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 программирования»**

**ПО «МДК 05.02 Разработка кода информационных систем»**

Выполнил: студент учебной группы

ИСПк-203-52-00

Широков Михаил Александрович

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2025

**1. Цель работы**

Получение базовых навыков реализации приложений с графическим интерфейсом пользователя на основе событийно-ориентированной парадигмы.

**2. Формулировка задания**

**Вариант 22:**

**1.** Реализовать приложение «Калькулятор площади фигур».

**2.** Предусмотреть выбор из фиксированного набора заданных фигур (не менее четырех).

**3. Описание алгоритма**

Разработанная программа состоит из 5 основных модулей, описывающих работу 5 форм: форма главного меню, форма круга, форма параллелограмма, форма прямоугольника, форма трапеции.

**3.1 Модуль для главного меню**

**Название модуля:** main\_manu.pas;

**Функция:** предоставление пользователю возможности выбора фигуры для расчета площади;

**Объекты формы:**

– TMenu\_form – объект формы главного меню;

– Select\_button – объект – кнопка, для перехода в форму для расчета площади выбранной фигуры;

– Figure\_box – объект – поле со списком, для выбора фигуры, площадь которой необходимо рассчитать;

– Head\_label – надпись для заголовка формы;

– information\_label – надпись, для пояснения действий для пользователя;

– Menu\_bg\_image – изображение заднего фона формы.

**Методы формы:**

– Select\_buttonClick – метод для кнопки Select\_button, осуществляющий переход на необходимую форму для расчета площади выбранной фигуры.

**Описание методов:**

**1.** Select\_buttonClick

Принимает параметры по значению: Sender: TObject – объект нажатой кнопки.

**Алгоритм работы:**

**a)** Метод начинает работу (нажата кнопка Select\_button)

**b)** Проверяется значение поля со списком Figure\_box. Если оно ровно «Круг», то запускается форма для расчета площади круга. Если оно ровно «Прямоугольник», то запускается форма для расчета площади прямоугольника, Если оно ровно «Параллелограмм». то запускается форма для расчета площади параллелограмма. Если оно ровно «Трапеция», то запускается форма для расчета площади трапеции.

**c)** Метод завершает работу.

**3.2 Модуль для расчета площади круга**

**Название модуля:** curcle\_module.pas;

**Функция:** предоставление пользователю возможности расчета площади круга;

**Объекты формы:**

– TCircle\_form – объект формы;

– R\_edL – надпись, для обозначения поля для ввода радиуса круга;

– Rezult\_button – кнопка для обработки данных и отображения результата;

R\_edit – поле, для ввода радиуса круга;

– S\_edit – поля для вывода площади круга;

– form\_image – изображение заднего фона формы;

– Head: TLabel – надпись заголовка формы;

– information\_label – надпись, содержащая информацию о круге;

– S\_edL1 – надпись для обозначения поля для вывода результата расчет площади круга;

**Методы формы:**

–Rezult\_buttonClick– метод для кнопки Rezult\_button, осуществляющий расчет площади круга по введенному радиусу и ее вывод в поле для вывода площади.

**Описание методов:**

**1.** Rezult\_buttonClick

Принимает параметры по значению: Sender: TObject – объект нажатой кнопки.

**Алгоритм работы:**

**a)** Метод начинает работу (нажата кнопка Rezult\_buttonClick)

**b)** Проверяется значение поля R-edit, если оно пусто, то пользователю будет выведено сообщение об этом;

**c)** Если значение этого поля не пусто, то производится расчет площади круга по формуле ПR2 и вывод в поле S-edit.

**d)** Метод завершает работу.

**3.3 Модуль для расчета площади прямоугольника**

**Название модуля:** Rectangel\_module.pas;

**Функция:** предоставление пользователю возможности расчета площади прямоугольника;

**Объекты формы:**

TRectangle\_form – объект формы;

bg\_image – изображение заднего фона формы;

Result\_button – кнопка для расчета площади прямоугольника;

S\_edit – поля для вывода площади прямоугольника;

B\_edit – поля для ввода длинны стороны прямоугольника;

A\_edit – поля для ввода длинны стороны прямоугольника;

Head – надпись для заголовка формы;

information\_label – надпись, содержащая информацию о прямоугольнике;

S\_L – надпись для обозначения поля вывода площади;

A\_L – надпись для обозначения поля вводы стороны А;

B\_L – напись для обозначения поля вводы сторонв B.

