Отдел образования Жлобинского райисполкома

Государственное учреждение образования

«Средняя общеобразовательная школа № 11 г. Жлобина»

Учебно-исследовательская работа по информатике

Создание мини-игры «Магический квадрат»

Выполнила

ученица 8 «А» класса

средней школы № 11 г. Жлобина

Забелова Екатерина

Руководитель:

Чупругина Наталья Анатольевна

учитель информатики

первой категории

СОШ № 11 г. Жлобина

2010г.

Введение

Тема моей научно-исследовательской работы «Создание мини-игры «Магический квадрат»». Я заинтересовалась идеей создания математической мини-игры. Имеющихся у меня знаний языка программирования Pascal оказалось недостаточно, хотя бы потому, что он не позволяет создавать программы с привлекательным и удобным интерфейсом. Передо мной встали две проблемы: 1) методы заполнения квадратной матрицы; 2) выбор языка программирования для реализации игры. Первая проблема легко решилась после изучения соответствующей литературы. Над второй пришлось поработать. Был выбран язык программирования Delphi, ядром которого является Pascal. Изучалась специальная литература, оказали помощь интернет-форумы по некоторым вопросам.

Работа актуальна для тех, кто хотел бы начать овладение языком программирования Delphi. Народная мудрость гласит: «Никого ничему нельзя научить, но всему можно научиться». На примере создания мини игры «Магический квадрат» можно научиться использовать компоненты, оформлять процедуры, менять свойства объектов, обрабатывать события. А практический вид познания – самый эффективный вид познания.

Лично для меня эта работа большой шаг в изучении и, самое главное, в практическом применении языка программирования высокого уровня.

Объектом исследования является программирование на языке Delphi.

Предмет исследования – разработка программ создания мини-игр на языке программирования Delphi.

Целью нашего исследования было:

теоретическое обоснование и разработка программы создания мини игры «Магический квадрат».

Гипотеза исследования:

Если

* проанализировать содержание понятия «магический квадрат» и методы построения магических квадратов;
* проанализировать назначение и возможности Delphi;
* проанализировать пути решения проблем создания мини-игр на языке Delphi;
* разработать программу мини-игры «Магический квадрат»

То это обеспечит

* + развитие и прикладное применение навыков программирования;
  + получение навыков программирования в популярной системе программирования Delphi;
  + неформальное усвоение материала по программированию на языке Delphi.

В соответствии с целью и гипотезой исследования ставились и решались следующие задачи:

Написать программу создания мини-игры «Магический квадрат», которая, во-первых, генерирует поле магического квадрата 3×3 с некоторыми пустыми клетками; во-вторых, проверяет, является ли заполненная квадратная матрица «магическим» квадратом.

# Теоретическая часть

МАГИЧЕСКИЙ КВАДРАТ, квадратная таблица из целых чисел, в которой суммы чисел вдоль любой строки, любого столбца и любой из двух главных диагоналей равны одному и тому же числу.

Магический квадрат – древнекитайского происхождения. Согласно легенде, во времена правления императора Ю (ок. 2200 до н.э.) из вод Хуанхэ (Желтой реки) всплыла священная черепаха, на панцире которой были начертаны таинственные иероглифы (рис. 1, а), и эти знаки известны под названием ло-шу и равносильны магическому квадрату, изображенному на рис. 1, б. В 11 в. о магических квадратах узнали в Индии, а затем в Японии, где в 16 в. магическим квадратам была посвящена обширная литература. Европейцев с магическими квадратами познакомил в 15 в. византийский писатель Э.Мосхопулос. Первым квадратом, придуманным европейцем, считается квадрат А.Дюрера (рис. 2), изображенный на его знаменитой гравюре Меланхолия 1. Дата создания гравюры (1514) указана числами, стоящими в двух центральных клетках нижней строки. Магическим квадратам приписывали различные мистические свойства. В 16 в. Корнелий Генрих Агриппа построил квадраты 3-го, 4-го, 5-го, 6-го, 7-го, 8-го и 9-го порядков, которые были связаны с астрологией 7 планет. Бытовало поверье, что выгравированный на серебре магический квадрат защищает от чумы. Даже сегодня среди атрибутов европейских прорицателей можно увидеть магические квадраты.

