлами

img\_index := 1;

Questions.Font.Size := 16;

Questions.Width := 700;

PictureDevice.Left := 800;

PictureDevice.Width := 545;

PictureDevice.Height := 400;

PictureDevice.Picture.LoadFromFile('resources\tests\device\_test\' + IntToStr(img\_index) + '.jpg');

end;

1: begin

if Reg.VeloLandUsers.FieldByName('UserDeviceTestScore').AsInteger + Reg.VeloLandUsers.FieldByName('UserPDDTestScore').AsInteger // проверка наличия нужного количества очков

+ Reg.VeloLandUsers.FieldByName('UserTypesTestScore').AsInteger < 700 then

begin

TestsVariant.ItemIndex := -1;

ShowMessage('Наберите минимум 700 очков для прохождения этого теста!');

exit;

end

else

begin

AssignFile( f ,'resources\tests\pdd\_test\pdd\_test.txt');

Questions.Font.Size := 12;

Questions.Width := 1000;

PictureDevice.Left := 1100;

PictureDevice.Width := 350;

PictureDevice.Height := 450;

PictureDevice.Picture.LoadFromFile('resources\tests\pdd\_test\traffic.png');

end;

end;

2: begin

if Reg.VeloLandUsers.FieldByName('UserDeviceTestScore').AsInteger + Reg.VeloLandUsers.FieldByName('UserPDDTestScore').AsInteger

+ Reg.VeloLandUsers.FieldByName('UserTypesTestScore').AsInteger < 1400 then

begin

TestsVariant.ItemIndex := -1;

ShowMessage('Наберите минимум 1400 очков для прохождения этого теста!');

exit;

end

else

begin

AssignFile( f ,'resources\tests\types\_test\types\_test.txt');

img\_index := 1;

Questions.Font.Size := 14;

Questions.Width := 700;

PictureDevice.Left := 800;

PictureDevice.Width := 700;

PictureDevice.Height := 400;

PictureDevice.Picture.LoadFromFile('resources\tests\types\_test\' + IntToStr(img\_index) + '.jpg');

end;

end;

end;

TestsVariant.Enabled:=false; //Выбор варианта становится недоступен

Questions.Enabled:=true; //Доступным становится поле с вопросом

NextButton.Enabled:=true; //Кнопка Далее

Score.Visible := False;

Questions.Visible := True;

PictureDevice.Visible := True;

i := 300;

Timer.Enabled := True;

Time.Visible := True;

reset(f); //Открываем файл для чтения

readln(f,s); //Считываем первую строку из файла

ball:=0; //изначально количество баллов 0

repeat

if (s[1]='-') then begin //Если первый символ строки ‘-‘ значит это вопрос

delete(s,1,1);

Questions.Caption:=s;

end

else if s[1]='\*' then begin //Если перв символ ‘\*’ значит это номер верного ответа

delete(s,1,1);

Nvern:=StrToInt(s);

end

else Questions.Items.Add(s); //Иначе это вариант ответа

readln(f,s); //Считываем следующую строку из файла

until (s[1]='-') or (Eof(f)); //Считывание и отправление вариантов ответов в RadiGroup до тех пор

// пока не достигнут следующий вопрос или конец файла

end;

procedure TTest.TimerTimer(Sender: TObject); // таймер ограничения времени

var H,M,S:integer;

begin

i := i - 1;

H := I div 3600;

M := (I mod 3600) div 60;

S := (I mod 3600) mod 60;

Time.Caption := 'Оставшееся время:' + Format('%.2d:',[H]) + Format('%.2d:',[M]) + Format('%.2d',[S]);

if i = 0 then // если время вышло, то начисляются очки, которые успел заработать пользователь

begin

Timer.Enabled := False;

Questions.Enabled := False;

NextButton.Enabled := False;

TestsVariant.Enabled := True;

Time.Visible := False;

