**Введение**

На курсовое проектирование была поставлена задача создать игровое приложение «Пазлы».

Цель курсового проекта заключается в создании этого игрового приложения и документации к нему.

Созданная программа рассчитана на любую аудиторию, которая интересуется данной тематикой.

Приведём краткое описание разделов пояснительной записки.

Первый раздел носит название «Анализ задачи». В нём можно ознакомиться с постановкой задачи, которая включает в себя: исследование предметной области поставленной задачи. Также в этом разделе можно узнать о том, как данная задача решается в настоящее время. Описаны все входные и выходные данные. В подразделе «Инструменты обработки» рассмотрена среда, в которой был создан курсовой проект. Здесь также установлены минимальные и оптимальные требования к аппаратным характеристикам, обеспечивающим правильное функционирование поставленной задачи.

В разделе «Проектирование задачи» рассмотрены основные аспекты разработки программного продукта. Здесь можно узнать об организации данных в контексте среды разработки. В данном разделе описан пользовательский интерфейс, составление алгоритмов процесса обработки информации, описана разработка системы справочной информации.

«Реализация задачи» − это третий раздел пояснительной записки, в котором описываются все элементы и объекты, которые использованы при реализации данного приложения. В этом разделе описаны функции пользователя и их структура. Также здесь можно увидеть таблицу, в которой представлена полная аннотация файлов, используемых в данном проекте.

Четвёртый раздел – «Тестирование». В нём описано полное и функциональное тестирование данной программы, т.е. протестирован каждый пункт меню, каждая операция, которая выполняется приложением. Смоделированы всевозможные действия пользователя при работе с программой, начиная от запуска до выхода.

В разделе «Применение» описано назначение, область применения, среды функционирования приложения. Также в нём описано использование справочной системы.

«Заключение» содержит краткую формулировку задачи, результаты проделанной работы, описание использованных методов и средств, описание степени автоматизации процессов на различных этапах разработки.

В разделе «Список используемых источников» приведён список используемых при разработке источников.

В приложении к пояснительной записке приведён листинг программы с необходимыми комментариями.

Схема работы системы представлена в графической части.

**1 Анализ задачи**

**1.1 Постановка задачи**

Темой данного курсового проекта является «Разработка игрового приложения «Пазлы»».

Целью данного проекта будет являться разработка игрового приложения «Пазлы»».

Задачей проекта будет являться предоставление возможности провести время за игрой в «пазлы», давать возможность людям загружать собственные изображения для использования их, как пазла, а также показывать время за которое был собран пазл.

Головоломка «Пазлы» предоставляет позволяет улучшить восприятие и предоставляет возможность весело провести время.

Данный проект будет интересен разным категориям людей, интересующимися различного рода головоломками.

Периодичность использования игрового приложения будет ограничена желанием пользователя играть в «Пазлы».

Разрабатываемый программный продукт должен будет позволить выполнить следующие действия:

* осуществит получение справочной информации;
* процесс разложения картинки, на количество ячеек, заданное пользователем;
* выведет таблицы рекордов;
* выведет время за которое был собран пазл;
* выведет изображение, которое должно будет получиться, при правильном размещении всех ячеек пазла.

В разработанной программе будут использованы три вида данных. К входной информации можно отнести имя, вводимое игроком, количество ячеек в пазле. К выходной – результат размещения ячеек, время за которое был собран пазл, таблица рекордов, изображение которое должно получиться. Постоянной информацией в проекте будет справочная система.

**1.2 Инструменты разработки**

Для разработки данного проекта будет выбрана среда Delphi 7, так как это самая удобная и доступная среда разработки. Delphi – язык программирования, относящийся к классу RAD- (Rapid Application Development – «Средство быстрой разработки приложений») средств CASE – технологии. Delphi сделал разработку приложений для Windows быстрым и приятным процессом, позволяющим разрабатывать сложные и интересные проекты одним человеком, использующим Delphi.

Выгоды от проектирования в среде Windows с помощью Delphi:

* устраняется необходимость в повторном вводе данных;
* обеспечивается согласованность проекта и его реализации;
* увеличивается производительность разработки и переносимость программ.

Yandex-браузер – нужен для нахождения информации и изображений.

Word 2007 – нужен для написания пояснительной записки.

Help&Manual – нужен для создания системы справочной информации.

Smart Install Maker – нужен для создания инсталляции приложения.

Параметры компьютера, на котором проводится разработка проекта:

* ОЗУ 1 ГБ;
* процессор Intel(R) Atom(TM)N2600;
* ОС Windows 7.

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

* процессор не ниже Pentium 500 Mhz;
* объем ОЗУ не меньше 128 Mb;
* объем места на жестком диске не менее 200 Mb;
* ОС – Windows 7,8,10.

Таким образом требования к приложению оптимальные и само приложение непривередливо.

**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;
* минимально-обходимое количество информации на каждой форме.

Так же следует обратить внимание на такие моменты:

* одинаковый, средний размер окна формы;
* небольшие кнопки;
* приятный глазу интерфейс.

Минимальные системные требования:

* 1 GHz процессор(или более быстрый);
* 128 Mb RAM;
* от 70 МB доступного места на жестком диске.

Рекомендуемые:

* 1.6 GHz процессор (или более быстрый);
* 1 GB RAM;
* от 70 МB доступного места на жестком диске;

Требования к интерфейсу: 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 легко и понятно принимать информацию.

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

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

**2 Проектирование задачи**

**2.1 Организация данных**

Данные о проекте сохраняются и хранятся в файле. Все файл сохраняются в \*.txt и \*.jpg формате и вводятся в программе.

Система справочной информации будет представлена файлом справки, который будет содержать информацию о приложении и правилах игры «пазлы».

Основными функциями приложения будут являться:

* задание размера пазла;
* собирание пазла;
* просмотр таблицы рекордов;
* просмотр справочной информации.

Воспользовавшись пунктом главного меню «Играть», можно:

* открыть предложенные калькуляторы с примерами задач;
* ввести имя игрока;
* выбрать уровень сложности;
* выйти на главное меню.

Воспользовавшись пунктом главного меню «Справка», можно открыть окно справочной информации о приложении.

Воспользовавшись пунктом главного меню «Рекорды», можно открыть таблицу рекордов, с информацией о самых быстрых сборов «пазлов».

Воспользовавшись пунктом главного меню «Выход», можно выйти из программы.

**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 будет являться:

* возможность выбор изображения для дальнейшего его разбения на ячейки;
* собирание из ячеек выбранного пользователем или установленного по умолчанию изображения.

Для реализации задач будут использоваться процедуры. С помощью процедуры, например, будет осуществляться считывание имен игроков и время за которое они собрали пазл из файла, их появление на форме таблицы рекордов.

**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.

Особое внимание следует уделить интерфейсу. Разработчик должен так организовать внешний вид своей программы, что бы пользователь понял, что от него требуется. При решении задачи будут использованы компоненты для ввода информации и её вывода, для вставки изображений, надписей и текста, и др.

