УЧРЕЖДЕНИЕ ОБРАЗОВАННИЯ «БОБРУЙСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНО-ЭКОНОМИЧЕСКИЙ КОЛЛЕДЖ»

**ПРИКЛАДНОЕ ПРИЛОЖЕНИЕ   
«ПОСТРОЕНИЕ МАГИЧЕСКИХ КВАДРАТОВ»**

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по дисциплине

«Основы алгоритмизации и программирования»   
  
  
**КП 509000.219П ПЗ**

**Разработчик**:

Учащаяся группы 19-П II курса

специальности «Программное обеспечение

информационных технологий» /Д.Н. Почерняева/

**Руководитель:**

Преподаватель высшей категории /С.В. Скрипко/

БОБРУЙСК 2019

# ОГЛАВЛЕНИЕ

ОГЛАВЛЕНИЕ 1

ВВЕДЕНИЕ 3

1. ПОСТАНОВКА ЗАДАЧИ 5

1.1. Назначение и функции программы 5

1.2. Математическая модель задачи 5

1.3. Информационная база задачи 6

1.4. Система меню 7

1.5. Система помощи 7

1.6. Контрольные примеры 9

1.7. Алгоритм программы 12

2. ОПИСАНИЕ ПРОГРАММЫ 12

2.1. Среда разработки 13

2.2. Пользовательский интерфейс 13

3. ОПИСАНИЕ ПРИМЕНЕНИЯ 16

3.1. Рекомендуемый состав технических средств 16

3.2. Рекомендуемый состав программных средств 17

3.3. Установка программы 17

ЗАКЛЮЧЕНИЕ 21

ЛИТЕРАТУРА 22

Нормативная литература 22

Учебная литература 22

Тематическая литература 22

ПРИЛОЖЕНИЯ 23

ПРИЛОЖЕНИЕ А 23

ПРИЛОЖЕНИЕ Б 24

ПРИЛОЖЕНИЕ В 28

# ВВЕДЕНИЕ

В данном курсовом проекте разработано приложение, которое позволяет реализовывать построение магических квадратов. Также в дополнение была создана проверка квадрата 3х3 на «магичность», т.е. на принадлежность к магическим квадратам.

Магический квадрат (волшебный квадрат или магическая матрица, МК) – это квадратная таблица из целых чисел, в которой суммы чисел любой строки, любого столбца и любой из двух диагоналей (главной и побочной) равны одному и тому же числу. Сумма чисел в каждой строке, столбце и на диагоналях называется магической константой (магической суммой) и зависит только от порядка квадрата.

Теория магических квадратов активно изучается с помощью методов высшей алгебры и операционного исчисления. На настоящее время магические квадраты (далее допустимо наименование «волшебные квадраты») применяются в криптографических алгоритмах защиты информации и приборостроении. Одно из высокоперспективных направлений применения магических квадратов является конструирование телевизоров, изображение для которых формируется на технологии «магического квадрата», позволяя организовать более плавные цветовые переходы.

Для курсового проекта была выбрана данная тема, так как в современном мире эта сфера до конца не изучена и есть необходимость оптимизации и упрощения алгоритмов построения магических квадратов.

Современные среды программирования позволяют находить новые и упрощенные решения магических квадратов, превзойдя долгие и многосложные методы перебора. Приложение по построению волшебных квадратов было создано в среде программирования Delphi, так как именно она позволяет быстро разрабатывать приложения с помощью уже существующих компонентов и инструментов, которых значительное множество. Также в основе Delphi лежит язык Object Pascal (расширение объектно-ориентированного языка Pascal), который имеет довольно простой синтаксис и является почти универсальным языком программирования.

Целью проекта была взята разработка алгоритма и усовершенствование уже существующих алгоритмов построения магических квадратов и реализация этих алгоритмов в приложении.

Задачами, в соответствии с целью, являются:

1. Изучение материала взятой темы: изучение видов магических квадратов, методы их построения, чтение дополнительных пособий, связанные с данной темой.
2. Изучение среды программирования Delphi 7: чтение соответствующей литературы, ознакомление с компонентами и инструментами среды разработки.
3. Создание алгоритмов построения магических квадратов: разработка и построение его графической записи (блок-схема). Оптимизация алгоритма и кода.
4. Разработка графической части проекта. Работа с графическими редакторами Adobe (Photoshop, Illustrator).
5. Тестирование и отладка (устранение ошибок) программы.
6. Ознакомление с правилами написания технической документации на программные средства. Создание справочной системы.
7. Ознакомление с утилитой Smart Install Maker. Создание установочного диска.

В состав пояснительной записки входят следующие разделы:

* 1. [Введение](#_ВВЕДЕНИЕ)
  2. [Постановка задачи](#_ПОСТАНОВКА_ЗАДАЧИ)
  3. [Описание программы](#_ОПИСАНИЕ_ПРОГРАММЫ)
  4. [Описание применения](#_ОПИСАНИЕ_ПРИМЕНЕНИЯ)
  5. [Заключение](#_ЗАКЛЮЧЕНИЕ)
  6. [Литература](#_ЛИТЕРАТУРА)
  7. [Приложения](#_ПРИЛОЖЕНИЯ)

# 1. ПОСТАНОВКА ЗАДАЧИ

В данном разделе рассматривается задача, определяемая темой проекта. А также описываются подзадачи, которые необходимо решить для достижения цели курсовой работы.

## **1.1. Назначение и функции программы**

Темой курсового проекта была взята разработка математической программы по построению магических квадратов.

Программа способна строить правильные магические квадраты, размерностью от 3 до 100. Правильным магическим квадратом называют квадрат, заполненный натуральными числами от 1 до , где n порядок магического квадрата.

Программа по построению магических квадратов может использоваться для различных целей, в которых необходимы уже готовые магические квадраты, например, для шифрования по методу магических квадратов.

Также волшебные квадраты положены в основу многих заданий, развивающих логику и математическое мышление. Для этого в программе есть небольшая головоломка, в которой можно попробовать самому составить магический квадрат размерностью 3х3. Игра не ограничивается необходимостью получить именно магическую константу и временем составление квадрата, чтобы не усложнять процесс заполнения для пользователя (то есть возможно получение нестандартного магического квадрата).

## **1.2. Математическая модель задачи**

### **1.2.1. Математическая формулировка задачи исследования**

Для решения данной математической задачи были использованы три разных метода составления магических квадратов, выбор которых зависит от порядка магического квадрата. Для магических квадратов нечетного порядка был изучен метод французского геометра А. де ла Лубера. Для магических квадратов одинарной четности (четно-нечетные или магические квадраты, порядок которых кратен 2, но не кратен 4) применен метод Ф. де ла Ира, который был немного модифицирован для упрощения реализации на Pascal и назван «методом четырёх квадратов». Чтобы построить квадраты двойной четности, т.е. четно-четные магические квадраты или квадраты, порядок которых кратен 4, использовался метод Рауз-Болла. Алгоритм метода решения задачи приведен в [приложении Б](#_ПРИЛОЖЕНИЕ_Б).

Для вычисления магический константы была использована универсальная формула нахождения (1.2.1.1):

, (1.2.1.1)

### **1.2.2. Выбор метода решения задачи**

Данная задача была решена с помощью компонентов StringGrid и StaticText. При полученном порядке квадрата, введенным пользователем с клавиатуры, после нажатия на кнопку «Построить магический квадрат» результат выводится в автоматически построенный StringGrid. Для того, чтобы получить значение магической константы необходимо нажать на кнопку «Магическая сумма», после чего она будет показана на форме под кнопкой «Магическая суммы». Листинг с кодом программы отображен в приложении В.

## **1.3. Информационная база задачи**

В данном подразделе описывается входная и выходная информация решения поставленной задачи.

### **1.3.1. Входная информация**

Входной информацией в программе является осуществление определенных действий при нажатии по кнопкам (переход по формам, построение МК, сохранение МК в сторонней программе Excel и т.д.), ввод данных в поля, представленных компонентом Edit.

Стартовой формой является форма GLAV, представляющая собой главное меню совокупностью кнопок, реализующие переход на формы MAG и GAME.

Для реализации переходов из главного меню на необходимую форму, использовался компонент Image, которые были созданы в формате значков ICO, который позволил создать фигуры необходимого размера без фона.

На форме MAG осуществляется построение магических квадратов, на которой пользователь прописывает размерность квадрата в поле ввода, после чего должен нажать на кнопку «Построить», при необходимости узнать магическую константу нажать на кнопку «Магическая сумма».

На форме GAME под названием «Заполнить магический квадрат» пользователь может сам построить магический квадрат размером 3 на 3. Также можно очистить квадрат полностью одним нажатием на кнопку «Очистить значения».

Попасть в главное меню можно с помощью стрелки на каждой форме.

### **1.3.2. Выходная информация**

Выходной информацией программы является построенные магические квадраты, реализованные в StringGrid, отображение магической суммы в StaticText (был использован этот компонент, а не Label, так как он удобнее в использовании), а также вывод МК в Excel, реализованный нажатием на кнопку «Сохранить в Excel». На форме с самостоятельным заполнением квадрата выходной информацией будут Image, которые появляются с данными о правильности/неправильности заполнения после нажатия на кнопку «Проверить».

## **1.4. Система меню**

Система меню представлена компонентом – PopupMenu, так называемым контекстным меню, в которое можно попасть, нажав на правую кнопку мыши или тачпада. Меню осуществляет переход по формам, вход в справку программы и выход из программы (см. рис. 1.4.1).

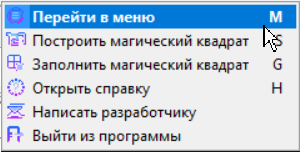


Рисунок 1.4.1 — Контекстное меню

Переход по формам может быть осуществлен при помощи горячих клавиш.

