**Содержание**

[Введение 2](#_Toc107797630)

[1 Анализ задачи 4](#_Toc107797631)

[1.1 Постановка задачи 4](#_Toc107797632)

[1.2 Инструменты разработки 5](#_Toc107797633)

[1.3 Требования к приложению 6](#_Toc107797634)

[2 Проектирование задачи 8](#_Toc107797635)

[2.2 Процессы 8](#_Toc107797636)

[2.3 Описание внешнего пользовательского интерфейса 9](#_Toc107797637)

[3 Реализация 12](#_Toc107797638)

[3.1 Реализация проекта 12](#_Toc107797639)

[3.1.1 Структура программы 12](#_Toc107797640)

[3.1.2 Структура и описание процедур и функций пользователя 12](#_Toc107797641)

[3.1.3 Описание использованных компонентов 14](#_Toc107797642)

[3.2 Спецификация программы 15](#_Toc107797643)

[4 Тестирование 17](#_Toc107797644)

[5 Руководство пользователя 19](#_Toc107797645)

[5.1 Общие сведения о программном продукте 19](#_Toc107797646)

[5.2 Инсталляция 19](#_Toc107797647)

[5.3 Выполнение программы 19](#_Toc107797648)

[5.3.1 Запуск программы 19](#_Toc107797649)

[5.3.2 Инструкции по работе с программой 20](#_Toc107797650)

[5.3.3 Завершение работы с программой 26](#_Toc107797651)

[5.4 Использование системы справочной информации 27](#_Toc107797652)

[Заключение 28](#_Toc107797653)

[Список использованных источников 29](#_Toc107797654)

[Приложение А 30](#_Toc107797655)

[Листинг программы 30](#_Toc107797656)

# **Введение**

На курсовое проектирование была поставлена задача разработать игровое приложение «Лабиринт».

Цель курсового проекта заключается в изучении основ проектирования игр, изучении основ создания 8-битных персонажей, практике работы с графическими объектами.

Создаваемое игровое приложение будет рассчитано для пользователей любого возраста и рода занятий.

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**

Темой данного курсового проекта является «Игровое приложение «Лабиринт».

Цель курсового проекта заключается в изучении основ проектирования игр, изучении основ создания 8-битных персонажей, практике работы с графическими объектами.

Создаваемое игровое приложение будет рассчитано для пользователей любого возраста и рода занятий.

За основу программы следует взять из соответствующих источников (сайтов, статей, энциклопедий) наиболее полную, достоверную и точную информацию о проектировании игр, о создании 8-битных изображений, организации работы с графическими объектами. При реализации проекта разработчик должен стараться сделать его простым и максимально доступным для пользователей. Приложение должно располагать своим интерфейсом и функционалом к пользователю, чтобы ему было комфортно.

Данный проект должен стать общедоступным для всех пользователей. В поставленной задаче необходимо реализовать максимально простой пользовательский интерфейс, позволяющий использовать проект пользователю, не обладающему дополнительными знаниями электронно-вычислительной техники.

В разрабатываемой программе будут использоваться три вида данных. К входной информации можно отнести данные пользователя при регистрации и авторизации. К выходной – сообщение с результатами прохождения игры.

Постоянной информацией в проекте будут являться графические изображения, медиа файлы и текстовые сообщения.

Программный продукт предоставит функционал пользователям, которые будут обладать равными правами в игровом приложении.

Разрабатываемый программный продукт должен позволять выполнять следующие действия:

* прохождение уровней;
* просмотр справки;
* просмотр пункта «о программе»;
* возможность управлять персонажем;
* вывод результатов игры.

# **1.2 Инструменты разработки**

1 Delphi 10 – главная среда разработки программы, в которой прописывается код и создаются формы.

2 Aseprite – это проприетарный редактор изображений с доступным исходным кодом, предназначенный в первую очередь для рисования и анимации в пиксельной графике.

3 Adobe Photoshop 2021 – редактор для редактирования изображений. Необходим для оформления некоторых элементов интерфейса.

4 Microsoft Word 2016 – редактор текста для написания документации.

5 Microsoft Power Point 2016 – программа для создания презентации.

6 Google-браузер – для нахождения информации о создании игровых приложений в среде Delphi 10, поиск информации о создании 8-битных персонажей, графических изображений.

7 Sublime text – проприетарный текстовый редактор. Поддерживает плагины на языке программирования Python. Разработчик позволяет бесплатно и без ограничений ознакомиться с продуктом, однако программа уведомляет о необходимости приобретения лицензии.

8 Dr.Explain – программа для быстрого создания файлов справки (help-файлов), справочных систем, on-line руководств пользователя, пособий и документации к программному обеспечению, изделиям, техническим и бизнес-системам

Для оптимального функционирования разрабатываемого приложения необходимо наличие у компьютера следующих параметров:

* 1.8 GHz процессор (или более быстрый);
* 1 GB RAM;
* от 75 MB доступного места на жестком диске.

Разработка ведется на ноутбуке HP. Со следующими параметрами:

* процессор Intel Core i5;
* объем ОЗУ 4 гб;
* объем места на HDD – 150 гб ;
* видеоподсистема 1366x768 точек с глубиной цвета 16 Bit;
* ОС – Windows 10.

Как видно разрабатываемое приложение не очень требовательно к аппаратным ресурсам, что, является большим плюсом.

# **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;

* небольшое количество информации на каждой форме;
* качественные графические 8-битные изображения.
* ненавязчивое звуковое сопровождение.
* Так же следует обратить внимание на такие моменты:
* оптимально подобранные размеры формы для удобного восприятия игры;
* визуально понятный и приятный глазу интерфейс.

Минимальные системные требования:

* 1.8 GHz процессор (или более быстрый);
* 1 GB RAM;
* от 75 MB доступного места на жестком диске.

Рекомендуемые:

* двухъядерный AMD Athlon или аналогичный процессор Intel (или более быстрый);
* 2 GB RAM;
* от 75 MB доступного места на жестком диске.

Требования к надежности:

Специальных требований к надежности не предъявляется.

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбое (не крахом) операционной системы, не должно превышать 30-ти минут при условии соблюдения условий эксплуатации технических и программных средств.

# **2 Проектирование задачи**

**2.1 Организация данных**

Основным средством хранения информации в приложении являются bmp- изображения, а также png-изображения для хранения текстур всех персонажей, фоновых изображений и текстур для кнопок. А также используются языки разметки html и css для хранения информации о бонусной системе. Информация о зарегистрированных пользователях храниться в текстовом файле. Звуковое оформление игрового процесса осуществляется при помощи файла формата mp3.

Система справочной информации представлена файлом справки, который содержит информацию о приложении и правилах его использования.

Основными функциями игрового приложения являются:

* просмотр информации о бонусной системе;
* вхождение в игровую сессию;
* прохождение уровня для перехода к следующему.

# **2.2 Процессы**

Согласно всем перечисленным требованиям и указаниям, которые были рассмотрены в разделе «Анализ задачи», было определено, чем конкретно должна заниматься разрабатываемая программа. Главной её задачей является прохождение игровой сессии.

Для реализации задач используются процедуры и функции. С помощью процедуры, например, будет осуществляться переход на следующий игровой уровень.

Персонажи, лабиринт и игровой уровень генерируются по определённым координатам. На форме с игрой загружаются изображения фоновых текстур, изображения текстур лабиринта, протагонист и призраки. Далее, после того, как пользователь начнёт игру, он может пройти уровень игры для того, чтобы перейти к новому уровню, в случае же смерти пользователя игровая сессия останавливается и выводится сообщение с результатами игры.

После прохождения уровня, пользователю высветится сообщение о переходе на следующий игровой уровень.

# **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 разрабатывает данную программу, какие компоненты использованы и какие методы автоматизированы.

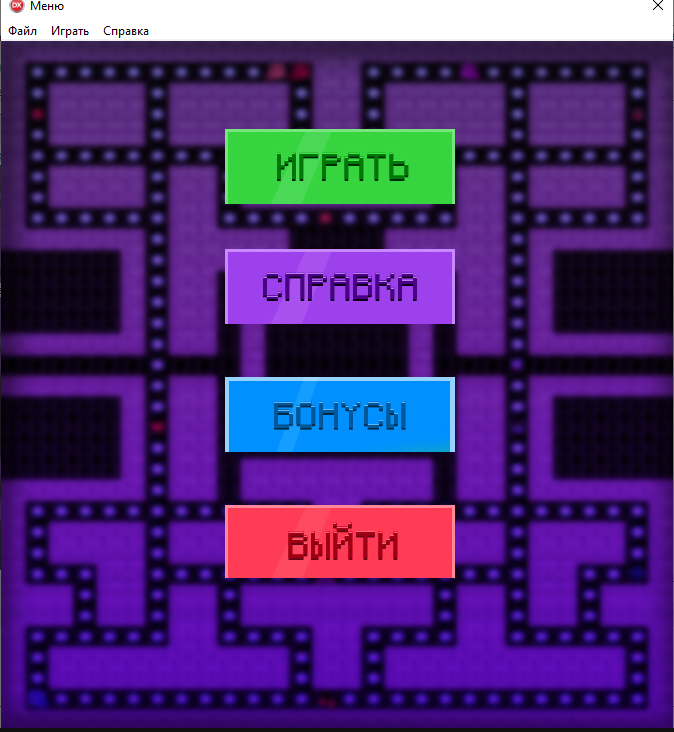
Особое внимание уделяется интерфейсу. Внешний вид программы организован так, чтобы пользователь понял, что от него требуется. Таким образом, ключевые кнопки графически выделены для визуализации интерактивных возможностей приложения (Рисунок 1).

Рисунок 1 – Визуализация интерактивных возможностей приложения

Для организации эффективной работы пользователя создаётся целостное приложение данной предметной области, в котором все компоненты приложения сгруппированы по функциональному назначению, со всех второстепенных форм можно вернуться на главную (Рисунок 2). При этом же главной целью является удобный и приятный графический интерфейс пользователя.