Для организации эффектной 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. Приложение должно позволить пользователю безпрепятственно собрать пазл .

Таким образом, для успешной работы всего проекта в целом следует предоставить удобные и интуитивно-понятные меню, заставку, а также учесть всевозможные вариант ошибок при работе над проектом.

**3 Реализация**

**3.1 Структура программы**

Данный курсовой проект содержит 8 модулей. Далее рассмотрим назначение каждого модуля:

Unit1 – это модуль, в котором происходит игровой процесс «пазлов».

Unit2 – это модуль, отображающий изображение которое должно получиться у пользователя в итоге в модуле Unit1.

Unit3 – это модуль застваки.

Unit4 – это модуль главного меню, в котором можно открыть модули Unit8, Unit7, справочную систему, выйти из приложения.

Unit 5 – это модуль выбора сложности пазла, после выбора сложности переносит пользователя на модуль Unit1.

Unit6 – это модуль ручного ввода сложности пазла, после корректного ввода переносит пользователя на модуль Unit1.

Unit7 – это модуль таблицы рекордов, который отображает в порядке возрастания имена игроков и время за которое они собрали пазл.

Unit8 – это модуль ввода имени игрока, после корректного ввода игрока переносит пользователя на модуль Unit5.

**3.1.1 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 выполнения закреплена? | Назначение |
| 1 | 2 | 3 | 4 |
| 1. procedure N3Click(Sender: TObject) | Unit1.pas | TMainMenu1 | Позволяет выбрать изображение, которое будет использовано для сбора. |
| 1. procedure FormCreate(Sender: TObject) | Unit1.pas | Form1 | Подготавливает модуль к делению изображения на ячейки. |

Продолжение Таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 1. procedure N221Click(Sender: TObject) | Unit1.pas | TMainMenu1 | Изменяет размер изображения на 2х2. |
| 1. procedure N441Click(Sender: TObject) | Unit1.pas | TMainMenu1 | Изменяет размер изображения на 4х4. |
| 1. procedure N881Click(Sender: TObject) | Unit1.pas | TMainMenu1 | Изменяет размер изображения на 8х8. |
| 1. procedure Timer1Timer(Sender: TObject) | Unit1.pas | TTimer1 | Выполняет основной игровой цикл, проверяет собран ли пазл и если собран, то показывает результат. |
| 1. procedure FormMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer); | Unit1.pas | Form1 | Проверяет где был нажат курсор и обеспечивает перемещение ячеек пазла по игровому полю. |
| 1. procedure N2Click(Sender: TObject) | Unit1.pas | TMainMenu1 | Открывает выбранное изображение и разбивает его на ячейки. |
| 1. procedure N5Click(Sender: TObject) | Unit1.pas | TMainMenu1 | Показывает изображение, которое должно получиться в итоге. |
| 1. procedure FormClose(Sender: TObject | Unit1.pas | Form1 | Закрывает приложение |
| 1. procedure N6Click(Sender: TObject) | Unit1.pas | TMainMenu1 | Выводит справку |
| 1. procedure FormShow(Sender: TObject) | Unit1.pas | Form1 | Запускает пазл, при запуске модуля Unit1 |
| 1. procedure N7Click(Sender: TObject) | Unit1.pas | TMainMenu1 | Возвращает пользователя на главное меню |
| 1. procedure Timer2Timer(Sender: TObject) | Unit1.pas | TTimer2 | Подсчитывает время, за которое собран пазл |
| 1. procedure FormHide(Sender: TObject) | Unit1.pas | Form1 | Закрывает окно с исходным изображением, если пользователь закончил собирать пазл. |
| 1. procedure pazlGo | Unit1.pas | Form1 | Разбивает изображение на части. |
| 1. procedure Setka(cellX,cellY, MX,MY,RazmerX,RazmerY,Tolshina:integer) | Unit1.pas | Form1 | Рисует сетку. |

Продолжение Таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 1. procedure FormShow(Sender: TObject) | Unit2.pas | Form2 | Загружает исходное изображение. |
| 1. procedure Button1Click(Sender: TObject) | Unit2.pas | TButton1 | Закрывает форму с исходным изображением. |
| 1. procedure Timer1Timer(Sender: TObject) | Unit3.pas | TTimer1 | Создание переменной, подсчитывающей точки в названии формы. |
| 1. procedure FormCreate(Sender: TObject) | Unit3.pas | Form3 | Создает анимацию загрузки. |
| 1. procedure FormClose(Sender: TObject | Unit4.pas | Form4 | Выходит из приложения. |
| 1. procedure Image3Click(Sender: TObject) | Unit4.pas | TImage3 |
| 1. procedure Image3MouseMove(Sender: TObject | Unit4.pas | TImage3 | Создает анимацию наведения на кнопку «Выход». |
| 1. procedure FormCreate(Sender: TObject) | Unit4.pas | Form4 | Присваивает переменным, проверки наведения курсора, false. |
| 1. procedure Image4MouseMove(Sender: TObject | Unit4.pas | TImage4 | Создает анимацию наведения на кнопку «Справка». |
| 1. procedure Image5MouseMove(Sender: TObject | Unit4.pas | TImage5 | Создает анимацию наведения на кнопку «Рекорды». |
| 1. procedure Image2MouseMove(Sender: TObject | Unit4.pas | TImage2 | Создает анимацию наведения на кнопку «Играть». |
| 1. procedure Image1MouseMove(Sender: TObject | Unit4.pas | TImage1 | Создает анимацию при выходе курсора за пределы одной из 4 кнопок. |
| 1. procedure Image4OnlClick(Sender: TObject) | Unit4.pas | TImage4 | Выводит справку. |
| 1. procedure Image2Click(Sender: TObject) | Unit4.pas | TImage2 | Переходит к форме выбора имени игрока. |
| 1. procedure Image5Click(Sender: TObject) | Unit4.pas | TImage5 | Отображает рекорды игроков. |
| 1. procedure Image6Click(Sender: TObject) | Unit5.pas | TImage6 | Возвращает в главное меню. |
| 1. procedure Image2Click(Sender: TObject) | Unit5.pas | TImage2 | Переходит на окно, где пользователь вводит n для пазла размером nxn |
| 1. procedure Image3Click(Sender: TObject) | Unit5.pas | TImage3 | Переходит на форму с пазлом, с размером – 2x2 |

