**Введение**

В рамках курсового проектирования была поставлена задача разработки игрового приложения под названием "Dodge the shapes". Целью данного проекта является создание игрового приложения, которое поможет развивать навыки управления и реакции у пользователей.

Игровое приложение рассчитано на широкую аудиторию и предлагает увлекательный геймплей для всех возрастных групп. Игрокам предстоит управлять персонажем, который должен уклоняться от появляющихся на экране разнообразных фигур. Целью игры является пройти все уровни, избегая столкновения с фигурами. Это приложение способствует развитию навыков быстроты принятия решений, координации движений и улучшает реакцию игрока. Благодаря увлекательному и динамичному игровому процессу.

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 задачи.

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 “Списке используемых источников” будет приведен список используемой литературы, нормативно- техническую и другую документацию.

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 разработка игрового приложения под названием "Dodge the shapes". Игровое приложение создано с целью предоставления увлекательного и обучающего опыта для пользователей, аудиторией которых являются любители компьютерных игр.

В настоящее время игры на компьютерах и мобильных устройствах являются популярным способом проведения свободного времени. Однако, в данном случае мы сосредоточимся на разработке игры, специально предназначенной для развития навыков управления и реакции.

Игровое приложение создано для широкой аудитории пользователей, и его целью является предоставление увлекательного опыта, который поможет развивать навыки управления, реакции и принятия решений. Приложение предлагает пять уровней сложности, на каждом из которых игроку предстоит уклоняться от появляющихся на экране фигур. Целью игры является набрать максимальное количество очков, избегая столкновения с фигурами.

Для реализации проекта необходимо создать удобный пользовательский интерфейс, который позволит пользователям комфортно играть в приложение. Игра должна быть доступной для широкой аудитории пользователей, включая тех, кто не обладает специальными знаниями в области компьютерных игр.

В рамках разрабатываемого программного продукта будут реализованы следующие функции:

1. Управление персонажем для уклонения от фигур.
2. Генерация случайных фигур на экране.
3. Обнаружение столкновений персонажа с фигурами.
4. Завершение игры и отображение результата.
5. Возможность выхода из игры.

Важным аспектом разработки приложения является создание эффективного и интуитивно понятного алгоритма генерации фигур и обработки столкновений, чтобы игровой процесс был динамичным и интересным для игрока.

В разрабатываемой программе будут использоваться три вида данных: входная, выходная и постоянная информация.

К входной информации можно отнести загрузку сохранения. К выходной информации можно отнести результат прохождения уровня. Постоянной информацией в проекте будут являться текстовые файлы, картинки и др.

**1.2 Инструменты разработки**

Для разработки данного проекта будет выбрана среда Delphi 11, так как

это самое удобная и доступная среда разработки на данный момент. Delphi 11-язык –программирования, относящийся к классу RAD- (Rapid Application Development – «Средство быстрой разработки приложений») средств CASE –тeхнологии. Delphi 11 сделал разработку приложений дляWindows быстрым и приятным процессом. Теперь разрабатывать сложные и интересные проекты можно только одним человеком, использующим Delphi 11

Интерфейс Windows обеспечивает полное перенесение CASE-технологии в интегральную систему поддержки работ по созданию прикладной системы на всех фазах жизненного цикла работы и проектирования системы.

Delphi 11 обладает широким набором возможностей, начиная от проектировщика форм и заканчивая поддержкой всех форматов популярных баз данных. Среда устраняет необходимость программировать такие компоненты Windows общего назначения, как метки, программы и даже диалоговые панели. Работая в Windows, можно видеть одинаковые «объекты» во многих разнообразных приложениях. Диалоговые панели (например, Choose File и Save File) являются примерами многократно-используемых компонентов, встроенных непосредственно в Delphi 11, который позволяет приспособить эти компоненты к имеющийся задаче, чтобы они работали именно так, как требуется создаваемому приложению. Также здесь имеются предварительно-определенные визуальные и не визуальные объекты.

Три основные части разработки интерфейса следующие: проектирование панели, проектирование диалога и представление окон. Для общего пользовательского доступа также должны учитываться условия применения архитектуры прикладных систем.

Сегодня появилась реальная возможность с помощью моделирования на современных многофункциональных средствах обработки и отображения информации таких как Delphi 11 конкретизировать тип и характеристики используемых информационных моделей, выявить основные особенности будущей деятельности операторов, сформулировать требования к параметрам аппаратно-программных средств интерфейса взаимодействия и т.д. Delphi 11 позволяет создать различные виды программ: консольные приложения, оконные приложения, приложения для работы с Интернетом и базами данных. То есть, Delphi 11 является не только средствами для работы с языком программирования Паскаль, но дополнительные инструменты, призванные для максимального упрощения и ускорения создание приложений.

К дополнительным инструментам можно отнести визуальный редактор форм, благодаря которому можно с легкостью создать полноценную программу, и другие визуальные составляющие разработки программного обеспечения. С Delphi вам не нужно вручную просчитывать расположение каждого элемента интерфейса пользователя, поэтому при разработке программы значительно экономится время.

Выгоды от проектирования в среде Windows с помощью Delphi 11:

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

Для создания инсталлятора будет использоваться мощное и удобное средство - Smart Install Maker. Программа обладает удобным и интуитивно понятным интерфейсом, а также полным набором необходимых функций для создания профессиональных инсталляторов с минимальным размером, высокой степенью сжатия файлов и приятным интерфейсом.

Помимо стандартного минимума, Smart Install Maker позволяет редактировать системный реестр и INI-файлы, создавать программные ярлыки, запускать ассоциируемые и исполняемые файлы, регистрировать новые шрифты и ActiveX компоненты, отображать тексты информации и лицензионного соглашения. Также, с помощью этой утилиты, можно создать мультиязыковые инсталляторы с поддержкой более 20-ти популярных языков мира.

Microsoft Word 2021 – редактор текста для написания документации.

Разработка ведется на ноутбуке Asus. У данного ноутбука следующие

параметры:

* процессор Ryzen 5;
* объем ОЗУ 16 гб;
* объем места на SSD – 512 гб;
* видеокарта AMD Radeon Vega 8 Graphics;
* ОС – Windows 11.

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 установлен целый ряд требований, которые предъявляются к разрабатываемой программе.

Накладываемые ограничения: приложение должно быть совместимо с операционными системами, поддерживающими Delphi 11, такими как Windows, macOS или Linux. Приложение не должно потреблять избыточные системные ресурсы, чтобы обеспечить плавную игровую производительность.

Требования к интерфейсу: приложение должно иметь главное меню, где игрок может выбрать начать новую игру или выйти из игры. Игровой экран должен содержать игровое поле, на котором отображаются игровые объекты, такие как игрок и враги. Приложение должно предоставить удобное и понятное управление игровым персонажем. Это может быть реализовано с помощью клавиатуры.

Пользователь не должен испытывать какого-либо дискомфорта в плане восприятия информация, отображённой на экране. Объекты (рисунки и символы) не должны быть слишком мелкие. Все окна приложения по возможности должны помещаться на экран полностью, так как использование в процессе работы полос прокруток достаточно неудобно.

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

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

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

**2. Проектирование задачи**

**2.1 Организация данных**

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

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

Необходимым условием при разработке данного приложения является описание организации данных, т.е. логическая и физическая структура данных

в контексте среды разработки. В разрабатываемой программе будут использоваться три вида данных.

Первым видом являются данные, которые будут введены разработчиком на этапе реализации задачи. Сюда можно отнести изображения (иконки), описание, исходные коды сортировок.

Вторым видом данных, используемых в программе, является вводимая пользователем информация. Входной информацией в разрабатываемой программе будут являться данные. Подходящей структурой данных для хранения элементов является: динамический одномерный массив.

Третьим видом данных является результат программы –отсортированные пользовательские данные. Его также относят к отдельному виду, так как ни пользователь, ни разработчик его не вводят, а программа сама получает его в результате выполнения определенных действий.