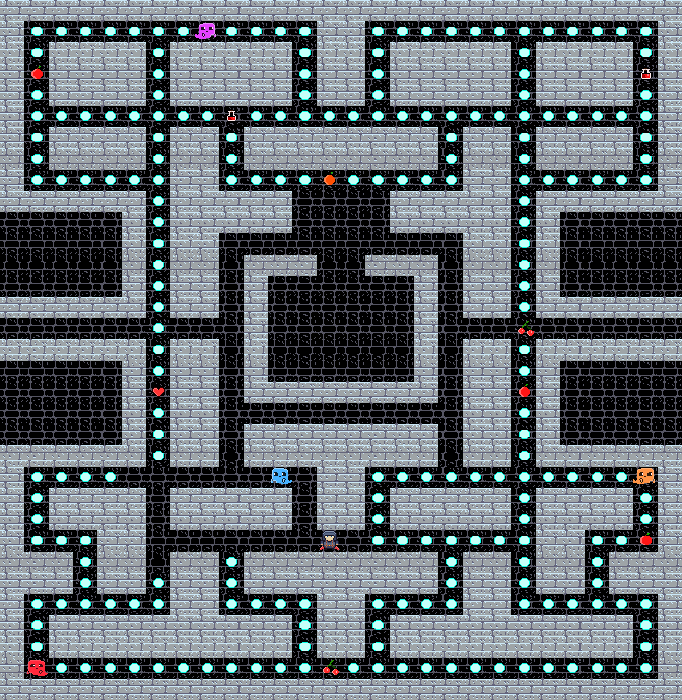


Рисунок 2 – Графический интерфейс игры

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 3.



Рисунок 3 – Структура навигации по проекту

# **3 Реализация**

# **3.1 Реализация проекта**

# **3.1.1 Структура программы**

Данный курсовой проект содержит 6 модулей. Далее рассмотрим назначение каждого модуля:

1 Модуль Unit1 – отвечает за игровой процесс игрового приложения;

2 Модуль Unit2 – представляет собой загрузочное меню;

3 Модуль Unit3 – данные модуль отвечает за игровое меню;

4 Модуль Unit4 – отвечает за вход пользователя в систему, при успешном входе в аккаунт пользователь может перейти на модуль игрового меню;

5 Модуль Unit5 – отвечает за регистрацию пользователя, после регистрации пользователь может перейти на модуль игрового меню;

6 Модуль Unit6 – данный модуль отвечает за отображения бонусной системы игрового приложения.

# **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 | 2 | 3 | 4 | 5 |
| 1 | procedure TimerImage1Timer(Sender: TObject); | Unit2 | TimerImage1 | Переход на следующую форму |
| 2 | procedure FormClose(Sender: TObject; var Action: TCloseAction); | Unit2 | Form2 | Закрытие формы |
| 3 | procedure Image3Click(Sender: TObject); | Unit4 | Image3 | Вход в аккаунт |
| 4 | procedure Image2Click(Sender: TObject); | Unit4 | Image2 | Переход на форму регистрации |
| 5 | procedure FormClose(Sender: TObject; var Action: TCloseAction); | Unit4 | Form4 | Закрытие формы |
| 6 | procedure Image2Click(Sender: TObject); | Unit5 | Image2 | Регистрация |
| 7 | procedure FormClose(Sender: TObject; var Action: TCloseAction); | Unit5 | Form5 | Закрытие формы |
| 8 | procedure TForm3.Image2Click(Sender: TObject); | Unit3 | Image2 | Переход на игровое поле |
| 9 | procedure Image3Click(Sender: TObject); | Unit3 | Image3 | Открывает справочный файл |
| 10 | procedure Image4Click(Sender: TObject); | Unit3 | Image4 | Закрытие из игрового приложения |
| 11 | procedure Image5Click(Sender: TObject); | Unit3 | Image5 | Просмотр информации о бонусной системе в игровом приложении |
| 12 | procedure N2Click(Sender: TObject); | Unit3 | MainMenu1 | Закрытие игрового приложения |
| 13 | procedure N3Click(Sender: TObject); | Unit3 | MainMenu1 | Открывает справочный файл |
| 14 | procedure N4Click(Sender: TObject); | Unit3 | MainMenu1 | Переход на игровое поле |
| 15 | procedure FormClose(Sender: TObject; var Action: TCloseAction); | Unit3 | Form3 | Закрытие игрового приложения |
| 16 | procedure FormCreate(Sender: TObject); | Unit1 | Form1 | Создание формы |
| 17 | procedure Timer1Timer(Sender: TObject); | Unit1 | Timer1 | Отображение игрового приложения |
| 18 | procedure FormKeyDown(Sender: TObject; var Key: Word; Shift: TShiftState); | Unit1 | Form1 | Отвечает за движения главного героя игрового приложения |
| 19 | procedure FormClose(Sender: TObject; var Action: TCloseAction); | Unit1 | Form1 | Закрытие игрового приложения |
| 20 | procedure lvl2; | Unit1 | Form1 | Осуществляет переход на 2 уровень |
| 21 | procedure lvl3; | Unit1 | Form1 | Осуществляет переход на 3 уровень |
| 22 | procedure chcolor; | Unit1 | Form1 | Изменяет текстуру фоновых графических изображений |
| 23 | procedure FormCreate(Sender: TObject); | Unit6 | Form6 | Создание формы, предназначенной для отображения бонусной системы игрового приложения |
| 24 | procedure FormClose(Sender: TObject; var Action: TCloseAction); | Unit6 | Form6 | Закрытие игрового приложения |

# **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 | TMainMenu | Form3 | Используется для создания  главного меню проекта |
| 2 | TImage | Form3,  Form2,  Form5,  Form4 | Используется как фоновое изображение, предназначен для кнопок |
| 3 | TTimer | Form1,  Form2 | Используется для определения длительности  загрузочного экрана, а также предназначен для отображения игрового поля |
| 4 | TWebBrowser | Form6 | Используется для отображения html-страниц  с информацией о бонусной системе игрового приложения |
| 5 | TLabel | Form4,  Form5 | Отображение надписей на  форме |
| 6 | TGauge | Form2 | Предназначен для отображения процесса загрузки |

# **3.2 Спецификация программы**

Точное название проекта и его состав приводится в Таблице 3.

Таблица 3 – Спецификация программы

|  |  |  |
| --- | --- | --- |
| № | Имя файла | Назначение |
| 1 | Labyrinth.exe | Исполняемый файл проекта |
| 2 | Labyrinth.dpr | Файл проекта, связывает все файлы, из которых состоит приложение |
| 3 | Unit1.pas | Файл программного модуля для формы Form1 |
| 4 | Unit2.pas | Файл программного модуля для формы  Form2 |
| 5 | Unit3.pas | Файл программного модуля для формы  Form3 |
| 6 | Unit4.pas | Файл программного модуля для формы  Form4 |
| 7 | Unit5.pas | Файл программного модуля для формы Form5 |
| 8 | Unit6.pas | Файл программного модуля для формы Form6 |
| 9 | Labyrinth.exe | Установочный файл |
| 10 | Delphi.chm | Файл справки, содержит помощь по работе с  программой |
| 11 | Unit1.dfm | Форма, предназначенная для отображения игрового  поля |
| 12 | Unit2.dfm | Форма на которой располагается загрузочное меню |
| 13 | Unit3.dfm | Форма на которой располагается игровое меню |
| 14 | Unit4.dfm | Форма, предназначенная для входа в аккаунт |
| 15 | Unit5.dfm | Форма, предназначенная для осуществления регистрации |
| 16 | Unit6.dfm | Форма, предназначенная для отображения информации о бонусной системе игрового приложения |
| 17 | Index.html | Файл, предназначенный для отображения информации о бонусной системе игрового приложения |
| 18 | Style1.css | Файл, предназначенный для описания внешнего вида документа |

# **4 Тестирование**

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

Отчёт о результатах тестирования предоставлен в Таблице 4.

Таблица 4 – Отчёт результатах тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Тест | Ожидаемый результат | Физический результат | Результат тестирования |
| 1 | Проверка входа пользователя в систему | Открытие главного окна | Открытие главного окна | Выполнено |
| 2 | Проверка кнопки «Бонусы» | В WebBrowser отображается информация о бонусной системе | В WebBrowser отображается информация о бонусной системе | Выполнено |
| 3 | Проверка кнопки «Играть» | Отображение игрового поля | Отображение игрового поля | Выполнено |
| 4 | Проверка кнопки «Справка» | Открытие справочного файла | Открытие справочного файла | Выполнено |
| 5 | Проверка кнопки «Выйти» | Закрытие игрового приложения | Закрытие игрового приложения | Выполнено |
| 6 | Проверка пункта меню «Играть» | Отображение игрового поля | Отображение игрового поля | Выполнено |
| 7 | Проверка пункта меню «Справка» | Открытие справочного файла | Открытие справочного файла | Выполнено |
| 8 | Проверка пункта меню «Выйти» | Закрытие игрового приложения | Закрытие игрового приложения | Выполнено |
| 9 | Проверка кнопки «Регистрация» | Добавление информации о пользователе в текстовый файл | Добавление информации о пользователе в текстовый файл | Выполнено |
| 10 | Проверка кнопки «Вход» | Если пользователь существует, то осуществляется открытие главного меню | Если пользователь существует, то осуществляется открытие главного меню | Выполнено |
| 11 | Проверка перехода на следующий уровень | Если все очки собраны, осуществляется переход на следующий уровень, изменение графических изображений, текстур, отображаются сообщения | Если все очки собраны, осуществляется переход на следующий уровень, изменение графических изображений, текстур, отображаются сообщения | Выполнено |
| 12 | Проверка на поражение | Отображение сообщений, закрытие игрового приложения | Отображение сообщений, закрытие игрового приложения | Выполнено |
| 13 | Проверка на прохождение игры | Отображение сообщений, закрытие игрового приложения | Отображение сообщений, закрытие игрового приложения | Выполнено |
| 14 | Проверка на непроходимость текстур лабиринта | Персонажи не проходят через текстуры лабиринта | Персонажи не проходят через текстуры лабиринта | Выполнено |
| 15 | Проверка на подсчёт очков силы | Увеличение очков силы | Увеличение очков силы | Выполнено |
| 16 | Проверка бонусной системы | Изменение скорости передвижения персонажей, изменение графических изображений, текстур, увеличение очков силы | Изменение скорости передвижения персонажей, изменение графических изображений, текстур, увеличение очков силы | Выполнено |

Когда я делал свой программный продукт, я столкнулся с рядом проблем, которые исправил. Например: я не знал как осуществить переход на следующий уровень, но проблема решена, и пользователь теперь может проходить уровни и наслаждаться игрой. Также была проблема с позиционированием элементов на html-странице, однако проблема решена и пользователь может доступно получать информацию о бонусной системе игрового приложения.