Продолжение Таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 1. procedure Image4Click(Sender: TObject) | Unit5.pas | TImage4 | Переходит на форму с пазлом, с размером 4x4 |
| 1. procedure Image5Click(Sender: TObject) | Unit5.pas | TImage5 | Переходит на форму с пазлом, с размером.8x8 |
| 1. procedure FormCreate(Sender: TObject) | Unit5.pas | Form5 | Присваивает переменным, проверки наведения курсора, false. |
| 1. procedure Image2MouseMove(Sender: TObject | Unit5.pas | TImage2 | Создает анимацию наведения на кнопку «0 уровень». |
| 1. procedure Image3MouseMove(Sender: TObject | Unit5.pas | TImage3 | Создает анимацию наведения на кнопку «1 уровень». |
| 1. procedure Image4MouseMove(Sender: TObject | Unit5.pas | TImage4 | Создает анимацию наведения на кнопку «2 уровень». |
| 1. procedure Image5MouseMove(Sender: TObject | Unit5.pas | TImage5 | Создает анимацию наведения на кнопку «3 уровень». |
| 1. procedure Image6MouseMove(Sender: TObject | Unit5.pas | TImage6 | Создает анимацию наведения на кнопку «Назад». |
| 1. procedure Image1MouseMove(Sender: TObject | Unit5.pas | TImage1 | Создает анимацию при выходе курсора за пределы одной из 5 кнопок. |
| 1. procedure FormClose(Sender: TObject | Unit5.pas | Form5 | Выходит из приложения |
| 1. procedure Button1Click(Sender: TObject); | Unit8.pas | TButton1 | Проверяет на корректность введенное имя игрока и если оно подходит по всем критериям, то переходит на форму выбора уровней. |
| 1. procedure FormClose(Sender: TObject; var Action: TCloseAction); | Unit8.pas | Form8 | Переходит в главное меню. |
| 1. procedure FormShow(Sender: TObject); | Unit8.pas | Form8 | Очищает поле для ввода имени игрока. |
| 1. procedure Button1Click(Sender: TObject); | Unit6.pas | TButton1 | При корректном вводе числа, переходит на форму с пазлом размером введенным пользователем. |
| 1. procedure FormShow(Sender: TObject) | Unit7.pas | Form7 | Выводит таблицу рекордов. |
| 1. procedure FormClose(Sender: TObject | Unit7.pas | Form7 | Переходит на главное меню и очищает поле для таблицы рекордов. |

Продолжение Таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| procedure Button1Click(Sender: TObject) | Unit7.pas | TButton1 | Закрывает форму с рекордами |
| procedure Button2Click(Sender: TObject) | Unit7.pas | TButton2 | Очищает таблицу рекордов. |

**3.1.2 Описание использованных компонентов**

Описание использованных для разработки приложения компонентов приводится в Таблице 2.

Таблица 2 – Использованные компоненты

|  |  |  |
| --- | --- | --- |
| Компонент | На какой форме расположен? | Назначение |
| 1 | 2 | 3 |
| 1 Timer1 | Form1 | Цикл игры |
| 2 Timer2 | Form1 | Таймер |
| 3 OpenDialog1 | Form1 | Позволяет найти изображение |
| 3 MainMenu1 | Form1 | Меню быстрого доступа |
| 4 Image1 | Form2 | Исходное изображение |
| 5 Button1 | Form2 | Кнопка «Ок» |
| 6 Timer1 | Form3 | Время загрузки |
| 7 ProgressBar1 | Form3 | Полоса загрузки |
| 8 Image1 | Form3 | Фон загрузочного меню |
| 9 Image1 | Form4 | Фон главного меню |
| 10 Image2 | Form4 | Кнопка «Играть» |
| 11 Image3 | Form4 | Кнопка «Выход» |
| 12 Image4 | Form4 | Кнопка «Справка» |
| 13 Image5 | Form4 | Кнопка «Рекорды» |
| 14 Image1 | Form5 | Фон формы выбора уровня |
| 15 Image2 | Form5 | Кнопка «0 уровень» |
| 16 Image3 | Form5 | Кнопка «1 уровень» |
| 17 Image4 | Form5 | Кнопка «2 уровень» |
| 18 Image5 | Form5 | Кнопка «3 уровень» |

Продолжение Таблицы 2

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 19 Image6 | Form5 | Кнопка «Назад» |
| 20 Lable1 | Form6 | Надпись «Введите n для пазла размером nxn» |
| 21 Edit1 | Form6 | Поле для ввода n |
| 22 Button1 | Form6 | Кнопка «Ok» |
| 23 Button1 | Form7 | Кнопка «Ok» |
| 24 Button2 | Form7 | Кнопка «Очистить» |
| 25 Image1 | Form7 | Фон для формы рекордов |
| 26 Memo1 | Form7 | Поле, куда вводится таблица рекордов |
| 27 Lable1 | Form8 | Надпись «Введите имя игрока» |
| 28 Edit1 | Form8 | Поле для ввода имени игрока |
| 29 Button1 | Form8 | Кнопка «Ok» |

**3.2 Спецификация программы**

Точное название проекта и его состав приводится в Таблице 3.

Таблица 3 – Спецификация программы

|  |  |
| --- | --- |
| Имя файла | Назначение |
| 1 | 2 |
| 1 Puzzle.exe | Исполняемый файл проекта, используется для запуска программы на выполнение |
| 2 Unit1.pas | Игра «пазл» |
| 3 Unit2.pas | Исходное изображение пазла |
| 4 Unit3.pas | Загрузочный экран |
| 5 Unit4.pas | Главное меню |
| 6 Unit5.pas | Выбор уровней |
| 7 Unit6.pas | Ввод пользователем количества ячеек в пазле |
| 8 Unit7.pas | Таблица рекордов |
| 9 Unit8.pas | Ввод имени игрока |
| 10 zastavka(ne udulyai pzpzpz).jpg | Картинка заставки и фона |

Продолжение Таблицы 3

|  |  |
| --- | --- |
| 1 | 2 |
| 11 ico.ico | Иконка проекта |
| 12 img | Папка с изображениями для пазла |
| 13 Appimg | Папка с изображениями для кнопок |
| 14 Help.chm | Справочник |

**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 о результатах тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| Тест | Ожидаемый результат | Физический результат | Результат тестирования |
| 1 | 2 | 3 | 4 |
| Проверка открытия приложения | Окно приложения появилось | Окно приложения появилось | Выполнено |
| Проверка открытия таблицы рекордов | Окно появилось с корректным расположением игроков по времени | Окно появилось с корректным расположением игроков по времени | Выполнено |
| Проверка открытия справки | Окно справки появилось | Окно справки появилось | Выполнено |
| Проверка открытия ввода имени игрока | Окно ввода имени игрока появилось, корректно обработало и передало данные | Окно ввода имени игрока появилось, корректно обработало и передало данные | Выполнено |
| Проверка открытия выбора уровня | Окно появилось и выдает пазл разделенный на заданное кнопками значение | Окно появилось и выдает пазл разделенный на заданное кнопками значение | Выполнено |
| Проверка окна ввода количество ячеек в пазле | Окно появилось и корректно обрабатывает и передает введенную информацию | Окно ввода имени игрока появилось, корректно обработало и передало данные | Выполнено |
| Проверка окна игры «пазлы» | Окно появилось и корректно обрабатывает передвижение ячеек пазлов, а также корректно принимает изображения при открытии их из папки img, также передает корректные значения | Окно появилось и корректно обрабатывает передвижение ячеек пазлов, а также корректно принимает изображения при открытии их из папки img, также передает корректные значения | Выполнено |
| Проверка закрытия приложения | Окно приложения исчезло | Окно приложения исчезло | Выполнено |

**5 Применение**

**5.1 Инсталляция**

Для того, чтобы установить программу, необходимо запустить puzzle.exe. Появится окно для установки программы (Рисунок 1).