Reg.VeloLandUsers.Edit;

if TestsVariant.ItemIndex = 0 then

if ball > Reg.VeloLandUsers.FieldByName('UserDeviceTestScore').AsInteger then

Reg.VeloLandUsers.FieldByName('UserDeviceTestScore').AsInteger := ball;

if TestsVariant.ItemIndex = 1 then

if ball > Reg.VeloLandUsers.FieldByName('UserPDDTestScore').AsInteger then

Reg.VeloLandUsers.FieldByName('UserPDDTestScore').AsInteger := ball;

if TestsVariant.ItemIndex = 2 then

if ball > Reg.VeloLandUsers.FieldByName('UserTypesTestScore').AsInteger then

Reg.VeloLandUsers.FieldByName('UserTypesTestScore').AsInteger := ball;

Reg.VeloLandUsers.Post;

TestsVariant.ItemIndex := -1;

Questions.Visible := False;

PictureDevice.Visible := False;

Questions.Items.Clear;

ShowMessage('Время вышло! Вам начислено ' + IntToStr(ball) + ' очков!');

end;

end;

end.

unit Game; // модуль игры "Парные картинки"

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Menus, Vcl.ExtCtrls, Vcl.StdCtrls, ShellAPI;

type

TPicturesGame = class(TForm)

Update: TTimer;

GameBg: TImage;

LevelOneButton: TImage;

LevelTwoButton: TImage;

LevelThreeButton: TImage;

LevelFourButton: TImage;

Field: TPaintBox;

Timer: TTimer;

Time: TLabel;

MoveCounter: TLabel;

BackToMainMenu: TImage;

MainMenuGame: TMainMenu;

N1: TMenuItem;

Help: TMenuItem;

AboutProgr: TMenuItem;

Title: TLabel;

TitleColor: TTimer;

procedure FormCreate(Sender: TObject);

procedure UpdateTimer(Sender: TObject);

procedure LevelOneButtonClick(Sender: TObject);

procedure FieldPaint(Sender: TObject);

procedure FieldMouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure LevelTwoButtonClick(Sender: TObject);

procedure LevelThreeButtonClick(Sender: TObject);

procedure LevelFourButtonClick(Sender: TObject);

procedure TimerTimer(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure BackToMainMenuClick(Sender: TObject);

procedure HelpClick(Sender: TObject);

procedure AboutProgrClick(Sender: TObject);

procedure TitleColorTimer(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

// объявление нового типа col\_row

col\_row = record

col: integer;

row: integer;

end;

const

MAX\_SIZE = 32; // максимальное кол-во парных картинок

MAX\_H = 8; // максимальный размер поля - 8х8

MAX\_W = 8;

var

PicturesGame: TPicturesGame;

Pole: array [1..MAX\_H,1..MAX\_W] of integer;

{ Pole[i,j] < 100 - код картинки, клетка закрыта;

Pole[i,j] > 100 и < 200 - клетка открыта, т.е. игрок видит картинку;

Pole[i,j] > 200 - игрок нашел пару для этой картинки }

Pictures: TBitmap; // картинки, загруженные из файла

n : integer; // колво открытых пар картинок

count: integer; // количества открытых в данный момент клеток

open1: col\_row; // координаты 1-й открытой клетки

open2: col\_row; // координаты 2-й открытой клетки

W: integer; // кол-во клеток по горизонтали

H: integer; // кол-во клеток по вертикали

// произведение W и H должно быть кратно 2-м

WK: integer; // ширина клетки

HK: integer; // высота клетки

i: integer;

Moves: integer;

SetLevel: string;

implementation

{$R \*.dfm}

uses RegistrationForm, Main, AboutProgram;

// рисует клетку поля

procedure Kletka(col,row: integer);

var

x,y: integer; // левый верхний угол клетки (координаты)

src, dst : Trect; // источник и получатель битового образа

begin

// преобразуем координаты клетки

// в координаты на поверхности формы

x := (col-1)\*WK;

y := (row-1)\*HK;

if Pole[col,row] > 200 then

// для этой клетки найдена пара

// клетку надо убрать с поля

begin

// установить цвет границы, закраски и текста

PicturesGame.Field.Canvas.Brush.Color := clGreen;

