

Octave Befehls Referenz

<https://github.com/Garbaz/octaveDE/blob/master/octave.md>

Grundlagen

Syntax

Ausgabe

Befehl *% Gibt das Resultat aus*

%z.B.

```
>> 1+2
```

```
ans = 3
```

```
>>
```

Befehl; *% Gibt das Resultat nicht aus*

% z.B.

```
>> a = 1+2;
```

```
>>
```

Befehle

<code>help</code> Befehl	<i>% Gibt Informationen zu Befehl</i>
<code>clear</code> [Variable]	<i>% Loescht Variable % (Oder alles wenn keine Variable angegeben)</i>
<code>who</code>	<i>% Gibt Informationen ueber aktuelle Variablen % & Funktionen (auch `whos`)</i>
<code>format</code> long/short	<i>% Aendert das Format (Anzahl der Fliesskommastellen)</i>
<code>save</code> Dateiname [V0 V1 ...]	<i>% Speichert den aktuellen Zustand (Variablen&Funktionen) % oder bestimmte Variablen/Funktionen als Datei</i>
<code>load</code> Dateiname	<i>% Laedt einen Zustand aus einer Datei</i>
<code>more</code> on/off	<i>% Aktiviert / Deaktiviert seitenweise Ausgabe</i>

Variablen & Listen

<code>a = Ausdruck</code>	<i>% Weist der Variable a einen Wert zu</i>
<code>[a0, a1, a2]</code>	<i>% Definiert eine Liste (auch [a0 a1 a2])</i>
<code>[a0; a1; a2]</code>	<i>% Definiert einen Vektor</i>
<code>[a00, a01, a02; a10, a11, a12]</code>	<i>% Definiert eine Matrix % (Im Beispiel 2 Zeilen a 3 Werten)</i>
<code>n:m</code>	<i>% Generiert eine Liste ganzer Zahlen von n bis m</i>

```

n:r:m          % Generiert eine Liste von Zahlen
               % mit Abstand r von n bis m
               % (z.B. 0:0.2:1 -> [0, 0.2, 0.4, 0.6, 0.8, 1])

linspace(n,m)  % Generiert eine Liste linear, gleichmaessig
               % verteilter Zahlen von n bis m

a(n)           % Gibt ntes Element von Liste zurueck
               % (auch a(n:m) -> Elemente n bis m)

a'             % Gibt die Transponierte von a.
               % (Aus Spalten werden Zeilen und anders herum)

```

Rechenoperatoren

Grundrechenarten / Matrix-Arithmetik

```

a+b  % Addition
a-b  % Subtraktion
a*b  % Multiplikation
a/b  % Division
a^b  % "a hoch b"

```

a & b koennen Konstanten, Variablen, Vektoren oder Matrizen sein. Fuer elementweise Verrechnung von Matrizen muss ein Punkt vor den Operator gesetzt werden.

z.B.:

```
a.*b
```

Integrierte Funktionen

```

sqrt(a)      % Quadratwurzel ("√")
nthroot(a,n) % nte Wurzel

exp(a)       % Exponential Funktion ("e hoch a")
log(a)       % Natuerlicher Logarithmus
log10(a)     % Logarithmus zur Basis 10

sin(a)       % Sinus Funktion (auch cos, tan, cot, csc & sec)
asin(a)      % Gegenfunktion von Sinus (auch acos, atan, acot, acsc & asec)

abs(a)       % Betrag (z.B. abs(-7) = 7)
sign(a)      % Vorzeichen (z.B. sign(-7) = -1)
round(a)     % Zur naechsten ganzen Zahl runden
floor(a)     % Abrunden
ceil(a)      % Aufrunden

mod(a,b)     % Modulo ("a mod b")

```

Integrierte Konstanten

```

pi      % Die Kreiszahl Pi (3.14159 ...)
e       % Eulersche Konstante (2.71828 ...)
i       % Komplexe Identitaet (√-1)
Inf     % Unendlich (∞)

ans     % Das Ergebnis der vorherigen Operation (Nicht wirklich eine Konstante)