На форму построения МК можно перейти клавишей «S» («Ы»).

На форму заполнения МК можно перейти клавишей «G» («П»).

На форму главного меню можно перейти клавишей «M» («Ь»).

Переход с помощью горячих клавиш возможен только при открытом приложении в активном режиме. Раскладка клавиатуры не влияет на выполнение процедуры открытия форм.

Далее основным способом открытия форм будут являться овальные кнопки на форме главного меню.

## **1.5. Система помощи**

Система помощи представлена в виде справки формата HTMLHelp с расширением chm. Для создания справки была использована утилита Help&Manual, а для создания .CHM-файла – Microsoft HTML Help Workshop. К программе справка была подключена с помощью встроенного модуля ShellAPI.

Открыть руководство пользователя можно кнопкой «Помощь» (см. рис.1.5.1) на главной форме, пунктом «Открыть справку» в контекстном меню или нажав горячую клавишу H (P) на клавиатуре.

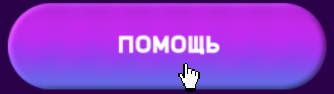


Рисунок 1.5.1 — Кнопка «Помощь»

В руководстве пользователя есть вся необходимая информация, благодаря которой неосведомленный пользователь сможет ознакомиться с принципами работы программы (см. рис. 1.5.2).

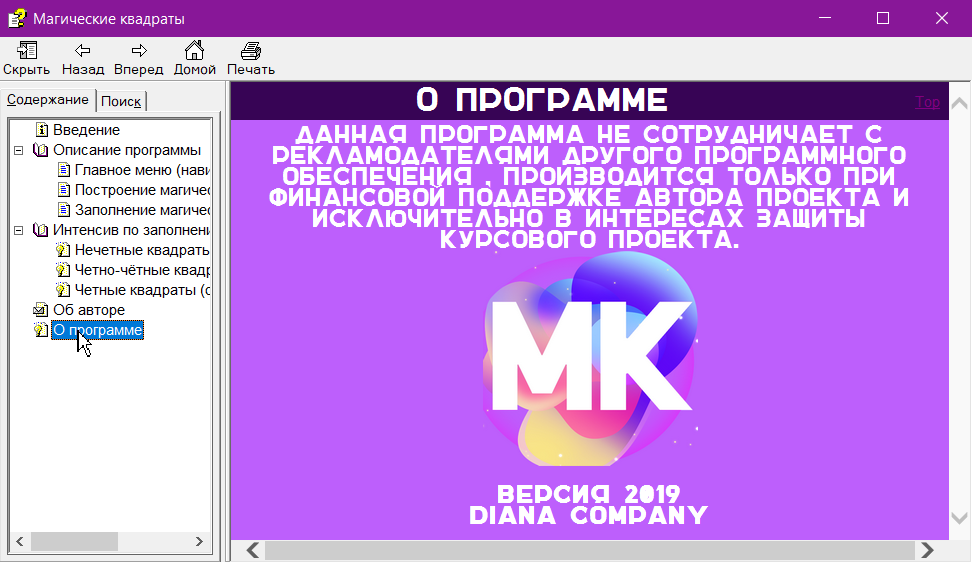


Рисунок 1.5.2 — Справка программы

В руководстве пользователя можно узнать описание программы о главном меню, форме построения и форме заполнения, прочитать интенсив по заполнению квадратов, а также узнать о программе и об авторе (см. рис. 1.5.3).

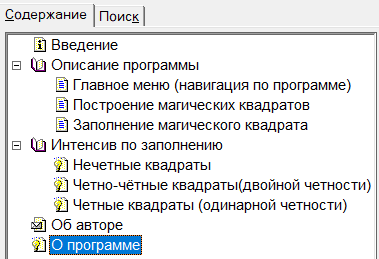


Рисунок 1.5.3 — Содержание справки

## **1.6. Контрольные примеры**

При запуске программы появляется стартовое окно формы: Главное меню. (см. рис. 1.6.1)

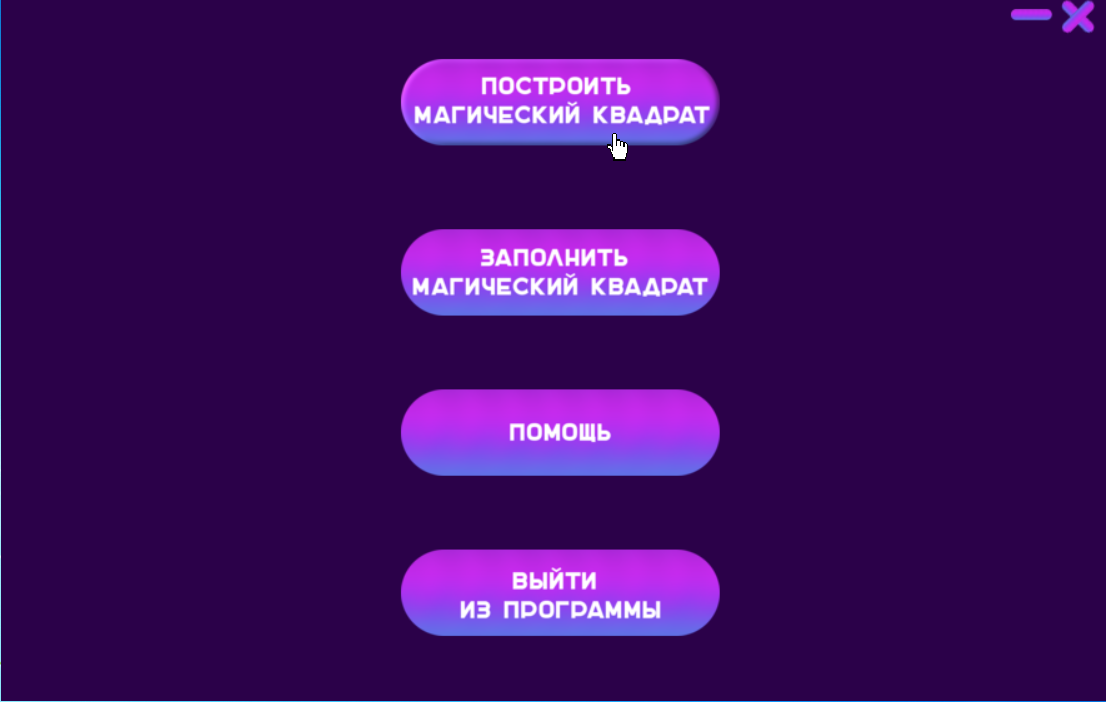


Рисунок 1.6.1 — Главное меню

При нажатии на кнопку «Построить магический квадрат» открывается форма, на которой при вводе порядка магического квадрата и нажатии на кнопку «Построить» реализуется магический квадрат. Если ввести порядок больше 21, то пользователю будет предложено сохранить квадрат в MS Excel, так как квадрат не помещается полностью на форму (см. рис. 1.6.2).

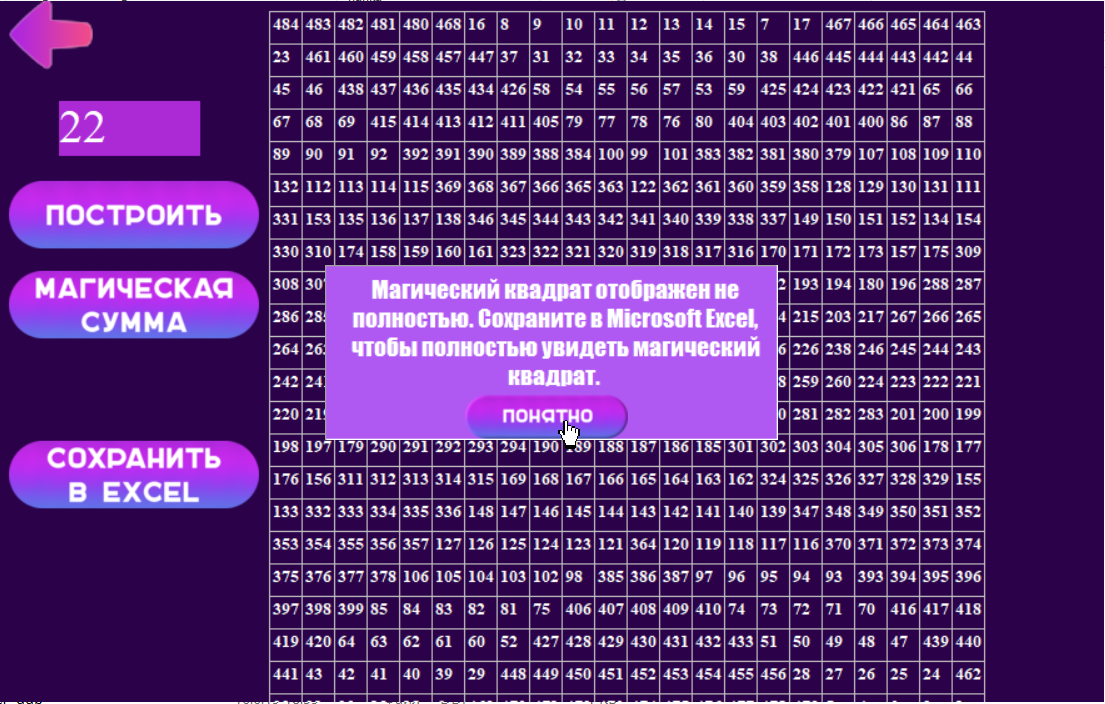


Рисунок 1.6.2 — Окно построения и информационное окно

При нажатии на кнопку «Магическая сумма» появляется значение магической суммы (см. рис.1.6.3).