Таким образом, организация данных является важной задачей при разработке данной и любой программы.

**2.2 Процессы**

Процессы в игровом приложении "Dodge the shapes" выполняют ряд основных функций, обеспечивающих игровой процесс и взаимодействие с игроком. Вот описание этих процессов в виде предложений:

1. Управление персонажем: Пользователь может управлять персонажем в игре, используя определенные клавиши. Этот процесс отслеживает ввод пользователя и передает соответствующие команды для перемещения персонажа по полю.
2. Обнаружение столкновений: Процесс обнаружения столкновений отслеживает взаимодействие персонажа с препятствиями. При пересечении границ персонажа с геометрическими фигурами происходит столкновение, что приводит к завершению уровня.
3. Сохранение прогресса: В игре реализован процесс сохранения прогресса, который позволяет игроку возобновлять игру с последнего пройденного уровня. При достижении нового уровня или завершении игры текущий прогресс сохраняется и может быть загружен в следующий раз.
4. Генерация врагов: Игровое приложение содержит три вида врагов. Процесс генерации врагов отвечает за создание и размещение вражеских объектов на поле, которые перемещаются в заданном направлении и представляют опасность для персонажа.

Эти процессы обеспечивают функциональность игры "Dodge the shapes" и позволяют игроку насладиться увлекательным игровым процессом, управлять персонажем, избегать препятствий и достигать новых уровней.

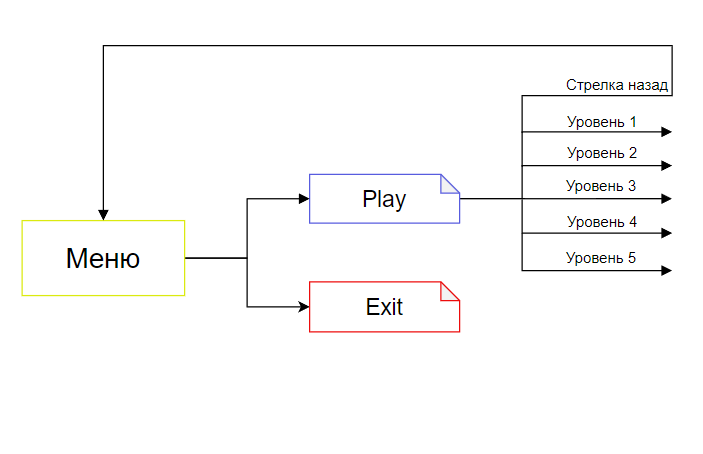
**2.3 Описание внешнего пользовательского интерфейса**

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

Для организации эффективной работы пользователя нужно создать целостное приложение данной предметной области, в которой все компоненты приложения будут сгруппированы по функциональному назначению. При этом необходимо обеспечить удобный графический интерфейс пользователя. Приложение должно позволить пользователю решать задачи, затрачивая значительно меньше усилий, чем при работе с разрозненными объектами.

Ниже на рисунке 1 представлена система организация навигации:

Рисунок 1 – Схема навигация



**3. Реализация**

**3.1 Реализация проекта**

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

Unit 2- является заставкой в курсовом проекте;

Unit 1-содержит код и компоненты для игры.

**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 закреплена | Назначение |
| 1 | 2 | 3 | 4 |
| 1 procedure TForm2.Timer1Timer(Sender: TObject); | Unit 2 | Timer1 | Отключение заставки, переход к Unit 1 |
| 2 procedure Media(text:string); | Unit 1 | - | Проигрывание музыки или звуков |
| 3 procedure TForm1.ApplicationEvents1Message(var Msg: tagMSG; var Handled: Boolean); | Unit 1 | ApplicationEvents1 | Считывание нажатия клавиш |
| 4 function CheckCollision(X1, Y1, Width1, Height1, X2, Y2, Width2, Height2: Integer): Boolean; | Unit 1 | - | Проверка столкновения игрока и врага |
| 5 procedure TForm1.TimerPlayerTimer(Sender: TObject); | Unit 1 | TimerPlayer | Обновление позиции игрока |
| 6 procedure TForm1.checkTimer(Sender: TObject); | Unit 1 | Check | Обновление проверки столкновения и прочее |
| 7 procedure TForm1.FormCreate(Sender: TObject); | Unit 1 | Form1 | Настройка формы, загрузка сохранения и создание массива таймеров |
| 8 procedure TForm1.DisableAllTimers; | Unit 1 | - | Отключение всех таймеров |
| 9 procedure TForm1.Image1Click(Sender: TObject); | Unit 1 | Image1 | Переход к уровням |

|  |  |  |  |
| --- | --- | --- | --- |
| Продолжение таблицы 1 | | | |
| 1 | 2 | 3 | 4 |
| 10 procedure TForm1.Image7Click(Sender: TObject); | Unit 1 | Image7 | Выход из игры |
| 11 procedure TForm1.backClick(Sender: TObject); | Unit 1 | Back | Кнопка «Назад» |
| 12 procedure TForm1.DeathPlayer; | Unit 1 | - | Возврат к выбору уровня |
| 13 procedure Tform1.normalaz; | Unit 1 | - | Возвращает всё как было до начала уровня |
| 14 procedure TForm1.winTTimer(Sender: TObject); | Unit 1 | Win | Отображение картинки победы |
| 15 procedure TForm1.loseTTimer(Sender: TObject); | Unit 1 | Lose | Отображение картинки поражения |
| 16 procedure TForm1.Image2Click(Sender: TObject); | Unit 1 | Image2 | Переход к первому уровню |
| 17 procedure TForm1.Image3Click(Sender: TObject); | Unit 1 | Image3 | Переход ко второму уровню |
| 18 procedure TForm1.Image4Click(Sender: TObject); | Unit 1 | Image4 | Переход к третьему уровню |
| 19 procedure TForm1.Image5Click(Sender: TObject); | Unit 1 | Image5 | Переход к четвертому уровню |
| 20 procedure TForm1.Image6Click(Sender: TObject); | Unit 1 | Image6 | Переход к пятому уровню |
| 21 procedure TForm1.LVL1\_5secTimer(Sender: TObject); | Unit 1 | LVL1\_5sec | Отображение управления и текста |
| 22 procedure CreateEnemy(AForm: TForm; postop,posleft:integer; direction:integer=0); | Unit 1 | - | Создание врага «Круг» |
| 23 procedure TForm1.createTimer(Sender: TObject); | Unit 1 | create | Создание врагов |
| 24 procedure TForm1.leftTimer(Sender: TObject); | Unit 1 | Left | Обновление позиции врага |
| 25 procedure TForm1.clearTimer(Sender: TObject); | Unit 1 | Clear | Очистка от врагов |
| 26 procedure TForm1.LVL1\_2\_5secTimer(Sender: TObject); | Unit 1 | LVL1\_2\_5sec | 5 секунд задержки |
| 27 procedure TForm1.create2Timer(Sender: TObject); | Unit 1 | Create2 | Создание врагов |
| 28 procedure TForm1.leftANDrightTimer(Sender: TObject); | Unit 1 | leftANDright | Обновление позиции врага |
| 29 procedure CreateSquare(AForm: TForm; posleft, postop: Integer; direction:integer=0 ); | Unit 1 | - | Создание врага «Квадрат» |
| 30 procedure TForm1.LVL2\_createTimer(Sender: TObject); | Unit 1 | LVL2\_create | Создание врагов |
| Продолжение таблицы 1 | | | |
| 1 | 2 | 3 | 4 |
| 31 procedure TForm1.LVL2\_3secTimer(Sender: TObject); | Unit 1 | LVL2\_3sec | 3 секунды задержки |
| 32 procedure TForm1.LVL2\_create2Timer(Sender: TObject); | Unit 1 | LVL2\_create2 | Создание врагов |
| 33 procedure TForm1.LVL2\_createSTimer(Sender: TObject); | Unit 1 | LVL2\_createS | Создание врагов |
| 34 procedure TForm1.LVL2\_moveTimer(Sender: TObject); | Unit 1 | LVL2\_move | Обновление позиции врага |
| 35 procedure CreateRectangle(AForm: TForm); | Unit 1 | - | Создание врага «Квадратик» |
| 36 procedure MoveEnemiesToPlayer(AForm: TForm; PlayerX, PlayerY: Integer); | Unit 1 | - | Движение «Квадратика» к игроку |
| 37 procedure TForm1.LVL3\_3secTimer(Sender: TObject); | Unit 1 | LVL3\_3sec | 3 секунды задержки |
| 38 procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction); | Unit 1 | Form1 | Закрытие игры, при ALT + F4 |
| 39 procedure TForm1.LVL3\_createTimer(Sender: TObject); | Unit 1 | LVL3\_create | Создание врагов |
| 40 procedure TForm1.LVL3\_moveTimer(Sender: TObject); | Unit 1 | LVL3\_move | Обновление позиции врага |
| 41 procedure TForm1.LVL4\_3secTimer(Sender: TObject); | Unit 1 | LVL4\_3sec | 3 секунды задержки |
| 42 procedure TForm1.LVL4\_3sec2Timer(Sender: TObject); | Unit 1 | LVL4\_3sec2 | 40 секундный таймер, время, которое нужно уворачиваться |
| 43 procedure TForm1.LVL4\_createTimer(Sender: TObject); | Unit 1 | LVL4\_create | Создание врагов |
| 44 procedure TForm1.LVL4\_create2Timer(Sender: TObject); | Unit 1 | LVL4\_create2 | Создание врагов |
| 45 procedure TForm1.LVL4\_create3Timer(Sender: TObject); | Unit 1 | LVL4\_create3 | Создание врагов |
| 46 procedure TForm1.LVL5\_3secTimer(Sender: TObject); | Unit 1 | LVL5\_3secTimer | 3 секунды задержки |
| 47 procedure TForm1.LVL5\_createTimer(Sender: TObject); | Unit 1 | LVL5\_create | Создание врагов |
| 48 procedure TForm1.LVL5\_create2Timer(Sender: TObject); | Unit 1 | LVL5\_create2 | Создание врагов |