Элементы программы были проверены, и было установлено, что все они работают правильно и выполняют задачи, указанные в процедурах.

При использовании данного игрового приложения пользователь не столкнется с ошибками системы.

**5 Руководство пользователя**

**5.1 Общие сведения о программном продукте**

Разрабатываемое игровое приложениеявляется продуктом сферы компьютерных развлечений. Игра относится к жанру аркадных компьютерных игр – одному из самых популярных жанров. Игра не имеет возрастных ограничений и может служить отличным способом времяпрепровождения.

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

Тестирование проводилось на разных классах ЭВМ и работать с данной программой было комфортно. Программа разработана на ПК со следующими характеристиками:

* процессор AMD Dual-Core A4-4300M (2.5 - 3.0 ГГц);
* объем ОЗУ 4Гб;
* графический адаптер AMD Radeon HD 7650M;
* операционная система Windows 10.

**5.2 Инсталляция**

Для того, чтобы установить программу необходимо запустить файл Setup.exe. Появится окно установки игрового приложения «Лабиринт».

При нажатии на кнопку Далее появляется окно выбора каталога установки.

Следуем инструкции и устанавливаем приложение.

**5.3 Выполнение программы**

**5.3.1 Запуск программы**

Данную программу можно запустить различными способами.

Первым из них является запуск с помощью ярлыка на рабочем столе. Необходимо дважды щелкнуть левой кнопкой мыши на ярлыке с названием Лабиринт.

Вторым способом является запуск из каталога, в который устанавливалось приложение (по умолчанию C:\Program Files\Labyrinth\Лабиринт.

По подготовленным тестам будет осуществляться функциональное и полное тестирование программного продукта. Отчет о результатах тестирования будет представлен в 4 разделе пояснительной записки.

**5.3.2 Инструкции по работе с программой**

После запуска на экране появляется загрузочное меню. (Рисунок 1).

Рисунок 1 – Загрузочное меню

После окончания загрузки открывается форма авторизации. Где гость может осуществить авторизацию или перейти на форму регистрации, нажав на кнопку «Регистрация». (Рисунок 2).

Рисунок 2 – Экран авторизации

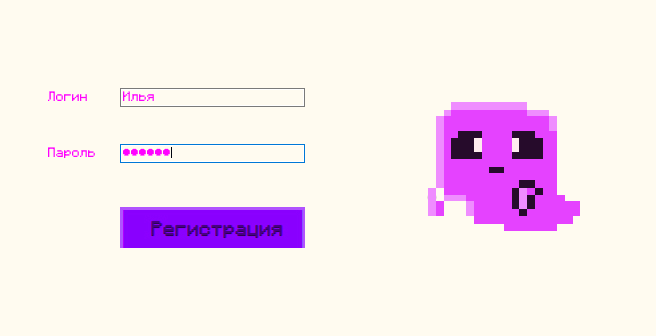
При нажатии на кнопку «Регистрация» будет осуществлён переход на форму регистрации. Где гость может ввести свой логин и пароль. (Рисунок 3).

Рисунок 3 – Экран регистрации

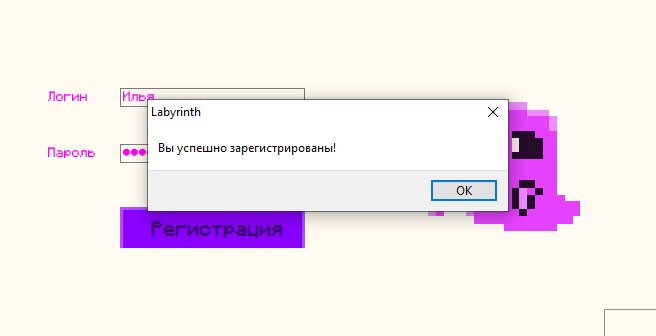
При корректном заполнении данных, будет выведено на экран сообщение о успешной регистрации. (Рисунок 4)

Рисунок 4 – Подтверждение регистрации

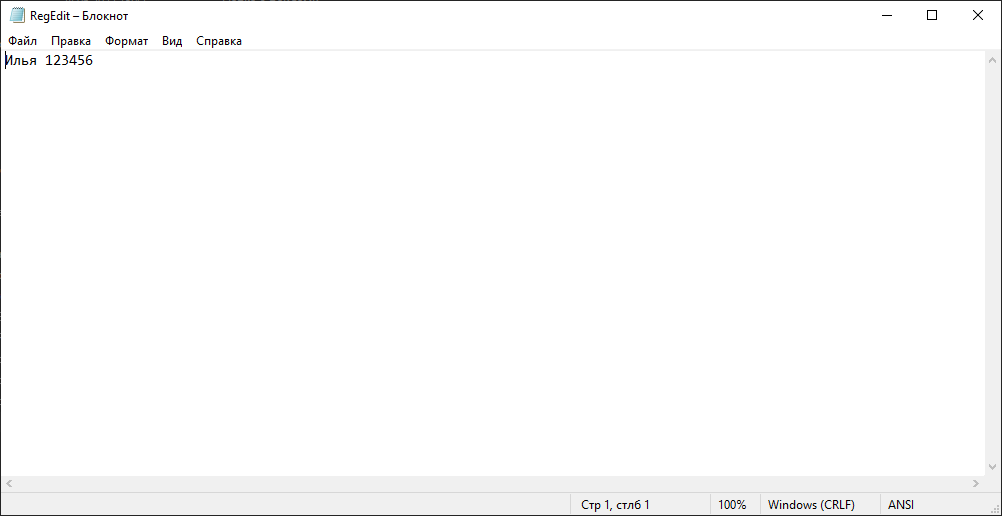
Также данные пользователей будут храниться в текстовом файле RegEdit.txt. (Рисунок 5)

Рисунок 5 – Добавление пользователя в текстовый файл

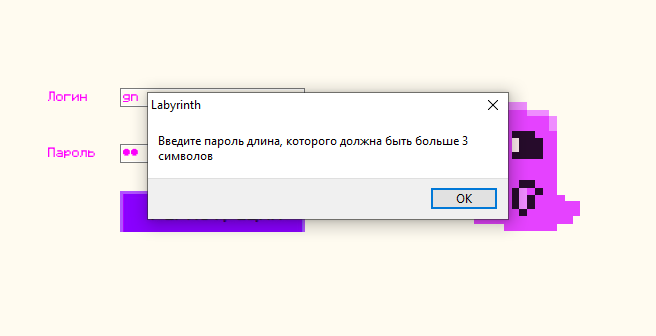
При некорректной регистрации будет выведено сообщение о повторном заполнении данных. (Рисунок 6).

Рисунок 6 – Вывод сообщение при некорректной регистрации

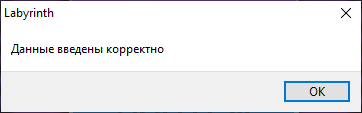
**При корректном заполнении данных на форме авторизации также будет выведено сообщение об успешной авторизации. (Рисунок 7).

Рисунок 7 – Подтверждение авторизации

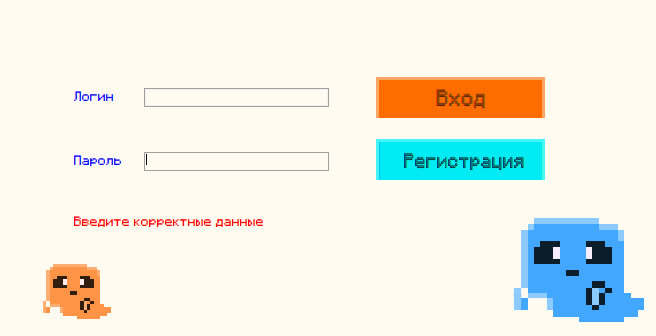
Если же пользователь введёт некорректные данные при авторизации будет выведено сообщение о повторном заполнении данных. (Рисунок 8).

Рисунок 8 – Некорректная авторизация

Соответственно, при успешной регистрации или авторизации будет осуществлён переход на форму главного меню. (Рисунок 9).

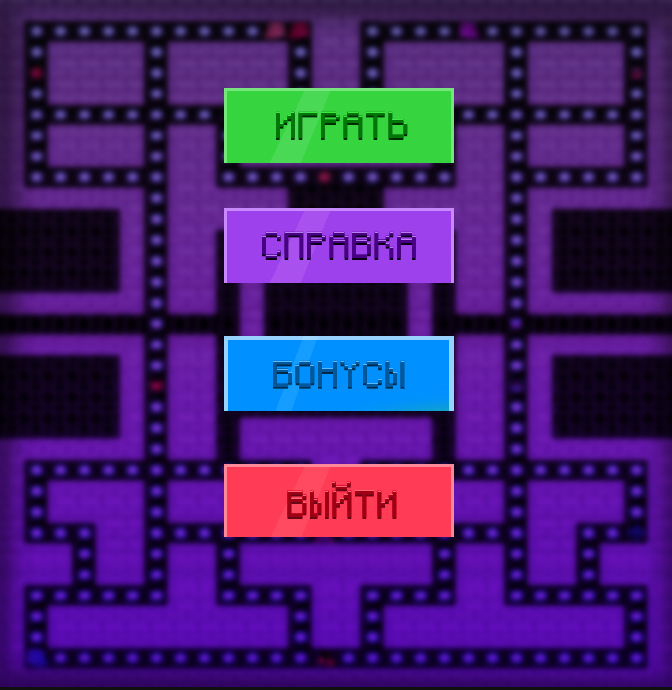


Рисунок 9 – Главное меню

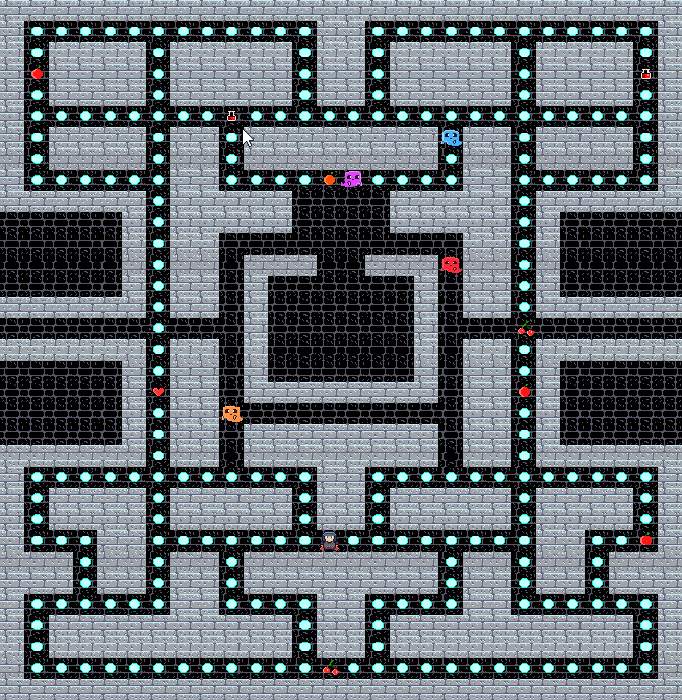
При нажатии на кнопку «Играть» будет запущена игровая сессия. (Рисунок 10).