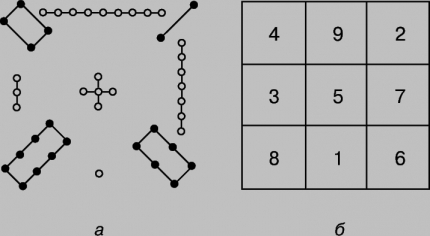


Рис. 1. ЛО-ШУ

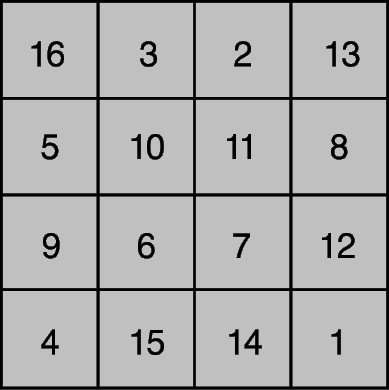


Рис. 2. КВАДРАТ ДЮРЕРА.

В 19 и 20 вв. интерес к магическим квадратам вспыхнул с новой силой. Их стали исследовать с помощью методов высшей алгебры и операционного исчисления.

Каждый элемент магического квадрата называется клеткой. Квадрат, сторона которого состоит из n клеток, содержит n2 клеток и называется квадратом n-го порядка. В большинстве магических квадратов используются первые n последовательных натуральных чисел. Сумма S чисел, стоящих в каждой строке, каждом столбце и на любой диагонали, называется постоянной квадрата и равна S = n(n2 + 1)/2. Доказано, что n ≥ 3. Для квадрата 3-го порядка S = 15, 4-го порядка – S = 34, 5-го порядка – S = 65.

Две диагонали, проходящие через центр квадрата, называются главными диагоналями. Ломаной называется диагональ, которая, дойдя до края квадрата, продолжается параллельно первому отрезку от противоположного края (такую диагональ образуют заштрихованные клетки на рис. 3). Клетки, симметричные относительно центра квадрата, называются кососимметричными. Таковы, например, клетки a и b на рис. 3.

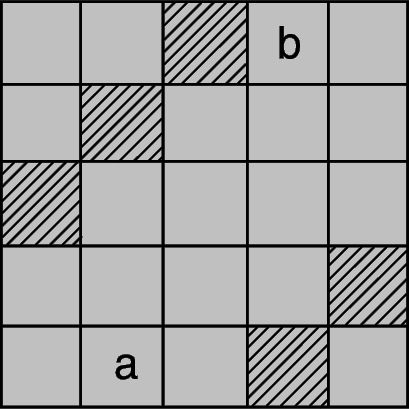


Рис. 3. ЛОМАНАЯ ДИАГОНАЛЬ И КОСОСИММЕТРИЧНЫЕ КЛЕТКИ

Правила построения магических квадратов делятся на три категории в зависимости от того, каков порядок квадрата: нечетен, равен удвоенному нечетному числу или равен учетверенному нечетному числу. Общий метод построения всех квадратов неизвестен, хотя широко применяются различные схемы. Мы рассмотрим ниже только один метод - метод построения магических квадратов нечетного порядка, который и использовали для заполнения матрицы числами.

Магические квадраты нечетного порядка можно построить с помощью метода французского геометра 17 в. А.де ла Лубера. Рассмотрим этот метод на примере квадрата 5-го порядка (рис. 4). Число 1 помещается в центральную клетку верхней строки. Все натуральные числа располагаются в естественном порядке циклически снизу вверх в клетках диагоналей справа налево. Дойдя до верхнего края квадрата (как в случае числа 1), продолжаем заполнять диагональ, начинающуюся от нижней клетки следующего столбца. Дойдя до правого края квадрата (число 3), продолжаем заполнять диагональ, идущую от левой клетки строкой выше. Дойдя до заполненной клетки (число 5) или угла (число 15), траектория спускается на одну клетку вниз, после чего процесс заполнения продолжается.



Рис. 4. МЕТОД ДЕ ЛА ЛУБЕРА.

# Практическая часть

Выражение «написать программу» отражает только один из этапов создания компьютерной программы, когда разработчик программы (программист) действительно пишет команды (инструкции) на бумаге или при помощи текстового редактора.

Программирование – это процесс создания (разработки) программы, который может быть представлен последовательностью следующих шагов:

1. Спецификация (определение, формулирование требований к программе).

2. Разработка алгоритма.

3. Кодирование (запись алгоритма на языке программирования).

4. Отладка.

5. Тестирование.

6. Создание справочной системы.

7. Создание установочного диска (CD-ROM).

## 

## Спецификация

Спецификация, определение требований к программе — один из важнейших этапов, на котором подробно описывается исходная информация, формулируются требования к результату, поведение программы в особых случаях (например, при вводе неверных данных), разрабатываются диалоговые окна, обеспечивающие взаимодействие пользователя и программы.

На этом этапе я определила последовательность этапов создания программы.

