Zum Programm: „Dreieck“

Vorhandene Programmteile

Variablendefinition

```
type
  punkt = record
    X: real;
    Y: real;
  end;

var
  a,
  b,
  c: punkt;  //global
```

Punkte einlesen

```
function Aeinlesen : punkt;
begin
  result.X := strtofloat(Form1.Edit1.text);
  result.Y := strtofloat(Form1.Edit2.text)
end; // Beispiel für Punkt A
```

Seitenlängenberechnung

```
function seitenlaenge(p, q: punkt) : real;
begin
  result := sqrt(sqr(p.X - q.X) + sqr(p.Y - q.Y))
end; // allgemeingültig
```

Winkelberechnung

```
function winkela(a, b, c: real) : real;
begin
  result := arccos((sqr(a) - sqr(b) - sqr(c)) / (-2 * b * c))
end; // Beispiel für Winkel Alpha .... !! Winkel in RAD!!!   !!
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Buttons, Math, Grids, ExtCtrls;
```

Flächenberechnung

```
function flaeche(sa, sb, wc: real) : real;
begin
  result := 0.5 * sa * sb * sin(wc)
end;
```

Ausgabe

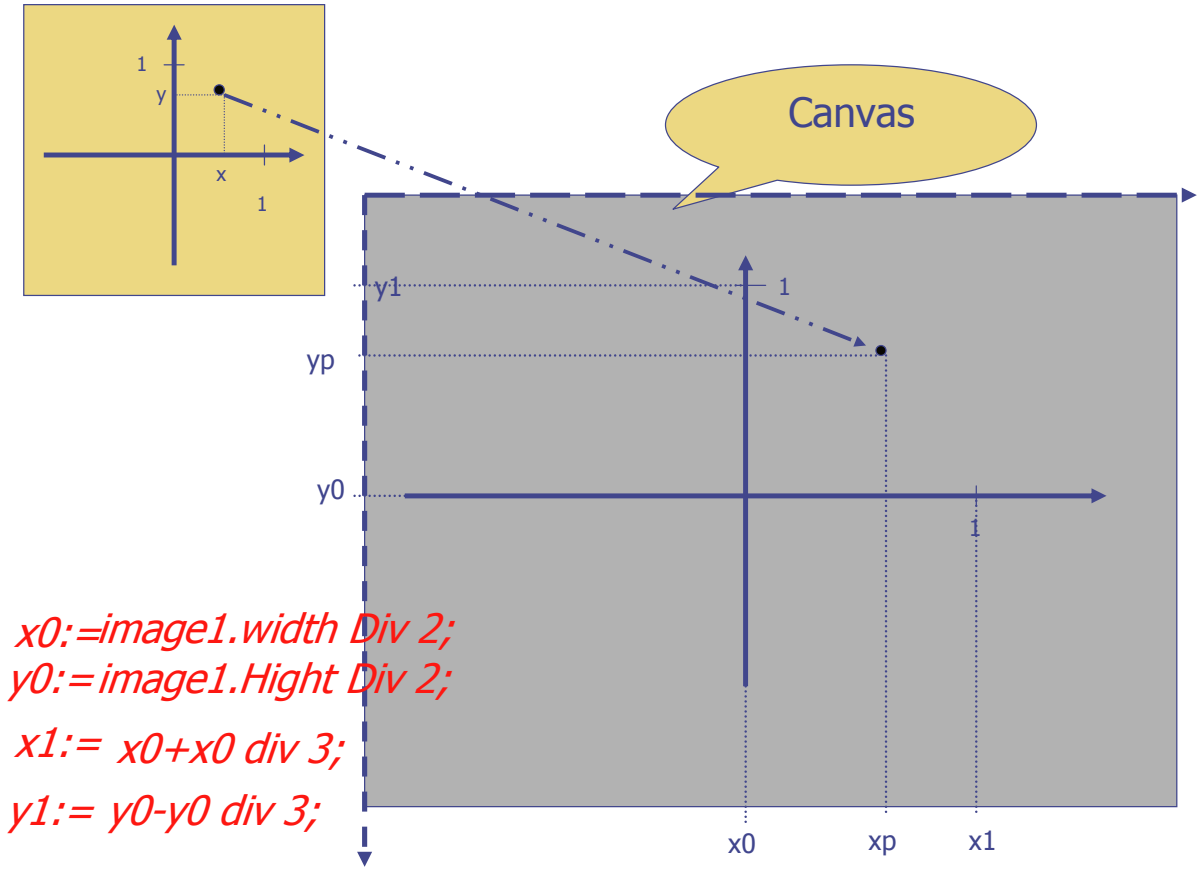
```
procedure ausgabe(a, b, c: punkt; sa, sb, sc, wa, wb, wc, f: real);
begin
  Form1.StringGrid1.cells[0, 0] := 'A';
  Form1.StringGrid1.cells[1, 0] := 'B';
  Form1.StringGrid1.cells[2, 0] := 'C';
  Form1.StringGrid1.cells[3, 0] := 'sa';