**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 |
| TMediaPlayer | Form1 | Проигрывание музыки, звуков |
| TImage | Form2, Form1 | Используется как кнопки, фон |
| TTimer | Form2, Form1 | Используется для задержки |
| TLabel | Form1 | Отображение надписей на форме |
| TApplicationEvents | Form1 | Используется для считывания нажатия клавиш |

**3.2 Спецификация программы**

Точное название проекта и его состав приводится в таблице 3.

Таблица 3 – Спецификация программы

|  |  |
| --- | --- |
| Имя файла | Назначение |
| 1 | 2 |
| DodheTheShapes.exe | Исполняемый файл проекта, используется для запуска программы на выполнение. |
| DodheTheShapes.dproj | Файл проекта, связывает все файлы из которых состоит приложение. |
| DodheTheShapes.exe | Ярлык исполняемого файла проекта |
| DodgeTheShapes.dpr | Главный файл проекта и содержит информацию о структуре проекта |
| DodgeTheShapes.dproj | Содержит информацию о настройках и параметрах проекта |
| Unit1.dfm | Хранит информации о формах пользовательского интерфейса |
| Unit2.dfm | Хранит информации о формах пользовательского интерфейса |
| Unit1.pas | Содержит исходные коды самой игры |
| Unit2.pas | Содержит исходные коды загрузочного экрнга |
| DodgeTheShapes.dproj.local | Используются для хранения локализации и перевода текстовых ресурсов приложения |
| DodgeTheShapes.identcache | Используется компилятором для ускорения процесса компиляции исходного кода |
| DodgeTheShapes.res | Используется для хранения различных типов ресурсов, таких как иконки, изображения, звуки, строки и другие элементы |
| progres\_point.txt | Хранит пройденные уровни |
| Uninstall.exe | Инсталятор игры |
| Unit2.vlb | Файл виртуальной библиотеки |
| Unit1.dcu | Используются для ускорения процесса сборки и повторной компиляции проекта |
| Продолжение таблицы 3 | | | |
| 1 | | 2 | |
| Unit2.dcu | | Используются для ускорения процесса сборки и повторной компиляции проекта | |
| Dodge The shapes.exe | | Установочный пакет программы | |
| 1.png | | Изображение | |
| 1z.png | | Изображение | |
| 2.png | | Изображение | |
| 2z.png | | Изображение | |
| 3.png | | Изображение | |
| 3z.png | | Изображение | |
| 4.png | | Изображение | |
| 4z.png | | Изображение | |
| 5.png | | Изображение | |
| 5z.png | | Изображение | |
| 04.mp3 | | Музыка | |
| 06.mp3 | | Музыка | |
| 08.mp3 | | Музыка | |
| 011.mp3 | | Музыка | |
| Win.mp3 | | Звук победы | |
| Lose.mp3 | | Звук поражения | |

**4. Тестирование**

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

Отчет о результатах тестирования представлен в таблице 4

Таблица 4- Отчет о результатах тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Идентификатор теста | Тест | Ожидаемый результат | Физический результат | Результат тестирование |
| 1 | Проверка входа  в приложение | Открытие главного окна | Открытие главного  окна | Выполнено |
| 2 | Проверка кнопки  «Play» | Появление уровней | Появление уровней | Выполнено |
| 3 | Проверка кнопки  «Exit» | Выход из игры | Выход из игры | Выполнено |
| 4 | Проверка комбинации ALT + F4 | Выход из игры | Выход из игры | Выполнено |
| 5 | Проверка кнопки первого уровня | Запуск первого уровня | Запуск первого уровня | Выполнено |
| 6 | Проверка кнопки второго уровня | Запуск второго уровня | Запуск второго уровня | Выполнено |
| 7 | Проверка кнопки третьего уровня | Запуск третьего уровня | Запуск третьего уровня | Выполнено |
| 8 | Проверка кнопки четвертого уровня | Запуск четвертого уровня | Запуск четвертого уровня | Выполнено |
| 9 | Проверка кнопки пятого уровня | Запуск пятого уровня | Запуск пятого уровня | Выполнено |
| 10 | Проверка передвижения игрока | Игрок двигается | Игрок двигается | Выполнено |
| 14 | Проверка создание врагов | Враги создаются | Враги создаются | Выполнено |
| 15 | Проверка передвижения врагов | Враги двигаются | Враги двигаются | Выполнено |
| 16 | Проверка проигрывания музыки и звуков | Музыка и звуки проигрывается | Музыка и звуки проигрывается | Выполнено |
| 17 | Проверка загрузки сохранения | Сохранение загружено | Сохранение загружено | Выполнено |

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

**5 Применение**

**5.1 Общие сведения о программном продукте**

Целью данного проекта является создание игрового приложения, где игрок управляет персонажем и избегает столкновения с различными фигурами, которые движутся по экрану. Цель игры - продержаться как можно дольше, пройдя уровень.

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 приложение запускалось. Программа разработана на ПК со следующими характеристиками:

* процессор AMD Ryzen 5 2500H with Radeon Graphics, 3600 МГц, ядер: 4, логических процессоров: 8
* объем ОЗУ 16gb;
* графический адаптер AMD Radeon Vega 8 Graphics 512mb;
* операционная система Windows 11.

**5.2 Инсталляция**

Для того, чтобы установить программу необходимо запустить файл-установщик «Dodge the shapes.exe». Появится окно установки приложения “ Установка Dodge the shapes”.

Затем достаточно следовать приведенной инструкции установки приложения.