1. Создание и оформление окна программы – главной формы.

2. Заполнение полей-клеток магического квадрата (предусмотреть защиту от ввода пользователем символов, кроме чисел).

3. Проверка квадратной матрицы на предмет, является ли она магическим квадратом.

4. Оформление диалогового окна «Магический квадрат», вызываемого одноименной командой главного меню.

5. Оформление диалогового окна «Правила игры», вызываемого одноименной командой главного меню.

Подобрала информацию необходимую для создания программы. Определила, как должен выглядеть результат (Рисунок 5).

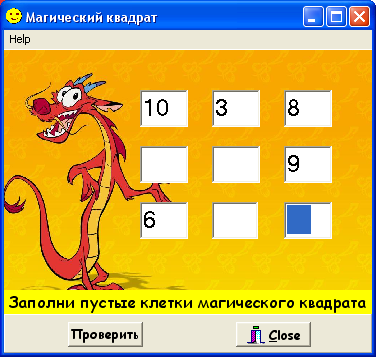


Рис. 5. «МАГИЧЕСКИЙ КВАДРАТ»

## 

## Разработка алгоритма

На этапе разработки алгоритма необходимо определить последовательность действий, которые надо выполнить для получения результата. Если задача может быть решена несколькими способами и, следовательно, возможны различные варианты алгоритма решения, то программист, используя некоторый критерий, например скорость решения алгоритма, выбирает наиболее подходящее решение. Результатом этапа разработки алгоритма является подробное словесное описание алгоритма или его блок-схема.

Далее приводится словесное описание алгоритма

1. Создание и оформление окна программы – главной формы

1.1 Выбор и размещение на форме компонентов

В Object Inspector на вкладке Properties значение свойства Position установить PoScreenCenter – для того, чтобы после запуска программы, форма появлялась в центре экрана. Name – Form1.

В редакторе Image Editor создать Icon File, нарисовать смайлик. Сохранить в файле Icon1.ico. В Object Inspector на вкладке Properties справа от свойства Icon щелкнуть мышью и в окне выбора файла иконы загрузить созданный файл.

Компоненты вкладки Standart:

1.1.1 TMainMenu – главное меню формы, Name – MainMenu1 (Содержит один пункт меню «Help» (Name – Help1) с двумя пунктами подменю «Магический квадрат» (Name – N1) и «Правила игры» (Name – N2)).

После установки компонента на форму создать пункт меню «Help». Для этого дважды щелкнуть на компоненте левой кнопкой мыши, в окне инспектора объектов вводим название пункта меню «Help» в строке свойства Caption, нажимаем клавишу Enter. Для создания пунктов подменю щелкнуть мышью в окне конструктора меню ниже пункта меню «Help», набрать «Магический квадрат», Enter, набрать «Правила игры», Enter. Чтобы связать с выбором пункта меню нужное действие, определяем процедуру обработки его события OnClick – вызов соответствующего диалогового окна.

1.1.2 TPanel – панель, Name – Panel1 (Для размещения кнопок). Значение свойства Align выбираем alBottom, чтобы панель разместить внизу формы. Свойство Caption очистить.

1.1.3 TButton – кнопка (Одна кнопка «Проверить», вторая, расположенная над ней, «Начать»). Кнопку Button1 размещаем на панели, Name – Button1. Свойство Caption – «Проверить». Кнопку Button2, Name – Button2, размещаем на кнопке Button1. Свойство Caption – «Начать».

1.1.4 TLabel – метка (Для вывода текста комментариев действий пользователя). Метку Label размещаем на форме. Name – Lb1. Свойство Caption очистить. Значение свойства Align выбираем alBottom. Свойство Color – clYellow. С помощью свойства Font корректируем шрифт.

Компоненты вкладки Additional:

1.1.5 TBitBtn – кнопка с изображением (Кнопка Close для закрытия окна); Name – BitBtn1. Кнопка Close размещается на панели. Свойство Kind – bkClose. Значение bkClose определяет типовой значок, надпись Close на кнопке, а также типовую функцию закрытия окна, которая будет связана с этой кнопкой.

1.1.6 TMaskEdit – поле с маской ввода (9 полей с маской ввода только цифровой информации). Устанавливаем ниже перечисленные значения свойств для всех девяти полей. Name – MEd1 .. MEd9. Раскрываем диалоговое окно свойства EditMask. В поле Input Mask вводим следующую последовательность символов: 9999;1;. В поле Character for Blanks вместо умалчиваемого символа-заполнителя, которым является символ подчеркивания (\_), указываем пробел и закрываем окно.