  Form1.StringGrid1.cells[4, 0] := 'sb';
  Form1.StringGrid1.cells[5, 0] := 'sc';
  Form1.StringGrid1.cells[6, 0] := 'Alpha';
  Form1.StringGrid1.cells[7, 0] := 'Beta';
  Form1.StringGrid1.cells[8, 0] := 'Gamma';
  Form1.StringGrid1.cells[9, 0] := 'Fläche';
  Form1.StringGrid1.cells[0, 1] := '(' + FloatToStr(a.X) + ' ; ' + FloatToStr(a.Y)
                                + ')';
  Form1.StringGrid1.cells[1, 1] := '(' + FloatToStr(b.X) + ' ; ' + FloatToStr(b.Y)
                                + ')';
  Form1.StringGrid1.cells[2, 1] := '(' + FloatToStr(c.X) + ' ; ' + FloatToStr(c.Y)
                                + ')';
  Form1.StringGrid1.cells[3, 1] := FloatToStr(sa);
  Form1.StringGrid1.cells[4, 1] := FloatToStr(sb);
  Form1.StringGrid1.cells[5, 1] := FloatToStr(sc);
  Form1.StringGrid1.cells[6, 1] := FloatToStr(wa * 180 / Pi) +
                                '°';
  Form1.StringGrid1.cells[7, 1] := FloatToStr(wb * 180 / Pi) +
                                '°';
  Form1.StringGrid1.cells[8, 1] := FloatToStr(wc * 180 / Pi) +
                                '°'; // Umrechnen in Gradmaß
  Form1.StringGrid1.cells[9, 1] := FloatToStr(f)
end;
```