**5.3 Выполнение программы**

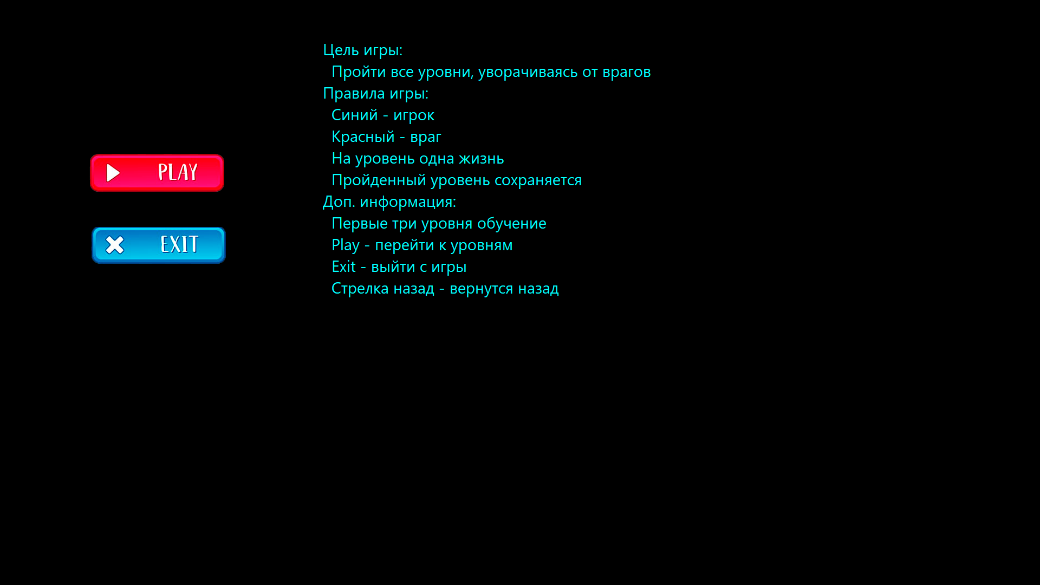
**5.3.1 Запуск программы**

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 «Dodge the shapes».

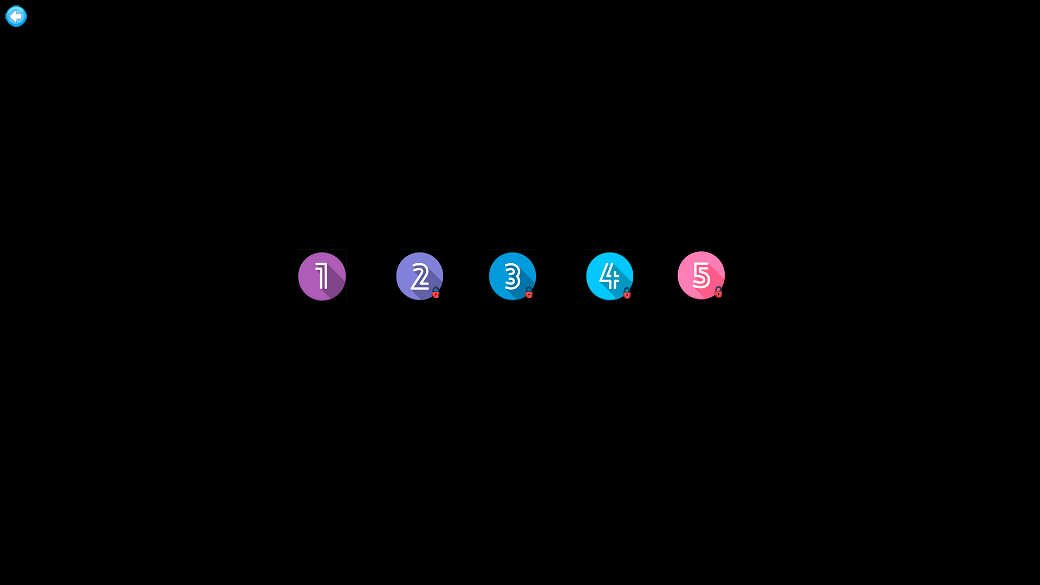
Вторым способом является запуск из каталога, в который устанавливалось приложение (по умолчанию C:\Program Files (x86)\Dodge the shapes).

**5.3.2 Инструкции по работе с программой**

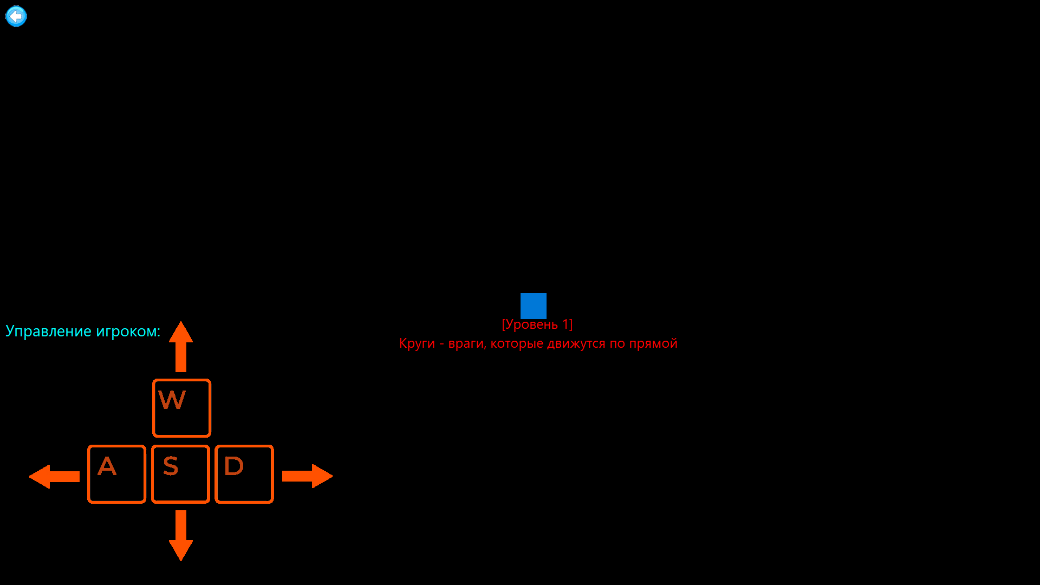
После запуска приложения на экране нас встречает заставка на рисунок 2.

Рисунок 2 – Заставка

При нажатии на кнопку “Play” будет осуществлен переход к выбору уровню, представленный на рисунке 3, при нажатии на кнопку “Exit” будет осуществлен выход из игры.

Рисунок 3- Выбор уровня

По нажатию на кнопку уровня будет осуществлен переход к уровню. При нажатии на стрелку назад произойдет возврат к главному меню. Далее рассмотрим сами уровень 1, представленный на рисунке 4.

Рисунок 4- Уровень 1

Кнопками W, A, S, D осуществляется управление игроком. При нажатии на стрелку назад произойдет переход к выбору уровней. Если игрок столкнётся с врагом или нажмёт на стрелку назад, то произойдет проигрыш уровня, представленный на рисунке 5. Если игрок сможет продержатся определённое время, то победит, победа уровня представлена на рисунке 6.

Рисунок 5- Проигрыш уровня

Рисунок 6- Победа уровня

Следующие уровни идентичны уровню 1.

**Заключение**

В рамках данного курсового проекта было разработано игровое приложение под названием "Dodge the shapes" с использованием Delphi 11. Игра содержит пять уровней и три вида врагов.

В поставленной задаче был реализован простой и понятный пользовательский интерфейс. Было создано пять уровней с постепенно увеличивающейся сложностью. Каждый уровень предлагает новые вызовы и препятствия для игрока

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

В процессе разработки программного продукта я научился создавать динамические компоненты на форме, проработал их взаимодействие, закрепил умение создания собственных процедур, научился обрабатывать все исключительные ситуации.

Исходя из этого, можно сделать вывод, что программа реализована успешно.

