FUNKTIONEN

Funktionen kennen Sie schon aus der Mathematik:
sin(30°) gibt einen Wert zurück, nämlich 0.5. sin(30°) können Sie verwenden wie irgend eine Variable.

z. B.
```
a:=sin(π/6);
b:= sin(π/6)+5;
Edit1.Text:=IntToStr(a);
```

Sie könnten aber die 1. und die 3. Anweisung zusammenfassen, und dabei eine Variable a sparen.

```
Edit1.Text:= IntToStr (sin(π/6));
```

Auch in Delphi kennen Sie schon eine ganze Reihe von vorgegebenen Variablen:

z. B. Arithmetische Funktionen: Abs(x), Int(x), sin(x), cos(x), Sqr(x), Sqrt(x), ArcTan(x), Random(x), Pi

Stringfunktionen: Length(x), IntToStr(x), Delete(wort, 2,1), LowerCase(x) und viele andere

Datums- und Zeitfunktionen: Date, DayOfWeek(datum)

Wie Sie sehen, gibt es Funktionen ohne Argument (Pi, Randomize), mit einem Argument (cos(x)) oder mit mehreren Argumenten (Delete, Insert).

Funktionen können wir auch selber definieren; das lohnt sich immer dann, wenn wir sie innerhalb einer Prozedur mehrmals benötigen, vielleicht auch, wenn wir eine Prozedur übersichtlicher gestalten möchten.

So sieht eine Funktion aus:

```
function Vzyl(r,h:Double):Double;
begin
   result:= pi*r*r*h;
end;
```

```
function Halb(a:Integer):Double;
begin
   Halb:=a/2;
end;
```

Zwischen **begin** und **end** können natürlich auch viel mehr Anweisungen stehen.

Sie sehen hier übrigens, dass in Funktionen eine Variable "result" benützt werden kann, die nicht definiert werden muss. Sie übernimmt in diesem Beispiel die Stelle von Vzyl

Aufgerufen (im Rahmen einer Prozedur) werden diese Funktionen z. B. mit:

```
Vzyl(3.75,1.8)
Halb(5)
```

Funktionen können nach **implementation**
{$R *.DFM}

eingefügt werden. Dann können sie von allen folgenden Prozeduren benützt werden.

Wenn Sie sie nur in einer einzigen Prozedur benützen, können Sie den Quelltext für die Funktion auch zwischen **procedure** und **begin** einfügen.

In "**Function** Halb(a:Integer):Double;" ist:

a ein Übergabeparameter vom Typ Integer.
a wird innerhalb der Funktionsdeklaration benützt.

Der Rückgabeparameter der Funktion, das wäre bei Halb(5) die Zahl 2.5, ist vom Typ Double.

Aufrufen lässt sich die Funktion mit jedem beliebigen Parameter:

```
b:=StrToInt(Edit1.Text);
Edit2.Text;=IntToStr(Halb(b));
```

Mehrere Variabeln können durch Komma getrennt gleichzeitig definiert werden, wenn Sie vom gleichen Typ sind. Variablen verschiedenen Typs trennt man durch Strichpunkt:

function Vzyl(r,h:Double):Double;
function Hoch(a:Real; n: Integer):Real; (Hoch(2.5, 2) ergibt $2.5^2 = 6.25$)

Übrigens:

Sie können in einer Prozedur auch andere Prozeduren aufrufen, die weiter oben definiert wurden.
Das ist ein Mittel, um komplizierte Strukturen übersichtlicher zu gestalten, indem man Teile davon in separate Prozeduren auslagert.

Eine Prozedur kann sich sogar selber wieder aufrufen, was aber mit Vorsicht zu bewerkstelligen ist, da das sehr leicht zu einer Endlosschleife führen kann. Dasselbe gilt für Funktionen.

Ein schönes Beispiel dafür ist die Funktion zur Berechnung einer Fakultät:

$n! = n \cdot (n-1) \cdot (n-2) \cdot \ldots \cdot 3 \cdot 2 \cdot 1$
z. B. $5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 120$

```
function fak(n:Integer):Integer;
begin
    if n=1 then result:=1 else result:=fak(n-1)*n;
end;
```