```
unit Unit1;

{$mode objfpc}{$H+}

interface

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

type

  { TForm1 }

  TForm1 = class(TForm)
    Button1: TButton;  { Кнопка запуска решения}
    Edit1: TEdit;      { Поле ввода стороны a}
    Edit2: TEdit;      { Поле ввода стороны b}
    Edit3: TEdit;      { Поле ввода угла gamma}
    Image1: TImage;    { Поясняющая картинка с принятыми обозначениями}
    Label1: TLabel;    { Пояснение к полю ввода стороны a Edit1}
    Label2: TLabel;    { Пояснение к полю ввода стороны b Edit2}
    Label3: TLabel;    { Пояснение к полю ввода угла gamma Edit3}
    Label4: TLabel;    { Пояснение пользователю}
    Label5: TLabel;    { Ответ сторона c}
    Label6: TLabel;    { Ответ площадь S}
    Label7: TLabel;    { Ответ радиус R}
    procedure Button1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);

  private
    { private declarations }
  public
    { public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.lfm}
```

```
{ TForm1 }

{Процедура выполняется при создании формы (запуске приложения) пока не надо.}
procedure TForm1.FormCreate(Sender: TObject);
begin

end;

{Процедура обработки щелчка по кнопке Button1 "РЕШЕНИЕ"}
procedure TForm1.Button1Click(Sender: TObject);
{
 a, b, c  : длины сторон треугольника
 S        : площадь треугольника
 R        : Радиус описанной окружности
 alpha, beta, gamma: углы треугольника
 kod1, kod2, kod3: коды индикаторы успешности преобразования
 en: Логический "флажок" true – данные правильные
                         false – данные некорректны.
}
Var
  a, b, c, R, S, gamma: real;
  kod1, kod2, kod3: integer;
  en: boolean;
Const
  MS1='В поля ввода вводите числа, а не всякую хрень! ';
  MS2='Угол между сторонами должен быть больше 0, но меньше 180 градусов!';

begin
  {Прежде всего преобразуем введенные данные и проверим их на корректность}
  en:=True; {Устанавливаем флаг. По умолчанию проверка пройдена}

  {Очищаем метки ответов от предыдущих результатов, если таковые были}
   Label5.Caption:='сторона c=?';
   Label6.Caption:='Площадь S=?';
   Label7.Caption:='Радиус R=?';

  { Проверяем корректность введенных данных (то что введены числа) всех скопом,
    и в случае некорректных данных сбрасываем флаг и просим повторить ввод
  }
```

```
  {Считываем длины сторон и угол из элементов Edit1, Edit2, Edit3.
 Т.е пытаемся преобразовать содежимое этих элементов в числа}
 Val(Edit1.Text, a, kod1);
 Val(Edit2.Text, b, kod2);
 Val(Edit3.Text, gamma, kod3);
{Коль пользователь ввел хрень, рявкнуть на него.}
 if (kod1<>0)or(kod2<>0)or(kod3<>0)
   then
     begin
       en:=false; {Сбрасываем флаг}
       MessageDlg(MS1, mtError, [mbOk],0);
     end;


 { Проверяем угол и, если он некорректен, сообщаем об этом.}
  if (gamma<=0)OR(gamma>=180)
    then
      begin
        en:=false; {Сбрасываем флаг}
        MessageDlg(MS2, mtWarning, [mbOk],0);
      end;

  {
   Только если данные корректны, то считаем площадь и радиус
   описанной окружности
   и выводим результаты.
  }
   if en
    then
      begin
         gamma:=gamma*pi/180;               { Переводим угол в
         радианы}
         c:=SQRT(a*a+b*b-2*a*b*COS(gamma)); {Считаем сторону c}
         S:=a*b*SIN(gamma)/2;               {Считаем площадь S}
         R:=a*b*c/(4*S);               {Считаем радиус описанной
         окружности R}

         {Выводим результаты в метки}
         Label5.Caption:='сторона c='+FloatToStrF(c, ffFixed, 6, 2);
         Label6.Caption:='Площадь S='+FloatToStrF(S, ffFixed, 6, 2);
         Label7.Caption:='Радиус R='+FloatToStrF(R, ffFixed, 6, 2);
      end;

  {XЪXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX}
```

```
end;


end.
```