**Список использованных источников**

1. Симонович С.В., Евсеев Г.А., Алексеев А.Г. Специальная информатика: Учебное пособие. - М.: АСТ-ПРЕСС: Инфорком-Пресс, 2001. - 480 с.
2. Архангельский, А. Я. Delphi 7. Справочное пособие. - Москва: Бином-Пресс, 2014. - 1024 с.
3. Вощинская, Г.Э. Разработка компонентов в DELPHI. - Воронеж: ИПЦ ВГУ, 2007. - 57 с.
4. Культин, Н. Основы программирования в Delphi 7. - Санкт-Петербург: Питер, 2009. - 640 с.
5. Архангельский, А. Я. Delphi 7. Справочное пособие. - Москва: Бином-Пресс, 2014. - 1024 с.
6. Культин, Н. Delphi 6. Программирование на Object Pascal / Н. Культин. - М.: БХВ-Петербург, 2012. - 528 c.
7. Музыка - <https://downloads.khinsider.com/game-soundtracks/album/just-shapes-beats-2018>
8. Звуки - https://zvukogram.com/category/

**Приложение А**

Листинг программы

unit Unit2;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.Imaging.pngimage;

type

TForm2 = class(TForm)

Timer1: TTimer;

Image1: TImage;

procedure Timer1Timer(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form2: TForm2;

implementation

{$R \*.dfm}

uses Unit1;

// закрытие заставки и открытие Unit1

procedure TForm2.FormCreate(Sender: TObject);

begin

end;

procedure TForm2.Timer1Timer (Sender: TObject);

begin

form1.show;

form2.Hide;

Timer1.Enabled:=false;

end;

end.

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.AppEvnts, Vcl.ComCtrls,

Vcl.StdCtrls, Vcl.Buttons, Vcl.Imaging.jpeg, math, System.ImageList,

Vcl.ImgList, Vcl.Imaging.pngimage, Vcl.MPlayer;

type

TForm1 = class(TForm)

Player: TShape;

TimerPlayer: TTimer;

ApplicationEvents1: TApplicationEvents;

Image1: TImage;

Image2: TImage;

Image3: TImage;

Image4: TImage;

Image5: TImage;

Image6: TImage;

check: TTimer;

create: TTimer;

left: TTimer;

create2: TTimer;

leftANDright: TTimer;

clear: TTimer;

MainText: TLabel;

LVL1\_5sec: TTimer;

LVL1\_2\_5sec: TTimer;

LVL2\_move: TTimer;

LVL2\_create: TTimer;

LVL2\_createS: TTimer;

Image7: TImage;

back: TImage;

LVL3\_create: TTimer;

LVL3\_move: TTimer;

wasd: TImage;

Label1: TLabel;

Label2: TLabel;

win: TImage;

lose: TImage;

winT: TTimer;

loseT: TTimer;

LVL2\_create2: TTimer;

LVL2\_3sec: TTimer;

LVL3\_3sec: TTimer;

LVL4\_3sec: TTimer;

LVL5\_3sec: TTimer;

LVL4\_create: TTimer;

LVL4\_3sec2: TTimer;

LVL4\_create2: TTimer;

LVL4\_create3: TTimer;

LVL5\_create: TTimer;

LVL5\_create2: TTimer;

MPlayer: TMediaPlayer;

procedure TimerPlayerTimer(Sender: TObject);

procedure ApplicationEvents1Message(var Msg: tagMSG; var Handled: Boolean);

procedure FormCreate(Sender: TObject);

procedure checkTimer(Sender: TObject);

procedure createTimer(Sender: TObject);

procedure Image1Click(Sender: TObject);

procedure Image2Click(Sender: TObject);

procedure create2Timer(Sender: TObject);

procedure leftANDrightTimer(Sender: TObject);

procedure clearTimer(Sender: TObject);

procedure leftTimer(Sender: TObject);

procedure LVL1\_5secTimer(Sender: TObject);

procedure LVL1\_2\_5secTimer(Sender: TObject);

procedure Image3Click(Sender: TObject);

procedure LVL2\_createTimer(Sender: TObject);

procedure LVL2\_moveTimer(Sender: TObject);

procedure LVL2\_createSTimer(Sender: TObject);

procedure Image7Click(Sender: TObject);

procedure backClick(Sender: TObject);

procedure LVL3\_moveTimer(Sender: TObject);

procedure LVL3\_createTimer(Sender: TObject);

procedure Image4Click(Sender: TObject);

procedure Image5Click(Sender: TObject);

procedure Image6Click(Sender: TObject);

procedure winTTimer(Sender: TObject);

procedure loseTTimer(Sender: TObject);

procedure LVL2\_create2Timer(Sender: TObject);

procedure LVL2\_3secTimer(Sender: TObject);

procedure LVL3\_3secTimer(Sender: TObject);

procedure LVL4\_3secTimer(Sender: TObject);

procedure LVL5\_3secTimer(Sender: TObject);

procedure LVL4\_createTimer(Sender: TObject);

procedure LVL4\_create2Timer(Sender: TObject);

procedure LVL4\_3sec2Timer(Sender: TObject);

procedure LVL4\_create3Timer(Sender: TObject);

procedure LVL5\_createTimer(Sender: TObject);

procedure LVL5\_create2Timer(Sender: TObject);

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

private

{ Private declarations }

Timers: array of TTimer;

procedure DeathPlayer;

procedure DisableAllTimers;

procedure normalaz;

public

{ Public declarations }

end;

const

WM\_KEYDOWN = $0100;

WM\_KEYUP = $0101;

var

PlayerSpeed:integer; // скорость игрока

Pspeed:integer=10;

EnemySpeed:integer = 16;

Form1: TForm1;

PlayerX: Integer; // Координаты игрока

PlayerY: Integer; // Координаты игрока

MoveVector: TPoint;

IsMovingUp, IsMovingDown, IsMovingLeft, IsMovingRight, IsSpace: Boolean; // Флаги направления движения

b,c,d:integer;

progres\_point:integer; // Сохранение уровней

F: Textfile;

Enemies: array of TShape; // массив врагов

Enemies2: array of TShape; // массив квадратиков

Square: TShape;

implementation

{$R \*.dfm}

uses Unit2;

// Проигрывание музыки и звуков

procedure Media(text:string);

begin

form1.MPlayer.FileName := text;

form1.MPlayer.Open;

form1.MPlayer.play;

end;

// считованние нажатия клавиш

procedure TForm1.ApplicationEvents1Message(var Msg: tagMSG; var Handled: Boolean);

begin

if (Msg.message = WM\_KEYDOWN) or (Msg.message = WM\_KEYUP) then

begin

case Msg.wParam of

Ord('W'), Ord('w'):

begin

if Msg.message = WM\_KEYDOWN then

IsMovingUp := True

else

IsMovingUp := False;

end;

Ord('A'), Ord('a'):

begin

if Msg.message = WM\_KEYDOWN then

IsMovingLeft := True

else

IsMovingLeft := False;

end;

Ord('S'), Ord('s'):

begin

if Msg.message = WM\_KEYDOWN then

IsMovingDown := True

else

IsMovingDown := False;

end;

Ord('D'), Ord('d'):

begin

if Msg.message = WM\_KEYDOWN then

IsMovingRight := True

else

IsMovingRight := False;

end;

32:

begin

if Msg.message = WM\_KEYDOWN then

IsSpace := true

else

IsSpace := False;

end;

end;

end;

end;

// функция проверки столкновения

function CheckCollision(X1, Y1, Width1, Height1, X2, Y2, Width2, Height2: Integer): Boolean;

begin

Result := (X1 < X2 + Width2) and (X1 + Width1 > X2) and

(Y1 < Y2 + Height2) and (Y1 + Height1 > Y2);

end;

//движение игрока

procedure TForm1.TimerPlayerTimer(Sender: TObject);

begin

MoveVector := Point(0, 0);