```

Graphen zeichnen ("Plotten")

```

plot(x,y)           % Zeichnet einen Graph zweier Listen (x->y),
                    %   wobei x die Wertemenge bildet und y die Funktionswerte

plot(x,y,Optionen)  % Wie plot(x,y), jedoch mit Optionen
                    %   wie der Graph gezeichnet wird.
                    %   (z.B. plot(x,y,'b--') -> Graph ist Blau & Gestrichelt)

plot(x0,y0,o0,x1,y1,o1) % Zeichnet mehrere Graphen in einem Fenster

title(Titel)        % Setzt den Titel, welcher ueber dem Graphen steht
xlabel(Label)        % Setzt den Namen der an der x-Achse steht
ylabel(Label)        % Setzt den Namen der an der y-Achse steht

legend(Name0, Name1) % Gibt eine Legende zu den einzelnen Graphen an

axis(v)             % Setzt die Skalierung der x-/y-Achsen manuell
                    %   (v = [x_min, x_max, y_min, y_max])

figure              % Der naechste plot() Befehl wird
                    %   in einem neuen Fenster angezeigt

figure(n)           % Der naechste plot() Befehl wird
                    %   im nten, bereits existierenden, Fenster angezeigt

print(Dateiname,Format) % Speichert den aktuellen Plot als Datei
                    %   (Fuer PNG Bild: Format = '-dpng')

```

Scripte

```

edit    % Startet / Wechselt zu einem Editor

Name    % Fuehrt einen Script mit Dateiname "Name.m" aus,
        %   welcher im aktuellen Verzeichnis liegt.

cd Pfad % Wechselt den den aktuellen Pfad (cd .. -> Einen Ordner zurueck)

pwd     % Gibt aktuelles Verzeichnis aus

```

Das erste Kommentar in einer Script-Datei wird ausgegeben wenn man `help Name` eingibt.

Funktionen, Logik & Programm-Fluss

Funktionen

Definition einer Funktion ohne Parameter:

```
function name
%Operationen
end
```

Definition einer Funktion mit Parametern:

```
function name (param0, param1, param2)
%Operationen
end
```

Definition einer Funktion mit Rueckgabewert(en):

```
%Ein Rueckgabewert:
function c = name(a,b)
%Operationen (z.B. c = a+b)
end
```

```
%Mehrere Rueckgabewerte:
function [c,d,e] = name(a,b)
%Operationen (z.B. c = a; d = b; e = 5)
end
```

Das erste Kommentar in einer Funktionszuweisung wird ausgegeben wenn man `help name` eingibt.

Aufruf einer Funktion:

```
name(a,b,c)      % Ruft die Funktion name mit den Parametern a, b und c auf
                  % und gibt das Resultat zurueck (Variable "ans").

r = name(a,b,c) % Das Resultat wird der Variable r zugewiesen
```

Logik

```
a == b  % Gleich
a ~= b  % Ungleich

a > b    % Groesser
a >= b   % Groesser/Gleich
a < b    % Kleiner
a <= b   % Kleiner/Gleich

e & f    % Und
e | f    % Oder
~ e      % Nicht
```

Programm-Fluss

Bedingung ("If")

```
if e      % Wenn e wahr ist
```

```
elseif f % Wenn e nicht wahr ist, jedoch f
else     % Wenn keine der Bedingungen wahr ist
end
```

Fallunterscheidung ("Switch")

```
switch a
case a0, % Fall "a == a0"
case a1, % Fall "a == a1"
otherwise, % Falls keine der Faelle zutrifft
end
```

Schleifen ("For", "While")

```
for a = Liste % Laeuft durch alle Elemente der Liste
               % und fuehrt gegebene Operationen mit {a = aktuelles Element} aus.
end

while e % Solang die Logische Bedingung e erfuehlt ist
        % werden gegebene Operationen ausgefuehrt
end
```