unit Unit1;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Menus, Vcl.StdCtrls, Vcl.Buttons,

Vcl.ExtCtrls, Vcl.Imaging.pngimage, Vcl.Imaging.GIFImg, Vcl.Mask, Vcl.MPlayer, MMSystem, ShellAPI;

type

TForm1 = class(TForm)

BitBtn1: TBitBtn;

BitBtn2: TBitBtn;

BitBtn3: TBitBtn;

Image2: TImage;

Image1: TImage;

Image3: TImage;

BitBtn4: TBitBtn;

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

N4: TMenuItem;

N5: TMenuItem;

MediaPlayer1: TMediaPlayer;

N6: TMenuItem;

procedure FormShow(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn1Click(Sender: TObject);

procedure N2Click(Sender: TObject);

procedure N4Click(Sender: TObject);

procedure N5Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure SetVolume(Volume: Integer);

procedure BitBtn4Click(Sender: TObject);

procedure MediaPlayer1Notify(Sender: TObject);

procedure N6Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

implementation

{$R \*.dfm}

uses Unit3, Unit2, Unit4, Unit5;

procedure TForm1.BitBtn1Click(Sender: TObject);

begin

form1.Hide;

form3.Show;

end;

procedure TForm1.BitBtn2Click(Sender: TObject);

begin

Form1.Hide;

Form4.Show;

end;

procedure TForm1.BitBtn3Click(Sender: TObject);

begin

MediaPlayer1.Close;

close;

end;

procedure TForm1.BitBtn4Click(Sender: TObject);

begin

Form1.Hide;

Form5.Show;

end;

procedure TForm1.FormCreate(Sender: TObject);

var fontway, Music1:string;

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\press-start-k.ttf'));

Music1 := ExtractFilePath(Application.ExeName) + 'terrariya-den.mp3';

MediaPlayer1.FileName := Music1;

MediaPlayer1.AutoOpen := True;

MediaPlayer1.Open;

MediaPlayer1.Play;

if MediaPlayer1.Mode = mpStopped then

begin

MediaPlayer1.Rewind;

MediaPlayer1.Play;

end;

end;

procedure TForm1.FormShow(Sender: TObject);

begin

try

form2.ShowModal;

form2.Destroy;

except

end;

end;

procedure TForm1.MediaPlayer1Notify(Sender: TObject);

begin

// Зацикливание музыки

if MediaPlayer1.Mode = mpStopped then

begin

MediaPlayer1.Rewind;

MediaPlayer1.Play;

end;

MediaPlayer1.Notify := True; // Включаем уведомления снова

end;

procedure TForm1.N2Click(Sender: TObject);

begin

close;

end;

procedure TForm1.N4Click(Sender: TObject);

begin

ShowMessage('Автор: Евгений Дашкевич');

end;

procedure TForm1.N5Click(Sender: TObject);

begin

ShowMessage('Версия: 1.0.0');

end;

procedure TForm1.N6Click(Sender: TObject);

begin

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

end;

procedure TForm1.SetVolume(Volume: Integer);

var

ConvertedVolume: DWORD;

begin

if (Volume < Form4.ScrollBar1.Min) or (Volume > Form4.ScrollBar1.Max) then

Exit;

// Конвертация значения громкости в диапазон для waveOutSetVolume

ConvertedVolume := Round((Volume / Form4.ScrollBar1.Max) \* $FFFF);

// Установка громкости для обоих каналов (левого и правого)

ConvertedVolume := (ConvertedVolume shl 16) or ConvertedVolume;

waveOutSetVolume(0, ConvertedVolume);

end;

end.

unit Unit2;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.ExtCtrls, Vcl.ComCtrls,

Vcl.Imaging.pngimage, Vcl.Menus, Vcl.MPlayer;

type

TForm2 = class(TForm)

Timer1: TTimer;

ProgressBar1: TProgressBar;

Image1: TImage;

Image2: TImage;

procedure Timer1Timer(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form2: TForm2;

implementation

{$R \*.dfm}

uses Unit1;

procedure TForm2.Timer1Timer(Sender: TObject);

begin

if ProgressBar1.Position < ProgressBar1.Max then

ProgressBar1.Position := ProgressBar1.Position + 45

else

begin

Form2.close;

Form1.Show;

timer1.Destroy;

end;

end;

end.

unit Unit3;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Grids, Vcl.StdCtrls, Vcl.ExtCtrls,

Vcl.Menus, Vcl.Buttons, Vcl.Imaging.pngimage, Vcl.Mask, Vcl.CheckLst, ShellAPI;

type

TDirection = (dirUp, dirDown, dirLeft, dirRight);