PlayerSpeed:=Pspeed;

if IsMovingUp then

MoveVector.Y := MoveVector.Y - 1;

if IsMovingLeft then

MoveVector.X := MoveVector.X - 1;

if IsMovingDown then

MoveVector.Y := MoveVector.Y + 1;

if IsMovingRight then

MoveVector.X := MoveVector.X + 1;

// границы для игрока

if Player.top < 1 then

MoveVector.Y := MoveVector.Y+1;

if Player.Left < 1 then

MoveVector.X:=MoveVector.X+1;

if Player.top+52 > Form1.Height then

MoveVector.Y:=MoveVector.Y-1;

if Player.Left+50 > Form1.Width then

MoveVector.X := MoveVector.X-1;

// уменьшение скорости для диагонального движения

if (IsMovingUp and IsMovingLeft) or (IsMovingUp and IsMovingRight) or

(IsMovingDown and IsMovingLeft) or (IsMovingDown and IsMovingRight)

then PlayerSpeed:=Round(PlayerSpeed\*0.7);

PlayerX := PlayerX + (MoveVector.X \* PlayerSpeed);

PlayerY := PlayerY + (MoveVector.Y \* PlayerSpeed);

// Обновляем позицию игрока на форме

Player.Left := PlayerX;

Player.Top := PlayerY;

end;

// таймер проверки

procedure TForm1.checkTimer(Sender: TObject);

var

I: Integer;

begin // запуск музыки в меню, если не играет другая музыка

if not (MPlayer.Mode = mpPlaying) then

Media('04.mp3');

// если есть кнопка play, то отключает кнопки уровней

if image1.Visible then

begin

label2.Visible:=true;

image2.Visible:=false;

image3.Visible:=false;

image4.Visible:=false;

image5.Visible:=false;

image6.Visible:=false;

end

else

label2.Visible:=false;

// проверка столкновения врагов с игроком

for I := 0 to Length(Enemies) - 1 do

begin

// Проверка столкновения врага с игроком

if CheckCollision(PlayerX, PlayerY, Player.Width, Player.Height,

Enemies[I].Left, Enemies[I].Top, Enemies[I].Width, Enemies[I].Height) then

begin

normalaz;

loseT.Enabled :=true;

lose.Visible :=true;

Media('lose.mp3');

end;

end;

// проверка столкновения квадратиков с игроком

for I := 0 to Length(Enemies2) - 1 do

begin

if CheckCollision(PlayerX, PlayerY, Player.Width, Player.Height,

Enemies2[I].Left, Enemies2[I].Top, Enemies2[I].Width, Enemies2[I].Height) then

begin

normalaz;

loseT.Enabled :=true;

lose.Visible :=true;

Media('lose.mp3');

end;

end;

end;

// главное меню

procedure TForm1.FormCreate(Sender: TObject);

begin

label2.Caption:=

'Цель игры:' + #13#10 +

' Пройти все уровни, уворачиваясь от врагов' + #13#10 +

'Правила игры:' + #13#10 +

' Синий - игрок' + #13#10 +

' Красный - враг' + #13#10 +

' На уровень одна жизнь'+ #13#10 +

' Пройденный уровень сохраняется'+ #13#10 +

'Доп. информация:' + #13#10 +

' Первые три уровня обучение'+ #13#10 +

' Play - перейти к уровням'+ #13#10 +

' Exit - выйти с игры'+ #13#10 +

' Стрелка назад - вернутся назад';

PlayerX:= round(Screen.Width/2);

PlayerY:= round(Screen.Height/2);

Form1.WindowState:=wsMaximized;

Form1.BorderStyle:=bsNone;

Form1.Width:=Screen.Width;

Form1.height:=Screen.Height;

AssignFile(F, 'progres\_point.txt');

Reset(F);

Readln(F,progres\_point);

closefile(F);

// загрузка сохранения

case progres\_point of

1: begin

Image3.Picture.LoadFromFile('2.jpg');

end;

2: begin

Image3.Picture.LoadFromFile('2.jpg');

Image4.Picture.LoadFromFile('3.jpg');

end;

3: begin

Image3.Picture.LoadFromFile('2.jpg');

Image4.Picture.LoadFromFile('3.jpg');

Image5.Picture.LoadFromFile('4.jpg');

end;

4: begin

Image3.Picture.LoadFromFile('2.jpg');

Image4.Picture.LoadFromFile('3.jpg');

Image5.Picture.LoadFromFile('4.jpg');

Image6.Picture.LoadFromFile('5.jpg');

end;

end;

// Массив таймеров

SetLength(Timers,22);

Timers[0] := create;

Timers[1] := left;

Timers[2] := create2;

Timers[3] := leftANDright;

Timers[4] := TimerPlayer;

Timers[5] := LVL1\_5sec;

Timers[6] := LVL1\_2\_5sec;

Timers[7] := LVL2\_create;

Timers[8] := LVL2\_create2;

Timers[9] := LVL2\_createS;

Timers[10] := LVL2\_3sec;

Timers[11] := LVL2\_move;

Timers[12] := LVL3\_create;

Timers[13] := LVL3\_move;

Timers[14] := LVL4\_create;

Timers[15] := LVL4\_create2;

Timers[16] := LVL4\_create3;

Timers[17] := LVL4\_3sec;

Timers[18] := LVL4\_3sec2;

Timers[19] := LVL5\_create;

Timers[20] := LVL5\_create2;

Timers[21] := LVL5\_3sec;

end;

// отключение всех таймеров

procedure TForm1.DisableAllTimers;

var

i: Integer;

begin

for i := Low(Timers) to High(Timers) do

begin

Timers[i].Enabled := False;

end;

end;

// переход к выбору уровней

procedure TForm1.Image1Click(Sender: TObject);

begin

image1.Visible:=false;

image2.Visible:=true;

image3.Visible:=true;

image4.Visible:=true;

image5.Visible:=true;

image6.Visible:=true;

image7.Visible:=false;

back.Visible:=true;

label2.Visible :=false;

end;

// вызод из игры

procedure TForm1.Image7Click(Sender: TObject);

begin

application.Terminate;

end;

// кнопка назад, если во время уровня нажать, то уровень проигран

procedure TForm1.backClick(Sender: TObject);

begin

if TimerPlayer.Enabled then

begin

Media('lose.mp3');

normalaz;

loseT.Enabled :=true;

lose.Visible :=true;

wasd.Visible :=false;

label1.Visible :=false;

end

else

begin

image1.Visible:=true;

image2.Visible:=false;

image3.Visible:=false;

image4.Visible:=false;

image5.Visible:=false;

image6.Visible:=false;

image7.Visible:=true;

back.Visible:=false;

wasd.Visible :=false;

label1.Visible :=false;

normalaz;

end;

end;

// переход к выбору уровней, если игрок столкнулся

procedure TForm1.DeathPlayer;

begin

image2.Visible:=true;

image3.Visible:=true;

image4.Visible:=true;

image5.Visible:=true;

image6.Visible:=true;

end;

// возращет все как было до уровня

procedure Tform1.normalaz;

begin

MainText.Caption:='';

player.Visible:=false;

DisableAllTimers;

clear.Enabled:=true;

b:=0;

PlayerX:= round(Screen.Width/2);

PlayerY:= round(Screen.Height/2);

end;

// победа

procedure TForm1.winTTimer(Sender: TObject);

begin

DeathPlayer;

win.Visible :=false;

winT.Enabled :=false;

end;

// поражение

procedure TForm1.loseTTimer(Sender: TObject);

begin

DeathPlayer;

lose.Visible :=false;

loseT.Enabled :=false;

end;

// проверка нажатия уровня 1

procedure TForm1.Image2Click(Sender: TObject);

begin

Media('08.mp3');

MainText.Caption:=

' [Уровень 1]' + #13#10 +

'Круги - враги, которые движутся по прямой';

image2.Visible:=false;

image3.Visible:=false;

image4.Visible:=false;

image5.Visible:=false;

image6.Visible:=false;

LVL1\_5sec.Enabled:=true;

player.Visible:=true;