Рисунок 10 – Игровая сессия

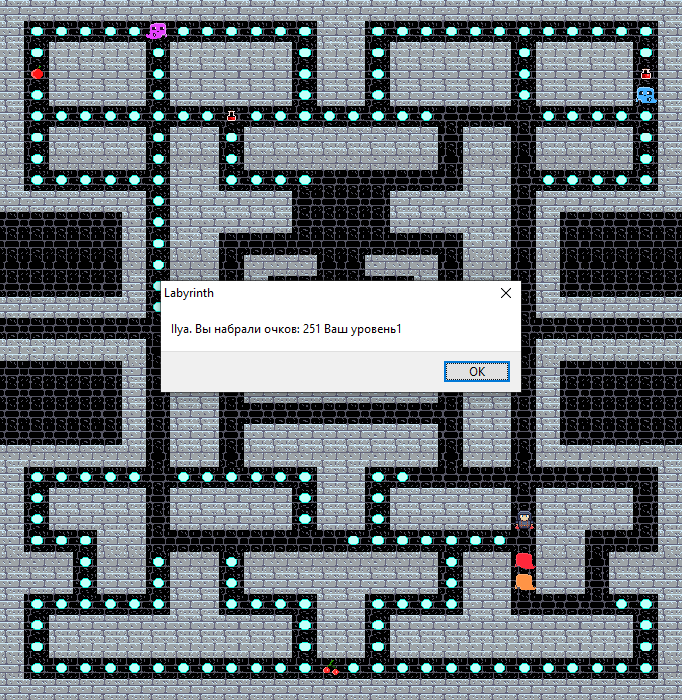
После окончания игровой сессии будет выведено сообщение с результатами. (Рисунок 11).

Рисунок 11 – Окончание игры

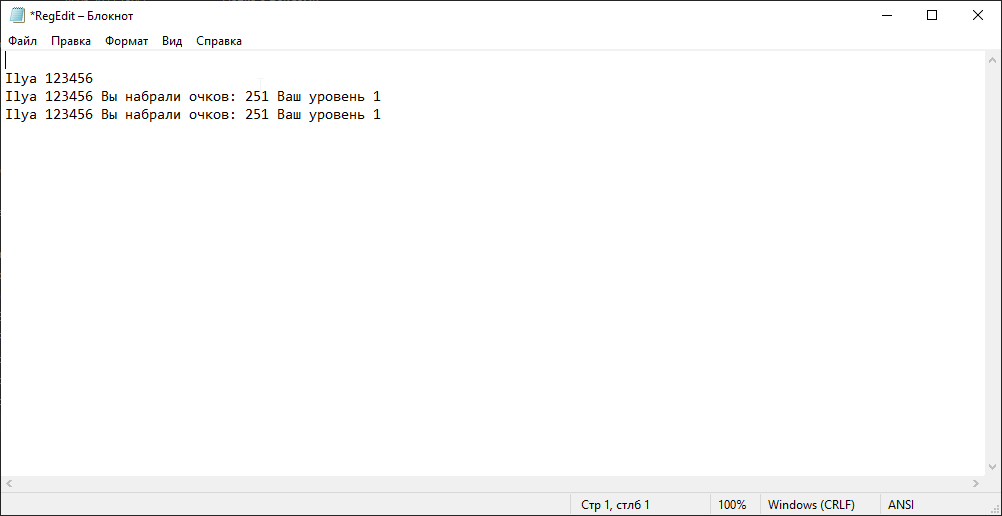
Результаты игровой сессии будет записываться в текстовый файл RegEdit.txt. (Рисунок 12).

Рисунок 12 – Запись в файл результатов игровой сессии

При нажатии на кнопку «Справка» открывается справочная система. (Рисунок 13).

Рисунок 13 – Справочная система

При нажатии на кнопку «Бонусы» открывается html документ с информацией о бонусной системе игрового приложения. (Рисунок 13).

Рисунок 14 – Бонусная система

При нажатии на кнопку «Выход» игровое приложение закрывается.

**5.3.3 Завершение работы с программой**

Завершить работу с программой можно несколькими способами:

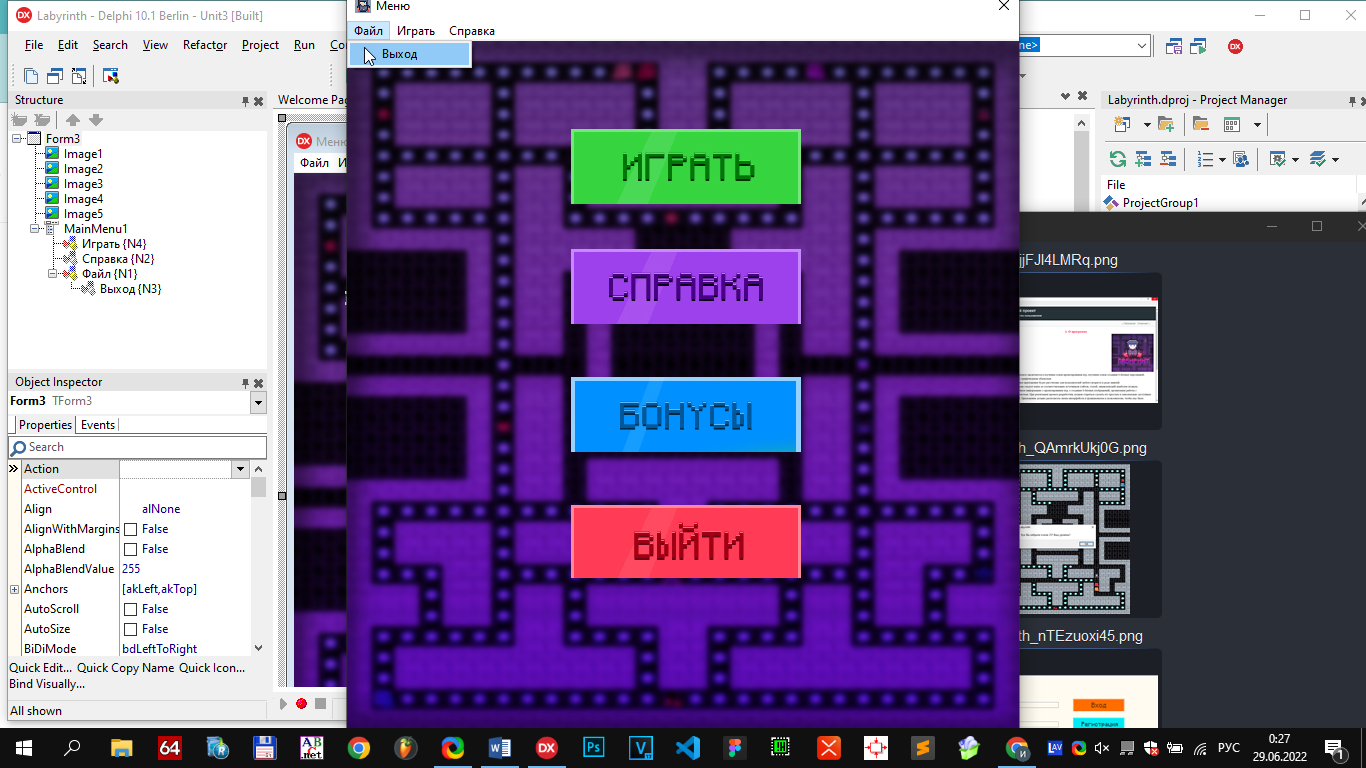
1. Нужно нажать на крестик в углу формы.
2. Также можно закрыть игровое приложение нажав пункт файл – Выход или нажать на кнопку главного меню «Выход»

Рисунок 15 – Выход из приложения

**5.4 Использование системы справочной информации**

Справочную систему можно запустить с помощью пункта меню Справка или нажать на кнопку главного меню «Справка». (Рисунок 15).

Рисунок 16 – Запуск справочной системы

## **Заключение**

Цель курсового проекта заключается в изучении основ проектирования игр, изучении основ создания 8-битных персонажей, практике работы с графическими объектами. Изучение основ создание аркадных игр.

В поставленной задаче был реализован простой, понятный, удобный, интересный, пользовательский интерфейс.

В ходе тестирования все исключительные ситуации были обработаны. Проект работает без сбоев и ошибок.

Продукт имеет дальнейшее развитие, так как на данный момент функционал недостаточно большой. Можно развить данный программный продукт, добавив новые уровни, новые графические изображения, текстуры. Несмотря на все это, данный программный продукт имеет законченный вид.

Исходя из этого, можно сделать вывод, что программа реализована успешно, а главное были приобретены новые знания в сфере создания 8-битных персонажей, текстур, игровых уровней и в целом в создании игровых приложений.

