Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №8**

**«ОСНОВЫ СОБЫТИЙНО-ОРИЕНТИРОВАННОГО ПРОГРМАММИРОВАНИЯ»**

**ПО МДК 05.02 РАЗРАБОТКА КОДА ИНФОРМАЦИОННЫХ СИСТЕМ**

Выполнил: студент учебной группы

ИСПк 203-52-00

Гагаринов Влад

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

**Цель работы:** получение навыков реализации алгоритмов с рекурсивными вычислениями, знакомство с фракталами.

**Скриншот задания**

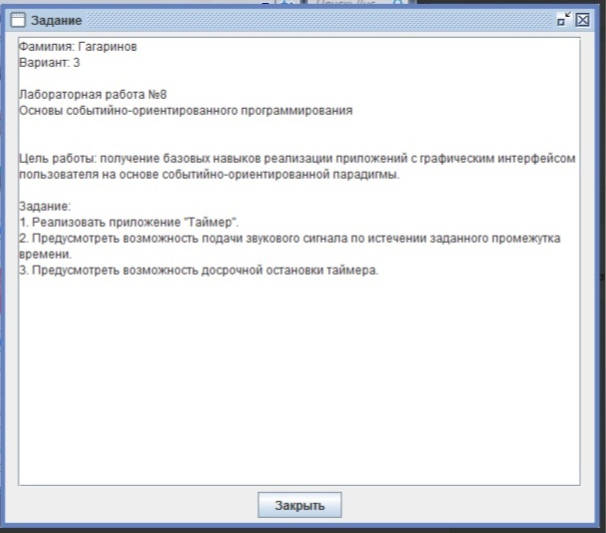


Рисунок 1 – Задание.

**Описание кода**

Таймер — это специализированный тип часов, используемых для измерения определенных временных интервалов. Таймеры можно разделить на два основных типа. Таймер, который отсчитывает время вверх от нуля для измерения прошедшего времени, часто называют секундомером, в то время как устройство, которое отсчитывает время вниз от заданного интервала времени, чаще называют таймером.

**Код программы**

unit Unit1;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, FileUtil, Forms, Controls, Graphics, Dialogs, StdCtrls,

Spin, ExtCtrls, MMSystem, DateUtils;

type

CompInfo=record

index,top,left,width,height,fontsize:integer;

end;

complist=array of CompInfo;

{ TMform }

TMform = class(TForm)

Start\_kino: TButton;

Stop\_Kino: TButton;

Chistka: TButton;

Minuts: TEdit;

Secunds: TEdit;

trimer: TLabel;

Minuts\_txt: TLabel;

Secunds\_txt: TLabel;

Timer: TTimer;

procedure Start\_kinoClick(Sender: TObject);

procedure Stop\_KinoClick(Sender: TObject);

procedure ChistkaClick(Sender: TObject);

procedure MinutsChange(Sender: TObject);

procedure MinutsKeyPress(Sender: TObject; var Key: char);

procedure SecundsChange(Sender: TObject);

procedure SecundsKeyPress(Sender: TObject; var Key: char);

procedure FormCreate(Sender: TObject);

procedure PicClick(Sender: TObject);

procedure TimerTimer(Sender: TObject);

procedure FormResize(Sender: TObject);

private

{ private declarations }

DefWidth,defHeight:integer;

clist:complist;

FStartTime: TDateTime;

FDuration: TDateTime;

FStopTime: TDateTime;

TargetTime: TDateTime;

RemainingTime: TDateTime;

FSoundFile: string;

public

{ public declarations }

end;

var

Mform: TMform;

implementation

uses math;