TForm3 = class(TForm)

DrawGrid1: TDrawGrid;

Image1: TImage;

BitBtn3: TBitBtn;

Image2: TImage;

Timer2: TTimer;

Label1: TLabel;

Image3: TImage;

BitBtn1: TBitBtn;

MainMenu1: TMainMenu;

N1: TMenuItem;

N6: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

N4: TMenuItem;

N5: TMenuItem;

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure DrawGrid1DrawCell(Sender: TObject; ACol, ARow: Integer;

Rect: TRect; State: TGridDrawState);

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

procedure Timer2Timer(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure N2Click(Sender: TObject);

procedure N5Click(Sender: TObject);

procedure CreateParams(var Params: TCreateParams) ; override;

procedure N4Click(Sender: TObject);

procedure N6Click(Sender: TObject);

private

{ Private declarations }

SnakeSpeed: Integer;

SnakeColor: Integer;

SnakeBorder: Integer;

Obstacles: array of TRect;

Snake: array of TPoint;

Food: TPoint;

Score: Integer;

Direction: TDirection;

GameInitialized: Boolean;

FSeconds: Integer;

FMinutes: Integer;

procedure MoveSnake;

procedure GenerateFood;

procedure CheckCollision;

procedure CheckFood;

procedure InitializeGame;

procedure UpdateScore;

procedure GenerateObstacles;

procedure SaveRecord(const Score: Integer);

public

{ Public declarations }

end;

var

Form3: TForm3;

implementation

{$R \*.dfm}

uses Unit1, Unit4;

procedure TForm3.BitBtn1Click(Sender: TObject);

begin

Timer2.Enabled := False;

Form3.Hide;

Form1.Show;

end;

procedure TForm3.BitBtn2Click(Sender: TObject);

begin

Form1.MediaPlayer1.Close;

Timer2.Enabled := False;

Close;

end;

procedure TForm3.BitBtn3Click(Sender: TObject);

var I: Integer;

begin

InitializeGame;

GenerateObstacles;

for I := High(Snake) downto 1 do

Snake[I] := Snake[I - 1];

case Direction of

dirUp: Dec(Snake[0].Y);

dirDown: Inc(Snake[0].Y);

dirLeft: Dec(Snake[0].X);

dirRight: Inc(Snake[0].X);

end;

end;

procedure TForm3.Timer2Timer(Sender: TObject);

begin

MoveSnake;

if GameInitialized then

begin

CheckCollision;

CheckFood;

end;

DrawGrid1.Repaint;

end;

procedure TForm3.GenerateObstacles; //Генерация препятствий

var

I, J: Integer;

Obstacle: TRect;

Collision: Boolean;

begin

Randomize;

SetLength(Obstacles, SnakeBorder); // Например, 5 препятствий

for I := 0 to High(Obstacles) do

begin

repeat

Collision := False;

Obstacle.Left := Random(DrawGrid1.ColCount - 1);

Obstacle.Top := Random(DrawGrid1.RowCount - 1);

Obstacle.Right := Obstacle.Left + 1;

Obstacle.Bottom := Obstacle.Top + 1;

// Проверка на совпадение с координатами змейки или еды

for J := 0 to High(Snake) do

if PtInRect(Obstacle, Snake[J]) then

begin

Collision := True;

Break;

end;

if PtInRect(Obstacle, Food) then

Collision := True;

until not Collision;

Obstacles[I] := Obstacle;

end;

end;

procedure TForm3.DrawGrid1DrawCell(Sender: TObject; ACol, ARow: Integer;

Rect: TRect; State: TGridDrawState);

var

I: Integer;

begin

with DrawGrid1.Canvas do

begin

// Отрисовка змейки

for I := 0 to High(Snake) do

begin

if (ACol = Snake[I].X) and (ARow = Snake[I].Y) then

begin

Brush.Color := SnakeColor;

FillRect(Rect);

Exit;

end;

end;

// Отрисовка еды

if (ACol = Food.X) and (ARow = Food.Y) then

begin

Brush.Color := clRed;

FillRect(Rect);

end

else

begin

Brush.Color := clWhite;

FillRect(Rect);

end;

// Отрисовка препятствий

for I := 0 to High(Obstacles) do

begin

if PtInRect(Obstacles[I], Point(ACol, ARow)) then

begin

Brush.Color := clGray;

FillRect(Rect);

end;

end;

end;

end;

procedure TForm3.InitializeGame;

begin

SetLength(Snake, 3);

Snake[0] := Point(14, 14); // Начальная позиция змейки

Direction := dirRight; // Начальное направление

GenerateFood;