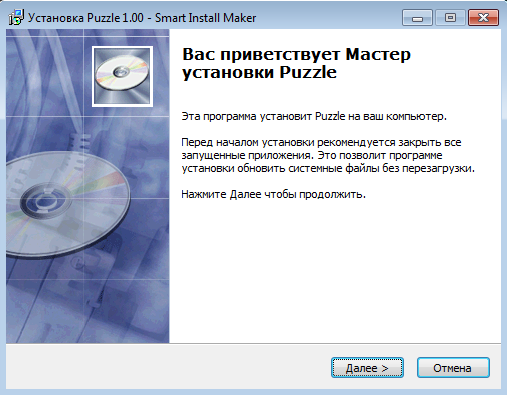
****

Рисунок 1 – Установка программы

Далее появится окно о запросе места установки (Рисунок 2).

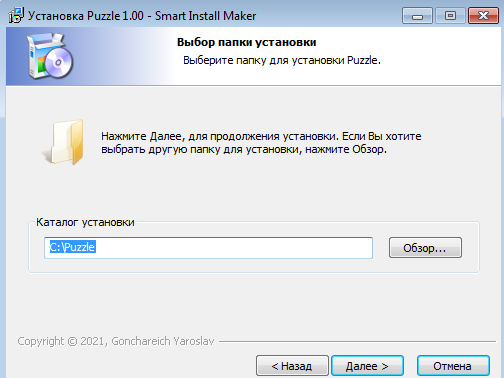


Рисунок 2 – Выбор места для установки

Далее выводится окно о том, что всё готово для начала установки (Рисунок 3).

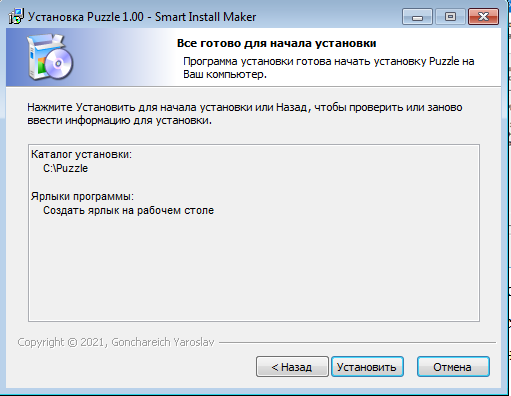


Рисунок 3 – Окно о том, что всё готово для начала установки

Следующим шагом идёт установка данного приложения и выведение сообщения об успешности её выполнения.

**5.2 Выполнение программы**

**5.2.1 Запуск программы**

Программу можно запустить несколькими способами.

Одним из них можно запустить программу, щёлкнув два раза левой кнопкой мыши по ярлыку на рабочем столе с названием «Puzzle» (Рисунок 4).



Рисунок 4 – Ярлык для захода в программу

Вторым способом является Пуск. Нужно нажать меню Пуск и нажать на приложение «Puzzle» (Рисунок 5).

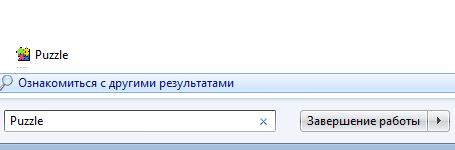


Рисунок 5 – Приложение для захода в программу

**5.2.3 Инструкции по работе с программой**

После запуска на экране появляется заставка (Рисунок 6).

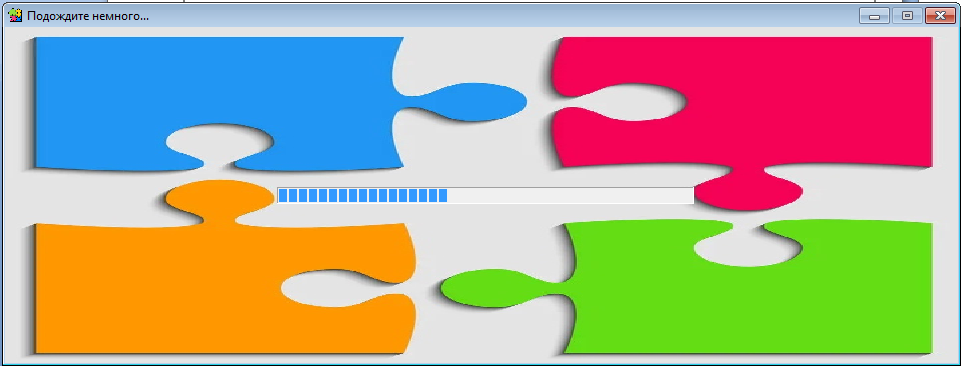


Рисунок 6 – Заставка

Затем появляется и главная форма (Рисунок 7), в которой пользователь сможет посмотреть таблицу рекордов, справочник, начать игру или выйти

****

Рисунок 7 – Главная форма

После того, как пользователь нажал на кнопку «Играть» он увидит окно с просьбой ввести имя игрока(Рисунок 8).

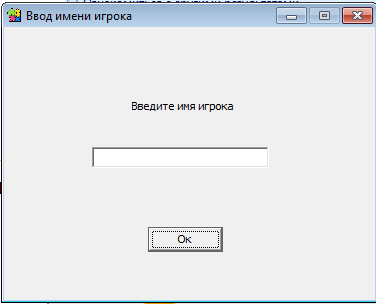


Рисунок 8 – Ввод имени игрока

После ввода имени пользователь выбирает уровень (Рисунок 9)