**Методы формы:**

–Rezult\_buttonClick– метод для кнопки Rezult\_button, осуществляющий расчет площади прямоугольника по введенным сторонам и ее вывод в поле для вывода площади.

**Описание методов:**

**1.** Rezult\_buttonClick

Принимает параметры по значению: Sender: TObject – объект нажатой кнопки.

**Алгоритм работы:**

**a)** Метод начинает работу (нажата кнопка Rezult\_buttonClick)

**b)** Проверяется значение поля A-edit, если оно пусто, то пользователю будет выведено сообщение об этом;

**c)** Проверяется значение поля B-edit, если оно пусто, то пользователю будет выведено сообщение об этом;

**d)** Если значение этого полей A-edit и B-edit не пусто, то производится расчет площади прямоугольника по формуле A\*B и вывод в поле S-edit.

**e)** Метод завершает работу.

**3.4 Модуль для расчета площади параллелограмма**

**Название модуля:** paralelogramm\_module.pas;

**Функция:** предоставление пользователю возможности расчета площади параллелограмма;

**Объекты формы:**

TPar\_form – объект формы;

bg\_image – объект изображения заднего фона формы;

Rezult\_button – кнопка для расчета и вывода площади параллелограмма в поле S\_edit;

S\_edit – поле для вывода площади параллелограмма;

H\_edit – поле для ввода высоты параллелограмма;

A\_edit – поле для ввода длинны основания параллелограмма;

Head – надпись заголовка формы;

information\_label – надпись для отображения информации о параллелограмме;

S\_L – надпись для обозначения поля для вывода площади;

H\_L – надпись для обозначения поля для ввода высоты;

A\_L – надпись для обозначения поля для ввода длинны основания;

**Методы формы:**

–Rezult\_buttonClick– метод для кнопки Rezult\_button, осуществляющий расчет площади параллелограмма по введенному основанию и высоте и ее вывод в поле для вывода площади.

**Описание методов:**

**1.** Rezult\_buttonClick

Принимает параметры по значению: Sender: TObject – объект нажатой кнопки.

**Алгоритм работы:**

**a)** Метод начинает работу (нажата кнопка Rezult\_buttonClick)

**b)** Проверяется значение поля A-edit, если оно пусто, то пользователю будет выведено сообщение об этом;

**c)** Проверяется значение поля H-edit, если оно пусто, то пользователю будет выведено сообщение об этом;

**d)** Если значение этого полей A-edit и H-edit не пусто, то производится расчет площади параллелограмма по формуле A\*H и вывод в поле S-edit.

**e)** Метод завершает работу.

**3.5 Модуль для расчета площади трапеции**

**Название модуля:** trapec\_module.pas;

**Функция:** предоставление пользователю возможности расчета площади трапеции;

**Объекты формы:**

Ttrap\_form – объект формы;

bg\_image – изображение заднего фона формы;

Rezult\_button – кнопка для расчета площади трапеции и вывода результата в S\_edit;

S\_Edit – поля для вывода площади трапеции;

H\_edit – поле для ввода высоты трапеции;

O1\_edit – поле для ввода первого основания трапеции;

O2\_Edit – поле для ввода второго основания трапеции;

Head – надпись заголовка формы;

information\_label – надпись, содержащая информацию о трапеции;

S\_L – надпись для обозначения поля вывода площади;

H\_L – надпись для обозначения поля ввода высоты;

O1\_L – надпись для обозначения поля ввода первого основания;

O2\_L – надпись для обозначения поля ввода второго основания;

**Методы формы:**

–Rezult\_buttonClick– метод для кнопки Rezult\_button, осуществляющий расчет площади трапеции по введенному основанию и высоте и ее вывод в поле для вывода площади.

**Описание методов:**

**1.** Rezult\_buttonClick

Принимает параметры по значению: Sender: TObject – объект нажатой кнопки.