**Список использованных источников**

1. Информация о классе TBitmap [электронный ресурс]. – Электронные данные. – Режим доступа: <https://bdrc.ru/articles/delphi-7/urok-31-klass-tbitmap.html>
2. Информация о создании 8-битных персонажей [электронный ресурс]. – Электронные данные. – Режим доступа: <https://dtf.ru/craft/121713-osnovy-pikselarta-chast-1>
3. Видеоинформация о создании 8-битных персонажей в редакторе изображений Aseprite [электронный ресурс]. – Электронные данные. – Режим доступа: <https://youtu.be/beV7kVOrhH8>
4. Основы создания игр при помощи языка программирования Delphi [электронный ресурс]. – Электронные данные. – Режим доступа: <https://delphischool.3dn.ru/load/delphi_i_igry/sozdanie_igr_na_delphi/21>

## 

## **Приложение А**

## **Листинг программы**

## unit Unit1;

## interface

## uses

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

## Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.MPlayer, unit2, unit4;

## type

## TKn = record

## X,Y,HP,Rotate,Anim,Score,Level,Time, Speed:Integer;

## end;

## TGhost = record

## X,Y,Rotate:Integer;

## Visible, Angry:Boolean;

## end;

## TForm1 = class(TForm)

## Timer1: TTimer;

## MediaPlayer1: TMediaPlayer;

## procedure FormCreate(Sender: TObject);

## procedure Timer1Timer(Sender: TObject);

## procedure FormKeyDown(Sender: TObject; var Key: Word; Shift: TShiftState);

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

## private

## { Private declarations }

## public

## { Public declarations }

## end;

## var

## Form1: TForm1;

## path:String;

## map:array[0..27, 0..32] of integer;

## Buf, Wall, Wall2, Bg, Wall3, freeze:TBitmap;

## i,j:integer;

## Power:array[1..13] of TBitmap;

## Kn:array[1..4, 1..3] of TBitmap;

## Gh:array[0..3] of TGhost;

## Ghosts:array[0..3, 1..5] of TBitmap;

## Knight:TKn;

## procedure lvl2;

## procedure lvl3;

## implementation

## uses unit5;

## {$R \*.dfm}

## procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);

## begin

## Application.Terminate;

## MediaPlayer1.Destroy;

## end;

## procedure TForm1.FormCreate(Sender: TObject);

## begin

## path:=extractFileDir(Application.ExeName);//получение пути к файлу

## MediaPlayer1.FileName:= (path+'\img\pixel.mp3');

## MediaPlayer1.Open;

## MediaPlayer1.AutoRewind := True;

## MediaPlayer1.Play;

## //создание буфера

## Buf:=TBitmap.Create;

## Buf.Width:=672;

## Buf.Height:=792;

## //фон

## Bg:=TBitmap.Create;

## Bg.Transparent:=True;

## Bg.LoadFromFile(path+'\img\bg.bmp');

## Wall3:=TBitmap.Create;

## Wall3.LoadFromFile(path+'\img\wall3.bmp');

## //создание стены

## Wall:=TBitmap.Create;

## Wall.LoadFromFile(path+'\img\wall.bmp');

## //изменение цвета

## freeze:=TBitmap.Create;

## freeze.LoadFromFile(path+'\img\wall4.bmp');

## //стена второго уровня

## Wall2:=TBitmap.Create;

## Wall2.LoadFromFile(path+'\img\wall2.bmp');

## //загрузка фото очков силы

## for i := 1 to 13 do

## begin

## Power[i]:=TBitmap.Create;

## Power[i].LoadFromFile(path+'\img\'+inttostr(i)+'.bmp');

## Power[i].Transparent:=True;

## end;

## for i := 1 to 4 do

## for j := 1 to 3 do

## begin

## Kn[i,j]:=TBitmap.Create;

## Kn[i,j].Transparent:=True;

## Kn[i,j].LoadFromFile(path+'\img\k'+inttostr(i)+inttostr(j)+'.bmp');

## end;

## for i := 0 to 3 do

## for j := 1 to 5 do

## begin

## Ghosts[i,j]:=TBitmap.Create;

## Ghosts[i,j].Transparent:=True;

## Ghosts[i,j].LoadFromFile(path+'\img\p'+inttostr(i)+inttostr(j)+'.bmp');

## end;

## //рыцарь

## Knight.X:=13;

## Knight.Y:=25;

## Knight.HP:=2;

## Knight.Rotate:=1;

## Knight.Anim:=1;

## Knight.Score:=0;

## Knight.Level:=1;

## Knight.Time:=-1;

## Knight.Speed:=Timer1.Interval;

## //призраки

## for i := 0 to 3 do

## begin

## Gh[i].Rotate:=2;

## Gh[i].Visible:=True;

## Gh[i].Angry:=True;

## end;

## Gh[0].X:=11;

## Gh[0].Y:=11;

## Gh[1].X:=12;

## Gh[1].Y:=11;

## Gh[2].X:=15;

## Gh[2].Y:=11;

## Gh[3].X:=16;

## Gh[3].Y:=11;

## begin

## //обнуление блок равен 0

## for i := 0 to 27 do

## for j := 0 to 32 do

## map[i,j]:=0;

## //создание боковых блоков(верхняя стена и нижняя)

## for i := 0 to 27 do

## begin

## //верхняя горизонталь

## map[i,0]:=-1;

## //нижняя горизонталь

## map[i,32]:=-1;

## end;

## for j := 0 to 9 do

## begin

## //верхние вертикалии

## map[0,j]:=-1;

## map[27,j]:=-1;

## end;

## for j := 0 to 11 do

## begin

## //нижние вертикалии

## map[0,32-j]:=-1;

## map[27,32-j]:=-1;

## end;

## for i := 0 to 5 do

## begin

## //стены по горизонтали

## map[i,9]:=-1;

## map[27-i,9]:=-1;

## map[i,14]:=-1;

## map[27-i,14]:=-1;

## map[i,21]:=-1;

## map[27-i,21]:=-1;

## map[i,16]:=-1;

## map[27-i,16]:=-1;

## end;

## for j := 0 to 5 do

## begin

## //средние вертикали

## map[5,j+9]:=-1;

## map[22,j+9]:=-1;

## map[5,j+16]:=-1;

## map[22,j+16]:=-1;

## map[7,j+16]:=-1;

## map[8,j+16]:=-1;

## map[20,j+16]:=-1;

## map[19,j+16]:=-1;

## end;

## //

## //центровая линия

## for i:=0 to 1 do

## for j:=1 to 4 do

## map[i+13,j]:=-1;

## //левый и правый квадрат(маленькие)

## for i := 0 to 3 do

## for j := 0 to 2 do

## begin

## map[i+2,j+2]:=-1;

## map[25-i,j+2]:=-1;

## end;

## //квадраты большие

## for i := 0 to 4 do

## for j := 0 to 2 do

## begin

## map[i+7,j+2]:=-1;

## map[20-i,j+2]:=-1;

## end;

## //квадраты меньше

## for i := 0 to 3 do

## for j := 0 to 1 do

## begin

## map[i+2,j+6]:=-1;

## map[25-i,j+6]:=-1;

## end;

## //преграды

## for j := 0 to 8 do

## for i := 0 to 1 do

## begin

## map[i+7,j+6]:=-1;

## map[i+19,j+6]:=-1;

## end;

## for i := 0 to 2 do

## for j := 0 to 1 do

## begin

## map[i+9,j+9]:=-1;

## map[18-i,j+9]:=-1;

## end;

## //двойная линия по центру

## for i := 0 to 7 do

## for j := 0 to 1 do

## begin

## map[i+10,j+6]:=-1;

## end;

## //нижняя часть

## for i := 0 to 9 do

## for j := 0 to 1 do

## begin

## map[i+2,30-j]:=-1;

## map[25-i,30-j]:=-1;

## end;

## //повёрнутая буква т

## for i := 0 to 1 do

## for j := 0 to 2 do

## begin

## map[i+7,28-j]:=-1;

## map[20-i,28-j]:=-1;

## end;

## //буквы т

## for i := 0 to 7 do

## for j := 0 to 1 do

## begin

## map[i+10,21-j]:=-1;

## map[i+10,27-j]:=-1;

## end;

## for i := 0 to 1 do

## for j := 0 to 2 do

## begin

## map[i+13,24-j]:=-1;

## map[i+13,30-j]:=-1;

## end;

## //боковые блоки по 1 блоку

## for i := 0 to 1 do

## for j := 0 to 1 do

## begin

## map[i+1,j+26]:=-1;

## map[26-i,j+26]:=-1;

## end;

## //буквы г

## for i := 0 to 1 do

## for j := 0 to 4 do

## begin

## map[i+4,j+23]:=-1;

## map[23-i,j+23]:=-1;

## end;

## for i := 0 to 1 do

## for j := 0 to 1 do

## begin

## map[i+2,j+23]:=-1;

## map[25-i,j+23]:=-1;

## end;

## //нижнии линии по горизонтали

## for i := 0 to 4 do

## for j := 0 to 1 do

## begin

## map[i+7,j+23]:=-1;

## map[20-i,j+23]:=-1;

## end;

## //домик

## for i := 0 to 7 do

## begin

## map[i+10,j+16]:=-1;

## end;

## //боковые стены домика

## for j := 0 to 4 do

## begin

## map[10,j+13]:=-1;

## map[17,j+13]:=-1;

## end;

## //крыша домика

## for i := 0 to 2 do

## begin

## map[i+10,12]:=-1;

## map[i+15,12]:=-1;

## end;

## //очки силы

## for i := 0 to 27 do

## for j := 0 to 32 do

## if map[i,j]=0 then map[i,j]:=1;

## //удаление точек

## for i := 7 to 20 do

## for j := 9 to 21 do

## if map[i,j]>0 then map[i,j]:=0;

## for i := 0 to 5 do

## for j := 9 to 21 do

## begin

## if map[i,j]>0 then map[i,j]:=0;

## if map[27-i,j]>0 then map[27-i,j]:=0;

## end;

## //очки силы

## map[1,3]:=3;

## map[26,3]:=2;

## map[13,8]:=4;

## map[26,25]:=3;

## map[26,25]:=3;