Рисунок 1.6.3 — Кнопка «Магическая сумма»

При нажатии на кнопку «Сохранить в Excel» открывается диалоговое окно (см. рис. 1.6.4) с сохранением магического квадрата, после чего открывается книга Excel с открытым магическим квадратом (см. рис.1.6.5).

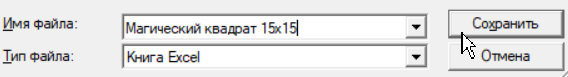


Рисунок 1.6.4 — Сохранение магического квадрата

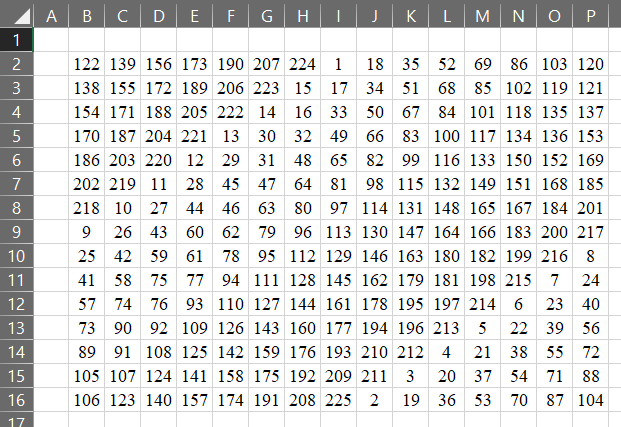


Рисунок 1.6.5 — Книга Excel с магическим квадратом

Попасть обратно в меню можно нажав на стрелку, направленную влево, или с помощью контекстного меню (см. рис. 1.6.6).

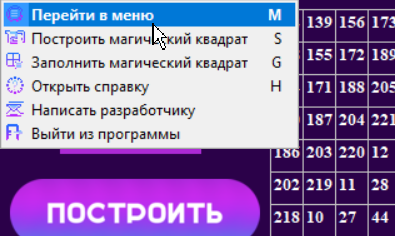


Рисунок 1.6.6 — Возврат в главное меню

При нажатии на кнопку «Заполнить магический квадрат» открывается форма (см. рис.1.6.7), на котором пользователь может заполнить МК 3х3, чтобы проверить квадрат на «магичность» нужно заполнить 9 полей числами, если же необходимо очистить квадрат можно нажать на кнопку «Очистить значения».

Если квадрат либо одна ячейка не заполнена появляется окно ошибки, аналогичное окну на рис. 1.6.2.

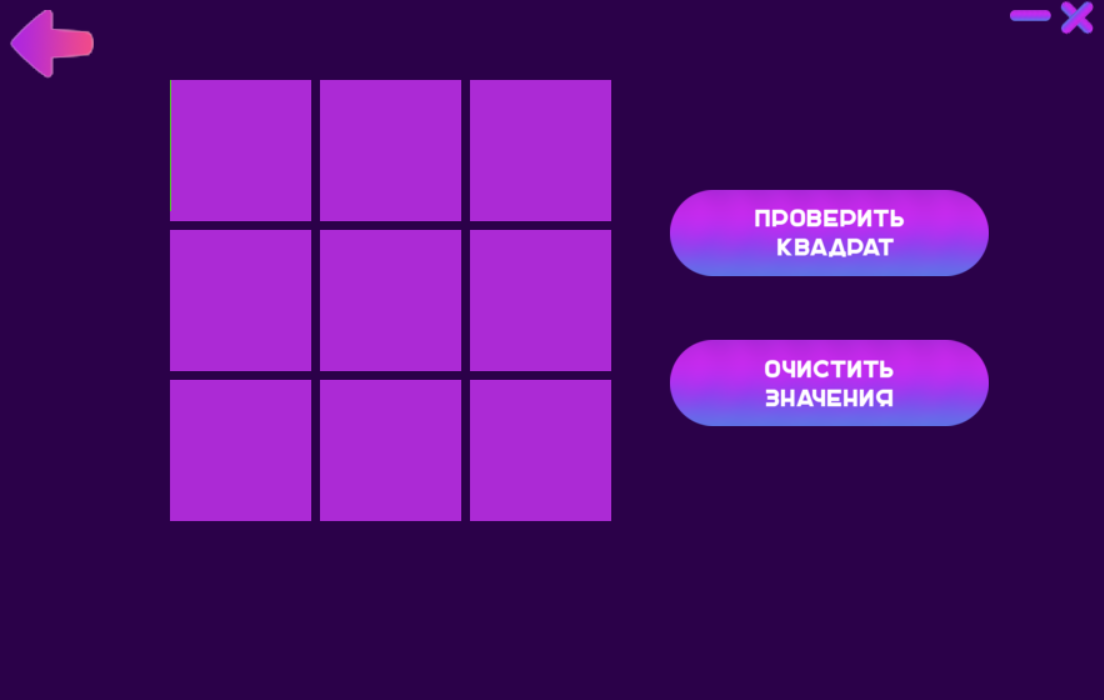


Рисунок 1.6.7 — Форма заполнения магического квадрата

При нажатии на кнопку помощь открывается справка программы (см. рис.1.6.8).

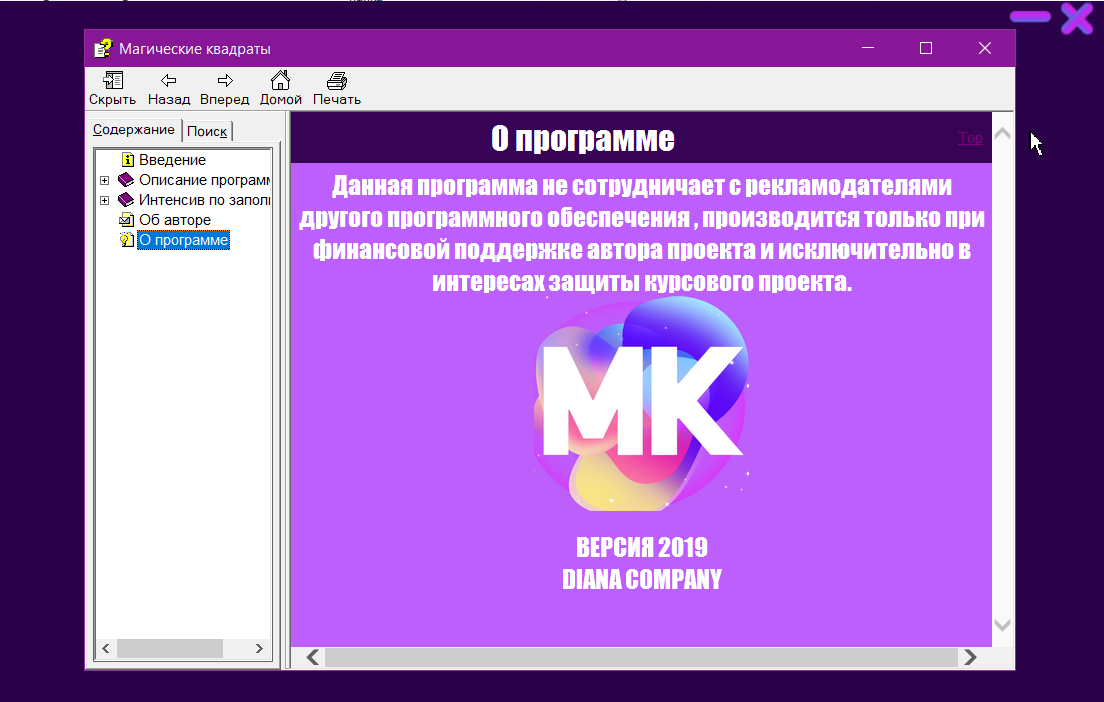


Рисунок 1.6.8 — Открытие справки

При необходимости можно свернуть приложение, нажав соответствующую кнопку.

Закрыть программу можно кнопкой в главном меню, в контекстном меню или крестиком в правом верхнем углу программы (см. рис. 1.6.9).



Рисунок 1.6.9 — Значки «Свернуть» и «Закрыть»

## **1.7. Алгоритм программы**

Связь форм программы между собой представлена на рис.1.7.1. Обобщенный алгоритм приведен в [*приложении А*](#_ПРИЛОЖЕНИЕ_А)*.*



Рисунок 1.7.1 — Связь форм между собой

# 2. ОПИСАНИЕ ПРОГРАММЫ

## **2.1. Среда разработки**

Средой разработка данного прикладного программного обеспечения является Delphi 7, в соответствии с планом изучаемой дисциплины. Система программирования Delphi версии 7 фирмы Enterprise (Borland) предоставляет наиболее широкие возможности для программирования приложений ОС Windows. Основным конкурентом Borland Delphi 7 является RAD-среда Borland C++ Builder, в которой код пишется на языке C++. Его недостатков пере языком Pascal является сложный синтаксис и объёмная спецификация языка.

При написании кода программы были использованы следующие компоненты:

Визуальные: Edit, Image, StringGrid, Label, StaticText, Bevel, Button, OleContainer.

Невизуальные: PopupMenu, SaveDialog.

## **2.2. Пользовательский интерфейс**

Интерфейс программы был тщательно проработан для удобства и простоты использования. Окна интерфейса программы служит для ввода и отображения информации.