**Алгоритм работы:**

**a)** Метод начинает работу (нажата кнопка Rezult\_buttonClick)

**b)** Проверяется значение поля O1-edit, если оно пусто, то пользователю будет выведено сообщение об этом;

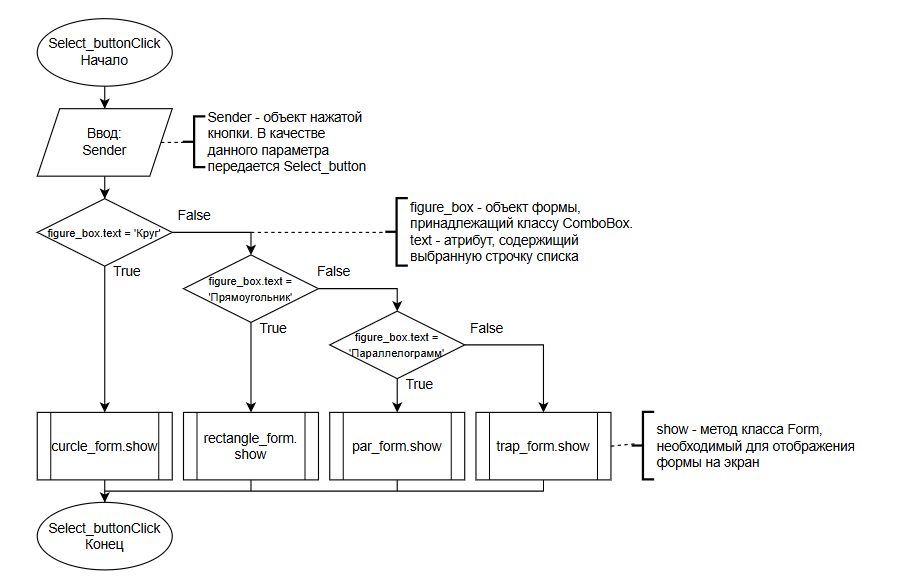
**c)** Проверяется значение поля H-edit, если оно пусто, то пользователю будет выведено сообщение об этом;

**d)** Проверяется значение поля O2-edit, если оно пусто, то пользователю будет выведено сообщение об этом;

**e)** Если значение этого полей O1-edit, O2-edit и H-edit не пусто, то производится расчет площади трапеции по формуле (O1 + O2) / 2 \* H и вывод в поле S-edit.