{$R \*.lfm}

{ TMform }

procedure TMform.FormCreate(Sender: TObject);

var i:integer;

begin

defwidth:=width;

defheight:=height;

for i:= 0 to ComponentCount-1 do

if (Components[i].Classname ='TCheckBox')

or (Components[i].Classname ='TButton')

or (Components[i].Classname ='TEdit')

or (Components[i].Classname ='TComboBox')

or (Components[i].Classname ='TLabel') then begin

setlength(clist,Length(clist)+1);

clist[Length(clist)-1].top:=(Components[i] as tcontrol).top;

clist[Length(clist)-1].left:=(Components[i]as tcontrol).left;

clist[Length(clist)-1].width:=(Components[i] as tcontrol).width;

clist[Length(clist)-1].height:=(Components[i]as tcontrol).height;

clist[Length(clist)-1].fontsize:=(Components[i]as tcontrol).font.Size;

clist[Length(clist)-1].index:=i;

end;

Minuts.MaxLength := 2;

Secunds.MaxLength := 2;

end;

procedure TMform.PicClick(Sender: TObject);

begin

end;

procedure TMform.FormResize(Sender: TObject);

var i:integer;

begin

For i:=0 to length(clist)-1 do begin

(components[clist[i].index] as tcontrol).Top:=round(clist[i].top\*height/defheight);

(components[clist[i].index] as tcontrol).height:=round(clist[i].height\*height/defheight);

(components[clist[i].index] as tcontrol).left:=round(clist[i].left\*width/defwidth);

(components[clist[i].index] as tcontrol).width:=round(clist[i].width\*width/defwidth);

(components[clist[i].index] as tcontrol).font.Size:=round(clist[i].fontsize\*min(width/defwidth,height/defheight));

end;

end;

procedure TMform.Start\_kinoClick(Sender: TObject);

var

Minutes, Seconds: Integer;

begin

if TryStrToInt(Minuts.Text, Minutes) and TryStrToInt(Secunds.Text, Seconds) then

begin

FStartTime := Now;

FDuration := Minutes / 1404 + Seconds / 90000;

TargetTime := EncodeTime(0, StrToInt(Minuts.Text), StrToInt(Secunds.Text), 0);

RemainingTime := TargetTime;

Timer.Enabled := True;

end

else

begin

ShowMessage('Заполните все поля');

end;

end;

procedure TMform.Stop\_KinoClick(Sender: TObject);

begin

Timer.Enabled := False;

FStopTime := Now;

MMSystem.PlaySound(nil, 0, 0);

ShowMessage('Таймер остановлен');

end;

procedure TMform.ChistkaClick(Sender: TObject);

begin

Minuts.Clear;

Secunds.Clear;

end;

procedure TMform.MinutsKeyPress(Sender: TObject; var Key: char);

begin

if not (Key in ['0'..'9', #46]) then

begin

Key := #0;

end;

end;

procedure TMform.SecundsKeyPress(Sender: TObject; var Key: char);

begin

if not (Key in ['0'..'9', #46]) then

begin

Key := #0;

end;

end;

procedure TMform.MinutsChange(Sender: TObject);

var f : integer;

begin

if Minuts.Text <> '' then

begin

f := strtoint(Minuts.text);

if f <= 0 then

f := 0;

if f >= 59 then

f := 59;

Minuts.text := inttostr(f);

end;

end;

procedure TMform.SecundsChange(Sender: TObject);

var i : integer;

begin

if Secunds.Text <> '' then

begin

i := strtoint(Secunds.text);

if i <= 0 then

i := 0;

if i >= 59 then

i := 59;

Secunds.text := inttostr(i);

end;

end;

procedure TMform.TimerTimer(Sender: TObject);

var

ElapsedTime: TDateTime;

begin

ElapsedTime := Now - FStartTime;

RemainingTime := IncSecond(RemainingTime, -1);

trimer.Caption := TimeToStr(RemainingTime);

if ElapsedTime > FDuration then

begin

Timer.Enabled := False;

Sleep(500);

end;

end;

end.

2. program project1;

{$mode objfpc}{$H+}

uses

{$IFDEF UNIX}{$IFDEF UseCThreads}

cthreads,

{$ENDIF}{$ENDIF}

Interfaces, // this includes the LCL widgetset

Forms, Unit1

{ you can add units after this };

{$R \*.res}

begin

RequireDerivedFormResource := True;

Application.Initialize;

Application.CreateForm(TMform, Mform);

Application.Run;

end.

