Ein- und Ausgabeparameter von Funktionen

Funktion zur Kontrolle der Ein- und Ausgabeparameter:

`nargin`	*Anzahl der Eingabeparameter*
`nargout`	*Anzahl der Ausgabeparameter*
`exist`	*prüft, ob eine Variable existiert*
`varargin`	*Eingabeparameterliste unbestimmter Länge (cell-array)*
`varargout`	*Ausgabeparameterliste unbestimmter Länge*
`nargchk`	*prüfen der Eingabeparameteranzahl*
`nargoutchk`	*prüfen der Ausgabeparameteranzahl*

Ein- und Ausgabeparameter von Funktionen

Funktion zur Kontrolle der Ein- und Ausgabeparameter:

`nargin`	*Anzahl der Eingabeparameter*
`nargout`	*Anzahl der Ausgabeparameter*
`exist`	*prüft, ob eine Variable existiert*
`varargin`	*Eingabeparameterliste unbestimmter Länge (cell-array)*
`varargout`	*Ausgabeparameterliste unbestimmter Länge*
`nargchk`	*prüfen der Eingabeparameteranzahl*
`nargoutchk`	*prüfen der Ausgabeparameteranzahl*

Übergabe von Funktionen als:

`string`	*Name der m-Datei, Aufruf mit feval*
`functionhandle`	

Beispiel

```
function [Vol, Ob] = zylinder(h, R, r)
% Volumen und Oberfläche (optional) eines Hohl- oder Vollzylinders
% mit Höhe h, Aussenradius R und Innenradius r (optional)

if nargin < 2
   error('zu wenige Parameter')
end

if nargin == 2
   r = 0
elseif r >= R
   error('Aussenradius nicht größer als Innenradius')
end

Vol = pi*h*(R^2 - r^2);

if nargout == 2
   Ob = 2*pi*(h*(r+R) + R^2-r^2);
end
```

Beispiel

```
function [x,fx] = steffensen(f,x,tol,max_it)
% STEFFENSEN zero of f(x) by Steffensen's method
% function [x,fx] = steffensen(f,x,tol,max_it)
% f: function handle
% x: approximation of a zero, updated
% tol: bound for |fx|, fx = f(x), for termination
% max_it: maximal number of iterations

% set defaults
if nargin < 4
   max_it = 100;
   if nargin < 3
      tol = 1e-10;
      if nargin < 2 disp('too few arguments'), return; end
   end
end
```

```
for n = 1:max_it
   fx = f(x);
   % Abbruch bei erreichter Genauigkeit
   if abs(fx) < tol
      return
   end
   d = f(x+fx)-fx;
   if abs(d) < eps*fx
      disp('nearly zero denominator');
      return
   end
   % Iterationsschritt
   x = x - fx*fx/d;
end
disp('no convergence');

end
```

```
>> x = steffensen(@cos,0)
   x =
       1.5708

>> fct = @(x) x^2-2
>> [x,fx] = steffensen(fct,1,1.0e-15,100)
   x =
       1.4142
   fx =
       4.4409e-16
```