PicturesGame.Field.Canvas.Pen.Color := clGreen;

PicturesGame.Field.Canvas.Font.Color := clGreen;

end;

if (Pole[col,row] > 100) and (Pole[col,row] < 200)

then

// клетка открыта - вывести картинку

begin

// Pole[col,row] = номер картинки + 100,

// где 100 - признак того, что клетка открыта

// определим положение картинки в Pictures

src := Bounds((Pole[col,row]-100 -1 )\*WK,0,WK,HK);

// координаты картинки (клетки) на форме

dst := Bounds(x,y,HK-2,WK-2);

// вывести картинку в клетку

PicturesGame.Field.Canvas.CopyRect(dst,Pictures.Canvas,src);

// установить цвет границы и цифры

PicturesGame.Field.Canvas.Pen.Color := clBlack;

PicturesGame.Field.Canvas.Font.Color := clBlack;

PicturesGame.Field.Canvas.Brush.Style := bsClear;

end;

if (Pole[col,row] > 0) and (Pole[col,row] < 100) then

// клетка закрыта, рисуем только контур

begin

PicturesGame.Field.Canvas.Brush.Color := clBtnFace;

PicturesGame.Field.Canvas.Pen.Color := clBlack;

PicturesGame.Field.Canvas.Font.Color := clBtnFace;

end;

// отрисовать клетку

PicturesGame.Field.Canvas.Rectangle(x,y,x+WK-2,y+HK-2);

PicturesGame.Field.Canvas.Brush.Color := clBtnFace;

end;

// отрисовывает поле

procedure ShowPole;

var

row,col: integer;

begin

for row:=1 to H do

for col:=1 to W do

Kletka(row,col);

end;

// новая игра

Procedure NewGame;

var

k: integer; // кол-во парных картинок

r: integer; // случайное число

buf: array[1..MAX\_SIZE] of integer;

// в buf[i] записываем, сколько чисел i

// записали в массив Pole

i,j: integer; // индексы массивов

begin

Randomize;

k := Trunc(H\*W/2);

for i:=1 to k do

buf[i] := 0;

// запишем в массив Pole случайные числа

// от 1 до 2

// каждое число должно быть записано два раза

for i:=1 to H do

for j:=1 to W do

begin

repeat

r := random (k) + 1;

until buf[r] < 2;

Pole[i,j] := r; // код картинки

inc(buf[r]);

end;

// здесь поле сгенерировано

n:=0;

ShowPole;

end;

procedure TPicturesGame.FormClose(Sender: TObject; var Action: TCloseAction); // начисление очков в БД при закрытии формы

begin

Timer.Enabled := False;

Time.Visible := False;

Field.Visible := False;

LevelOneButton.Visible := True;

LevelTwoButton.Visible := True;

LevelThreeButton.Visible := True;

LevelFourButton.Visible := True;

Title.Visible := True;

MoveCounter.Visible := False;

BackToMainMenu.Left := 700;

Reg.VeloLandUsers.Edit;

SetLevel := '';

Reg.VeloLandUsers.FieldByName('UserScore').AsInteger := Reg.VeloLandUsers.FieldByName('UserDeviceTestScore').AsInteger

+ Reg.VeloLandUsers.FieldByName('UserPDDTestScore').AsInteger

+ Reg.VeloLandUsers.FieldByName('UserTypesTestScore').AsInteger

+ Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger;

end;

procedure TPicturesGame.FormCreate(Sender: TObject); // базовое расположение элементов и загрузка картинок

begin

Application.Title := 'Планета велосипедов';

GameBg.Picture.LoadFromFile('resources/screens\_and\_bg/device\_test\_bg.png');

LevelOneButton.Picture.LoadFromFile('resources/buttons/game\_button\_lvl1.png');

LevelTwoButton.Picture.LoadFromFile('resources/buttons/game\_button\_lvl2.png');

LevelThreeButton.Picture.LoadFromFile('resources/buttons/game\_button\_lvl3.png');

LevelFourButton.Picture.LoadFromFile('resources/buttons/game\_button\_lvl4.png');