## map[6,18]:=5;

## map[13,31]:=7;

## map[21,15]:=7;

## map[21,18]:=3;

## map[9,5]:=2;

## end;

## end;

## procedure TForm1.FormKeyDown(Sender: TObject; var Key: Word;

## Shift: TShiftState);

## begin

## if Key = VK\_Up then Knight.Rotate:=2; //вверх

## if Key = VK\_Down then Knight.Rotate:=1; //вниз

## if Key = VK\_Left then Knight.Rotate:=3; //влево

## if Key = VK\_Right then Knight.Rotate:=4; //вправо

## end;

## //уровень 2

## procedure lvl2;

## begin

## //создание боковых блоков(верхняя стена и нижняя)

## for i := 0 to 27 do

## begin

## //верхняя горизонталь

## map[i,0]:=-2;

## //нижняя горизонталь

## map[i,32]:=-2;

## end;

## for j := 0 to 9 do

## begin

## //верхние вертикалии

## map[0,j]:=-2;

## map[27,j]:=-2;

## end;

## for j := 0 to 11 do

## begin

## //нижние вертикалии

## map[0,32-j]:=-2;

## map[27,32-j]:=-2;

## end;

## for i := 0 to 5 do

## begin

## //стены по горизонтали

## map[i,9]:=-2;

## map[27-i,9]:=-2;

## map[i,14]:=-2;

## map[27-i,14]:=-2;

## map[i,21]:=-2;

## map[27-i,21]:=-2;

## map[i,16]:=-2;

## map[27-i,16]:=-2;

## end;

## for j := 0 to 5 do

## begin

## //средние вертикали

## map[5,j+9]:=-2;

## map[22,j+9]:=-2;

## map[5,j+16]:=-2;

## map[22,j+16]:=-2;

## map[7,j+16]:=-2;

## map[8,j+16]:=-2;

## map[20,j+16]:=-2;

## map[19,j+16]:=-2;

## end;

## //

## //центровая линия

## for i:=0 to 1 do

## for j:=1 to 4 do

## map[i+13,j]:=-2;

## //левый и правый квадрат(маленькие)

## for i := 0 to 3 do

## for j := 0 to 2 do

## begin

## map[i+2,j+2]:=-2;

## map[25-i,j+2]:=-2;

## end;

## //квадраты большие

## for i := 0 to 4 do

## for j := 0 to 2 do

## begin

## map[i+7,j+2]:=-2;

## map[20-i,j+2]:=-2;

## end;

## //квадраты меньше

## for i := 0 to 3 do

## for j := 0 to 1 do

## begin

## map[i+2,j+6]:=-2;

## map[25-i,j+6]:=-2;

## end;

## //преграды

## for j := 0 to 8 do

## for i := 0 to 1 do

## begin

## map[i+7,j+6]:=-2;

## map[i+19,j+6]:=-2;

## end;

## for i := 0 to 2 do

## for j := 0 to 1 do

## begin

## map[i+9,j+9]:=-2;

## map[18-i,j+9]:=-2;

## end;

## //двойная линия по центру

## for i := 0 to 7 do

## for j := 0 to 1 do

## begin

## map[i+10,j+6]:=-2;

## end;

## //нижняя часть

## for i := 0 to 9 do

## for j := 0 to 1 do

## begin

## map[i+2,30-j]:=-2;

## map[25-i,30-j]:=-2;

## end;

## //повёрнутая буква т

## for i := 0 to 1 do

## for j := 0 to 2 do

## begin

## map[i+7,28-j]:=-2;

## map[20-i,28-j]:=-2;

## end;

## //буквы т

## for i := 0 to 7 do

## for j := 0 to 1 do

## begin

## map[i+10,21-j]:=-2;

## map[i+10,27-j]:=-2;

## end;

## for i := 0 to 1 do

## for j := 0 to 2 do

## begin

## map[i+13,24-j]:=-2;

## map[i+13,30-j]:=-2;

## end;

## //боковые блоки по 1 блоку

## for i := 0 to 1 do

## for j := 0 to 1 do

## begin

## map[i+1,j+26]:=-2;

## map[26-i,j+26]:=-2;

## end;

## //буквы г

## for i := 0 to 1 do

## for j := 0 to 4 do

## begin

## map[i+4,j+23]:=-2;

## map[23-i,j+23]:=-2;

## end;

## for i := 0 to 1 do

## for j := 0 to 1 do

## begin

## map[i+2,j+23]:=-2;

## map[25-i,j+23]:=-2;

## end;

## //нижнии линии по горизонтали

## for i := 0 to 4 do

## for j := 0 to 1 do

## begin

## map[i+7,j+23]:=-2;

## map[20-i,j+23]:=-2;

## end;

## //домик

## for i := 0 to 7 do

## begin

## map[i+10,j+16]:=-2;

## end;

## //боковые стены домика

## for j := 0 to 4 do

## begin

## map[10,j+13]:=-2;

## map[17,j+13]:=-2;

## end;

## //крыша домика

## for i := 0 to 2 do

## begin

## map[i+10,12]:=-2;

## map[i+15,12]:=-2;

## end;

## //очки силы

## for i := 0 to 27 do

## for j := 0 to 32 do

## if map[i,j]=0 then map[i,j]:=6;

## //удаление точек

## for i := 7 to 20 do

## for j := 9 to 21 do

## if map[i,j]>0 then map[i,j]:=0;

## for i := 0 to 5 do

## for j := 9 to 21 do

## begin

## if map[i,j]>0 then map[i,j]:=0;

## if map[27-i,j]>0 then map[27-i,j]:=0;

## end;

## map[1,3]:=2;

## map[26,3]:=5;

## map[13,8]:=7;

## map[26,25]:=8;

## map[26,25]:=3;

## map[6,18]:=13;

## map[13,31]:=7;

## map[21,18]:=9;

## end;

## //уровень 3

## procedure lvl3;

## begin

## //создание боковых блоков(верхняя стена и нижняя)

## for i := 0 to 27 do

## begin

## //верхняя горизонталь

## map[i,0]:=-3;

## //нижняя горизонталь

## map[i,32]:=-3;

## end;

## for j := 0 to 9 do

## begin

## //верхние вертикалии

## map[0,j]:=-3;

## map[27,j]:=-3;

## end;

## for j := 0 to 11 do

## begin

## //нижние вертикалии

## map[0,32-j]:=-3;

## map[27,32-j]:=-3;

## end;

## for i := 0 to 5 do

## begin

## //стены по горизонтали

## map[i,9]:=-3;

## map[27-i,9]:=-3;

## map[i,14]:=-3;

## map[27-i,14]:=-3;

## map[i,21]:=-3;

## map[27-i,21]:=-3;

## map[i,16]:=-3;

## map[27-i,16]:=-3;

## end;

## for j := 0 to 5 do

## begin

## //средние вертикали

## map[5,j+9]:=-3;

## map[22,j+9]:=-3;

## map[5,j+16]:=-3;

## map[22,j+16]:=-3;

## map[7,j+16]:=-3;

## map[8,j+16]:=-3;

## map[20,j+16]:=-3;

## map[19,j+16]:=-3;

## end;

## //

## //центровая линия

## for i:=0 to 1 do

## for j:=1 to 4 do

## map[i+13,j]:=-3;

## //левый и правый квадрат(маленькие)

## for i := 0 to 3 do

## for j := 0 to 2 do

## begin

## map[i+2,j+2]:=-3;

## map[25-i,j+2]:=-3;

## end;

## //квадраты большие

## for i := 0 to 4 do

## for j := 0 to 2 do

## begin

## map[i+7,j+2]:=-3;

## map[20-i,j+2]:=-3;

## end;

## //квадраты меньше

## for i := 0 to 3 do

## for j := 0 to 1 do

## begin

## map[i+2,j+6]:=-3;

## map[25-i,j+6]:=-3;

## end;

## //преграды

## for j := 0 to 8 do

## for i := 0 to 1 do

## begin

## map[i+7,j+6]:=-3;

## map[i+19,j+6]:=-3;

## end;

## for i := 0 to 2 do

## for j := 0 to 1 do

## begin

## map[i+9,j+9]:=-3;

## map[18-i,j+9]:=-3;

## end;

## //двойная линия по центру

## for i := 0 to 7 do

## for j := 0 to 1 do

## begin

## map[i+10,j+6]:=-3;

## end;

## //нижняя часть

## for i := 0 to 9 do

## for j := 0 to 1 do

## begin

## map[i+2,30-j]:=-3;

## map[25-i,30-j]:=-3;

## end;

## //повёрнутая буква т

## for i := 0 to 1 do

## for j := 0 to 2 do

## begin

## map[i+7,28-j]:=-3;

## map[20-i,28-j]:=-3;

## end;

## //буквы т

## for i := 0 to 7 do

## for j := 0 to 1 do

## begin

## map[i+10,21-j]:=-3;

## map[i+10,27-j]:=-3;

## end;

## for i := 0 to 1 do

## for j := 0 to 2 do

## begin

## map[i+13,24-j]:=-3;

## map[i+13,30-j]:=-3;

## end;

## //боковые блоки по 1 блоку

## for i := 0 to 1 do

## for j := 0 to 1 do

## begin

## map[i+1,j+26]:=-3;

## map[26-i,j+26]:=-3;

## end;

## //буквы г

## for i := 0 to 1 do

## for j := 0 to 4 do

## begin

## map[i+4,j+23]:=-3;

## map[23-i,j+23]:=-3;

## end;

## for i := 0 to 1 do

## for j := 0 to 1 do

## begin

## map[i+2,j+23]:=-3;

## map[25-i,j+23]:=-3;

## end;

## //нижнии линии по горизонтали

## for i := 0 to 4 do

## for j := 0 to 1 do

## begin

## map[i+7,j+23]:=-3;

## map[20-i,j+23]:=-3;

## end;

## //домик

## for i := 0 to 7 do

## begin

## map[i+10,j+16]:=-3;

## end;

## //боковые стены домика

## for j := 0 to 4 do

## begin

## map[10,j+13]:=-3;

## map[17,j+13]:=-3;

## end;

## //крыша домика

## for i := 0 to 2 do

## begin

## map[i+10,12]:=-3;

## map[i+15,12]:=-3;

## end;

## //очки силы

## for i := 0 to 27 do

## for j := 0 to 32 do

## if map[i,j]=0 then map[i,j]:=10;

## //удаление точек

## for i := 7 to 20 do

## for j := 9 to 21 do

## if map[i,j]>0 then map[i,j]:=0;

## for i := 0 to 5 do

## for j := 9 to 21 do

## begin

## if map[i,j]>0 then map[i,j]:=0;

## if map[27-i,j]>0 then map[27-i,j]:=0;

## end;

## map[1,3]:=4;

## map[26,3]:=5;

## map[13,8]:=8;

## map[26,25]:=2;