1.2 Процедура обработки события OnPaint (для того, чтобы сделать рисунок с расширением .bmp фоновым). Свойство Bitmap содержит ссылку на объект класса TBitmap. Перед загрузкой объекта Bitmap его надо создать. Метод StretchDraw() растягивает или сжимает картинку так, чтобы она заполнила всю форму.

1.3 Процедура обработки события OnCanResize (запрещающая изменение размеров формы).

1.4 Процедура обработки события OnActivate (программно задаем заголовок формы, обеспечиваем показ кнопки «Начать», скрываем кнопку «Проверить»).

2. Заполнение полей-клеток магического квадрата

2.1 Процедура обработки события OnClick кнопки Button2 («Начать»).

2.1.1 Процедура zapolnenie(x,y:integer) (По входящему параметру х заполняет клетки квадрата, по входящему параметру у одним из четырех способов очищает некоторые клетки, выводит в метке соответственную надпись, показывает кнопку «Проверить», прячет кнопку «Начать»).

2.1.2 В теле главной процедуры проверяем, была ли ситуации «Попробуй еще раз» (z=true). Если была, то вызываем процедуру zapolnenie(x,y:integer) со старыми значениями х и у; если нет (z=false), генерируем у – способ удаления клеток, х – начальное число х. Заполняем матрицу.

3. Проверка квадратной матрицы на предмет, является ли она магическим квадратом

Процедура обработки события OnClick кнопки Button1 («Проверить»). Пробуем преобразовать к числовому виду значения полей TMaskEdit. Если получилось (не осталось пустых полей), находим суммы по строкам, по столбцам, по диагоналям. Если полученные суммы равны, то выводим в метке надпись «Молодец! Все верно!», прячем кнопку «Проверить», показываем кнопку «Начать», z:=false (следующее заполнение будет сгенерировано). Если полученные суммы не равны, то выводим в метке надпись «Попробуй еще раз», прячем кнопку «Проверить», показываем кнопку «Начать», z:=true (следующее заполнение будет тем же, х и у не изменятся). Если не получилось преобразовать к числовому виду значения полей TMaskEdit, подаем звуковой сигнал, выводим сообщение «Не все клетки заполнены», выходим из процедуры.

4. Оформление диалогового окна «Магический квадрат», вызываемого одноименной командой главного меню

Caption – Магический квадрат, Name – OKBottomDlg. Размещаем компонент TMemo – многострочное редактируемое текстовое поле, вкладки Standart. Щелкаем мышью справа от свойства Lines. В окне String List Editor вводим текст о магическом квадрате. Размещаем компонент TStringGrid – текстовая таблица, вкладки Additional. Процедура обработки события OnPaint (для того, чтобы сделать рисунок с расширением .bmp фоновым) содержит процедуру пользователя для заполнения таблицы числами.

В процедуре создаем объект Bitmap, загружаем его из файла. Методом StretchDraw() заполняем всю форму рисунком Bitmap.

5. Оформление диалогового окна «Правила игры», вызываемого одноименной командой главного меню

Caption – Правила игры, Name – OKBottomDlg1. Размещаем компонент TMemo – многострочное редактируемое текстовое поле, вкладки Standart. Щелкаем мышью справа от свойства Lines. В окне String List Editor вводим текст с правилами игры. Процедура обработки события OnPaint (для того, чтобы сделать рисунок с расширением .bmp фоновым).

## Кодирование

После того как определены требования к программе и составлен алгоритм решения, алгоритм записывается на выбранном языке программирования. В результате получается исходная программа. На этом этапе я выбрала язык программирования Delphi и записала алгоритм, листинг которого приводится в приложении.

## Анализ и тестирование

Программа запускается с помощью файла Magic\_sqr.exe. После запуска открывается окно, в котором необходимо заполнить матрицу нажатием кнопки «Начать» или покинуть программу.



Рис. 6. ПРОГРАММА ПОСЛЕ ЗАПУСКА

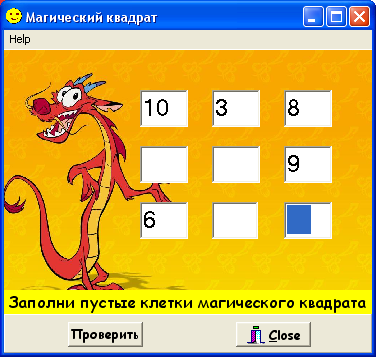


Рис. 7. ПРОГРАММА ПОСЛЕ ЩЕЛЧКА МЫШИ ПО КНОПКЕ «НАЧАТЬ»

После заполнения матрицы нажимаем на кнопку «Проверить», и на экране выводится сообщение (рисунок 8 или 9).