**e)** Метод завершает работу.

**4. Схема алгоритма с комментариями**

Рисунок 1. – Схема метода Select\_buttonClick модуля main\_manu

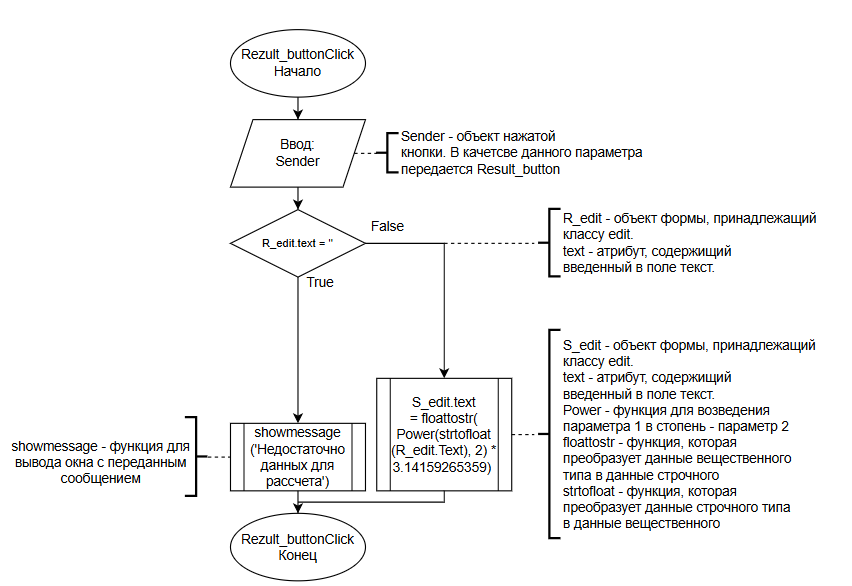


Рисунок 2. – Схема метода Result\_buttonClick модуля curcle\_modile

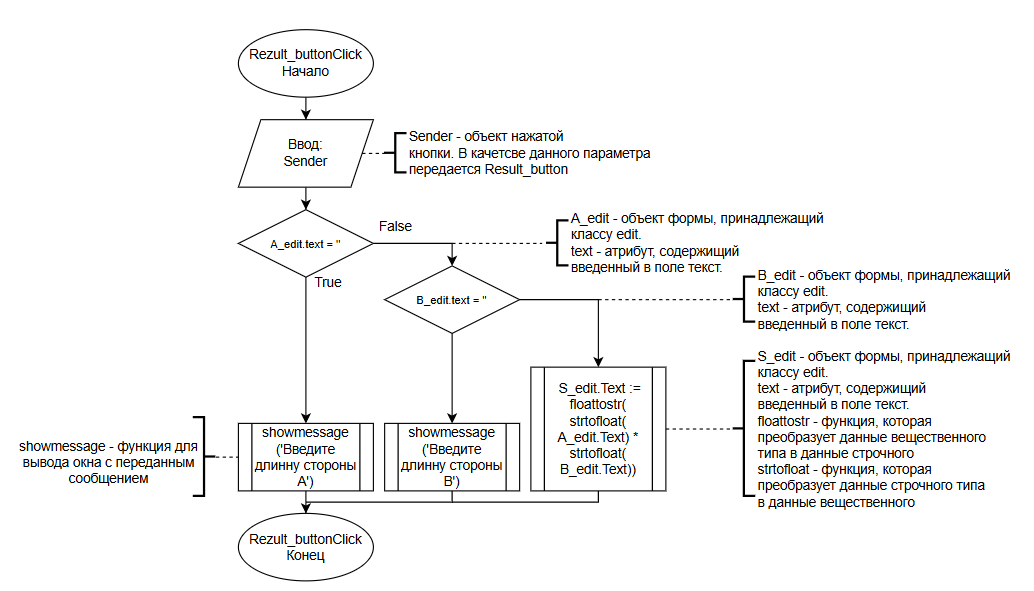
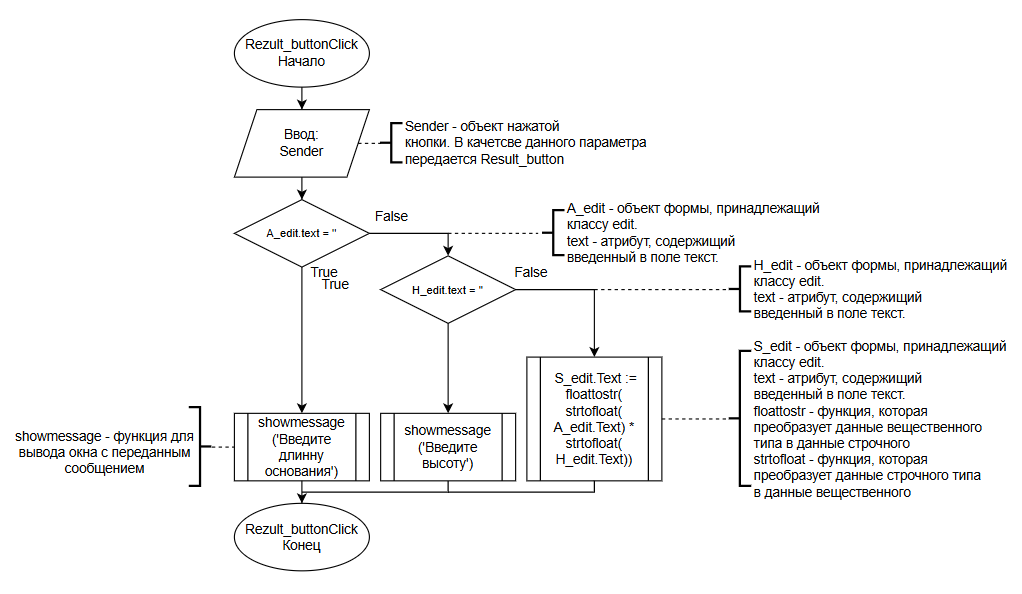