Главное меню представлено рядом кнопок (рис.2.2.1):

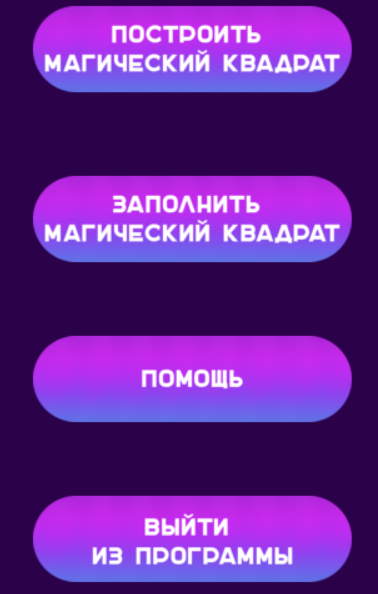


Рисунок 2.2.1 — Кнопки главного меню

Каждая из этих кнопок выполняет действия, соответствующие надписям на них. Кнопки представлены компонентом Image в виде закругленных прямоугольников, которые были созданы в графическом редакторе Photoshop и Illustrator. Также при наведении курсором на кнопку пользователь может наблюдать эффект нажима на кнопку, что было реализовано изменениями видимости двух компонентов Image, представленных одной кнопкой.

Рассмотрим окно, на которое пользователь попадает после нажатия на кнопку «Построить магический квадрат». Окно исполнено в одной цветовой пастельной палитре, что привлекает внимание пользователя и не отвлекает его от процесса. Поле ввода представлено компонентом Edit, в который можно вписывать только числа, что защищает пользователя от ошибок. Кнопки аналогичны кнопкам из главного меню. Также на форме нет никаких лишних компонентов и сторонней информации (см. рис. 2.2.2).

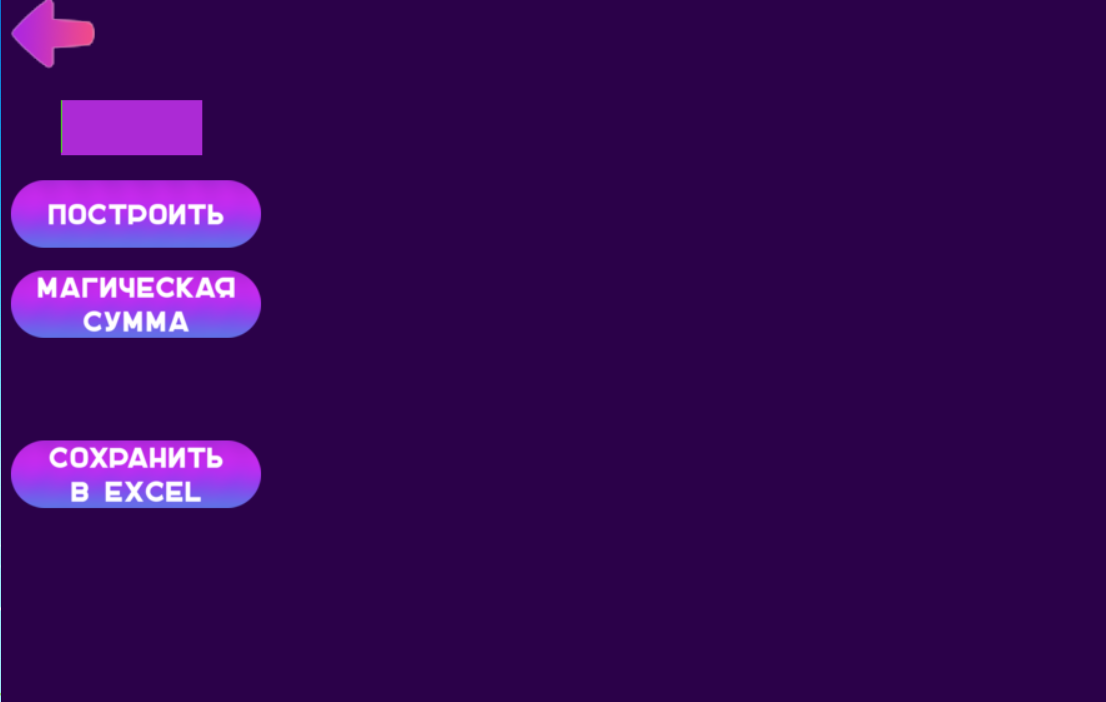


Рисунок 2.2.2 — Интерфейс формы построения

Форма заполнения МК соответствует цветовой схеме программы. Компоненты Edit, стоящие в виде квадрата упрощают представление магического квадрата, также по ячейкам квадрата можно перемещаться с помощью клавиши на клавиатуре TAB. Надписей, находящихся на кнопках достаточно для понимания действий программы без чтения справки (см. рис.2.2.3).

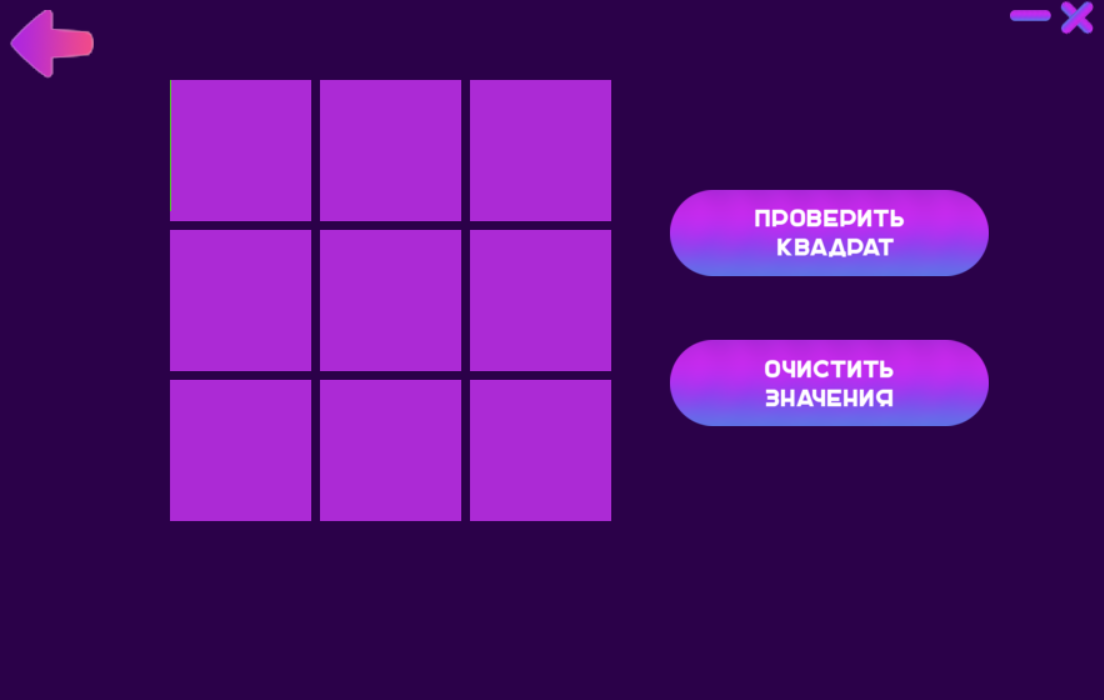


Рисунок 2.2.3 — Интерфейс формы заполнения

Справка представлена в отдельных окнах, соответствующих основным цветам программы. Справка написана доступным для пользователя языком без технических терминов и имеет всю необходимую информацию для рационального пользования программой (см. рис. 2.2.4).

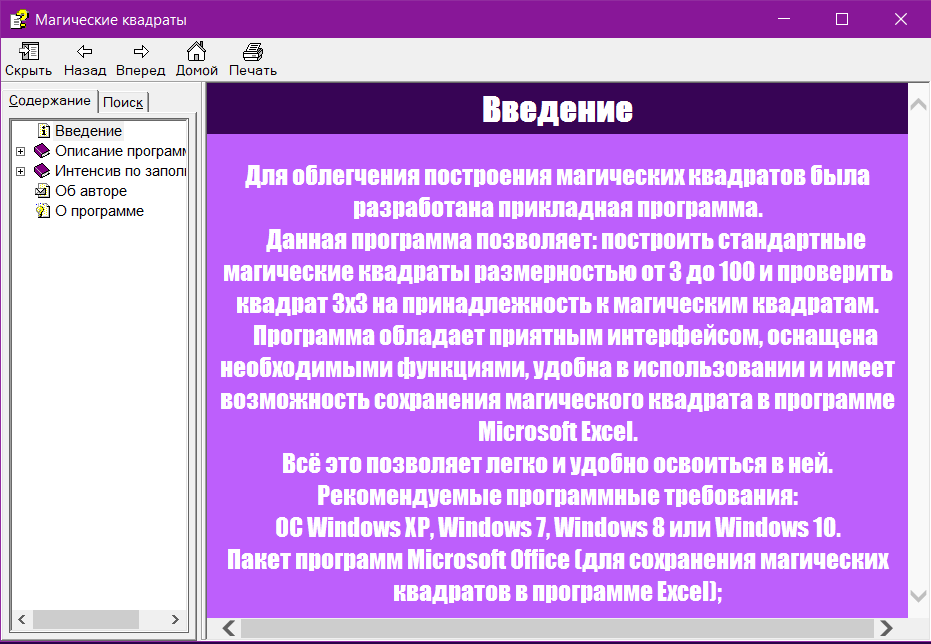


Рисунок 2.2.4 — Разбор интерфейса справки

# 3. ОПИСАНИЕ ПРИМЕНЕНИЯ

## **3.1. Рекомендуемый состав технических средств**

В состав технических средств должен входить персональный

компьютер (ПЭВМ), включающий в себя:

* Процессор с тактовой частотой не менее 0,8 ГГц;
* Операционную систему Windows XP (см. рис.3.3.1), Windows 7 (см. рис. 3.3.2), Windows 8, Windows 10.
* Оперативную память объемом не менее 128 Мб;
* Свободное место на жестком диске не менее 10 Мб;
* Рекомендуется использовать стандартное соотношение сторон монитора, так как возможно некорректное отображение программы.
* Привод оптических дисков (CD, DVD).