GenerateObstacles; // Генерация препятствий

Timer2.Enabled := True;

Score := 0;

UpdateScore;

GameInitialized := True; // Игра инициализирована

if Form4.RadioButton1.Checked = True then SnakeSpeed := 400

else

if Form4.RadioButton2.Checked = True then SnakeSpeed := 200

else

if Form4.RadioButton3.Checked = True then SnakeSpeed := 100;

SnakeSpeed := Timer2.Interval;

if Form4.RadioButton4.Checked = True then SnakeColor := clGreen

else

if Form4.RadioButton5.Checked = True then SnakeColor := clLime

else

if Form4.RadioButton6.Checked = True then SnakeColor := clYellow;

if Form4.RadioButton7.Checked = True then SnakeBorder := 0

else

if Form4.RadioButton8.Checked = True then SnakeBorder := 15

else

if Form4.RadioButton9.Checked = True then SnakeBorder := 30;

end;

procedure TForm3.FormCreate(Sender: TObject);

begin

DrawGrid1.ColCount := 28;

DrawGrid1.RowCount := 28;

SetLength(Snake, 3);

Snake[0] := Point(14, 14); // Начальная позиция змейки

Direction := dirRight; // Начальное направление

GenerateFood;

GenerateObstacles; // Генерация препятствий

SnakeSpeed := 200;

Timer2.Interval := 200; // Интервал таймера (скорость змейки)

Timer2.Enabled := True;

KeyPreview := True; // Для обработки нажатий клавиш

GameInitialized := False; // Игра еще не инициализирована

end;

procedure TForm3.FormKeyDown(Sender: TObject; var Key: Word; Shift: TShiftState);

begin

case Key of

VK\_UP, ord('W'): if Direction <> dirDown then Direction := dirUp;

VK\_DOWN, ord('S'): if Direction <> dirUp then Direction := dirDown;

VK\_LEFT, ord('A'): if Direction <> dirRight then Direction := dirLeft;

VK\_RIGHT, ord('D'): if Direction <> dirLeft then Direction := dirRight;

end;

end;

procedure TForm3.MoveSnake;

var

I: Integer;

begin

for I := High(Snake) downto 1 do

Snake[I] := Snake[I - 1];

case Direction of

dirUp: Dec(Snake[0].Y);

dirDown: Inc(Snake[0].Y);

dirLeft: Dec(Snake[0].X);

dirRight: Inc(Snake[0].X);

end;

end;

procedure TForm3.N2Click(Sender: TObject);

begin

close;

end;

procedure TForm3.N4Click(Sender: TObject);

begin

ShowMessage('Автор: Дашкевич Евгений');

end;

procedure TForm3.N5Click(Sender: TObject);

begin

ShowMessage('Версия: 1.0.0');

end;

procedure TForm3.N6Click(Sender: TObject);

begin

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

end;

procedure TForm3.CheckCollision;

var

I: Integer;

begin

// Проверка столкновения с границами

if (Snake[0].X < 0) or (Snake[0].X >= DrawGrid1.ColCount) or

(Snake[0].Y < 0) or (Snake[0].Y >= DrawGrid1.RowCount) then

begin

Timer2.Enabled := False;

ShowMessage('Игра закончена! Очки: ' + IntToStr(Score));

SaveRecord(Score); // сохраняем рекорд при завершении игры

Exit;

end;

// Проверка столкновения с самим собой

for I := 1 to High(Snake) do

if (Snake[0].X = Snake[I].X) and (Snake[0].Y = Snake[I].Y) then

begin

Timer2.Enabled := False;

ShowMessage('Вы проиграли! Очки: ' + IntToStr(Score));

SaveRecord(Score); // сохраняем рекорд при завершении игры

Exit;

end;

// Проверка столкновения с препятствиями

for I := 0 to High(Obstacles) do

if PtInRect(Obstacles[I], Snake[0]) then

begin

Timer2.Enabled := False;

ShowMessage('Вы врезались в препятствие! Очки: ' + IntToStr(Score));

SaveRecord(Score); // сохраняем рекорд при завершении игры

Exit;

end;

end;

procedure TForm3.UpdateScore;

begin

Label1.Caption := IntToStr(Score);

end;

procedure TForm3.CheckFood;

begin

if (Snake[0].X = Food.X) and (Snake[0].Y = Food.Y) then

begin

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

Snake[High(Snake)] := Snake[High(Snake) - 1];

GenerateFood;

Inc(Score);

UpdateScore;

SnakeSpeed := Round(SnakeSpeed - 2);

Timer2.Interval := SnakeSpeed;

end;

end;

procedure TForm3.SaveRecord(const Score: Integer);

var

FileName: string;