Рисунок 3. – Схема метода Result\_buttonClick модуля rectangel\_modile

Рисунок 4. – Схема метода Result\_buttonClick модуля paralelogramm\_module

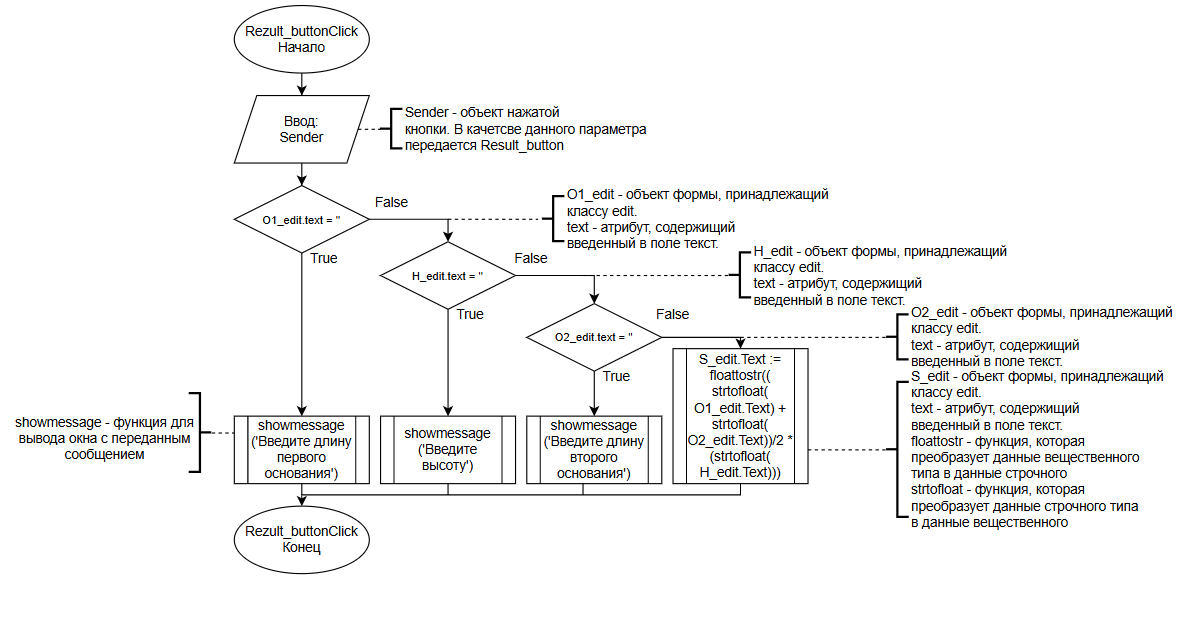
****

Рисунок 5. – Схема метода Result\_buttonClick модуля trapec\_module

**5. Код программы**

**5.1 Код модуля main\_manu**

unit Main\_manu;

{$mode objfpc}{$H+}

interface

uses

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

Rectangel\_module, curcle\_module, paralelogramm\_module, trapec\_module;

type

{ TMenu\_form }

TMenu\_form = class(TForm)

Select\_button: TButton;

Figure\_box: TComboBox;

Head\_label: TLabel;

information\_label: TLabel;

Menu\_bg\_image: TImage;

procedure Select\_buttonClick(Sender: TObject);

private

public

end;

var

Menu\_form: TMenu\_form;

implementation

{$R \*.lfm}

{ TMenu\_form }

procedure TMenu\_form.Select\_buttonClick(Sender: TObject);

begin

if Figure\_box.Text = 'Круг' then

Circle\_form.show

else if Figure\_box.Text = 'Прямоугольник' then

Rectangle\_form.show

else if Figure\_box.Text = 'Параллелограмм' then

Par\_form.show

else

Trap\_form.show;

end;

end.

**5.2 Код модуля curcle\_module**

unit curcle\_module;

{$mode ObjFPC}{$H+}

interface

uses

Classes, SysUtils, Forms, Controls, Graphics, Dialogs, ExtCtrls, StdCtrls, Math;

type

{ TCircle\_form }

TCircle\_form = class(TForm)

R\_edL: TLabel;

Rezult\_button: TButton;

R\_edit: TEdit;

S\_edit: TEdit;

form\_image: TImage;

Head: TLabel;

information\_label: TLabel;

S\_edL1: TLabel;

procedure Rezult\_buttonClick(Sender: TObject);

private

public

end;

var

Circle\_form: TCircle\_form;

implementation

{$R \*.lfm}

{ TCircle\_form }

procedure TCircle\_form.Rezult\_buttonClick(Sender: TObject);

begin

if R\_edit.Text = '' then

showmessage('Недостаточно данных для рассчета :(')

else

try

S\_edit.Text := floattostr(Power(strtofloat(R\_edit.Text), 2) \* 3.14159265359);

except

showmessage('Радиуc введен неверно :(');

end;

end;

end.