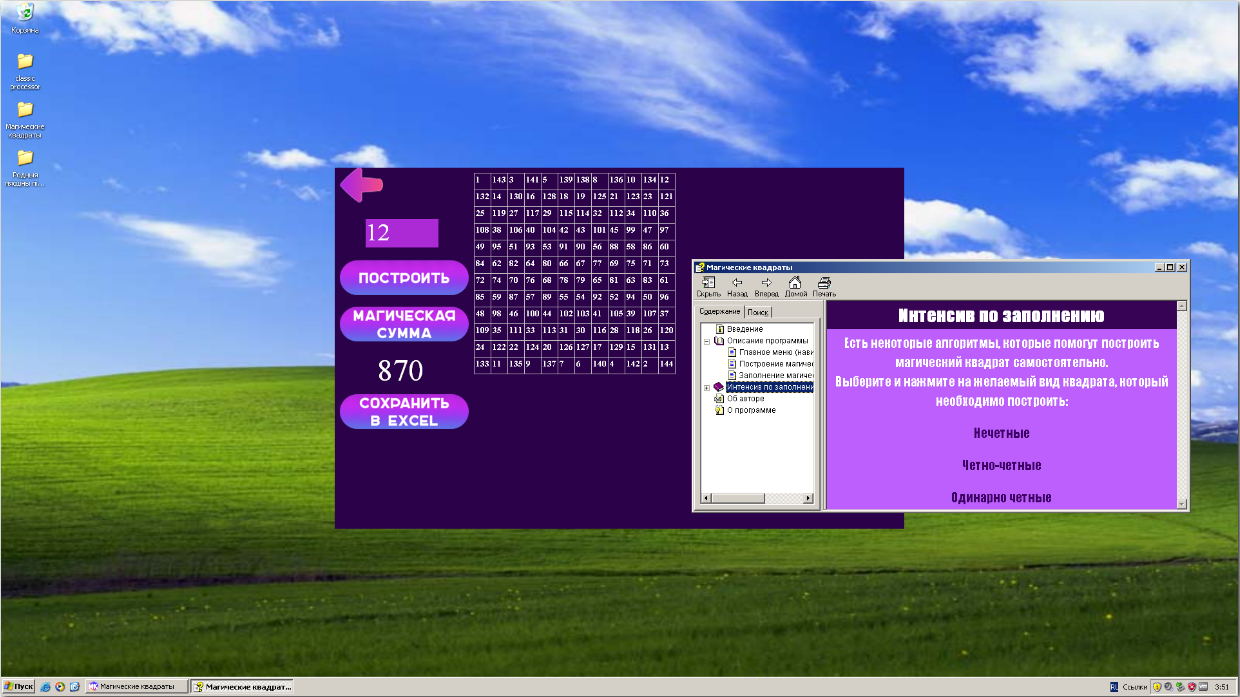


Рисунок 3.3.1 — Демонстрация работы программы на Windows XP

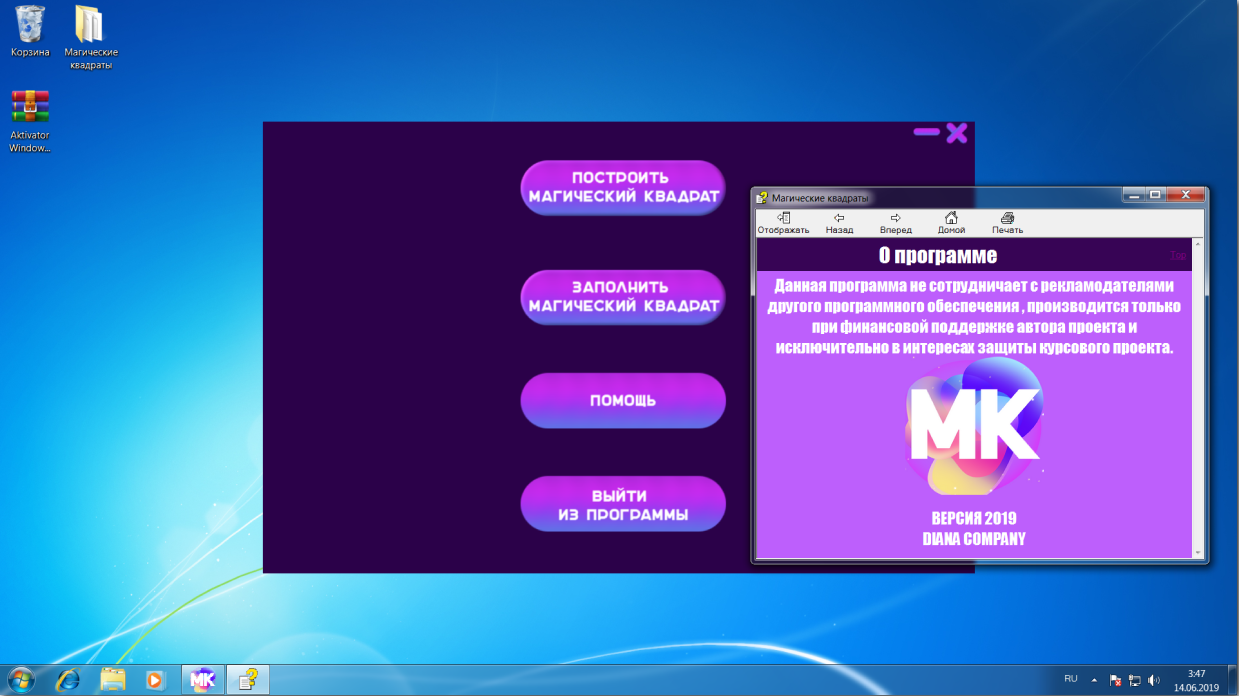


Рисунок 3.3.2 — Демонстрация работы программы на Windows 7

## **3.2. Рекомендуемый состав программных средств**

Системные программные средства, используемые программой, должны быть представлены операционной системой Windows. Для реализации всех возможностей программы требуется предустановленный пакет программ Microsoft Office, а именно Microsoft Excel. Также рекомендуется иметь клиента электронной почты для связи с разработчиком программы, соответственно доступ в интернет (см. рис. 3.2.1).

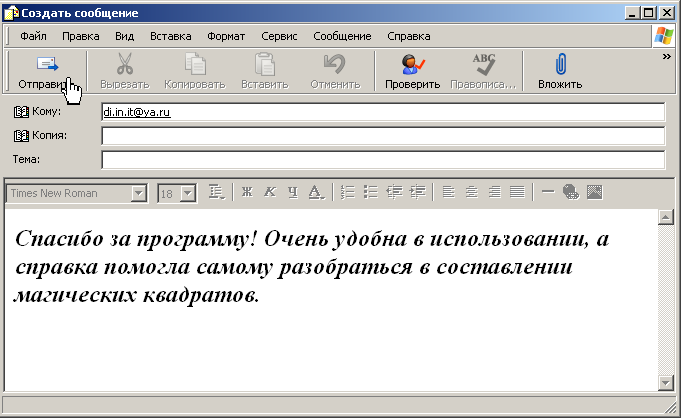


Рисунок 3.3.1 — Пример письма разработчику на Windows XP

## **3.3. Установка программы**

Для пользования программой необходимо установить её на ПЭВМ с помощью готового инсталлятора. Не рекомендуется использовать программу, скаченную из интернета, так как она может содержать вредоносное программное обеспечение. Мастера установки рекомендуется загружать с диска, на котором находится проект (см. рис.3.3.1).

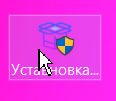


Рисунок 3.3.1 — Инсталлятор программы

После запуска инсталлятора пользователь видит окно приветствия мастера установки, который предложит закрыть работающие программы и нажать на кнопку «Далее» (см. рис.3.3.2).

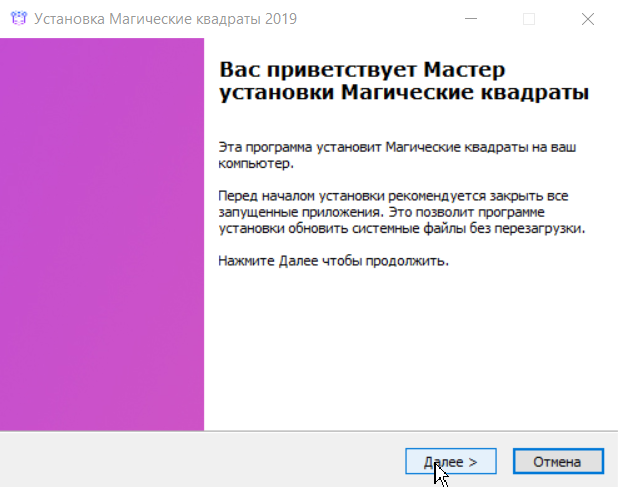


Рисунок 3.3.2 — Стартовое окно установки

В следующем окне мастер установки покажет, в какую папку он собирается установить программу. Необходимо щелкните по кнопке «Далее». Можно изменить папку назначения, то есть укажите свою, нажав на кнопку «Обзор», указать нужную папку и нажать кнопку «Далее» (см. рис.3.3.3).