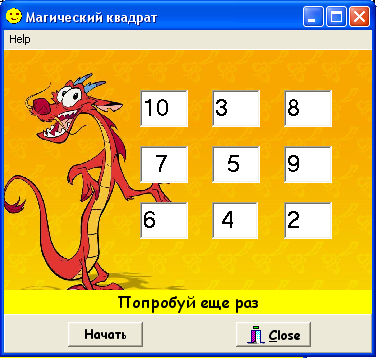


Рис. 8. ПРОГРАММА ПОСЛЕ ЩЕЛЧКА МЫШИ ПО КНОПКЕ «ПРОВЕРИТЬ»

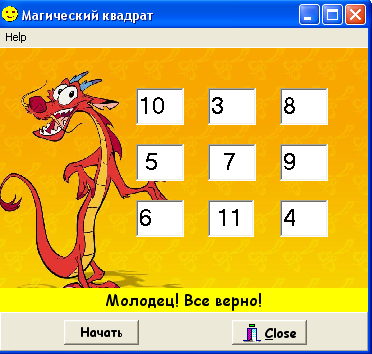


Рис. 9. ПРОГРАММА ПОСЛЕ ЩЕЛЧКА МЫШИ ПО КНОПКЕ «ПРОВЕРИТЬ»

На этом этапе программа была неоднократно протестирована, ошибок не выявлено.

## Создание справочной системы

Программа имеет меню «Help», которое содержит два пункта: «Магический квадрат» и «Правила игры».

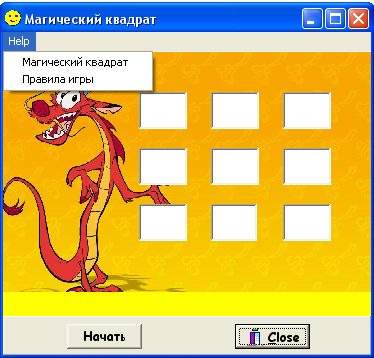


Рис. 10. ПРОГРАММА ПОСЛЕ ЩЕЛЧКА МЫШИ ПО МЕНЮ «HELP»

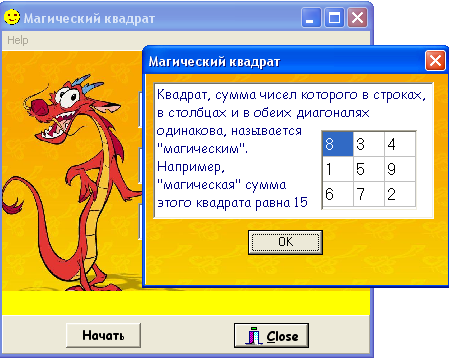


Рис. 11. ПРОГРАММА ПОСЛЕ ВЫБОРА ПУНКТА МЕНЮ «МАГИЧЕСКИЙ КВАДРАТ»

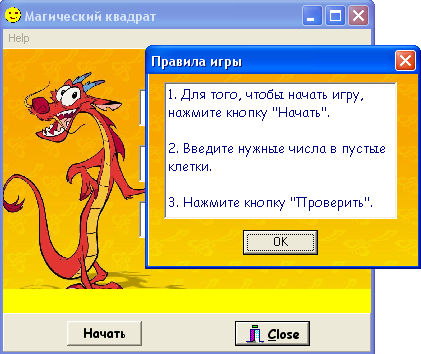


Рис. 12. ПРОГРАММА ПОСЛЕ ВЫБОРА ПУНКТА МЕНЮ «ПРАВИЛА ИГРЫ»

## Создание установочного диска

Из-за невысокой сложности программы, создание установочного диска нецелесообразно.

# Заключение

В настоящее время в нашу жизнь широко внедряются информационные технологии. На сегодняшний день, наверное, не осталось такой области, где бы они ни применялись.

Постоянно растет объем и сложность обрабатываемой информации, требуются все новые и новые виды ее представления. Как показывает практика, большинство пользователей уже не представляют себе, как бы они выполняли свою работу, с которой прекрасно справлялись еще 5 – 10 лет назад, без помощи компьютера.

В ходе исследовательской работы разработана тестирующая программа. Проектирование выполнено в системе Delphi. В работе:

* представлена информация о магическом квадрате и используемом методе заполнения магического квадрата;
* приведен теоретический материал о компонентах, используемых в программе;
* представлен листинг программы;
* приведен анализ и тестирование программы с подробным описанием.