**5.3 Код модуля rectangel\_module**

unit Rectangel\_module;

{$mode ObjFPC}{$H+}

interface

uses

Classes, SysUtils, Forms, Controls, Graphics, Dialogs, ExtCtrls, StdCtrls;

type

{ TRectangle\_form }

TRectangle\_form = class(TForm)

bg\_image: TImage;

Result\_button: TButton;

S\_edit: TEdit;

B\_edit: TEdit;

A\_edit: TEdit;

Head: TLabel;

information\_label: TLabel;

S\_L: TLabel;

A\_L: TLabel;

B\_L: TLabel;

procedure information\_labelClick(Sender: TObject);

procedure Result\_buttonClick(Sender: TObject);

private

public

end;

var

Rectangle\_form: TRectangle\_form;

implementation

{$R \*.lfm}

{ TRectangle\_form }

procedure TRectangle\_form.Result\_buttonClick(Sender: TObject);

begin

if A\_edit.Text = '' then

showmessage('Введите длинну стороны A :(')

else if B\_edit.Text = '' then

showmessage('Введите длинну стороны B :(')

else

try

S\_edit.Text := floattostr(strtofloat(A\_edit.Text) \* strtofloat(B\_edit.Text));

except

showmessage('Длины сторон введены неверно :(');

end;

end;

procedure TRectangle\_form.information\_labelClick(Sender: TObject);

begin

end;

end.

**5.4 Код модуля paralelogramm\_module**

unit paralelogramm\_module;

{$mode ObjFPC}{$H+}

interface

uses

Classes, SysUtils, Forms, Controls, Graphics, Dialogs, ExtCtrls, StdCtrls;

type

{ TPar\_form }

TPar\_form = class(TForm)

bg\_image: TImage;

Rezult\_button: TButton;

S\_edit: TEdit;

H\_edit: TEdit;

A\_edit: TEdit;

Head: TLabel;

information\_label: TLabel;

S\_L: TLabel;

H\_L: TLabel;

A\_L: TLabel;

procedure Rezult\_buttonClick(Sender: TObject);

private

public

end;

var

Par\_form: TPar\_form;

implementation

{$R \*.lfm}

{ TPar\_form }

procedure TPar\_form.Rezult\_buttonClick(Sender: TObject);

begin

if A\_edit.Text = '' then

showmessage('Введите длинну основания :(')

else if H\_edit.Text = '' then

showmessage('Введите высоту :(')

else

try

S\_edit.Text := floattostr(strtofloat(A\_edit.Text) \* strtofloat(H\_edit.Text));

except

showmessage('Параметры введены неверно :(');

end;

end;

end.

**5.5 Код модуля trapec\_module**

unit trapec\_module;

{$mode ObjFPC}{$H+}

interface

uses

Classes, SysUtils, Forms, Controls, Graphics, Dialogs, ExtCtrls, StdCtrls;

type

{ Ttrap\_form }

Ttrap\_form = class(TForm)

bg\_image: TImage;

Rezult\_button: TButton;

S\_Edit: TEdit;

H\_edit: TEdit;

O1\_edit: TEdit;

O2\_Edit: TEdit;

Head: TLabel;

information\_label: TLabel;

S\_L: TLabel;

H\_L: TLabel;

O1\_L: TLabel;

O2\_L: TLabel;

procedure Rezult\_buttonClick(Sender: TObject);

private

public

end;

var

trap\_form: Ttrap\_form;

implementation

{$R \*.lfm}

{ Ttrap\_form }

procedure Ttrap\_form.Rezult\_buttonClick(Sender: TObject);

begin

if O1\_edit.Text = '' then

showmessage('Введите длинну первого основания :(')

else if H\_edit.Text = '' then

showmessage('Введите высоту :(')

else if O1\_edit.Text = '' then

showmessage('Введите длинну первого основания :(')

else

try

S\_edit.Text := floattostr((strtofloat(O1\_edit.Text) + strtofloat(O2\_edit.Text)) / 2 \* (strtofloat(H\_edit.Text)));

except

showmessage('Параметры введены неверно :(');

end;

end;

end.