TimerPlayer.Enabled:=true;

wasd.visible :=true;

label1.Visible :=true;

end;

// проверка нажатия уровня 2

procedure TForm1.Image3Click(Sender: TObject);

begin

if progres\_point > 0 then

begin

Media('08.mp3');

MainText.Caption:=

' [Уровень 2]' + #13#10 +

'Квадраты - враги, создающие вокруг себя круги';

image2.Visible:=false;

image3.Visible:=false;

image4.Visible:=false;

image5.Visible:=false;

image6.Visible:=false;

player.Visible:=true;

TimerPlayer.Enabled:=true;

Lvl2\_create.Enabled:=true;

Lvl2\_move.Enabled:=true;

end;

end;

// проверка нажатия уровня 3

procedure TForm1.Image4Click(Sender: TObject);

begin

if progres\_point > 1 then

begin

Media('08.mp3');

MainText.Caption:=

' [Уровень 3]' + #13#10 +

'Квадратики - враги, которые приследуют тебя';

image2.Visible:=false;

image3.Visible:=false;

image4.Visible:=false;

image5.Visible:=false;

image6.Visible:=false;

player.Visible:=true;

TimerPlayer.Enabled:=true;

LVL3\_3sec.Enabled:=true;

end;

end;

// проверка нажатия уровня 4

procedure TForm1.Image5Click(Sender: TObject);

begin

if progres\_point > 2 then

begin

Media('011.mp3');

MainText.Caption:=

' [Уровень 4]';

image2.Visible:=false;

image3.Visible:=false;

image4.Visible:=false;

image5.Visible:=false;

image6.Visible:=false;

player.Visible:=true;

TimerPlayer.Enabled:=true;

LVL4\_3sec.Enabled :=true;

LVL4\_3sec2.Enabled:=true;

end;

end;

// проверка нажатия уровня 5

procedure TForm1.Image6Click(Sender: TObject);

begin

if progres\_point > 3 then

begin

Media('06.mp3');

MainText.Caption:=

' [Уровень 5]';

image2.Visible:=false;

image3.Visible:=false;

image4.Visible:=false;

image5.Visible:=false;

image6.Visible:=false;

player.Visible:=true;

TimerPlayer.Enabled:=true;

LVL5\_3sec.Enabled :=true;

end;

end;

// ===================================level 1=====================================

// 5 секунд задержки и включение картинки с управлением

procedure TForm1.LVL1\_5secTimer(Sender: TObject);

begin

MainText.Caption:='';

left.Enabled:=true;

create.Enabled:=true;

wasd.visible :=false;

label1.Visible :=false;

LVL1\_5sec.Enabled:=false;

end;

// создание врагов кругов

procedure CreateEnemy(AForm: TForm; postop,posleft:integer; direction:integer=0);

var

Enemy: TShape;

begin

Enemy := TShape.Create(AForm);

Enemy.Parent := AForm;

Enemy.Shape := stCircle;

Enemy.Brush.Color := clRed;

Enemy.Width := 40;

Enemy.Height := 40 ;

Enemy.Left := posleft; // Начальная позиция

Enemy.Top := postop;

SetLength(Enemies, Length(Enemies) + 1);

Enemies[Length(Enemies) - 1] := Enemy;

case direction of

1: Enemy.Tag := 1; // Движение влево

2: Enemy.Tag := 2; // Движение вправо

3: Enemy.Tag := 3; // Движение вверх

4: Enemy.Tag := 4; // Движение вниз

end;

end;

// создание врагов

procedure TForm1.createTimer(Sender: TObject);

begin

begin

if b>Screen.Height+100 then

begin

create.Enabled:=false;

left.Enabled:=false;

clear.Enabled:=true;

MainText.Caption:=' А теперь посложнее';

LVL1\_2\_5sec.Enabled:=true;

b:=0;

end;

b:=b+100;

CreateEnemy(Self,b,Form1.ClientWidth);

end;

end;

// обновление позиции

procedure TForm1.leftTimer(Sender: TObject);

begin

var

I: Integer;

for I := 0 to Length(Enemies) - 1 do

Enemies[I].Left := Enemies[I].Left - EnemySpeed;

end;

// очистка от врагов и очистка массива врагов

procedure TForm1.clearTimer(Sender: TObject);

begin

for var I := 0 to Length(Enemies) - 1 do

Enemies[I].free;

SetLength(Enemies, 0);

clear.Enabled:=false;

for var I := 0 to Length(Enemies2) - 1 do

Enemies2[I].free;

SetLength(Enemies2, 0);

clear.Enabled:=false;

end;

// 5 секунд задержки

procedure TForm1.LVL1\_2\_5secTimer(Sender: TObject);

begin

create2.Enabled:=true;

leftANDright.Enabled:=true;

MainText.Caption:='';

LVL1\_2\_5sec.Enabled:=false;

end;

// создание врагов

procedure TForm1.create2Timer(Sender: TObject);

begin

if b>Screen.Height+400 then

begin

Image3.Picture.LoadFromFile('2.jpg');

if progres\_point < 1 then

begin

progres\_point:=1;

AssignFile(F, 'progres\_point.txt');

Rewrite(F);

write(F,progres\_point);

Closefile(F);

end;

Media('win.mp3');

normalaz;

winT.Enabled :=true;

win.Visible :=true;

end;

b:=b+100;

CreateEnemy(Self,b,Form1.ClientWidth);

CreateEnemy(Self,b,0);

end;

// обновление позиции врагов

procedure TForm1.leftANDrightTimer(Sender: TObject);

begin

var

I: Integer;

for I := 0 to Length(Enemies) - 1 do

begin

if i mod 2=0 then

Enemies[I].Left := Enemies[I].Left - EnemySpeed

else

Enemies[I].Left := Enemies[I].Left + EnemySpeed;

end;

end;

//====================================== level 2 ===============================================

// создание врага квадрат

procedure CreateSquare(AForm: TForm; posleft, postop: Integer; direction:integer=0 );

begin

Square := TShape.Create(AForm);

Square.Parent := AForm;

Square.Shape := stRectangle;

Square.Brush.Color := clRed;

Square.Width := 50;

Square.Height := 50;

Square.Left := posleft;

Square.Top := postop;

SetLength(Enemies, Length(Enemies) + 1);

Enemies[Length(Enemies) - 1] := Square;

case direction of

5: Square.Tag := 5; // Движение влево

6: Square.Tag := 6; // Движение вправо

7: Square.Tag := 7; // Движение вверх

8: Square.Tag := 8; // Движение вниз

end;

end;

// создание врагов

procedure TForm1.LVL2\_createTimer(Sender: TObject);

begin

MainText.Caption:='';

b:=b+400;

CreateSquare(self,b,0,8);

LVL2\_createS.Enabled:=true;

if b>=2400 then

begin

LVL2\_3sec.Enabled:=true;

LVL2\_create.Enabled :=false;

b:=0;

end;

end;

// 3 секунды задержки

procedure TForm1.LVL2\_3secTimer(Sender: TObject);

begin

LVL2\_create2.Enabled:=true;

end;

// создание врагов

procedure TForm1.LVL2\_create2Timer(Sender: TObject);

begin

MainText.Caption:='';

b:=b+400;

CreateSquare(self,b,form1.Height,7);

LVL2\_createS.Enabled:=true;

if b>=2000 then

begin

Image4.Picture.LoadFromFile('3.jpg');

if progres\_point < 2 then

begin

progres\_point:=2;

AssignFile(F, 'progres\_point.txt');

Rewrite(F);

write(F,progres\_point);

Closefile(F);

end;

Media('win.mp3');

normalaz;

winT.Enabled :=true;

win.Visible :=true;

end;

end;

// создание врагов возле квадрата

procedure TForm1.LVL2\_createSTimer(Sender: TObject);

begin

CreateEnemy(Self, Square.Top - 20, Square.Left + 4, 3); // Создаем врага над квадратом, движущегося вверх