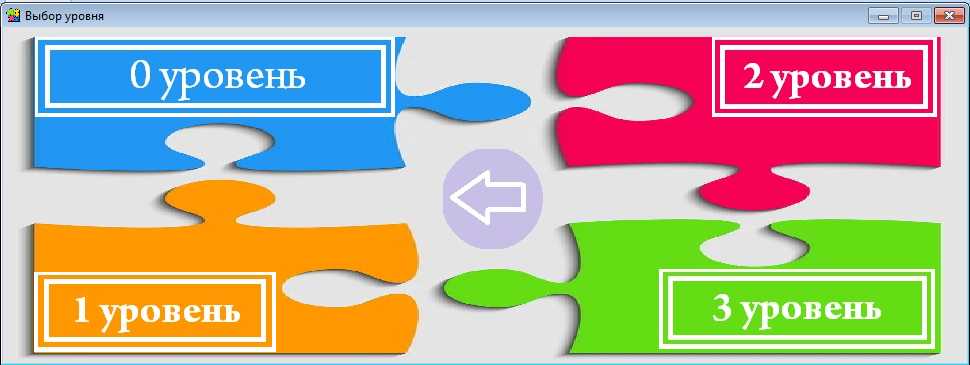
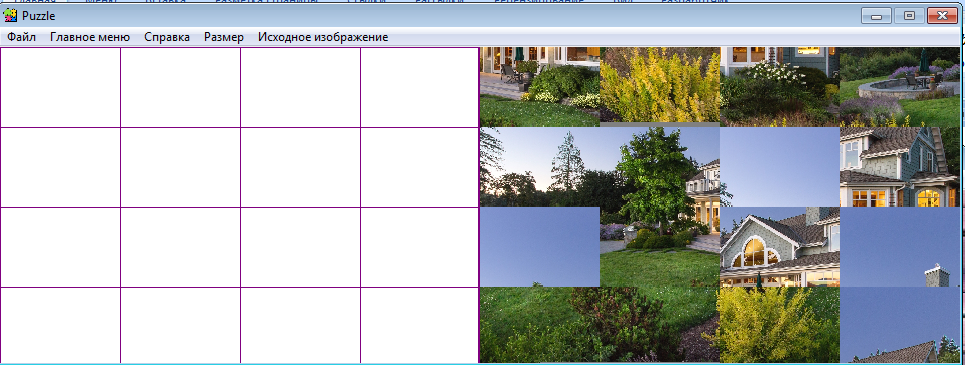


Рисунок 9 – Выбор уровня

После выбора урованя пользователь играет в пазл(Рисунок 10), пока не соберет его или пока не выйдет.



В главом меню и во время игры пользователь может зайти в справку

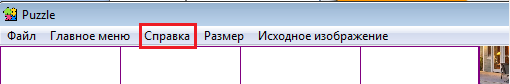
****

Рисунок 11 – Переход в справку при помощи подпункта «Справка»

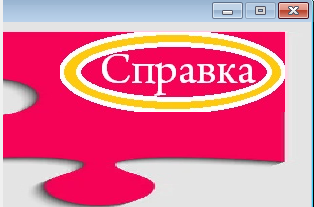


Рисунок 12 – Переход в справку из главного меню

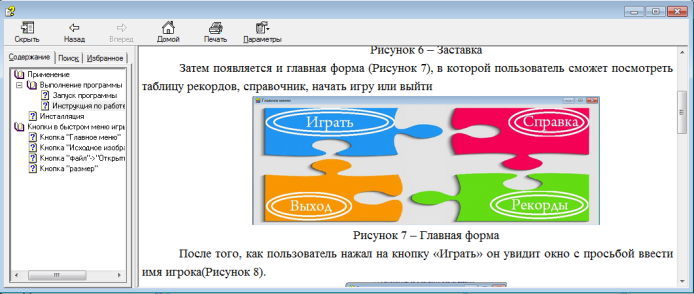


Рисунок 13 – Справка **Заключение**

Целью данного курсового проекта являлась разработка игрового приложения «Пазлы».

В ходе реализации поставленной задачи был укреплён пройденный курс программирования в среде Delphi 7, а так же получено много дополнительной информации о её возможностях.

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

После тщательного тестирования приложения были выявлены некоторые недоработки, которые были частично исправлены на стадии проектирования, и полностью исключены на стадии тестирования программы. В целом при реализации программы, были выполнены все условия, перечисленные в предыдущих разделах пояснительной записки. Таким образом, можно сказать, что программа была реализована вполне успешно.

**Список использованных источников**

1. Сайт о программировании на Delphi [Электронный ресурс]: – Режим доступа: <https://webdelphi.ru>. – Дата доступа: 05.06.2021.
2. Форум о программировани [Электронный ресурс]: – Режим доступа: <https://www.cyberforum.ru>. – Дата доступа: 09.06.2021.
3. Видеохостинг [Электронный ресурс]: – Режим Доступа [https://www.youtube.com](https://www.youtube.com/watch?v=UvG-BTKYypc). – Дата доступа: 27.06.2021.
4. Сайт с изображениями [Электронный ресурс]: – Режим доступа: <https://pixabay.com>. – Дата доступа: 29.06.2021.

**Приложение А**

**Листинг программы**

**Форма 1**

unit Unit1;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, Menus, JPEG, ShellAPI,ExtCtrls;

type

TForm1 = class(TForm)

OpenDialog1: TOpenDialog;

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

N4: TMenuItem;

N221: TMenuItem;

N441: TMenuItem;

N881: TMenuItem;

Timer1: TTimer;

N5: TMenuItem;

N6: TMenuItem;

N7: TMenuItem;

Timer2: TTimer;

procedure N3Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure N221Click(Sender: TObject);

procedure N441Click(Sender: TObject);

procedure N881Click(Sender: TObject);

procedure Timer1Timer(Sender: TObject);

procedure FormMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);

procedure N2Click(Sender: TObject);

procedure N5Click(Sender: TObject);

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

procedure N6Click(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure N7Click(Sender: TObject);

procedure Timer2Timer(Sender: TObject);

procedure FormHide(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

fullpath:string;

end;

type TPlayer = record

nickname:string[20];

time:string[8];

timer:integer;

end;

var

Form1: TForm1;

buf,original:TBitmap;

open:TJPEGImage;

pazl:array[1..1024] of TBitmap;

kopia,pole:array[0..31,0..31] of integer;

path:string;

razmer,XX,YY,Aktiv:integer;

tmr:integer;

procedure pazlGo;

procedure Setka(cellX,cellY,MX,MY,RazmerX,RazmerY,Tolshina:integer;SetkaColor:TColor;Buf:TBitmap);

implementation

uses unit2, Unit3, Unit5, Unit4, Unit6, Unit7, Unit8;

{$R \*.dfm}

procedure TForm1.N3Click(Sender: TObject);

begin

form1.Close;

end;

procedure TForm1.FormCreate(Sender: TObject);

var i,j:integer;

begin

tmr:=0;

Form1.Hide;

Form3.Show;

path:=ExtractFileDir(Application.ExeName);

buf:=TBitmap.Create;

buf.width:=960;

buf.Height:=320;

open:=TJpegImage.Create;

open.LoadFromFile(path+'\img\1.jpg');

fullpath:=path+'\img\1.jpg';

original:=TBitmap.Create;

original.width:=480;

original.Height:=320;

original.Canvas.StretchDraw(Rect(0,0,480,320),open);

for i:=0 to 31 do

for j:=0 to 31 do

begin

kopia[i,j]:=0;

pole[i,j]:=0;

end;

razmer:=4;

Aktiv:=0;

pazlGo;

end;

procedure TForm1.N221Click(Sender: TObject);

begin

razmer:=2;

pazlgo;