## map[26,25]:=3;

## map[6,18]:=4;

## map[13,31]:=9;

## map[21,18]:=13;

## map[6,13]:=11;

## end;

## //бонус изменение цвета

## procedure chcolor;

## begin

## for i := 0 to 27 do

## begin

## //верхняя горизонталь

## map[i,0]:=-4;

## //нижняя горизонталь

## map[i,32]:=-4;

## end;

## for j := 0 to 9 do

## begin

## //верхние вертикалии

## map[0,j]:=-4;

## map[27,j]:=-4;

## end;

## for j := 0 to 11 do

## begin

## //нижние вертикалии

## map[0,32-j]:=-4;

## map[27,32-j]:=-4;

## end;

## for i := 0 to 5 do

## begin

## //стены по горизонтали

## map[i,9]:=-4;

## map[27-i,9]:=-4;

## map[i,14]:=-4;

## map[27-i,14]:=-4;

## map[i,21]:=-4;

## map[27-i,21]:=-4;

## map[i,16]:=-4;

## map[27-i,16]:=-4;

## end;

## for j := 0 to 5 do

## begin

## //средние вертикали

## map[5,j+9]:=-4;

## map[22,j+9]:=-4;

## map[5,j+16]:=-4;

## map[22,j+16]:=-4;

## map[7,j+16]:=-4;

## map[8,j+16]:=-4;

## map[20,j+16]:=-4;

## map[19,j+16]:=-4;

## end;

## //

## //центровая линия

## for i:=0 to 1 do

## for j:=1 to 4 do

## map[i+13,j]:=-4;

## //левый и правый квадрат(маленькие)

## for i := 0 to 3 do

## for j := 0 to 2 do

## begin

## map[i+2,j+2]:=-4;

## map[25-i,j+2]:=-4;

## end;

## //квадраты большие

## for i := 0 to 4 do

## for j := 0 to 2 do

## begin

## map[i+7,j+2]:=-4;

## map[20-i,j+2]:=-4;

## end;

## //квадраты меньше

## for i := 0 to 3 do

## for j := 0 to 1 do

## begin

## map[i+2,j+6]:=-4;

## map[25-i,j+6]:=-4;

## end;

## //преграды

## for j := 0 to 8 do

## for i := 0 to 1 do

## begin

## map[i+7,j+6]:=-4;

## map[i+19,j+6]:=-4;

## end;

## for i := 0 to 2 do

## for j := 0 to 1 do

## begin

## map[i+9,j+9]:=-4;

## map[18-i,j+9]:=-4;

## end;

## //двойная линия по центру

## for i := 0 to 7 do

## for j := 0 to 1 do

## begin

## map[i+10,j+6]:=-4;

## end;

## //нижняя часть

## for i := 0 to 9 do

## for j := 0 to 1 do

## begin

## map[i+2,30-j]:=-4;

## map[25-i,30-j]:=-4;

## end;

## //повёрнутая буква т

## for i := 0 to 1 do

## for j := 0 to 2 do

## begin

## map[i+7,28-j]:=-4;

## map[20-i,28-j]:=-4;

## end;

## //буквы т

## for i := 0 to 7 do

## for j := 0 to 1 do

## begin

## map[i+10,21-j]:=-4;

## map[i+10,27-j]:=-4;

## end;

## for i := 0 to 1 do

## for j := 0 to 2 do

## begin

## map[i+13,24-j]:=-4;

## map[i+13,30-j]:=-4;

## end;

## //боковые блоки по 1 блоку

## for i := 0 to 1 do

## for j := 0 to 1 do

## begin

## map[i+1,j+26]:=-4;

## map[26-i,j+26]:=-4;

## end;

## //буквы г

## for i := 0 to 1 do

## for j := 0 to 4 do

## begin

## map[i+4,j+23]:=-4;

## map[23-i,j+23]:=-4;

## end;

## for i := 0 to 1 do

## for j := 0 to 1 do

## begin

## map[i+2,j+23]:=-4;

## map[25-i,j+23]:=-4;

## end;

## //нижнии линии по горизонтали

## for i := 0 to 4 do

## for j := 0 to 1 do

## begin

## map[i+7,j+23]:=-4;

## map[20-i,j+23]:=-4;

## end;

## //домик

## for i := 0 to 7 do

## begin

## map[i+10,j+16]:=-4;

## end;

## //боковые стены домика

## for j := 0 to 4 do

## begin

## map[10,j+13]:=-4;

## map[17,j+13]:=-4;

## end;

## //крыша домика

## for i := 0 to 2 do

## begin

## map[i+10,12]:=-4;

## map[i+15,12]:=-4;

## end;

## //очки силы

## for i := 0 to 27 do

## for j := 0 to 32 do

## if map[i,j]=0 then map[i,j]:=12;

## //удаление точек

## for i := 7 to 20 do

## for j := 9 to 21 do

## if map[i,j]>0 then map[i,j]:=0;

## for i := 0 to 5 do

## for j := 9 to 21 do

## begin

## if map[i,j]>0 then map[i,j]:=0;

## if map[27-i,j]>0 then map[27-i,j]:=0;

## end;

## end;

## procedure TForm1.Timer1Timer(Sender: TObject);

## var win,m:boolean;

## begin

## //уменьшение времени

## if Knight.Time>0 then dec(Knight.Time);

## if Knight.Time=0 then

## begin

## for i := 0 to 3 do

## Gh[i].Angry:=True;

## Knight.Time:=-1;

## end;

## //фон

## Buf.Canvas.Brush.Color:=clBlack;

## Buf.Canvas.Rectangle(0,0, 672, 792);

## for i := 0 to 27 do

## for j := 0 to 32 do

## begin

## if map[i,j]=0 then Buf.Canvas.StretchDraw(Rect(i\*24, j\*24, i\*24+24, j\*24+24),Bg);

## if map[i,j]<0 then Buf.Canvas.StretchDraw(Rect(i\*24, j\*24, i\*24+24, j\*24+24),Wall);

## if map[i,j]=-2 then Buf.Canvas.StretchDraw(Rect(i\*24, j\*24, i\*24+24, j\*24+24),Wall2);

## if map[i,j]=-3 then Buf.Canvas.StretchDraw(Rect(i\*24, j\*24, i\*24+24, j\*24+24),Wall3);

## if map[i,j]>=1 then Buf.Canvas.StretchDraw(Rect(i\*24, j\*24, i\*24+24, j\*24+24),Power[map[i,j]]);

## if map[i,j]=-4 then Buf.Canvas.StretchDraw(Rect(i\*24, j\*24, i\*24+24, j\*24+24),freeze);

## end;

## //призрак

## for i := 0 to 3 do

## begin

## if Gh[i].Angry = True then

## Buf.Canvas.StretchDraw(Rect(Gh[i].X\*24, Gh[i].Y\*24, Gh[i].X\*24+24, Gh[i].Y\*24+24), Ghosts[i, Gh[i].Rotate])

## else

## Buf.Canvas.StretchDraw(Rect(Gh[i].X\*24, Gh[i].Y\*24, Gh[i].X\*24+24, Gh[i].Y\*24+24), Ghosts[i,5]);

## m:=false;

## //движение призраков

## case Gh[i].Rotate of

## 1:if map[Gh[i].X, Gh[i].Y+1]>=0 then begin Gh[i].Y:=Gh[i].Y+1; m:=True; end;

## 2:if map[Gh[i].X, Gh[i].Y-1]>=0 then begin Gh[i].Y:=Gh[i].Y-1; m:=true; end;

## 3:if map[Gh[i].X-1, Gh[i].Y]>=0 then begin Gh[i].X:=Gh[i].X-1; m:=true; end;

## 4:if map[Gh[i].X+1, Gh[i].Y]>=0 then begin Gh[i].X:=Gh[i].X+1; m:=true; end;

## end;

## if m=False then Gh[i].Rotate:=Random(4)+1;

## end;

## //прорисовка рыцаря

## Buf.Canvas.StretchDraw(Rect(Knight.X\*24, Knight.Y\*24, Knight.X\*24+24, Knight.Y\*24+24),Kn[Knight.Rotate, Knight.Anim]);

## Inc(Knight.Anim);

## case Knight.Rotate of

## 1:if map[Knight.X, Knight.Y+1]>=0 then Knight.Y:=Knight.Y+1;

## 2:if map[Knight.X, Knight.Y-1]>=0 then Knight.Y:=Knight.Y-1;

## 3:if map[Knight.X-1, Knight.Y]>=0 then Knight.X:=Knight.X-1;

## 4:if map[Knight.X+1, Knight.Y]>=0 then Knight.X:=Knight.X+1;

## end;

## if Knight.Anim>3 then Knight.Anim:=1;

## //агрессивность призраков

## if map[Knight.X,Knight.Y]>0 then

## begin

## if map[Knight.X, Knight.Y]=2 then

## begin

## for i := 0 to 3 do

## begin

## Gh[i].Angry:=false;

## Knight.time:=50;

## end;

## end;

## //увеличение hp

## if map[Knight.X,Knight.Y]>0 then

## begin

## if map[Knight.X, Knight.Y]=5 then

## Knight.HP:=Knight.HP+5;

## end;

## //увеличение очков на 20

## if map[Knight.X,Knight.Y]>0 then

## begin

## if map[Knight.X, Knight.Y]=3 then

## Knight.Score:=Knight.Score+20;

## end;

## //увеличение очков на 50

## if map[Knight.X,Knight.Y]>0 then

## begin

## if map[Knight.X, Knight.Y]=4 then

## begin

## Knight.Score:=Knight.Score+50;

## end;

## //увеличение очков в 2 раза

## if map[Knight.X,Knight.Y]>0 then

## begin

## if map[Knight.X, Knight.Y]=7 then

## Knight.Score:=Knight.Score\*2;

## end;

## //бомба

## if map[Knight.X,Knight.Y]>0 then

## begin

## if map[Knight.X, Knight.Y]=8 then

## begin

## Gh[0].X:=1;

## Gh[0].Y:=1;

## Gh[1].X:=26;

## Gh[1].Y:=1;

## Gh[2].X:=1;

## Gh[2].Y:=31;

## Gh[3].X:=26;

## Gh[3].Y:=31;

## Knight.Score:=Knight.Score+10;

## end;

## //замедление

## if map[Knight.X,Knight.Y]>0 then

## begin

## if map[Knight.X, Knight.Y]=9 then

## Timer1.Interval:=200;

## end;

## //изменение карты

## if map[Knight.X,Knight.Y]>0 then

## begin

## if map[Knight.X, Knight.Y]=11 then

## chcolor;

## end;

## if map[Knight.X,Knight.Y]>0 then

## begin

## if map[Knight.X, Knight.Y]=13 then

## Timer1.Interval:=100;

## end;

## map[Knight.X, Knight.Y]:=0;

## Knight.Score:=Knight.Score+1;

## end;

## end;

## end;

## form1.Caption:='Уровень = ' +inttostr(Knight.Level) + ' | ' + 'Кол-во HP= ' + inttostr(Knight.HP) + ' | ' + 'Количество очков = ' + inttostr(Knight.Score);