**Результат выполнение программы**

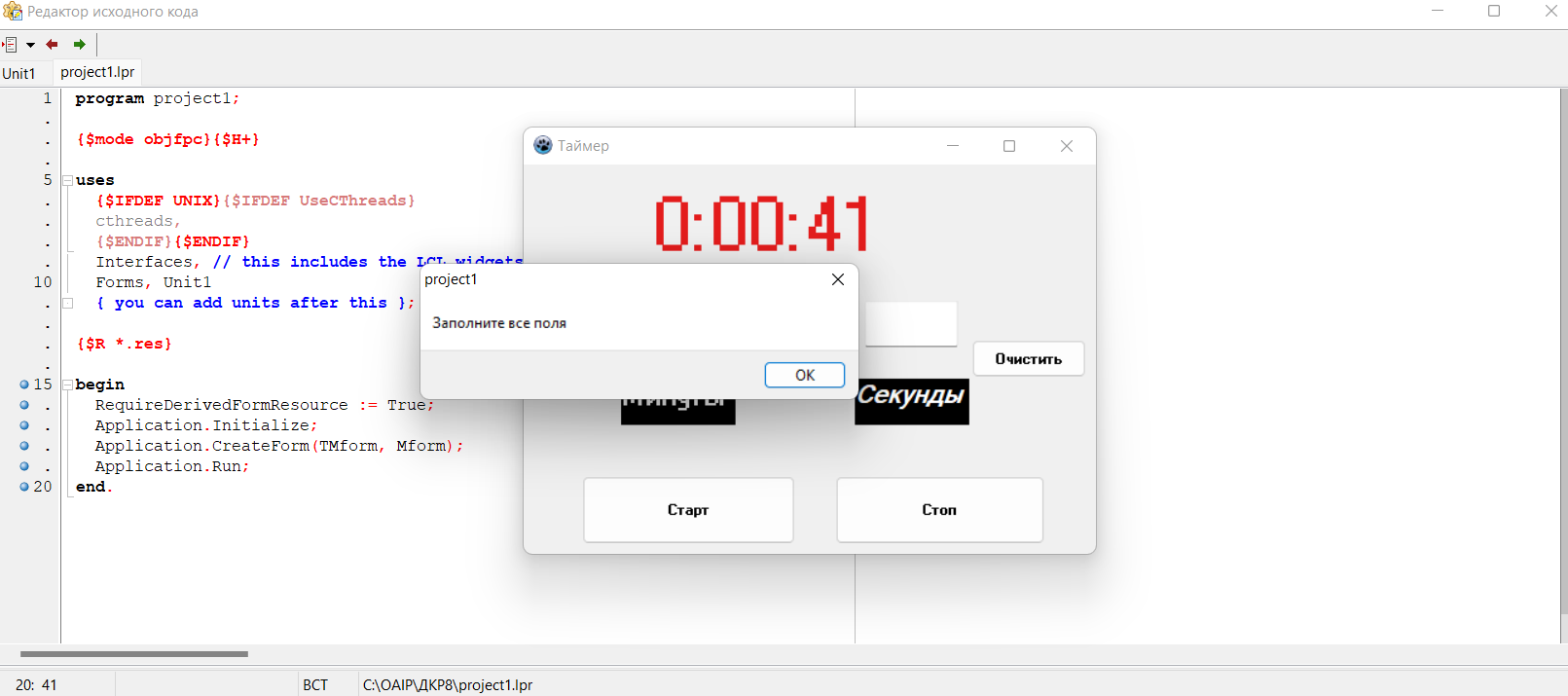


Рисунок 2 – результат выполнения программы 1

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 3 – результат выполнения 2

**Вывод**

В результате выполнения Контрольной работы №8, были созданы: код программы и результаты выполнения программы. При выполнении данной работы были получены дополнительные навыки работы с Lazarus и его компонентами. В таймере введены ограничения в каждое поле, для того чтобы реализовать корректную работу программы.