В ходе работы я познакомилась с более широкими возможностями языка программирования Delphi, по сравнению с языком Pascal. Постараюсь применить полученные знания при участии в олимпиадном движении по программированию. А также очень хотелось бы создать серию математических мини-игр.

# Библиографический список

1. Архангельский А.Я. Delphi 6. Справочное пособие – М.: ЗАО «Издательство БИНОМ», 2001.

2. Болл У., Коксетер Г. «Математические эссе и развлечения» - М.: Мир, 1986.

3. Культин Н.Б. Основы программирования в Delphi 7 – СПб.: БХВ-Петербург, 2003.

4. Постников М.М. «Магические квадраты» - М.: Наука, 1964.

5. Фаронов В.В. Delphi. Программирование на языке высокого уровня: Учебник для вузов – СПб.: Питер, 2008. -640 с.

# Приложение

## 

## Листинг программы создания главного окна

unit mag2;

interface

uses

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

Dialogs, Buttons, StdCtrls, Mask, ExtCtrls, Menus;

type

TForm1 = class(TForm)

Panel1: TPanel; //Панель, на которой размещены кнопки

Lb1: TLabel; //Метка, для вывода комментариев к программе

MEd1: TMaskEdit; //Поля с маской ввода - для заполнения квадрата

MEd2: TMaskEdit;

MEd3: TMaskEdit;

MEd4: TMaskEdit;

MEd5: TMaskEdit;

MEd6: TMaskEdit;

MEd7: TMaskEdit;

MEd8: TMaskEdit;

MEd9: TMaskEdit;

Button1: TButton; //Кнопка "Проверить"

BitBtn1: TBitBtn; //Кнопка "Close"

Button2: TButton; //Кнопка "Начать"

MainMenu1: TMainMenu; //Главное меню формы

Help1: TMenuItem; //Один пункт меню "Help"

N1: TMenuItem; //Пункт подменю "Магический квадрат"

N2: TMenuItem; //Пункт подменю "Правила игры"

procedure FormActivate(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure FormCanResize(Sender: TObject; var NewWidth,

NewHeight: Integer; var Resize: Boolean);

procedure N1Click(Sender: TObject);

procedure N2Click(Sender: TObject);

procedure FormPaint(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

Bitmap:TBitmap; //фоновая картинка

implementation

uses dialog\_mag, dialog\_pravila;

var

x,y:integer; //х - начальное число, с которого заполняем таблицу,

//у - номер способа очистки клеток

z:boolean; //Индикатор правильности заполнения пользователем пустых клеток

{$R \*.dfm}

procedure TForm1.FormPaint(Sender: TObject); //Процедура рисования фона формы

begin

Bitmap:=TBitmap.Create; //Создание объекта-битового образа

Bitmap.LoadFromFile('ris.bmp'); //Загрузка битового изображения из файла

Canvas.StretchDraw(Rect(0, 0, Form1.clientWidth, Form1.clientHeight),Bitmap);

//Рисование и заполнение формы рисунком Bitmap

end;

procedure TForm1.FormCanResize(Sender: TObject; var NewWidth, //Процедура,

NewHeight: Integer; var Resize: Boolean); //запрещающая изменение размеров формы

begin

Resize:=false;

end;

procedure TForm1.FormActivate(Sender: TObject); //Активизация формы

begin

Caption:='Магический квадрат'; //Установить заголовок формы "Магический квадрат"

Button2.Show; //Показать кнопку "Начать"

Button1.Hide; //Спрятать кнопку "Проверить"

end;

procedure TForm1.Button2Click(Sender: TObject); //Обработка нажатия кнопки "Начать"

procedure zapolnenie(x,y:integer);//Процедура заполнения

begin

MEd1.ReadOnly:=false; //Очищаем поля ввода

MEd2.ReadOnly:=false;

MEd3.ReadOnly:=false;

MEd4.ReadOnly:=false;

MEd5.ReadOnly:=false;

MEd6.ReadOnly:=false;

MEd7.ReadOnly:=false;

MEd8.ReadOnly:=false;

MEd9.ReadOnly:=false;

MEd2.Text:=IntToStr(x); //Заполняем поля ввода числами,

MEd9.Text:=IntToStr(x+1); //которые преобразовываем

MEd4.Text:=IntToStr(x+2); //к строковому представлению

MEd7.Text:=IntToStr(x+3);

MEd5.Text:=IntToStr(x+4);

MEd3.Text:=IntToStr(x+5);

MEd6.Text:=IntToStr(x+6);

MEd1.Text:=IntToStr(x+7);

MEd8.Text:=IntToStr(x+8);