FileHandle: TextFile;

begin

FileName := ExtractFilePath(Application.ExeName) + 'snake\_record.txt'; // имя файла, в который будем сохранять рекорд

AssignFile(FileHandle, FileName);

try

// Проверяем, существует ли файл

if FileExists(FileName) then

Append(FileHandle) // Открываем файл для добавления данных

else

Rewrite(FileHandle); // Создаем новый файл

// Записываем рекорд в файл

Writeln(FileHandle, Form4.Edit1.Text + '-' + IntToStr(Score));

except

on E: EInOutError do

begin

ShowMessage('Произошла ошибка при работе с файлом. Подробности: ' + E.Message);

Exit; // Выходим из процедуры, чтобы избежать попытки закрытия неоткрытого файла

end;

end;

// Закрываем файл в блоке finally

try

CloseFile(FileHandle);

except

on E: EInOutError do

begin

ShowMessage('Ошибка при закрытии файла. Подробности: ' + E.Message);

end;

end;

end;

procedure TForm3.GenerateFood;

var

FoodCollision: Boolean;

I, J: Integer;

begin

Randomize;

repeat

FoodCollision := False;

Food.X := Random(DrawGrid1.ColCount);

Food.Y := Random(DrawGrid1.RowCount);

// Проверка на совпадение с координатами змейки

for I := 0 to High(Snake) do

begin

if (Food.X = Snake[I].X) and (Food.Y = Snake[I].Y) then

begin

FoodCollision := True;

Break;

end;

end;

// Проверка на совпадение с координатами препятствий

if not FoodCollision then

begin

for I := 0 to High(Obstacles) do

begin

for J := Obstacles[I].Left to Obstacles[I].Right do

if (Food.X = J) and ((Food.Y = Obstacles[I].Top) or (Food.Y = Obstacles[I].Bottom)) then

begin

FoodCollision := True;

Break;

end;

if FoodCollision then

Break;

end;

end;

until not FoodCollision;

end;

procedure TForm3.CreateParams(var Params: TCreateParams) ;// процедура для того чтобы ярлык в панели задач не пропадал

begin

inherited;

Params.ExStyle := Params.ExStyle or WS\_EX\_APPWINDOW;

Params.WndParent := 0;

end;

procedure TForm3.FormShow(Sender: TObject);

begin

DrawGrid1.Repaint;

end;

end.

unit Unit4;

interface

uses

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

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

Vcl.Menus, Vcl.StdCtrls, Vcl.Buttons, MMSystem, ShellAPI;

type

TForm4 = class(TForm)

Image1: TImage;

BitBtn1: TBitBtn;

ScrollBar1: TScrollBar;

CheckBox1: TCheckBox;

Image5: TImage;

RadioButton1: TRadioButton;

RadioButton2: TRadioButton;

RadioButton3: TRadioButton;

Panel1: TPanel;

RadioButton4: TRadioButton;

RadioButton5: TRadioButton;

RadioButton6: TRadioButton;

Edit1: TEdit;

Panel2: TPanel;

Panel3: TPanel;

RadioButton7: TRadioButton;

RadioButton8: TRadioButton;

RadioButton9: TRadioButton;

MainMenu1: TMainMenu;

N1: TMenuItem;

N6: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

N4: TMenuItem;

N5: TMenuItem;

procedure N4Click(Sender: TObject);

procedure N5Click(Sender: TObject);

procedure N2Click(Sender: TObject);

procedure BitBtn1Click(Sender: TObject);

procedure CheckBox1Click(Sender: TObject);

procedure ScrollBar1Change(Sender: TObject);

procedure CreateParams(var Params: TCreateParams) ; override;

procedure N6Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form4: TForm4;

implementation

{$R \*.dfm}

uses Unit1;

procedure TForm4.BitBtn1Click(Sender: TObject);

begin

Form4.Hide;

Form1.Show;

end;

procedure TForm4.CheckBox1Click(Sender: TObject);

begin

if CheckBox1.Checked = True then Form1.MediaPlayer1.Pause

else

if CheckBox1.Checked = False then Form1.MediaPlayer1.Play;

end;

procedure TForm4.N2Click(Sender: TObject);

begin

close;

end;

procedure TForm4.N4Click(Sender: TObject);

begin

ShowMessage('Автор: Дашкевич Евгений');

end;

procedure TForm4.N5Click(Sender: TObject);

begin

ShowMessage('Версия: 1.0.0 ');

end;

procedure TForm4.N6Click(Sender: TObject);

begin

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

end;

procedure TForm4.ScrollBar1Change(Sender: TObject);

begin

Form1.SetVolume(ScrollBar1.Position);

end;

procedure TForm4.CreateParams(var Params: TCreateParams) ;// процедура для того чтобы ярлык в панели задач не пропадал

begin

inherited;