## //проверка на победу

## win:=True;

## for i := 0 to 27 do

## for j := 0 to 32 do

## if map[i,j]>0 then win:=False;

## if win=True then

## begin

## inc(Knight.Level);

## if Knight.Level=2 then lvl2;

## Timer1.Interval:=170;

## Knight.X:=13;

## Knight.Y:=25;

## Knight.Rotate:=1;

## Knight.Anim:=1;

## Gh[0].X:=11;

## Gh[0].Y:=11;

## Gh[1].X:=12;

## Gh[1].Y:=11;

## Gh[2].X:=15;

## Gh[2].Y:=11;

## Gh[3].X:=16;

## Gh[3].Y:=11;

## if Knight.Level=3 then lvl3;

## Timer1.Interval:=130;

## Knight.X:=13;

## Knight.Y:=25;

## Knight.Rotate:=1;

## Knight.Anim:=1;

## Gh[0].X:=11;

## Gh[0].Y:=11;

## Gh[1].X:=12;

## Gh[1].Y:=11;

## Gh[2].X:=15;

## Gh[2].Y:=11;

## Gh[3].X:=16;

## Gh[3].Y:=11;

## if Knight.Level=2 then

## begin

## ShowMessage('Вы молодец!!!Уровень 2 начинается');

## end;

## if Knight.Level=3 then

## begin

## ShowMessage('Ваааууу!!! Как это возможно? Уровень 3');

## end;

## if Knight.Level>3 then

## begin

## Timer1.Enabled:=False;

## if LogAndPass<>'' then

## begin

## ShowMessage(Log+'.'+' Вы набрали очков: '+IntToStr(Knight.Score)+' Ваш уровень '+IntToStr(Knight.Level-1));

## SaveTxt.Add(Log+' '+' Вы набрали очков: '+IntToStr(Knight.Score)+' Ваш уровень '+IntToStr(Knight.Level-1));

## SaveTxt.SaveToFile(ExtractFilePath(Application.ExeName)+'RegEdit.txt');

## end

## else

## begin

## ShowMessage(LoginUsr+'.'+' Вы набрали очков: '+IntToStr(Knight.Score)+' Ваш уровень '+IntToStr(Knight.Level-1));

## SaveTxt.Add(LoginUsr+' '+PassUsr+' Вы набрали очков: '+IntToStr(Knight.Score)+' Ваш уровень '+IntToStr(Knight.Level-1));

## SaveTxt.SaveToFile(ExtractFilePath(Application.ExeName)+'RegEdit.txt')

## end;

## Form1.Close;

## end;

## end;

## form1.Canvas.StretchDraw(Rect(0,0,Form1.ClientWidth, Form1.ClientHeight),Buf);

## //поражение

## for i := 0 to 3 do

## begin

## if (Gh[i].X=Knight.X) and (Gh[i].Y=Knight.Y) and (Gh[i].Angry=True) and (Timer1.Enabled=true) then

## Knight.HP:=Knight.HP-1;

## if Knight.HP<=0 then

## begin

## Timer1.Enabled:=False;

## if LogAndPass<>'' then

## begin

## ShowMessage(Log+'.'+' Вы набрали очков: '+IntToStr(Knight.Score)+' Ваш уровень '+IntToStr(Knight.Level));

## end

## else

## begin

## ShowMessage(LoginUsr+'.'+' Вы набрали очков: '+IntToStr(Knight.Score)+' Ваш уровень'+IntToStr(Knight.Level));

## SaveTxt.Add(LoginUsr+' '+PassUsr+' Вы набрали очков: '+IntToStr(Knight.Score)+' Ваш уровень '+IntToStr(Knight.Level));

## SaveTxt.SaveToFile(ExtractFilePath(Application.ExeName)+'RegEdit.txt');

## end;

## ShowMessage('Не расстраиваетесь! Совсем немного не хватило');

## Form1.Close;

## end;

## end;

## //если призрак не злой, то рыцарь его убивает

## if (Gh[i].X=Knight.X) and (Gh[i].Y=Knight.Y) and (Gh[i].Angry=False) then

## begin

## Gh[i].X:=13;

## Gh[i].Y:=25;

## Knight.Score:=Knight.Score+5;

## end;

## end;

## end.

unit Unit2;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Samples.Gauges, Vcl.Imaging.jpeg, unit4,

Vcl.ExtCtrls, Vcl.MPlayer;

type

TForm2 = class(TForm)

Image1: TImage;

Gauge1: TGauge;

TimerImage1: TTimer;

procedure TimerImage1Timer(Sender: TObject);

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

var

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form2: TForm2;

implementation

uses unit3, unit1;

{$R \*.dfm}

procedure TForm2.FormClose(Sender: TObject; var Action: TCloseAction);

begin

Application.Terminate;

end;

procedure TForm2.TimerImage1Timer(Sender: TObject);

begin

Gauge1.Progress:=Gauge1.Progress+1;

if (Gauge1.Progress=100) then

begin

TimerImage1.Enabled:=False;

Form4.Show;

Form2.Hide;

end;

end;

end.

unit Unit3;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Imaging.jpeg, Vcl.ExtCtrls, unit1,

Vcl.Samples.Gauges, ShellAPI, Vcl.Menus, Vcl.OleCtrls, SHDocVw;

type

TForm3 = class(TForm)

Image1: TImage;

Image2: TImage;

Image3: TImage;

Image4: TImage;

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

Image5: TImage;

N3: TMenuItem;

N4: TMenuItem;

procedure Image2Click(Sender: TObject);

procedure Image4Click(Sender: TObject);

procedure Image3Click(Sender: TObject);

procedure N2Click(Sender: TObject);

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

procedure Image5Click(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure N4Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form3: TForm3;

implementation

{$R \*.dfm}

uses unit6;

procedure TForm3.FormClose(Sender: TObject; var Action: TCloseAction);

begin

Application.Terminate;

end;

procedure TForm3.Image2Click(Sender: TObject);

begin

ShowMessage('Игра началась! Уровень первый');

Form1.Timer1.Enabled:=True;

Form1.Show;

Form3.Hide;

end;

procedure TForm3.Image3Click(Sender: TObject);

begin

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

end;

procedure TForm3.Image4Click(Sender: TObject);

begin

Form3.Hide;

end;

procedure TForm3.Image5Click(Sender: TObject);

begin

Form6.Show;

Form3.Hide;

end;

procedure TForm3.N2Click(Sender: TObject);

begin

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

end;

procedure TForm3.N3Click(Sender: TObject);

begin

Application.Terminate;

end;

procedure TForm3.N4Click(Sender: TObject);

begin

Form3.Hide;

Form1.Show;

end;

end.

unit Unit4;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, unit5, Vcl.Mask,

Vcl.Imaging.pngimage, Vcl.ExtCtrls, Vcl.Imaging.jpeg;

type

TForm4 = class(TForm)

Edit1: TEdit;

Label1: TLabel;

Label2: TLabel;

MaskEdit1: TMaskEdit;

Label3: TLabel;

Image1: TImage;

Image2: TImage;

Image3: TImage;

Image4: TImage;

procedure Image2Click(Sender: TObject);

procedure Image3Click(Sender: TObject);

procedure FormClose(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form4: TForm4;

Log, Pass, LogAndPass:string;

implementation

uses unit3, Unit2, unit1;

{$R \*.dfm}

procedure TForm4.FormClose(Sender: TObject);

begin

Application.Terminate;

end;

procedure TForm4.Image2Click(Sender: TObject);

begin

Form5.Show;

Form4.Hide;

end;

procedure TForm4.Image3Click(Sender: TObject);

var

SaveTxt:TStringList;

i:integer;

begin

SaveTxt:=TStringList.Create;

SaveTxt.LoadFromFile(ExtractFilePath(Application.ExeName)+'RegEdit.txt');

Log:=Edit1.Text;

Pass:=MaskEdit1.Text;

LogAndPass:=Edit1.Text+' '+MaskEdit1.Text;

for i := 0 to SaveTxt.Count-1 do

begin

if LogAndPass = SaveTxt[i] then

begin

Form3.Show;

Form4.Hide;

ShowMessage('Данные введены корректно');

end

else

begin

Label3.Visible:=True;

Label3.Caption:='Введите корректные данные';

Edit1.Clear;

MaskEdit1.Clear;

end;

end;

end;

end.

unit Unit5;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Mask,

Vcl.Imaging.pngimage, Vcl.ExtCtrls, Vcl.Imaging.jpeg;

type

TForm5 = class(TForm)

Edit1: TEdit;

Label1: TLabel;

Label2: TLabel;

MaskEdit1: TMaskEdit;

Image1: TImage;

Image2: TImage;

procedure Image2Click(Sender: TObject);

procedure FormClose(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form5: TForm5;

SaveTxt:TStringList;

LoginUsr, PassUsr:string;

implementation

uses unit3, unit1;

{$R \*.dfm}

procedure TForm5.FormClose(Sender: TObject);

begin

Application.Terminate;

end;

procedure TForm5.Image2Click(Sender: TObject);

begin

SaveTxt:=TStringList.Create;

SaveTxt.LoadFromFile(ExtractFilePath(Application.ExeName)+'RegEdit.txt');

if (Edit1.Text<>'') and (MaskEdit1.Text<>'') and (Length(MaskEdit1.Text)>=4) then

begin

LoginUsr:=Edit1.Text;

PassUsr:=MaskEdit1.Text;

SaveTxt.Add(LoginUsr+' '+PassUsr);

SaveTxt.SaveToFile(ExtractFilePath(Application.ExeName)+'RegEdit.txt');

ShowMessage('Вы успешно зарегистрированы!');

Form3.Show;

Form5.Hide;

end

else

begin

ShowMessage('Введите пароль длина, которого должна быть больше 3 символов');

Edit1.Clear;

MaskEdit1.Clear;

end;

end;

end.

unit Unit6;

interface

uses

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

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

type

TForm6 = class(TForm)

WebBrowser1: TWebBrowser;

procedure FormCreate(Sender: TObject);

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

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form6: TForm6;

implementation

{$R \*.dfm}

procedure TForm6.FormClose(Sender: TObject; var Action: TCloseAction);

begin

Application.Terminate;

end;

procedure TForm6.FormCreate(Sender: TObject);

begin

WindowState:=wsMaximized;

WebBrowser1.Navigate(ExtractFilePath(paramStr(0))+'file\bonuses.html');

end;

end.