BackToMainMenu.Picture.LoadFromFile('resources/buttons/return\_button.png');

BackToMainMenu.AutoSize := True;

Time.Font.Name := 'Minecraft Rus Regular';

Time.Font.Size := 14;

Moves := 0;

Time.Visible := False;

MoveCounter.Font.Name := 'Minecraft Rus Regular';

MoveCounter.Font.Size := 14;

MoveCounter.Caption := 'Количество ходов';

Title.Font.Name := 'Minecraft Rus Regular';

Title.Font.Size := 22;

Title.Caption := 'Игра "Парные картинки с велосипедами"';

MoveCounter.Visible := False;

PicturesGame.Position := poDesktopCenter;

Timer.Enabled := False;

LevelOneButton.AutoSize := True;

LevelTwoButton.AutoSize := True;

LevelThreeButton.AutoSize := True;

LevelFourButton.AutoSize := True;

Field.Visible := False;

Update.Enabled := False;

Update.Interval := 100;

end;

procedure TPicturesGame.LevelOneButtonClick(Sender: TObject); //выбор первого уровня игры

begin

LevelOneButton.Visible := False;

LevelTwoButton.Visible := False;

LevelThreeButton.Visible := False;

LevelFourButton.Visible := False;

Title.Visible := False;

Pictures := TBitmap.Create;

// загрузить картинки из файла

Pictures.LoadFromFile('resources/game\_pictures/Level1.bmp');

HK := 256; // высота картинки

WK := 256; // ширина картинки

// установить размера поля

H := 2;

W := 2;

Field.Visible := True;

Field.Height := H\* HK;

Field.Width := W \* WK;

MoveCounter.Caption := 'Количество ходов';

i := 30;

Field.Top := 200;

Field.Left := 700;

BackToMainMenu.Left := 700;

SetLevel := 'One';

Timer.Enabled := True;

Time.Visible := True;

MoveCounter.Visible := True;

n := 0;

NewGame;

end;

procedure TPicturesGame.LevelTwoButtonClick(Sender: TObject); //выбор второго уровня игры

begin

if Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger < 200 then

begin

ShowMessage('Для прхождения этого уровня Вы должны пройти предыдущий!');

exit;

end;

LevelOneButton.Visible := False;

LevelTwoButton.Visible := False;

LevelThreeButton.Visible := False;

LevelFourButton.Visible := False;

Title.Visible := False;

Pictures := TBitmap.Create;

// загрузить картинки из файла

Pictures.LoadFromFile('resources/game\_pictures/Level2.bmp');

HK := 128; // высота картинки

WK := 128; // ширина картинки

// установить размера поля

H := 4;

W := 4;

Field.Visible := True;

Field.Height := H \*HK;

Field.Width := W \* WK;

Field.Left := 700;

Field.Top := 200;

BackToMainMenu.Left := 700;

MoveCounter.Caption := 'Количество ходов';

i := 120;

SetLevel := 'Two';

Timer.Enabled := True;

MoveCounter.Visible := True;

Time.Visible := True;

n := 0;

NewGame;

end;

procedure TPicturesGame.HelpClick(Sender: TObject); // вызов справка

begin

ShellExecute(0, PChar('Open'),PChar('VeloLandHelp.chm'),nil,nil,SW\_SHOW);

end;

procedure TPicturesGame.AboutProgrClick(Sender: TObject); // вызов окна "О программе"

begin

AProgramm.ShowModal;

end;

procedure TPicturesGame.LevelThreeButtonClick(Sender: TObject); // выбор третьего уровня игры

begin

if Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger < 600 then // проверка пройден ли предыдущий уровень

begin

ShowMessage('Для прхождения этого уровня Вы должны пройти предыдущий!');

exit;

end;

LevelOneButton.Visible := False;

LevelTwoButton.Visible := False;

LevelThreeButton.Visible := False;

LevelFourButton.Visible := False;

Title.Visible :=False;

Pictures := TBitmap.Create;

// загрузить картинки из файла

Pictures.LoadFromFile('resources/game\_pictures/Level3.bmp');