Params.ExStyle := Params.ExStyle or WS\_EX\_APPWINDOW;

Params.WndParent := 0;

end;

end.

unit Unit5;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Menus, Vcl.Imaging.pngimage,

Vcl.ExtCtrls, Vcl.StdCtrls, Vcl.Buttons, ShellAPI;

type

TForm5 = class(TForm)

Image1: TImage;

Memo1: TMemo;

BitBtn1: TBitBtn;

Image2: TImage;

MainMenu1: TMainMenu;

N1: TMenuItem;

N6: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

N4: TMenuItem;

N5: TMenuItem;

procedure FormShow(Sender: TObject);

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure N2Click(Sender: TObject);

procedure N4Click(Sender: TObject);

procedure N5Click(Sender: TObject);

procedure N6Click(Sender: TObject);

private

procedure LoadRecords;

procedure SortRecords;

public

{ Public declarations }

end;

var

Form5: TForm5;

implementation

Uses Unit1;

{$R \*.dfm}

procedure TForm5.BitBtn1Click(Sender: TObject);

var

FileName: string;

FileHandle: TextFile;

begin

FileName := ExtractFilePath(Application.ExeName) + 'snake\_record.txt'; // имя файла, который будем очищать

AssignFile(FileHandle, FileName);

try

try

Rewrite(FileHandle); // Открываем файл в режиме записи, чтобы очистить его

except

on E: EInOutError do

begin

ShowMessage('Произошла ошибка при открытии файла для записи. Подробности: ' + E.Message);

Exit; // Выходим из процедуры, чтобы избежать попытки закрытия неоткрытого файла

end;

end;

finally

try

CloseFile(FileHandle);

except

on E: EInOutError do

begin

ShowMessage('Произошла ошибка при закрытии файла. Подробности: ' + E.Message);

end;

end;

end;

Memo1.Clear; // Очищаем Memo1

end;

procedure TForm5.BitBtn2Click(Sender: TObject);

begin

Form5.Hide;

Form1.Show;

end;

procedure TForm5.FormShow(Sender: TObject);

begin

LoadRecords; // Загружаем и сортируем записи при показе формы

end;

procedure TForm5.LoadRecords;

var

FileName: string;

FileHandle: TextFile;

Line: string;

begin

FileName := ExtractFilePath(Application.ExeName) + 'snake\_record.txt';

if FileExists(FileName) then

begin

Memo1.Clear;

AssignFile(FileHandle, FileName);

try

Reset(FileHandle); // Открываем файл для чтения

try

while not EOF(FileHandle) do

begin

ReadLn(FileHandle, Line);

Memo1.Lines.Add(Line);

end;

finally

CloseFile(FileHandle); // Закрываем файл в любом случае

end;

except

on E: EInOutError do

ShowMessage('Произошла ошибка при чтении файла. Подробности: ' + E.Message);

end;

SortRecords; // Сортируем записи после загрузки

end

else

begin

ShowMessage('Файл не найден: ' + FileName);

end;

end;

procedure TForm5.N2Click(Sender: TObject);

begin

Close;

end;

procedure TForm5.N4Click(Sender: TObject);

begin

ShowMessage('Автор: Дашкевич Евгений');

end;

procedure TForm5.N5Click(Sender: TObject);

begin

ShowMessage('Версия: 1.0.0');

end;

procedure TForm5.N6Click(Sender: TObject);

begin

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

end;

procedure TForm5.SortRecords;

var

i, j: Integer;

Temp: string;

Score1, Score2: Integer;

begin

// Простая сортировка пузырьком для примера; замените на более эффективный алгоритм, если нужно

for i := 0 to Memo1.Lines.Count - 2 do

begin

for j := Memo1.Lines.Count - 2 downto i do

begin

// Получаем числовые значения результатов для сравнения

Score1 := StrToIntDef(Trim(Copy(Memo1.Lines[j], Pos('-', Memo1.Lines[j]) + 1, MaxInt)), 0);

Score2 := StrToIntDef(Trim(Copy(Memo1.Lines[j+1], Pos('-', Memo1.Lines[j+1]) + 1, MaxInt)), 0);

// Сортируем по убыванию

if Score1 < Score2 then

begin

Temp := Memo1.Lines[j];

Memo1.Lines[j] := Memo1.Lines[j+1];

Memo1.Lines[j+1] := Temp;

end;

end;

end;

end;

end.