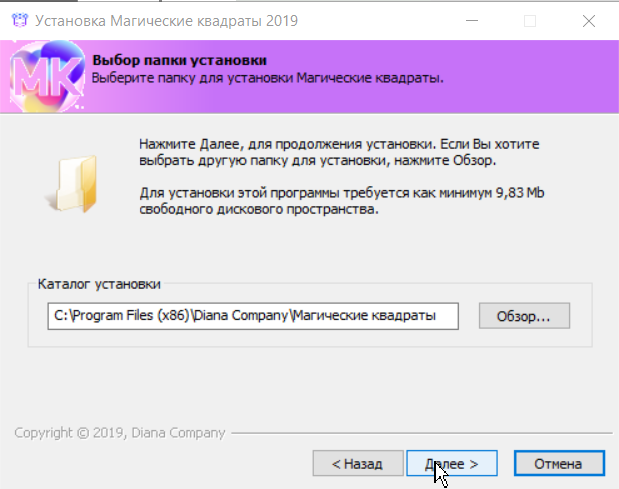


Рисунок 3.3.3 — Указание директории установки

После необходимо проверить информацию об установке и нажать кнопку «Установить» (см. рис.3.3.4).

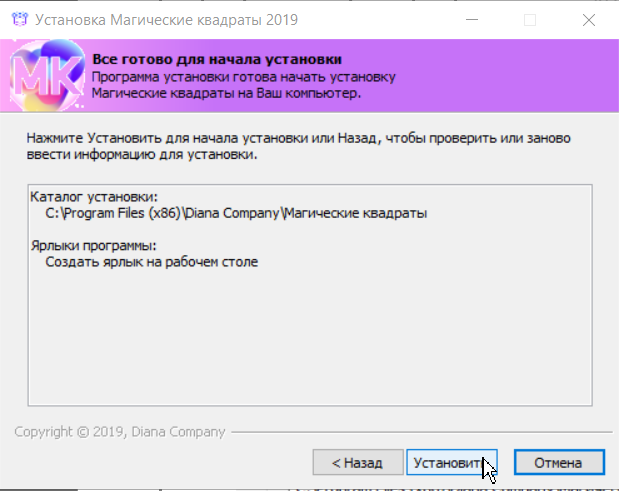


Рисунок 3.3.4 — Проверка информации об установке

Начнется процесс установки (инсталляции) программы, по окончании которого мастер сообщит о завершении процесса и попросит нажать кнопку «Готово» для выхода из программы установщика.

Также в этом окне можно сразу запустить установленную программу и прочитать ReadMe в виде файла справки (см. рис. 3.3.5).

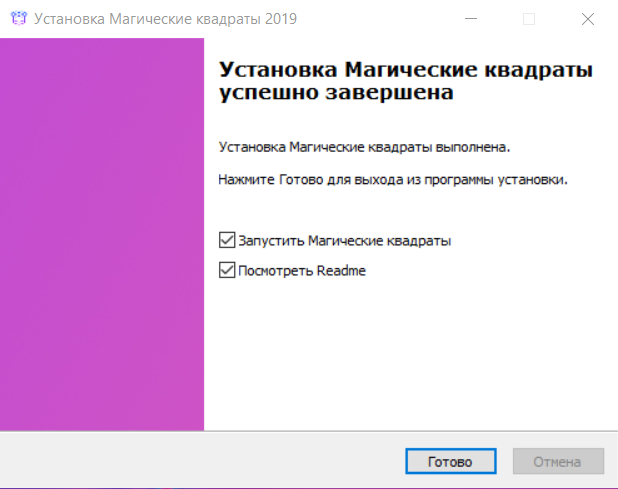


Рисунок 3.3.5 — Успешная установка программы

Теперь можно запустить приложение щелком по его ярлыку на рабочем столе (рис.3.3.6) или через главное меню и ознакомиться со всеми его возможностями.

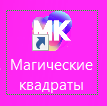


Рисунок 3.3.6 — Ярлык программы

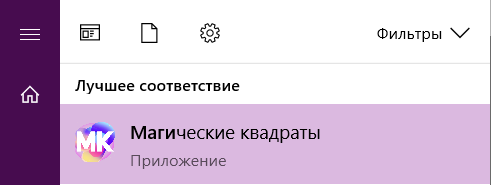


Рисунок 3.3.7 — Программа в главном меню

# ЗАКЛЮЧЕНИЕ

Актуальность темы проекта определяется тем, что в настоящее время магические квадраты применяются в криптографических алгоритмах защиты информации и приборостроении, оптимизации алгоритмов построения магических квадратов.

Цель курсового проекта: создание прикладной программы «Построение магических квадратов» была достигнута.

Были решены следующие задачи, поставленные в соответствии с целью:

1) Были изучены виды МК, множество различных методов их построения.

2) Основательно изучена среда программирования Delphi 7.

3) Созданы алгоритмы построения магических квадратов: разработаны и построены его графическая запись (блок-схема). Были несколько раз оптимизированы алгоритм и код.

4) Разработана графическая часть проекта. Детально создан дизайн в одной стилистике, спроектировано взаимодействие пользователя с интерфейсом и обеспечена интуитивная навигация по программе.

5) Была протестирована и отлажена программа. В процессе тестирования были исправлены ошибки при работе с программой, созданы окна подсказок, предусмотрены меры, чтобы уберечь пользователя от некорректных действий.

6) Были изучены правила написания технической документации на программные средства. Создана интуитивно понятная справочная система с теоретической частью для пользователя, которая поможет самому строить магические квадраты.

Программа отлажена, протестирована и готова для демонстрации и практического использования. Пояснительная записка написана в соответствие с ГОСТами и требованиями, предъявляемыми к технической документации.

Достоинствами программы являются:

1. Возможность построения магических квадратов высокого порядка, по сравнению с подобными программами.
2. Дополнительная головоломка для пользователя.
3. Невысокие системные требования.
4. Привлекательный и оригинальный дизайн программы.
5. Наличие учебного пособия по построению магических квадратов.
6. Простота и понятность использования программы.

Выявленные недостатки в ходе тестирования:

1) Возможно некорректное отображение программы на устаревших или с низкими характеристиками устройствах.

# ЛИТЕРАТУРА

## **Нормативная литература**

1. ГОСТ 19.102-77. ЕСПД. Стадии разработки.
2. ГОСТ 19.106-78. ЕСПД. Требования к программным документам, выполненным печатным способом.
3. ГОСТ 19.602-78. ЕСПД. Правила дублирования, учета и хранения программных документов, выполненных печатным способом.
4. ГОСТ 19.402-78. ЕСПД. Описание программы.
5. ГОСТ 19.404-79. ЕСПД. Пояснительная записка. Требования к содержанию и оформлению.
6. ГОСТ 19.003-80. ЕСПД. Схемы алгоритмов и программ. Обозначения условные графические.

## **Учебная литература**

1. Программирование в Delphi. Трюки и эффекты. / *Чиртик А.А. –* 2010.
2. Занимательные уроки с паскалем, или PascalABC.NET для начинающих. / *Рубанцев В. –* 2012.
3. Delphi. Справочник. / *Лишнер Р.*- 2001. / ISBN 5‑93286‑019‑7.
4. Delphi 7. Справочное пособие. / *Архангельский А.Я*. – 2004. / ISBN 5-9518-0027-7
5. Delphi. Программирование на языке высокого уровня. / *Фаронов В.В. –* 2003. / ISBN 5-8046-0008-7

## **Тематическая литература**

1. Solving Magic Squares: Generic Solutions to Solving Magic Squares. / *Donald C. Simpson –* 2001/ ISBN 0-75960-428-2.
2. Магические квадраты. / *Постников М.М.* – 1964.

# ПРИЛОЖЕНИЯ

## **ПРИЛОЖЕНИЕ А**

### **Обобщенный алгоритм программы**



## **ПРИЛОЖЕНИЕ Б**

### **Алгоритм метода решения математической части программы**



### **Соединители 1 и А. Продолжение Приложения Б.**



### **Соединители 2 и В. Продолжение Приложения Б.**



### **Соединители 3 и С. Продолжение Приложения Б.**



## **ПРИЛОЖЕНИЕ В**

**Листинг программы**

### **Форма 1. Главное меню программы**

unit Main;

interface

uses

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

Dialogs, StdCtrls, jpeg, ExtCtrls, Menus, ShellAPI;

type

TGLAV = class(TForm)

Button2: TButton;

Button3: TButton;

PopupMenu1: TPopupMenu;

N1: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

N4: TMenuItem;

N5: TMenuItem;

Postr1: TImage;

Postr2: TImage;

Game1: TImage;

Game2: TImage;

Button1: TButton;

Help1: TImage;

Help2: TImage;

Back1: TImage;

Back2: TImage;

Min: TImage;

Exit: TImage;

N6: TMenuItem;

procedure Button1Click(Sender: TObject);

procedure Postr2Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Game2Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure Back2Click(Sender: TObject);

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

procedure FormShortCut(var Msg: TWMKey; var Handled: Boolean);

procedure N1Click(Sender: TObject);

procedure N5Click(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure Postr1MouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer);

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

procedure Game1MouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer);

procedure Help1MouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer);

procedure Back1MouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer);

procedure N2Click(Sender: TObject);

procedure MinClick(Sender: TObject);

procedure ExitClick(Sender: TObject);

procedure N6Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Help2Click(Sender: TObject);

procedure N4Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

GLAV: TGLAV;

implementation

uses MSCreate, MiniGame;

{$R \*.dfm}

procedure TGLAV.Button2Click(Sender: TObject); //открытие формы с игрой

begin

GAME.Show;

GLAV.Hide;

end;

procedure TGLAV.FormCreate(Sender: TObject); //создание формы

begin

GLAV.Color:=$0049012B;

end;

procedure TGLAV.FormMouseUp(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer); //контекстное меню

begin

if Button = mbRight then PopupMenu1.Popup(326, 151);

end;

procedure TGLAV.FormShortCut(var Msg: TWMKey; var Handled: Boolean);

begin

if Msg.CharCode=(Ord('M')) then //горячая клавиша, открывающая форму построения

begin

Handled:=true;

Game.Hide;

Glav.Show;