HK := 128; // высота картинки

WK := 128; // ширина картинки

H := 6;

W := 6;

// установить размера поля

Field.Visible := True;

Field.Height := H \*HK;

Field.Width := W \* WK;

Field.Left := 600;

Field.Top := 2;

BackToMainMenu.Left := 700;

i := 180;

SetLevel := 'Three';

Timer.Enabled := True;

MoveCounter.Caption := 'Количество ходов';

Time.Visible := True;

MoveCounter.Visible := True;

n := 0;

NewGame;

end;

procedure TPicturesGame.LevelFourButtonClick(Sender: TObject); // выбор четвёртого уровня игры

begin

if Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger < 1200 then

begin

ShowMessage('Для прхождения этого уровня Вы должны пройти предыдущий!');

exit;

end;

LevelOneButton.Visible := False;

LevelTwoButton.Visible := False;

LevelThreeButton.Visible := False;

LevelFourButton.Visible := False;

Title.Visible := False;

Pictures := TBitmap.Create;

// загрузить картинки из файла

Pictures.LoadFromFile('resources/game\_pictures/Level4.bmp');

HK := 128; // высота картинки

WK := 128; // ширина картинки

H := 8;

W := 8;

// установить размера поля

Field.Visible := True;

Field.Height := H \*HK;

Field.Width := W \* WK;

Field.Left := 500;

Field.Top := 0;

BackToMainMenu.Left := 0;

i := 300;

SetLevel := 'Four';

MoveCounter.Caption := 'Количество ходов';

Timer.Enabled := True;

Time.Visible := True;

MoveCounter.Visible := True;

n := 0;

NewGame;

end;

procedure TPicturesGame.BackToMainMenuClick(Sender: TObject); // кнопка возвращения на главное меню

begin

PicturesGame.Close;

MainForm.Show;

end;

procedure TPicturesGame.FieldMouseDown(Sender: TObject; Button: TMouseButton; //нажатие на клетку поля

Shift: TShiftState; X, Y: Integer);

var

col\_: integer; // номер клетки по горизонтали

row\_: integer; //номер клетки по вертикали

begin

col\_ := Trunc(X/WK) + 1;

row\_ := Trunc(Y/HK) + 1;

if Pole[col\_,row\_] > 200 then

// щелчок в на месте одной из двух

// уже найденных парных картинок

exit;

// открытых клеток нет

if count = 0 then

begin

count := 1;

open1.col := col\_;

open1.row := row\_;

// клетка помечается как открытая

Pole[open1.col,open1.row] := Pole[open1.col,open1.row] + 100;

Kletka(open1.col,open1.row);

exit;

end;

// открыта одна клетка, надо открыть вторую

if count = 1 then begin

open2.col := col\_;

open2.row := row\_;

// если открыта одна клетка и щелчок сделан

// в этой клетке, то ничего не происходит

if (open1.col = open2.col) and (open1.row = open2.row)

then exit

else begin

count := 2; // теперь открыты две клетки

Moves := Moves + 1;

MoveCounter.Caption := 'Количество ходов: ' + IntToStr(Moves);

Pole[open2.col,open2.row] :=

Pole[open2.col,open2.row] + 100;

Kletka(open2.col,open2.row); // отрисуем вторую клетку

// проверим, открытые картинки одинаковые?

if Pole[open1.col,open1.row] = Pole[open2.col,open2.row] then

// открыты две одинаковые картинки

begin

n := n+1;

Update.Enabled := True; // запустить таймер

// процедура обработки события OnTimer

// "сотрет" две одинаковые картинки

end;

end;

exit;

end;

if count = 2 then

begin

// открыты 2 клетки с разными картинками

// закроем их и откроем новую, в которой

// сделан щелчок

// закрыть открытые клетки

Pole[open1.col,open1.row] := Pole[open1.col,open1.row] - 100;

Pole[open2.col,open2.row] := Pole[open2.col,open2.row] - 100;

Kletka(open1.col,open1.row);

Kletka(open2.col,open2.row);