**6. Результаты выполнения программы**

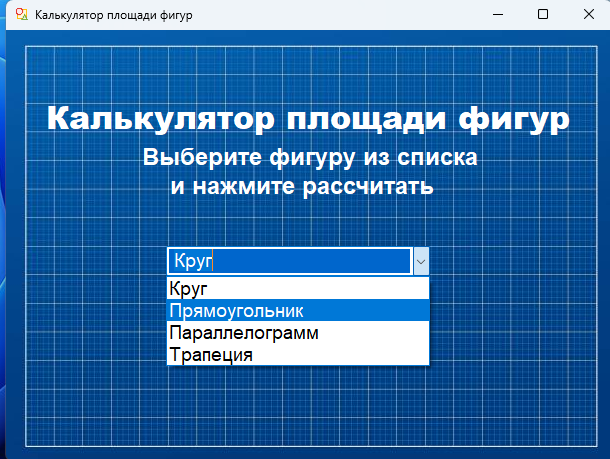
****

Рис 6.1 – Результат выполнения программы

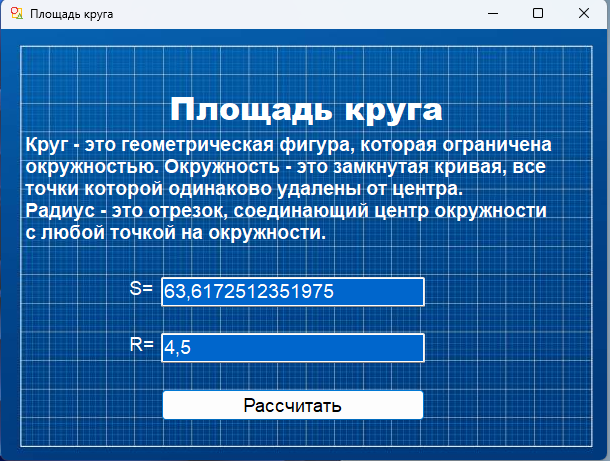


Рис 6.2 – Результат выполнения программы

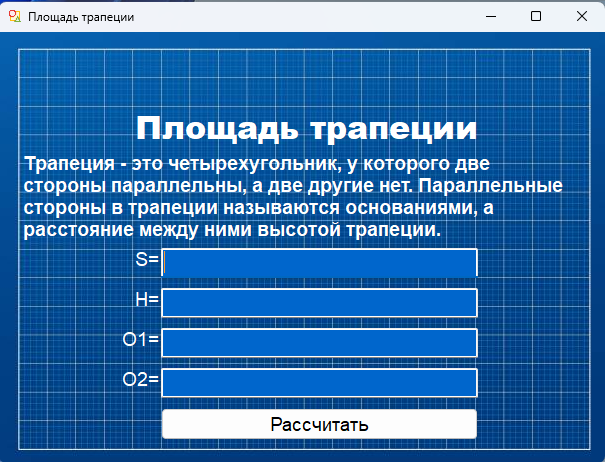


Рис 6.3 – Результат выполнения программы

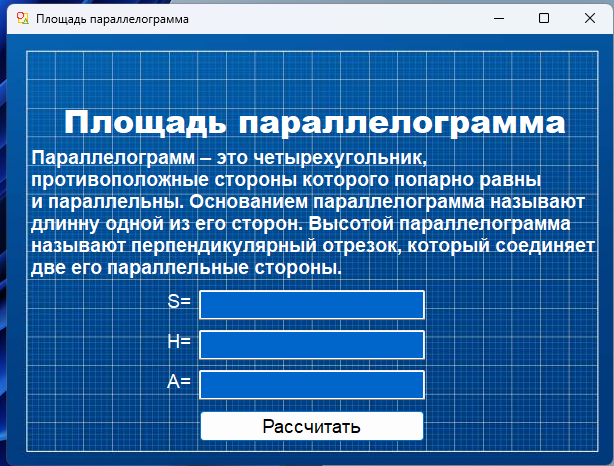


Рис 6.4 – Результат выполнения программы

**7. Вывод**

Работа завершена. В ходе работы были изучены основные принципы и аспекты событийно-ориентированного программирования в интегрированной среде разработки Lazarus. В процессе работы были освоены основные методы работы с виджетами Form, Label, Button, Combobox, Edit, Memo, Picture и т.д. и получены навыки разработки приложений с графическим пользовательским интерфейсом. В результате работы была разработана программа для расчета площадей геометрических фигур.