Mag.Hide;

end;

if Msg.CharCode=(Ord('S')) then //горячая клавиша, открывающая форму построения

begin

Button1.Click;

Handled:= true;

Game.Hide;

Glav.Hide;

end;

if Msg.CharCode =(Ord('G'))then //горячая клавиша, открывающая игру

begin

Button2.Click;

Handled:= true;

Mag.Hide;

Glav.Hide;

end;

if Msg.CharCode =(Ord('H')) then //горячая клавиша, открывающая справку

begin

Button3.Click;

Handled:= true;

end;

end;

procedure TGLAV.N5Click(Sender: TObject);

begin

Application.Terminate;

end;

procedure TGLAV.N3Click(Sender: TObject);

begin

Button2.Click;

end;

procedure TGLAV.Postr1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

Postr2.Visible:=True;

Postr1.Visible:=False;

end;

procedure TGLAV.FormMouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

Postr1.Visible:=True; Postr2.Visible:=False;

Game1.Visible:=True; Game2.Visible:=False;

Help1.Visible:=True; Help2.Visible:=False;

Back1.Visible:=True; Back2.Visible:=False;

end;

procedure TGLAV.Postr2Click(Sender: TObject);

begin

Button1.Click;

end;

procedure TGLAV.Game1MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

begin

Game2.Visible:=True; Game1.Visible:=False;

end;

procedure TGLAV.Button1Click(Sender: TObject);

begin

MAG.Show;

GLAV.Hide;

end;

procedure TGLAV.Game2Click(Sender: TObject);

begin

Button2.Click;

end;

procedure TGLAV.Help1MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

begin

Help2.Visible:=True; Help1.Visible:=False;

end;

procedure TGLAV.Back1MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

begin

Back2.Visible:=True; Back1.Visible:=False;

end;

procedure TGLAV.Back2Click(Sender: TObject);

begin

Application.Terminate;

end;

procedure TGLAV.N1Click(Sender: TObject);

begin

Glav.Show;

end;

procedure TGLAV.N2Click(Sender: TObject);

begin

Glav.Hide;

MAG.Show;

end;

procedure TGLAV.MinClick(Sender: TObject); //свернуть

begin

Application.Minimize;

end;

procedure TGLAV.ExitClick(Sender: TObject); //закрыть

begin

Application.Terminate;

end;

procedure TGLAV.N6Click(Sender: TObject); //написать разработчику

begin

ShellExecute(Handle, nil, 'mailto:di.in.it@ya.ru', nil, nil, SW\_SHOW);

end;

procedure TGLAV.Button3Click(Sender: TObject);

begin

ShellExecute(0,PChar('Open'),PChar('Руководство пользователя.chm'), nil, nil, SW\_SHOWNORMAL);

end;

procedure TGLAV.Help2Click(Sender: TObject);

begin

Button3.Click;

end;

procedure TGLAV.N4Click(Sender: TObject);

begin

Button3.Click;

end;

end.

### **Форма 2. Форма построения магических квадратов**

unit MSCreate;

interface

uses

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

Dialogs, Grids, StdCtrls, jpeg, ExtCtrls, OleCtnrs, ComObj;

type

TMAG = class(TForm)

Edit1: TEdit;

Button1: TButton;

StringGrid1: TStringGrid;

Button2: TButton;

StaticText1: TStaticText;

Back1: TImage;

Back2: TImage;

OleContainer1: TOleContainer;

Str1: TImage;

Str2: TImage;

SaveDialog1: TSaveDialog;

Save1: TImage;

Save2: TImage;

Sum1: TImage;

Sum2: TImage;

procedure FormCreate(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Back2Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Back1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

procedure FormMouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Str1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

procedure Str2Click(Sender: TObject);

procedure Save1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

procedure Save2Click(Sender: TObject);

procedure FormMouseUp(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Sum1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

procedure Sum2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

const mn=100; //максимальный размер МК

var

MAG: TMAG;

p: Boolean;

n: Integer;

implementation

uses Main, ErrorSize, WindowSave;

{$R \*.dfm}

{Используется сокращение "магический квадрат" как МК в сносках}

procedure Obmen(var a, b: Integer);

//меняет местами значения двух переменных в массиве в МК

var

T: Integer;

begin

T:=a;

a:=b;

b:=T;

end;

procedure TMAG.Button1Click(Sender: TObject);

//построене МК

var

i, j, k, s, b, r, m: Integer;

MG: Extended;

a: Array[1 .. mn, 1 .. mn] of Integer; //Массив, макс.размера 100

BEGIN{1}

begin {2}

if (Length(Trim(Edit1.Text))=0) then begin

Error.Show;

Exit; end;

n:=StrToInt(Edit1.Text);

if n<3 then Error.Show;

if n>21 then SaveMS.Visible:=true;

if n>mn then begin

Error.Show; SaveMs.Visible:=false; Exit;

end;

p:=True;

//НЕЧЕТНЫЕ (3,5,7...)

if Odd(n) Then

begin {3}

i:=1;

j:=Succ(n div 2);

for k:=1 To Sqr(n) do

begin {4}

a[i,j]:=k;

if k mod n=0 Then Inc(i)

Else

begin {5}

Dec(i);

Inc(j);

if i=0 Then i:=n;

if j>n Then j:=1

end{->5}

end; {->4}

end {->3}

Else

//ЧЕТНО-ЧЕТНЫЕ(двойная четность, 4,8,12,16...)

if n mod 4=0 Then

begin {6}

k:= 1;

for i:=1 To n Do

for j:=1 To n Do

begin {7}

a[i,j]:=k;

Inc(k)

end; {->7}

j:=2;

m:=n div 2;

for i:=1 To m Do

for k:=1 To m Div 2 Do

begin {8}

if j=Succ(m) Then j:=2

Else

if j=(m+2) Then j:=1;

s:=Succ(n-i);

b:=Succ(n-j);

Obmen(a[i,j], a[s,b]);

Obmen(a[i,b], a[s,j]);

Inc(j,2)

end {->8}

end {->6}

Else

if n<>2 Then //ОДНОЧЕТНЫЕ МК(кроме двойных, 6,10,14,18...)

begin {9}

k:=1;

for i:=1 To n Do

for j:=1 To n Do

begin {10}

a[i,j]:=k;

Inc(k)

end; {->10}

r:=Pred(n div 2) div 2;

m:=n div 2;

for i:=1 To m Do

begin {11}

j:=i;

for k:=1 To r Do

begin {12}

if j>m Then j:=1;

s:=Succ(n-i); b:=Succ(n-j);

Obmen(a[i,j], a[s,b]);

Obmen(a[i,b], a[s,j]);

Inc(j)

end {->12}

end; {->11}

i:=1;

j:=Succ(r);

for k:=1 To m Do

begin {13}

if j>m Then j:=1;

s:=Succ(n-i);

Obmen(a[i,j], a[s,j]);

Inc(i);

Inc(j)

end; {->13}

i:=1;

j:=r+2;

for k:=1 To m Do

begin {14}

if j>m Then j:=1;

b:=Succ(n-j);

Obmen(a[i,j], a[i,b]);

Inc(i);

Inc(j)

end {->14}

end {->9}

Else

p:=False;

begin {15} //вывод построенных МК

for i:=1 To n Do

begin {16}

for j:=1 To n Do

begin {17}

if n>=3 then begin

StringGrid1.ColWidths[0]:= 0;

StringGrid1.RowHeights[0]:= 0;

StringGrid1.ColCount:= n + 1;

StringGrid1.RowCount:= n + 1;

StringGrid1.Cells[i,j]:= IntToStr(a[j,i]);

StringGrid1.Visible:=true; end;

end; {->17}

end {->16}

end; {->15}

end;

//расчет магической константы(суммы)

if n>=3 then begin

MG:=(n\*(n\*n+1)/2);

StaticText1.Caption:=FloatToStr(MG);

end; {->2}

END; {->1}

procedure TMAG.Button2Click(Sender: TObject); //сохранение МК в Excel

var

Excel, WorkBook, Sheet:variant;

i, j: integer;

FileExcel: string;

begin

if SaveDialog1.Execute then

FileExcel:= SaveDialog1.FileName

else

Exit;

Excel:=CreateOleObject('Excel.Application');

Excel.DisplayAlerts:=False;

Excel.Visible:=true;

Workbook:=Excel.Workbooks.Add;

Workbook.SaveAs(FileExcel);

Sheet:= Workbook.ActiveSheet;

for i:= 0 to StringGrid1.RowCount-1 do

begin

for j:= 0 to StringGrid1.ColCount-1 do

Sheet.Cells[i+1, j+1]:= StringGrid1.Cells[j, i];

end;

Workbook.Save;

{Workbook.Close;}

{Excel.Quit;}

Excel:= UnAssigned;

end;

procedure TMAG.FormMouseUp(Sender: TObject; Button: TMouseButton; //Появление контекстного меню

Shift: TShiftState; X, Y: Integer);

begin

if Button = mbRight then GLAV.PopupMenu1.Popup(326, 151);

end;

procedure TMAG.Edit1KeyPress(Sender: TObject; var Key: Char); //Маска ввода только чисел

begin

if not (Key in ['0'..'9', #8])then Key:=#0;

end;

procedure TMAG.Back2Click(Sender: TObject);

begin

MAG.Close;

GLAV.Show;

end;

procedure TMAG.FormCreate(Sender: TObject);

begin

MAG.Color:=GLAV.Color;

StaticText1.Font.Color:=clWhite; StringGrid1.Font.Color:=clWhite;

StaticText1.Color:=MAG.Color; StringGrid1.Color:=MAG.Color;

Edit1.Color:=$00D52AAC; StringGrid1.Font.Style:=[fsBold];

end;

procedure TMAG.Back1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

Back1.Visible:=false; Back2.Visible:=true;

end;

procedure TMAG.FormMouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

Back2.Visible:=false; Back1.Visible:=true;

Str2.Visible:=false; Str1.Visible:=true;

Save2.Visible:=false; Save1.Visible:=true;

Sum2.Visible:=false; Sum1.Visible:=true;

end;

procedure TMAG.Str1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

Str1.Visible:=false; Str2.Visible:=true;

end;

procedure TMAG.Str2Click(Sender: TObject);

begin

Button1.Click;

end;

procedure TMAG.Save1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

Save1.Visible:=false; Save2.Visible:=true;

end;

procedure TMAG.Save2Click(Sender: TObject);

begin

Button2.Click;

end;

procedure TMAG.Sum1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

Sum1.Visible:=false; Sum2.Visible:=true;

end;

procedure TMAG.Sum2Click(Sender: TObject);

begin

StaticText1.Visible:=true;

end;

end.

