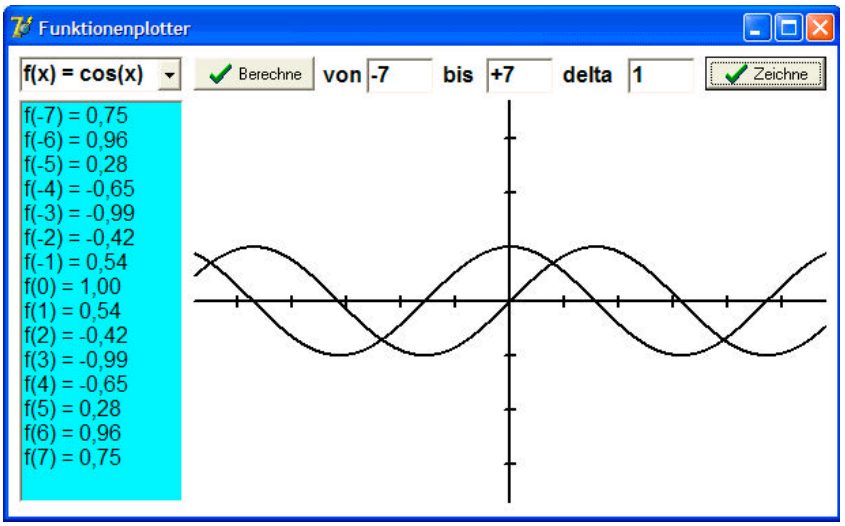
Projekt Funktionenplotter

In dem folgenden Projekt werden alle in der zweiten Programmiereinheit neu erworbenen Kenntnisse wiederholt und angewendet: Grafik, Schleifen, Prozeduren und Funktionen.

1. Starten Sie Delphi, wählen Sie **Datei Neue Anwendung** und speichern Sie mit **Projekt speichern unter**. Legen Sie einen **Ordner "Plotter"** unter Ihrem Namen an.
2. Geben Sie der FORM die **Caption "Funktionenplotter"** und den **Namen "FoPlotter"** und wählen Sie als Hintergrundfarbe **`clWhite`.**
3. Erstellen Sie das Layout mit folgenden Komponenten:
   - **ComboBox** (links oben) zur Funktionenauswahl (Items vorbelegen!)
   - **ListBox** (links unten) für die Wertetabelle
   - **PaintBox** (aus der Komponentenliste System) fürs Schaubild.



---

4. Die Wertetabelle erscheint analog zur letzten Stunde innerhalb einer Listbox.

```
ListBox1.Clear;
for i := start to ende do
  ListBox1.Items.Add(f(i));
```

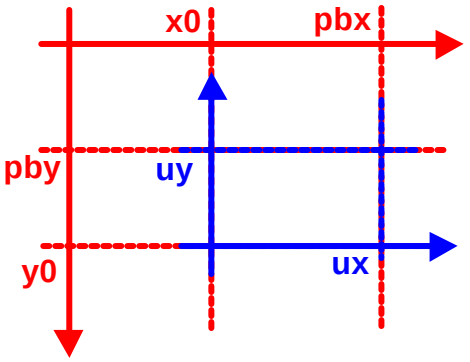
5. Die darzustellende Funktion wird über die ComboBox gewählt. Belegen Sie dazu die Items der ComboBox mit entsprechenden Funktionen. Je nach ItemIndex wird dann in der Funktionsdefinition mit einer case-Anweisung der jeweilige Term zugewiesen (s.r.):

```
function TFoPlotter.f(x:real):real;
begin
  case ComboBox1.ItemIndex of
     0: Result := sin(x);
     1: Result := cos(x);
  end;
end;
```

---

6. Die Funktionen für die Umrechnung der Koordinatensysteme müssen erstellt werden:

   Das **Paintbox-Koordinatensystem (pbx, pby)**
   - rechnet in Integer-Werten (Pixeln) und hat den
   - Ursprung in der linken oberen Bildschirmecke;
   - Das **User-Koordinatensystem (ux, uy)**
     - rechnet in "logischen Zentimetern" (real) und
     - mathematischer Koordinatenachsendefinition.



   Die Umrechnung wird durch drei Variablen x0, y0 und ppcm vollständig festgelegt:
   - Die Integer-Variablen **x0 und y0** geben den Ursprung des User-Koordinatensystems im Paintbox-Koordinaten-System an, und
   - die Real-Variable **ppcm** (Pixel per cm) regelt die Umrechnung der Einheiten.

a) Deklarieren Sie die globalen Variablen x0, y0 und ppcm im "**private**"-Abschnitt der Formular-Deklaration. Bevor Sie die Umrechnungsfunktionen implementieren können, müssen diese drei Variablen mit sinnvollen Werten belegt werden. Dies sollten Sie gleich beim Programmstart in der OnCreate-Methode des Formulars erledigen:

```
procedure TfoPlotter.FormCreate(Sender: TObject);
begin
  ppcm := 40;
  x0   := PaintBox1.Width  Div 2;
  y0   := PaintBox1.Height Div 2;
end;
```

Damit wird der Ursprung des User-Koordinatensystems in die Mitte des Zeichenbereichs geschoben; außerdem wird festgelegt, dass die Einheitsstrecke 40 Pixel lang ist.

b) Schreiben Sie nun die Umrechnungsfunktionen zwischen den Koordinatensystemen.

`function TFoPlotter.ux (pbx: Integer): Real;`	`function TFoPlotter.pbx (ux: Real): Integer;`
`function TFoPlotter.uy (pby: Integer): Real;`	`function TFoPlotter.pby (uy: Real): Integer;`

Fertigen Sie hierzu eine Zeichnung an, mit deren Hilfe Sie den mathematischen Zusammenhang zwischen den Paintbox- und den User-Koordinaten eines Punktes konstruieren können.

c) Für die praktische Arbeit ist es vorteilhaft, noch vier weitere Variablen zu vereinbaren, die beschreiben, welcher Bereich des Usersystems im Zeichenbereich der Paintbox dargestellt wird. Deklarieren Sie vier Real-Variablen **xmin, xmax, ymin** und **ymax**

im "private"-Abschnitt der Formular-Deklaration (zusätzlich zu x0, y0, ppcm) und initialisieren Sie sie (siehe rechts) in der OnCreate-Methode (siehe oben):

```
xmin := ux(0);
xmax := ux(PaintBox1.Width);
ymin := uy(PaintBox1.Height);
ymax := uy(0);
```

6. Die Koordinatenachsen und ihre Beschriftung müssen gezeichnet werden: Geben Sie hierzu in der Ereignisbehandlungsroutine OnClick des Zeichnen-Buttons (Doppelklick!)

mit Hilfe der eben definierten Umrechnungsfunktionen und Bereichsgrenzen das Koordinatensystem auf **PaintBox1.Canvas** aus.

Die Achsen werden z.B. durch die nebenstehenden Befehle gezeichnet:

```
With PaintBox1.Canvas do
begin
  MoveTo(pbx(xmin), pby(0));
  LineTo(pbx(xmax), pby(0));
  MoveTo(pbx(0), pby(ymin));
  LineTo(pbx(0), pby(ymax));
end;
```

Erzeugen Sie in einer Schleife, die von **Round(xmin)** bis **Round(xmax)** läuft, alle Skalierungsstriche auf der x-Achse. Erzeugen Sie danach die Skalierung der y-Achse.

Ein Skalierungsstrich mit der Länge 0,2 bei der Position (4) wird z.B. durch den folgenden Code ausgegeben:

```
MoveTo(pbx(4), pby(0.1));
LineTo(pbx(4), pby(-0.1));
```