Case y of //Выбираем способ удаления значений некоторых клеток

0:begin

MEd4.Text:=''; MEd4.SetFocus; //Удаляем число из поля

MEd5.Text:=''; MEd5.SetFocus; //и передаем ему фокус ввода

MEd8.Text:=''; MEd8.SetFocus;

MEd9.Text:=''; MEd9.SetFocus;

MEd1.ReadOnly:=true; //Запрещаем ввод чисел и изменение остальных полей

MEd2.ReadOnly:=true;

MEd3.ReadOnly:=true;

MEd6.ReadOnly:=true;

MEd7.ReadOnly:=true;

end;

1:begin

MEd2.Text:=''; MEd2.SetFocus;

MEd5.Text:=''; MEd5.SetFocus;

MEd6.Text:=''; MEd6.SetFocus;

MEd8.Text:=''; MEd8.SetFocus;

MEd1.ReadOnly:=true;

MEd3.ReadOnly:=true;

MEd4.ReadOnly:=true;

MEd7.ReadOnly:=true;

MEd9.ReadOnly:=true;

end;

2:begin

MEd1.Text:=''; MEd1.SetFocus;

MEd2.Text:=''; MEd2.SetFocus;

MEd4.Text:=''; MEd4.SetFocus;

MEd6.Text:=''; MEd6.SetFocus;

MEd9.Text:=''; MEd9.SetFocus;

MEd3.ReadOnly:=true;

MEd5.ReadOnly:=true;

MEd7.ReadOnly:=true;

MEd8.ReadOnly:=true;

end;

3:begin

MEd3.Text:=''; MEd3.SetFocus;

MEd4.Text:=''; MEd4.SetFocus;

MEd6.Text:=''; MEd6.SetFocus;

MEd7.Text:=''; MEd7.SetFocus;

MEd8.Text:=''; MEd8.SetFocus;

MEd1.ReadOnly:=true;

MEd2.ReadOnly:=true;

MEd5.ReadOnly:=true;

MEd9.ReadOnly:=true;

end;

end;

Lb1.Caption:='Заполни пустые клетки магического квадрата';

//Появление соответствующей надписи в метке

Button1.Show; //Показать кнопку "Проверить"

Button2.Hide; //Спрятать кнопку "Начать"

end;

BEGIN

if z then //Проверка правильности вввода недостающих чисел

zapolnenie(x,y) //Заполнение "Попробуй еще раз" старыми значениями х и у

else //Заполняем в первый раз и при каждом правильном решении

begin

y:=random(4); //Выбираем случайным образом способ удаления значений полей

x:=Random(100); //Выбираем число, которым начинаем заполнение полей

zapolnenie(x,y); //заполняем квадрат

end;

end;

procedure TForm1.Button1Click(Sender: TObject); //Обработка нажатия кнопки "Проверить"

var s1,s2,s3,s4,s5,s6,s7,s8:integer; //Суммы в строках, столбцах, диагоналях

begin

try //Пробуем преобразовать в числовое представление значения полей

//и находим соответствующие суммы

s1:=StrToInt(Trim(MEd1.Text))+ StrToInt(Trim(MEd2.Text))+ StrToInt(Trim(MEd3.Text));

s2:=StrToInt(Trim(MEd4.Text))+ StrToInt(Trim(MEd5.Text))+ StrToInt(Trim(MEd6.Text));

s3:=StrToInt(Trim(MEd7.Text))+ StrToInt(Trim(MEd8.Text))+ StrToInt(Trim(MEd9.Text));

s4:=StrToInt(Trim(MEd1.Text))+ StrToInt(Trim(MEd4.Text))+ StrToInt(Trim(MEd7.Text));

s5:=StrToInt(Trim(MEd2.Text))+ StrToInt(Trim(MEd5.Text))+ StrToInt(Trim(MEd8.Text));

s6:=StrToInt(Trim(MEd3.Text))+ StrToInt(Trim(MEd6.Text))+ StrToInt(Trim(MEd9.Text));

s7:=StrToInt(Trim(MEd3.Text))+ StrToInt(Trim(MEd5.Text))+ StrToInt(Trim(MEd7.Text));

s8:=StrToInt(Trim(MEd1.Text))+ StrToInt(Trim(MEd5.Text))+ StrToInt(Trim(MEd9.Text));

If (s1=s2) and (s2=s3) and (s3=s4) and (s4=s5) and (s5=s6) and (s6=s7) and (s7=s8)

then //Если все суммы равны

begin Lb1.Caption:='Молодец! Все верно!' ; //Выводим в метке

//соответствующую надпись

z:=false; //z определяет, что заполнение следующих значений квадрата

//нужно начать с новой генерации начального значения х и способа удаления у

Button2.Show; //Показать кнопку "Начать"