Aktiv:=0;

end;

procedure TForm1.N441Click(Sender: TObject);

begin

razmer:=4;

pazlgo;

Aktiv:=0;

end;

procedure TForm1.N881Click(Sender: TObject);

begin

razmer:=8;

pazlgo;

Aktiv:=0;

end;

procedure TForm1.Timer1Timer(Sender: TObject);

var chekWhile,kolvo,min,Tolshina,i,j,n,MX,MY:integer;

timeItogo,timerS,timerM,timerH:string;

bufff,playerWrite,playerWrite1:TPlayer;

player:file of TPlayer;

SetkaColor:TColor;

win:boolean;

begin

win:=true;

n:=1;

for i:=0 to Razmer-1 do

for j:=0 to Razmer-1 do

begin

if pole[i,j]<>n then win:=false;

n:=n+1;

end;

if win=True then

begin

Timer1.Enabled:=false;

Timer2.Enabled:=false;

form1.Hide;

if not(FileExists('Players.txt')) then

begin

assignFile(player,'Players.txt');

rewrite(player);

closeFile(player);

end;

timerS:=IntToStr(tmr mod 60);

timerM:=IntToStr(tmr div 60);

timerH:=IntToStr(tmr div 3600);

if StrToInt(timerS)<10 then timerS:='0'+timerS;

if StrToInt(timerM)<10 then timerM:='0'+timerM;

if StrToInt(timerH)<10 then timerH:='0'+timerH;

timeItogo:=timerH+':'+timerM+':'+timerS;

assignFile(player,'Players.txt');

reset(player);

chekWhile:=0;

kolvo:=0;

while(chekWhile=0)and(not(eof(player))) do begin

read(player,playerWrite);

if playerWrite.timer>tmr then chekWhile:=1;

kolvo:=kolvo+1;

end;

closeFile(player);

assignFile(player,'Players.txt');

playerWrite.timer:=tmr;

playerWrite.time:=timeItogo;

playerWrite.nickname:=form8.nickname;

reset(player);

seek(player,fileSize(player));

write(player,playerWrite);

seek(player,0);

if filesize(player)>1 then begin

for i:=0 to filesize(player) do

begin

for j:=0 to filesize(player)-2 do

begin

seek(player,j);

read(player,PlayerWrite);

seek(player,j+1);

read(player,PlayerWrite1);

if PlayerWrite.timer>playerWrite1.timer then

begin

bufff:=PlayerWrite;

PlayerWrite:=playerWrite1;

playerWrite1:=bufff;

end;

seek(player,j);

write(player,PlayerWrite);

seek(player,j+1);

write(player,PlayerWrite1);

end;

end;

end;

closeFile(player);

if form8.pas\_h=2 then ShellExecute(Self.Handle, 'open', PChar('appImg\recordAktiv.jpg'), nil, nil, SW\_SHOWNORMAL);

form7.Show;

end;

buf.canvas.Brush.Color:=clWhite;

buf.canvas.Rectangle(0,0,960,320);

MY:=0;

MX:=0;

Tolshina:=1;

SetkaColor:=clPurple;

Setka(XX,YY,MX,MY,Razmer,Razmer,Tolshina,SetkaColor,Buf);

MX:=480;

Setka(XX,YY,MX,MY,Razmer,Razmer,Tolshina,SetkaColor,Buf);

for i:=0 to razmer-1 do

for j:=0 to razmer-1 do

BEGIN

if Pole[i,j]>0 then buf.canvas.Draw(i\*XX,j\*YY,pazl[Pole[i,j]]);

if Aktiv>0 then buf.Canvas.Draw(Mouse.CursorPos.X-Form1.Left-(XX div 2),Mouse.CursorPos.Y-Form1.Top-25-(YY div 2),pazl[aktiv]);

if kopia[i,j]>0 then buf.Canvas.Draw(i\*XX+480,j\*YY,pazl[kopia[i,j]]);

end;

form1.Canvas.Draw(0,0,buf);

end;

procedure pazlGo;

var i,j,n:integer;

begin

for i:=0 to 31 do

for j:=0 to 31 do

begin

kopia[i,j]:=0;

pole[i,j]:=0;

end;

XX:=480 div razmer;

YY:=320 div razmer;

n:=1;

for i:=0 to razmer-1 do

for j:=0 to razmer-1 do

begin

pazl[n]:=TBitmap.Create;

pazl[n].Width:=XX;

pazl[n].Height:=YY;

pazl[n].Canvas.CopyRect(rect(0,0,XX,YY),original.Canvas,Rect(i\*XX,j\*YY,i\*XX+XX-1,j\*YY+YY-1));

n:=n+1;

end;

n:=1;

randomize;

while(n<(razmer\*razmer)+1)do

begin

i:=random(razmer);

j:=random(razmer);

if kopia[i,j]=0 then

begin

kopia[i,j]:=n;

n:=n+1;

end;

end;

end;

procedure Setka(cellX,cellY,MX,MY,RazmerX,RazmerY,Tolshina:integer;SetkaColor:TColor;Buf:TBitmap);

var i:integer;

begin

Buf.Canvas.Pen.Color:=SetkaColor;

Buf.Canvas.Pen.Width:=Tolshina;

for i:=0 to RazmerY do

begin

Buf.Canvas.MoveTo(0+MX,i\*cellY+MY);

Buf.Canvas.LineTo(RazmerX\*cellX+MX,i\*cellY+MY);

end;

for i:=0 to RazmerX do

begin

Buf.Canvas.MoveTo(i\*cellX+MX,0+MY);

Buf.Canvas.LineTo(i\*cellX+MX,RazmerY\*cellY+MY);

end;

Buf.Canvas.Brush.Color:=clPurple;

buf.Canvas.FillRect(Rect(478,0,482,316));

end;

procedure TForm1.FormMouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

var uzhe:boolean;

boof,i,j:integer;

begin

if (Button=MBRight) and (aktiv>0) then

begin

uzhe:=false;

for i:=0 to razmer-1 do

for j:=0 to razmer-1 do

begin

if (uzhe=false) and (kopia[i,j]=0) then

begin

uzhe:=true;

kopia[i,j]:=aktiv;

aktiv:=0;

end;

end;

end

else

begin

uzhe:=false;

if (X>=480) and (X<=960) and (Y>=0) and (Y<=320) then

begin

if (kopia[(X-480) div XX, Y div YY]>0) and (aktiv=0) then

begin

Aktiv:=kopia[(X-480) div XX, Y div YY];

kopia[(X-480) div XX, Y div YY]:=0;

uzhe:=true;

end;

if (kopia[(X-480) div XX, Y div YY]=0) and (aktiv>0) and (uzhe=false) then

begin

kopia[(X-480) div XX, Y div YY]:=Aktiv;