// запись в open1 номера текущей клетки

open1.col := col\_;

open1.row := row\_;

count := 1; // счетчик открытых клеток

// открыть текущую клетки

Pole[open1.col,open1.row] := Pole[open1.col,open1.row] + 100;

Kletka(open1.col,open1.row);

end;

end;

procedure TPicturesGame.FieldPaint(Sender: TObject); // рисование поля

begin

ShowPole;

end;

// обработка события таймера

procedure TPicturesGame.UpdateTimer(Sender: TObject);

begin

// в массиве Pole клетки помечаются как совпавшие

Pole[open1.col,open1.row] := Pole[open1.col,open1.row] + 100;

Pole[open2.col,open2.row] := Pole[open2.col,open2.row] + 100;

count := 0;

// отрисовать клетки

Kletka(open2.col,open2.row);

Kletka(open1.col,open1.row);

// остановка таймера

Update.Enabled := False;

if n = Trunc(W\*H/2)

then // открыты все пары

begin

Field.Visible := False;

LevelOneButton.Visible := True;

LevelTwoButton.Visible := True;

LevelThreeButton.Visible := True;

LevelFourButton.Visible := True;

Title.Visible := True;

Timer.Enabled := False;

MoveCounter.Visible := False;

Time.Visible := False;

Reg.VeloLandUsers.Edit;

if (SetLevel = 'One') and (Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger < 200) then // начисление очков в зависимости от того, проходился ли уровень раньше

Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger := 200

else if (SetLevel = 'Two') and (Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger < 600) then

Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger := 600

else if (SetLevel = 'Three') and (Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger < 1200) then

Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger := 1200

else

Reg.VeloLandUsers.FieldByName('UserGameScore').AsInteger := 2000;

Reg.VeloLandUsers.Post;

SetLevel := '';

MoveCounter.Caption := 'Количество ходов';

ShowMessage('Вы успешно прошли уровень! Поздравляем!');

Moves := 0;

end;

end;

procedure TPicturesGame.TimerTimer(Sender: TObject); // таймер для ограничения по времени

var H,M,S:integer;

begin

i := i - 1;

H := I div 3600;

M := (I mod 3600) div 60;

S := (I mod 3600) mod 60;

Time.Caption := 'Оставшееся время:' + Format('%.2d:',[H]) + Format('%.2d:',[M]) + Format('%.2d',[S]);

if i = 0 then

begin

LevelOneButton.Visible := True;

LevelTwoButton.Visible := True;

LevelThreeButton.Visible := True;

LevelFourButton.Visible := True;

Timer.Enabled := False;

Time.Visible := False;

Field.Visible := False;

Title.Visible := True;

Moves := 0;

MoveCounter.Visible := False;

MoveCounter.Caption := 'Количество ходов';

SetLevel := '';

ShowMessage('Время вышло! Попробуйте ещё раз!');

end;

end;

procedure TPicturesGame.TitleColorTimer(Sender: TObject); // переливание цветов текста "Игра "Парные картинки с велосипедами"

begin

Title.Font.Color := rgb(random(255),random(255), random(255));

end;

end.

unit AboutProgram; // модуль окна "О программе"

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.StdCtrls;

type

TAProgramm = class(TForm)

AppImg: TImage;

Version: TLabel;

Developer: TLabel;

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

AProgramm: TAProgramm;

implementation

{$R \*.dfm}

procedure TAProgramm.FormCreate(Sender: TObject); // загрузка информации "О программе"

begin

AppImg.Picture.LoadFromFile('resources/app\_icon/app\_icon.png');

AppImg.AutoSize := True;

AProgramm.Position := poDesktopCenter;

Version.Font.Name := 'Minecraft Rus Regular';

Version.Font.Size := 14;

Version.Caption := 'Версия: Планета велосипедов 1.0';

Developer.Font.Name := 'Minecraft Rus Regular';

Developer.Font.Size := 14;

Developer.Caption := 'Разработчик: Сырель Владислав, учащийся УО ГГПК группы ПЗТ-34';

end;

end.