Button1.Hide; //Спрятать кнопку "Проверить"

end

else //Если хоть одна сумма отлична от других

begin

Lb1.Caption:='Попробуй еще раз'; //Выводим в метке соответствующую надпись

z:=true; //z определяет, что заполнение квадрата

//нужно произвести старыми значениями х и у

Button2.Show; //Показать кнопку "Начать"

Button1.Hide; //Спрятать кнопку "Проверить"

end;

Except //Если преобразовать в числовое представление значения полей

// не удалось (какое-то поле осталось не заполнено)

MessageBeep(mb\_Ok); //Звуковой сигнал

ShowMessage('Не все клетки заполнены!'); //Вывод окна сообщения

Exit; //Выход из процедуры

end;

end;

procedure TForm1.N1Click(Sender: TObject); //Выбор меню "Магический квадрат"

begin

OKBottomDlg.Show; //Показ диалогового окна "Магический квадрат"

end;

procedure TForm1.N2Click(Sender: TObject); //Выбор меню "Правила игры"

begin

OKBottomDlg1.Show; //Показ диалогового окна "Правила игры"

end;

end.

## Листинг модуля вызова диалогового окна «Магический квадрат»

unit dialog\_mag;

interface

uses Windows, SysUtils, Classes, Graphics, Forms, Controls, StdCtrls,

Buttons, ExtCtrls, Grids;

type

TOKBottomDlg = class(TForm)

OKBtn: TButton; //Кнопка "ОК"

Bevel1: TBevel; //Кромка

Memo1: TMemo; //Текстовое поле

StringGrid1: TStringGrid; //Текстовая таблица

procedure OKBtnClick(Sender: TObject);

procedure FormPaint(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

OKBottomDlg: TOKBottomDlg;

Bitmap1:TBitmap; //фоновая картинка

StringGrid1:TStringGrid;

implementation

{$R \*.dfm}

procedure TOKBottomDlg.FormPaint(Sender: TObject);//Процедура рисования фона формы

procedure zapoln;

begin

StringGrid1.Cells[0,0]:='8'; //Заполнение таблицы числами

StringGrid1.Cells[0,1]:='1';

StringGrid1.Cells[0,2]:='6';

StringGrid1.Cells[1,0]:='3';

StringGrid1.Cells[1,1]:='5';

StringGrid1.Cells[1,2]:='7';

StringGrid1.Cells[2,0]:='4';

StringGrid1.Cells[2,1]:='9';

StringGrid1.Cells[2,2]:='2';

end;

BEGIN

Bitmap1:=TBitmap.Create; //создание объекта-битового образа

Bitmap1.LoadFromFile('fon.bmp'); //Загрузка битового изображения из файла

Canvas.StretchDraw(Rect(0, 0, OKBottomDlg.clientWidth, OKBottomDlg.clientHeight),Bitmap1);

//Рисование и заполнение формы рисунком Bitmap

zapoln; //Вызов процедуры заполнения таблицы числами

end;

procedure TOKBottomDlg.OKBtnClick(Sender: TObject);

begin

OKBottomDlg.Close; //Закрытие окна по щелчку кнопки "OK"

end;

end.

## Листинг модуля вызова диалогового окна «Правила игры»

unit dialog\_pravila;

interface

uses Windows, SysUtils, Classes, Graphics, Forms, Controls, StdCtrls,

Buttons, ExtCtrls;

type

TOKBottomDlg1 = class(TForm)

OKBtn: TButton; //Кнопка "ОК"

Bevel1: TBevel; //Кромка

Memo1: TMemo; //Текстовое поле

procedure OKBtnClick(Sender: TObject);

procedure FormPaint(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

OKBottomDlg1: TOKBottomDlg1;

Bitmap2:TBitmap; //фоновая картинка

implementation

{$R \*.dfm}

procedure TOKBottomDlg1.FormPaint(Sender: TObject);//Процедура рисования фона формы

begin

Bitmap2:=TBitmap.Create; //создание объекта-битового образа

//Загрузка битового изображения из файла

Bitmap2.LoadFromFile('fon.bmp');

Canvas.StretchDraw(Rect(0, 0, OKBottomDlg1.clientWidth, OKBottomDlg1.clientHeight),Bitmap2);

//Рисование и заполнение формы рисунком Bitmap

end;

procedure TOKBottomDlg1.OKBtnClick(Sender: TObject);

begin

OKBottomDlg1.Close; //Закрытие окна по щелчку кнопки "OK"

end;

end.