Aktiv:=0;

end;

if (aktiv>0) and (kopia[(X-480) div XX, Y div YY]>0) then

begin

boof:= aktiv;

aktiv:=kopia[(X-480) div XX, Y div YY];

kopia[(X-480) div XX, Y div YY]:=boof;

end;

end;

if (X>=0) and (X<=480) and (Y>=0) and (Y<=320) then

begin

if (pole[X div XX, Y div YY]>0) and (aktiv=0) then

begin

Aktiv:=pole[X div XX, Y div YY];

pole[X div XX, Y div YY]:=0;

uzhe:=true;

end;

if (pole[X div XX, Y div YY]=0) and (aktiv>0) and (uzhe=false) then

begin

pole[X div XX, Y div YY]:=Aktiv;

Aktiv:=0;

end;

if (aktiv>0) and (pole[X div XX, Y div YY]>0) then

begin

boof:= aktiv;

aktiv:=pole[X div XX, Y div YY];

pole[X div XX, Y div YY]:=boof;

end;

end;

end;

end;

procedure TForm1.N2Click(Sender: TObject);

begin

opendialog1.InitialDir:=path+'\img';

opendialog1.Execute;

opendialog1.Filter:='Изобржения формата jpg';

if opendialog1.Filename<>'' then

begin

if opendialog1.Filename=path+'\img\'+'7.jpg' then Form8.pas\_h:=Form8.pas\_h+1;

open.loadfromfile(opendialog1.FileName);

original.Canvas.StretchDraw(Rect(0,0,480,320),open);

fullpath:=opendialog1.Filename;

pazlgo;

timer1.enabled:=true;

timer2.Enabled:=true;

tmr:=0;

end;

end;

procedure TForm1.N5Click(Sender: TObject);

begin

Form2.Show;

end;

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

begin

Form3.Close;

end;

procedure TForm1.N6Click(Sender: TObject);

begin

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

end;

procedure TForm1.FormShow(Sender: TObject);

begin

timer1.Enabled:=True;

timer2.Enabled:=True;

tmr:=0;

pazlgo;

end;

procedure TForm1.N7Click(Sender: TObject);

begin

Form1.Hide;

Form4.Show;

end;

procedure TForm1.Timer2Timer(Sender: TObject);

begin

tmr:=tmr+1;

end;

procedure TForm1.FormHide(Sender: TObject);

begin

Form2.Hide

end;

end.

**Форма 2**

unit Unit2;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, ExtCtrls;

type

TForm2 = class(TForm)

Image1: TImage;

Button1: TButton;

procedure FormShow(Sender: TObject);

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form2: TForm2;

implementation

uses Unit1;

{$R \*.dfm}

procedure TForm2.FormShow(Sender: TObject);

begin

Image1.Picture.LoadFromFile(form1.fullpath);

end;

procedure TForm2.Button1Click(Sender: TObject);

begin

Form2.Hide;

Form1.Show;

end;

end.

**Форма 3**

unit Unit3;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, ComCtrls, ExtCtrls, jpeg;

type

TForm3 = class(TForm)

Timer1: TTimer;

ProgressBar1: TProgressBar;

Image1: TImage;

procedure Timer1Timer(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form3: TForm3;

n:integer;

implementation

uses Unit1, Unit4;

{$R \*.dfm}

procedure TForm3.FormCreate(Sender: TObject);

begin

n:=1;

end;

procedure TForm3.Timer1Timer(Sender: TObject);

var dviz, x,y:integer;

begin

randomize;

dviz:=random(20);

y:=n mod 4;

Form3.Caption:='Подождите немного';

for x:=1 to y do

Form3.Caption:=Form3.Caption+'.';

if (ProgressBar1.Position < ProgressBar1.Max) then

ProgressBar1.Position:=ProgressBar1.Position+dviz

else

begin

Form3.hide;

Form4.Show;

Timer1.Enabled:=False;

end;

n:=n+1;

end;

end.

**Форма 4**

unit Unit4;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, ExtCtrls, jpeg,ShellAPI;

type

TForm4 = class(TForm)

Image1: TImage;

Image2: TImage;

Image3: TImage;

Image4: TImage;

Image5: TImage;

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

procedure Image3Click(Sender: TObject);

procedure Image3MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

procedure FormCreate(Sender: TObject);

procedure Image4MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

procedure Image5MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

procedure Image2MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

procedure Image1MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

procedure Image4DblClick(Sender: TObject);

procedure Image2Click(Sender: TObject);

procedure Image5Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

type TPlayer = record

nickname:string[20];

time:string[8];

timer:integer;

end;

var

Form4: TForm4;

btnChek1,btnChek2,btnChek3,btnChek4:boolean;

player:file of TPlayer;

implementation

uses Unit3, Unit5, Unit7, Unit8;

{$R \*.dfm}

procedure TForm4.FormClose(Sender: TObject; var Action: TCloseAction);

begin

Form3.Close;

end;

procedure TForm4.Image3Click(Sender: TObject);

begin

Form4.Close;

end;

procedure TForm4.Image3MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

if btnChek2=false then

begin

Image3.Picture.LoadFromFile('appImg\выходAktiv.jpg');

btnChek2:=True;

end;

end;

procedure TForm4.FormCreate(Sender: TObject);

begin

btnChek1:=false;

btnChek2:=false;

btnChek3:=false;

btnChek4:=false;

end;

procedure TForm4.Image4MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

if btnChek3=false then

begin

Image4.Picture.LoadFromFile('appImg\справкаAktiv.jpg');

btnChek3:=True;

end;

end;

procedure TForm4.Image5MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

if btnChek4=false then

begin

Image5.Picture.LoadFromFile('appImg\рекордыAktiv.jpg');

btnChek4:=True;

end;

end;

procedure TForm4.Image2MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

if btnChek1=false then

begin

Image2.Picture.LoadFromFile('appImg\игратьAktiv.jpg');

btnChek1:=True;

end;

end;

procedure TForm4.Image1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

if (btnChek1=true) then

begin

btnChek1:=false;

Image2.Picture.LoadFromFile('appImg\играть.jpg');

end;

if (btnChek2=true) then

begin

btnChek2:=false;

Image3.Picture.LoadFromFile('appImg\выход.jpg');

end;

if (btnChek3=true) then

begin

btnChek3:=false;

Image4.Picture.LoadFromFile('appImg\справка.jpg');

end;

if (btnChek4=true) then

begin

btnChek4:=false;

Image5.Picture.LoadFromFile('appImg\рекорды.jpg');

end;

end;

procedure TForm4.Image4DblClick(Sender: TObject);

begin

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

end;

procedure TForm4.Image2Click(Sender: TObject);

begin

form4.Hide;

form8.show;

end;

procedure TForm4.Image5Click(Sender: TObject);

begin

if not(FileExists('Players.txt')) then

begin

assignFile(player,'Players.txt');

rewrite(player);

closeFile(player);

end;

form7.Show;

end;

end.