CreateEnemy(Self, Square.Top + 20, Square.Left + 4, 4); // Создаем врага под квадратом, движущегося вниз

CreateEnemy(Self, Square.Top , Square.Left - 10, 1); // Создаем врага слева от квадрата, движущегося влево

CreateEnemy(Self, Square.Top , Square.Left + 20, 2); // Создаем врага справа от квадрата, движущегося вправо

end;

// обновление их позиции

procedure TForm1.LVL2\_moveTimer(Sender: TObject);

var

i: Integer;

begin

for i := 0 to Length(Enemies) - 1 do

begin

case Enemies[i].Tag of

1: Enemies[i].Left := Enemies[i].Left - EnemySpeed; // Движение влево

2: Enemies[i].Left := Enemies[i].Left + EnemySpeed; // Движение вправо

3: Enemies[i].Top := Enemies[i].Top - EnemySpeed; // Движение вверх

4: Enemies[i].Top := Enemies[i].Top + EnemySpeed; // Движение вниз

7: Enemies[i].Top := Enemies[i].Top - Round(EnemySpeed/2.2); // Движение вверх для квадрата

8: Enemies[i].Top := Enemies[i].Top + Round(EnemySpeed/2.2); // Движение вниз для кв

end;

end;

end;

//========================================= level 3 ===============================================

// создание квадратика

procedure CreateRectangle(AForm: TForm);

var

Enemy: TShape;

begin

Enemy := TShape.Create(AForm);

Enemy.Parent := AForm;

Enemy.Shape := stRectangle;

Enemy.Brush.Color := clRed;

Enemy.Width := 20;

Enemy.Height := 20;

Enemy.Left := Random(AForm.ClientWidth - Enemy.Width); // Случайная начальная позиция по горизонтали

Enemy.Top := Random(AForm.ClientHeight - Enemy.Height); // Случайная начальная позиция по вертикали

SetLength(Enemies2, Length(Enemies2) + 1);

Enemies2[Length(Enemies2) - 1] := Enemy;

end;

// Движение квадратика к игроку

procedure MoveEnemiesToPlayer(AForm: TForm; PlayerX, PlayerY: Integer);

var

i: Integer;

EnemyX, EnemyY: Integer;

DirectionX, DirectionY: Integer;

Speed: Integer;

begin

Speed := 4; // скорость

for i := 0 to Length(Enemies2) - 1 do

begin

EnemyX := Enemies2[i].Left + Enemies2[i].Width div 2;

EnemyY := Enemies2[i].Top + Enemies2[i].Height div 2;

DirectionX := PlayerX - EnemyX;

DirectionY := PlayerY - EnemyY;

// Вычисляем длину вектора направления

var DirectionLength := Sqrt(Sqr(DirectionX) + Sqr(DirectionY));

if DirectionLength > 0 then

begin

DirectionX := Round(DirectionX / DirectionLength \* Speed);

DirectionY := Round(DirectionY / DirectionLength \* Speed);

end;

Enemies2[i].Left := Enemies2[i].Left + DirectionX;

Enemies2[i].Top := Enemies2[i].Top + DirectionY;

end;

end;

// 3 секунды задержки

procedure TForm1.LVL3\_3secTimer(Sender: TObject);

begin

MainText.Caption:='';

LVL3\_create.Enabled:=true;

LVL3\_move.Enabled:=true;

LVL3\_3sec.Enabled:=false;

end;

// если форма закрыта с помощью ALT+F4 то закрывает игру

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

begin

application.Terminate;

end;

// создание врагов

procedure TForm1.LVL3\_createTimer(Sender: TObject);

begin

if Length(Enemies2) > 25 then

begin

Image5.Picture.LoadFromFile('4.jpg');

if progres\_point < 3 then

begin

progres\_point:=3;

AssignFile(F, 'progres\_point.txt');

Rewrite(F);

write(F,progres\_point);

Closefile(F);

end;

Media('win.mp3');

normalaz;

winT.Enabled :=true;

win.Visible :=true;

end;

CreateRectangle(self);

end;

// обновление позиции врагов

procedure TForm1.LVL3\_moveTimer(Sender: TObject);

var I: integer;

begin

MoveEnemiesToPlayer(Self, Player.Left + Player.Width div 2, Player.Top + Player.Height div 2);

end;

//============================================ level 4 ==================================================

// 3 секунды задержки

procedure TForm1.LVL4\_3secTimer(Sender: TObject);

begin

MainText.Caption:='';

LVL4\_create.Enabled :=true; // включение прошлых таймеров для обновления позиции

LVL2\_move.Enabled :=true;

LVL4\_3sec.Enabled :=false;

c:=Form1.ClientWidth;

d:=1;

b:=1;

end;

// 3 секунды задержки

procedure TForm1.LVL4\_3sec2Timer(Sender: TObject);

begin

Image6.Picture.LoadFromFile('5.jpg');

if progres\_point < 4 then

begin

progres\_point:=4;

AssignFile(F, 'progres\_point.txt');

Rewrite(F);

write(F,progres\_point);

Closefile(F);

end;

Media('win.mp3');

normalaz;

winT.Enabled :=true;

win.Visible :=true;

end;

// создание врагов

procedure TForm1.LVL4\_createTimer(Sender: TObject);

begin

if b mod 10=0 then

begin

c:=0;

d:=2;

LVL4\_create2.Enabled:=true;

end;

b:=b+1;

CreateEnemy(Self,random(Screen.Height),c,d);

end;

// создание врагов

procedure TForm1.LVL4\_create2Timer(Sender: TObject);

begin

CreateEnemy(Self,Screen.width,random(Screen.width),3);

CreateEnemy(Self,0,random(Screen.width),4);

if b mod 10=0 then LVL4\_create3.Enabled :=true;

end;

// создание врагов

procedure TForm1.LVL4\_create3Timer(Sender: TObject);

begin

CreateSquare(self,random(Screen.width),0,8);

LVL2\_createS.Enabled:=true;

end;

//============================================ level 5 ==================================================

// 5 секунд задержки

procedure TForm1.LVL5\_3secTimer(Sender: TObject);

begin

MainText.Caption:='';

LVL5\_3sec.Enabled :=false; // включение прошлых таймеров для обновление позиции

LVL5\_create.Enabled :=true;

LVL3\_move.Enabled :=true;

LVL2\_move.Enabled:=true;

end;

// создание врагов

procedure TForm1.LVL5\_createTimer(Sender: TObject);

begin

CreateEnemy(Self,Screen.Height,random(Screen.width),3);

CreateEnemy(Self,0,random(Screen.width),4);

CreateEnemy(Self,random(Screen.Height),Screen.width, 1);

CreateEnemy(Self,random(Screen.Height),0, 2);

LVL5\_create2.Enabled :=true;

LVL4\_3sec2.Interval:=60000;

LVL4\_3sec2.Enabled :=true;

end;

// каждые 5 секунды создает один квадратик

procedure TForm1.LVL5\_create2Timer(Sender: TObject);

begin

CreateRectangle(self);

end;

end.begin

MainText.Caption:='';

LVL5\_3sec.Enabled :=false;

LVL5\_create.Enabled :=true;

LVL3\_move.Enabled :=true;

LVL2\_move.Enabled:=true;

end;

procedure TForm1.LVL5\_createTimer(Sender: TObject);

begin

CreateEnemy(Self,Screen.Height,random(Screen.width),3);

CreateEnemy(Self,0,random(Screen.width),4);

CreateEnemy(Self,random(Screen.Height),Screen.width, 1);

CreateEnemy(Self,random(Screen.Height),0, 2);

LVL5\_create2.Enabled :=true;

LVL4\_3sec2.Interval:=60000;

LVL4\_3sec2.Enabled :=true;

end;

procedure TForm1.LVL5\_create2Timer(Sender: TObject);

begin

CreateRectangle(self);

end;

end.close;

end;

end;

end.