### **Форма 3. Форма заполнения магического квадрата**

unit MiniGame;

interface

uses

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

Dialogs, StdCtrls, jpeg, ExtCtrls;

type

TGAME = class(TForm)

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

Edit4: TEdit;

Edit5: TEdit;

Edit6: TEdit;

Edit7: TEdit;

Edit8: TEdit;

Edit9: TEdit;

Button1: TButton;

Button2: TButton;

Back1: TImage;

Back2: TImage;

Prov1: TImage;

Prov2: TImage;

Clean1: TImage;

Clean2: TImage;

Exit1: TImage;

Min: TImage;

ImTrue: TImage;

ImFalse: TImage;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure Back1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

procedure Back2Click(Sender: TObject);

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

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Edit2KeyPress(Sender: TObject; var Key: Char);

procedure Edit3KeyPress(Sender: TObject; var Key: Char);

procedure Edit4KeyPress(Sender: TObject; var Key: Char);

procedure Edit5KeyPress(Sender: TObject; var Key: Char);

procedure Edit6KeyPress(Sender: TObject; var Key: Char);

procedure Edit7KeyPress(Sender: TObject; var Key: Char);

procedure Edit8KeyPress(Sender: TObject; var Key: Char);

procedure Edit9KeyPress(Sender: TObject; var Key: Char);

procedure Prov1MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

procedure Clean1MouseMove(Sender: TObject; Shift: TShiftState; X,Y: Integer);

procedure Clean2Click(Sender: TObject);

procedure Prov2Click(Sender: TObject);

procedure FormMouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure FormMouseUp(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure MinClick(Sender: TObject);

procedure Exit1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

GAME: TGAME;

implementation

uses ErrorCheck, Main;

{$R \*.dfm}

procedure TGAME.Button1Click(Sender: TObject);

var

a:array [1..3,1..3] of integer;

pr:boolean;

i,sum: integer;

BEGIN //проверка на пустые поля

if (Length(Trim(Edit1.Text))=0)or

(Length(Trim(Edit2.Text))=0)or

(Length(Trim(Edit3.Text))=0)or

(Length(Trim(Edit4.Text))=0)or

(Length(Trim(Edit5.Text))=0)or

(Length(Trim(Edit6.Text))=0)or

(Length(Trim(Edit7.Text))=0)or

(Length(Trim(Edit8.Text))=0)or

(Length(Trim(Edit9.Text))=0)then

begin Prover.Show; Exit; end;

a[1,1]:=StrToInt(Edit1.Text); //считывание чисел

a[1,2]:=StrToInt(Edit2.Text);

a[1,3]:=StrToInt(Edit3.Text);

a[2,1]:=StrToInt(Edit4.Text);

a[2,2]:=StrToInt(Edit5.Text);

a[2,3]:=StrToInt(Edit6.Text);

a[3,1]:=StrToInt(Edit7.Text);

a[3,2]:=StrToInt(Edit8.Text);

a[3,3]:=StrToInt(Edit9.Text);

pr:=true;

if a[1,1]<>a[1,2] then begin //проверка суммы

sum:=0;

for i:=1 to 3 do

sum:=sum+a[1,i];

for i:=1 to 3 do

if(sum <> a[i,1]+a[i,2]+a[i,3]) then

pr:=false;

for i:=1 to 3 do

if(sum <> a[1,i]+a[2,i]+a[3,i]) then

pr:=false;

if(sum <> a[1,1]+a[2,2]+a[3,3]) then

pr:=false;

if (pr=false) then //вывод результата

ImFalse.Visible:=true; ImTrue.Visible:=false;

if (pr=true) then

ImTrue.Visible:=true; ImFalse.Visible:=false;

end;

if a[1,1]=a[1,2] then begin

ImFalse.Visible:=true; ImTrue.Visible:=false;

end;

END;

procedure TGAME.Button2Click(Sender: TObject); //очистка полей

begin

Edit1.Clear; Edit2.Clear; Edit3.Clear;

Edit4.Clear; Edit5.Clear; Edit6.Clear;

Edit7.Clear; Edit8.Clear; Edit9.Clear;

end;

procedure TGAME.MinClick(Sender: TObject);

begin

Application.Minimize;

end;

procedure TGAME.Exit1Click(Sender: TObject);

begin

Application.Terminate;

end;

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

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Glav.perform(WM\_SysCommand,$F012,0);

end;

procedure TGAME.FormMouseUp(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer); //контекстное меню

begin

if Button = mbRight then GLAV.PopupMenu1.Popup(326, 151);

end;

procedure TGAME.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9', #8])then Key:=#0;

end;

procedure TGAME.Edit2KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9', #8])then Key:=#0;

end;

procedure TGAME.Edit3KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9', #8])then Key:=#0;

end;

procedure TGAME.Edit4KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9', #8])then Key:=#0;

end;

procedure TGAME.Edit5KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9', #8])then Key:=#0;

end;

procedure TGAME.Edit6KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9', #8])then Key:=#0;

end;

procedure TGAME.Edit7KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9', #8])then Key:=#0;

end;

procedure TGAME.Edit8KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9', #8])then Key:=#0;

end;

procedure TGAME.Edit9KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in ['0'..'9', #8])then Key:=#0;

end;

procedure TGAME.FormCreate(Sender: TObject);

begin

Game.Color:=GLAV.Color;

Edit1.Color:=$00D52AAC; Edit2.Color:=Edit1.Color; Edit3.Color:=Edit1.Color;

Edit4.Color:=Edit1.Color; Edit5.Color:=Edit1.Color; Edit6.Color:=Edit1.Color;

Edit7.Color:=Edit1.Color; Edit8.Color:=Edit1.Color; Edit9.Color:=Edit1.Color;

ImTrue.Visible:=false; ImFalse.Visible:=false;

end;

procedure TGAME.Prov1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

Prov1.Visible:=false; Prov2.Visible:=true;

end;

procedure TGAME.FormMouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

Back2.Visible:=false;

Prov2.Visible:=false; Prov1.Visible:=true;

Clean2.Visible:=false; Clean1.Visible:=true;

end;

procedure TGAME.Back1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

Back2.Visible:=true;

end;

procedure TGAME.Back2Click(Sender: TObject);

begin

GAME.Close;

GLAV.Show;

end;

procedure TGAME.Clean1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

Clean1.Visible:=false; Clean2.Visible:=true;

end;

procedure TGAME.Clean2Click(Sender: TObject);

begin

Button2.Click;

end;

procedure TGAME.Prov2Click(Sender: TObject);

begin

Button1.Click;

end;

end.

### **Форма 4. Ошибка размерности**

unit ErrorSize;

interface

uses

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

Dialogs, StdCtrls, ExtCtrls;

type

TError = class(TForm)

StaticText1: TStaticText;

Button1: TButton;

Bevel1: TBevel;

Ok1: TImage;

procedure FormCreate(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Ok1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Error: TError;

implementation

{$R \*.dfm}

procedure TError.FormCreate(Sender: TObject);

begin

Error.Color:=$00F258B0;

StaticText1.Caption:='Введена неправильная размерность. Минимальная размерность 3х3, максимальная 100х100.';

end;

procedure TError.Button1Click(Sender: TObject);

begin

Error.Close;

end;

procedure TError.Ok1Click(Sender: TObject);

begin

Button1.Click;

end;

end.

### **Форма 5. Ошибка заполнения**

unit ErrorCheck;

interface

uses

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

Dialogs, StdCtrls, ExtCtrls;

type

TProver = class(TForm)

StaticText1: TStaticText;

Bevel1: TBevel;

Okey: TImage;

procedure OkeyClick(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Prover: TProver;

implementation

{$R \*.dfm}

procedure TProver.OkeyClick(Sender: TObject);

begin

Prover.Close;

end;

procedure TProver.FormCreate(Sender: TObject);

begin

Prover.Color:=$00F258B0;

StaticText1.Caption:='Квадрат заполнен не полностью. Пожалуйста проверьте ещё раз.';

end;

end.

### **Форма 6. Информационное окно сохранения**

unit WindowSave;

interface

uses

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

Dialogs, ExtCtrls, StdCtrls;

type

TSaveMS = class(TForm)

Bevel1: TBevel;

Sav1: TImage;

StaticText1: TStaticText;

procedure FormCreate(Sender: TObject);

procedure Sav1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

SaveMS: TSaveMS;

implementation

uses MSCreate;

{$R \*.dfm}

procedure TSaveMS.FormCreate(Sender: TObject);

begin

StaticText1.Caption:='Магический квадрат отображен не полностью. Сохраните в Microsoft Excel, чтобы полностью увидеть магический квадрат. ';

SaveMS.Color:=$00F258B0;

end;

procedure TSaveMS.Sav1Click(Sender: TObject);

begin

SaveMS.Close;

end;

end.