**Форма 5**

unit Unit5;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, jpeg, ExtCtrls;

type

TForm5 = class(TForm)

Image1: TImage;

Image2: TImage;

Image3: TImage;

Image4: TImage;

Image5: TImage;

Image6: TImage;

procedure Image6Click(Sender: TObject);

procedure Image2Click(Sender: TObject);

procedure Image3Click(Sender: TObject);

procedure Image4Click(Sender: TObject);

procedure Image5Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure Image2MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

procedure Image3MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

procedure Image4MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

procedure Image5MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

procedure Image6MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

procedure Image1MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

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

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form5: TForm5;

btnChek1,btnChek2,btnChek3,btnChek4,btnChek5:boolean;

procedure start();

implementation

uses Unit1, Unit4, Unit3, Unit6;

{$R \*.dfm}

procedure start();

begin

Form5.Hide;

Form1.Show;

end;

procedure TForm5.Image4MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

if btnChek3=false then

begin

Image4.Picture.LoadFromFile('appImg\2 уровеньAktiv.jpg');

btnChek3:=True;

end;

end;

procedure TForm5.Image5MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

if btnChek4=false then

begin

Image5.Picture.LoadFromFile('appImg\3 уровеньAktiv.jpg');

btnChek4:=True;

end;

end;

procedure TForm5.Image6MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

if btnChek5=false then

begin

Image6.Picture.LoadFromFile('appImg\назадAktiv.jpg');

btnChek5:=True;

end;

end;

procedure TForm5.Image6Click(Sender: TObject);

begin

Form5.Hide;

Form4.Show;

end;

procedure TForm5.Image2Click(Sender: TObject);

begin

Form6.show;

end;

procedure TForm5.Image3Click(Sender: TObject);

begin

razmer:=2;

start();

end;

procedure TForm5.Image4Click(Sender: TObject);

begin

razmer:=4;

start();

end;

procedure TForm5.Image5Click(Sender: TObject);

begin

razmer:=8;

start();

end;

procedure TForm5.FormCreate(Sender: TObject);

begin

btnChek1:=false;

btnChek2:=false;

btnChek3:=false;

btnChek4:=false;

end;

procedure TForm5.Image2MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

if btnChek1=false then

begin

Image2.Picture.LoadFromFile('appImg\0 уровеньAktiv.jpg');

btnChek1:=True;

end;

end;

procedure TForm5.Image3MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

if btnChek2=false then

begin

Image3.Picture.LoadFromFile('appImg\1 уровеньAktiv.jpg');

btnChek2:=True;

end;

end;

procedure TForm5.Image1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

if (btnChek1=true) then

begin

btnChek1:=false;

Image2.Picture.LoadFromFile('appImg\0 уровень.jpg');

end;

if (btnChek2=true) then

begin

btnChek2:=false;

Image3.Picture.LoadFromFile('appImg\1 уровень.jpg');

end;

if (btnChek3=true) then

begin

btnChek3:=false;

Image4.Picture.LoadFromFile('appImg\2 уровень.jpg');

end;

if (btnChek4=true) then

begin

btnChek4:=false;

Image5.Picture.LoadFromFile('appImg\3 уровень.jpg');

end;

if (btnChek5=true) then

begin

btnChek5:=false;

Image6.Picture.LoadFromFile('appImg\назад.jpg');

end;

end;

procedure TForm5.FormClose(Sender: TObject; var Action: TCloseAction);

begin

Form3.Close;

end;

end.

**Форма 6**

unit Unit6;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls;

type

TForm6 = class(TForm)

Button1: TButton;

Edit1: TEdit;

Label1: TLabel;

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form6: TForm6;

implementation

uses Unit1, Unit5;

{$R \*.dfm}

procedure TForm6.Button1Click(Sender: TObject);

var vedRaz:integer;

begin

if edit1.Text='' then showmessage('Введите целое число')

else if TryStrToInt(edit1.Text,vedRaz) then

begin

if vedRaz>32 then showmessage('Число должны быть меньше 32')else

if vedRaz=0 then showmessage('Число должно быть не равно нулю') else

if vedRaz<0 then

begin

vedraz:=vedRaz\*(-1);

razmer:=vedRaz;

edit1.Text:='';

pazlgo;

form6.Hide;

form5.Hide;

form1.Show;

end else

begin

razmer:=vedRaz;

edit1.Text:='';

pazlgo;

form6.Hide;

form5.Hide;

form1.Show;

end

end

else showmessage('Введите целое цисло');

edit1.Text:='';

end;

end.

**Форма 7**

unit Unit7;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, jpeg, ExtCtrls, StdCtrls;

type

TForm7 = class(TForm)

Image1: TImage;

Memo1: TMemo;

Button1: TButton;

Button2: TButton;

procedure FormShow(Sender: TObject);

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

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

type TPlayer = record

nickname:string[20];

time:string[8];

timer:integer;

end;

var

Form7: TForm7;

player:file of TPlayer;

playerRead:TPlayer;

implementation

uses Unit4;

{$R \*.dfm}

procedure TForm7.FormShow(Sender: TObject);

var n:integer;

begin

assignFile(player,'Players.txt');

reset(player);

n:=1;

while(not eof(player)) do

begin

read(player,playerRead);

Memo1.Lines.Add(IntToStr(n)+'. '+playerRead.nickname+' - '+playerRead.time);

n:=n+1;

end;

end;

procedure TForm7.FormClose(Sender: TObject; var Action: TCloseAction);

begin

Form4.Show;

Memo1.Clear;

end;

procedure TForm7.Button1Click(Sender: TObject);

begin

Form7.Close;

end;

procedure TForm7.Button2Click(Sender: TObject);

begin

assignFile(player,'Players.txt');

rewrite(player);

closeFile(player);

memo1.clear;

end;

end.

**Форма 8**

unit Unit8;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls;

type

TForm8 = class(TForm)

Label1: TLabel;

Button1: TButton;

Edit1: TEdit;

procedure Button1Click(Sender: TObject);

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

procedure FormShow(Sender: TObject);

private

{ Private declarations }

public

nickname:string[20];

pas\_h:integer;

{ Public declarations }

end;

var

Form8: TForm8;

implementation

uses Unit5, Unit4;

{$R \*.dfm}

procedure TForm8.Button1Click(Sender: TObject);

begin

pas\_h:=0;

if Edit1.Text='' then

ShowMessage('Введите хотя бы 1 символ')

else if Length(Edit1.Text)>20 then

begin

ShowMessage('Имя должны быть не длиннее 20 символов');

Edit1.Text:='';

end else

begin

nickname:=Edit1.Text;

if (nickname='smoke') or (nickname='смок')or (nickname='смоук') then pas\_h:=1;

form8.Hide;

form5.Show;

end;

end;

procedure TForm8.FormClose(Sender: TObject; var Action: TCloseAction);

begin

Form4.Show;

end;

procedure TForm8.FormShow(Sender: TObject);

begin

Edit1.Text:='';

end;

end.