1. Einbindung in BERECHNEN- Button

```
procedure TForm1.Button1Click(Sender: TObject);
var
  sa,
  sb,
  sc,
  wa,
  wb,
  wc,
  f: real;
begin
  a := Aeinlesen;
  b := Beinlesen;
  c := Ceinlesen;
  sa := seitenlaenge(b, c);
  sb := seitenlaenge(a, c);
  sc := seitenlaenge(a, b);
  wa := winkela(sa, sb, sc);
  wb := winkelb(sa, sb, sc);
  wc := winkelc(sa, sb, sc);
  f := flaeche(sa, sb, wc);
  ausgabe(a, b, c, sa, sb, sc, wa, wb, wc, f)
end;
```

Darstellung des Dreiecks

Zur Darstellung ist eine Image-Komponente notwendig.
(Einstellung Höhe: 300 Breite: 400



Globale Definition der notwendigen Variablen:

```
var
  breite,
  hoehe,
  x0,
  y0,
  x1,
  y1: Integer;
```

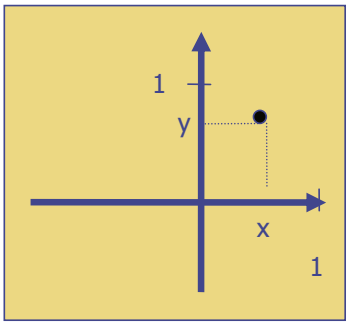
Berechnung der Größen

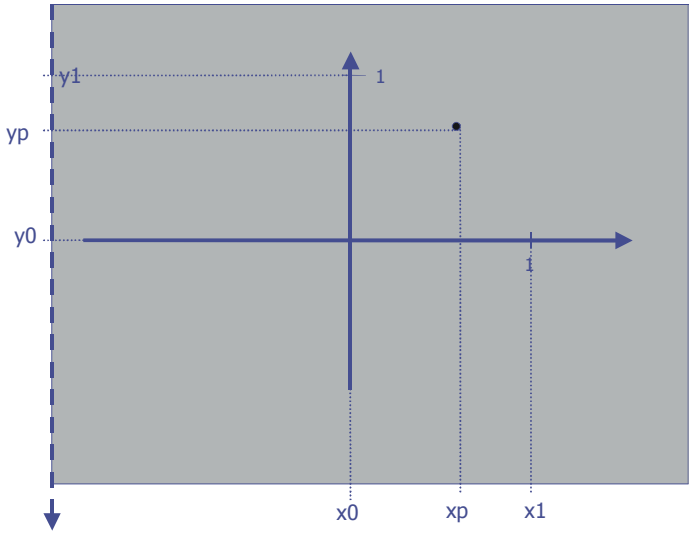
```
  breite := Image1.clientwidth;
  hoehe := Image1.clientheight;
  x0 := Image1.Width div 2;
  y0 := Image1.height div 2;
  x1 := x0 + x0 div 10;
  y1 := y0 - y0 div 10;
```

Zeichnen des Koordinatensystems

```
procedure koordinatensystem;
begin
  with Form1 do   //Verwendung von with ..do zur Vereinfachung des Programmtextes
  begin
    Image1.canvas.pen.color := clblack;
    Image1.canvas.MoveTo(0, y0);
    Image1.canvas.lineto(2 * x0, y0);
    Image1.canvas.MoveTo(x0, 0);
    Image1.canvas.lineto(x0, 2 * y0);
    Image1.canvas.MoveTo(x1, y0 + 5);
    Image1.canvas.lineto(x1, y0 - 5);
    Image1.canvas.MoveTo(x0 - 4, y1);
    Image1.canvas.lineto(x0 + 5, y1){Koordinatensystem
 wird formatiert und dargestellt}
  end;
end;
```

Transformation der Koordinaten von x-y-Koordinatensystem in Canvas-Koordinaten





$$\frac{x}{1} = \frac{xp - x0}{x1 - x0} \Rightarrow xp = x \cdot (x1 - x0) + x0$$

$$\frac{y}{1} = \frac{yp - y0}{y1 - y0} \Rightarrow yp = y \cdot (y1 - y0) + y0$$

Koordinatentransformation für das Dreieck

```
  a.X := a.X * (x1 - x0) + x0;
  b.X := b.X * (x1 - x0) + x0;
  c.X := c.X * (x1 - x0) + x0;
  a.Y := a.Y * (y1 - y0) + y0;
  b.Y := b.Y * (y1 - y0) + y0;
  c.Y := c.Y * (y1 - y0) + y0;
```

Darstellung des Dreiecks

```
procedure dreieckzeichnen(a, b, c: punkt);
begin
  with Form1 do
  begin
    a.X := a.X * (x1 - x0) + x0;
    b.X := b.X * (x1 - x0) + x0;
    c.X := c.X * (x1 - x0) + x0;
    a.Y := a.Y * (y1 - y0) + y0;
    b.Y := b.Y * (y1 - y0) + y0;
    c.Y := c.Y * (y1 - y0) + y0;
    Image1.canvas.MoveTo(round(a.X), round(a.Y));
    Image1.canvas.lineto(round(b.X), round(b.Y));
    Image1.canvas.MoveTo(round(b.X), round(b.Y));
    Image1.canvas.lineto(round(c.X), round(c.Y));
    Image1.canvas.MoveTo(round(c.X), round(c.Y));
    Image1.canvas.lineto(round(a.X), round(a.Y));
    Image1.canvas.TextOut(round(a.X), round(a.Y) + 5, 'A');
    Image1.canvas.TextOut(round(b.X), round(b.Y) + 5, 'B');
    Image1.canvas.TextOut(round(c.X), round(c.Y) + 5, 'C')
  end;
end;
```

Einbindung in Zeichnen- Button

```
procedure TForm1.Button2Click(Sender: TObject);
begin
  breite := Image1.clientwidth;
  hoehe := Image1.clientheight;
  x0 := Image1.Width div 2;
  y0 := Image1.height div 2;
  x1 := x0 + x0 div 10;
  y1 := y0 - y0 div 10;
  koordinatensystem;
  dreieckzeichnen(a, b, c)
end;
```