

# 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

```
help Befehl           % Gibt Informationen zu Befehl
clear [Variable]      % Löscht Variable
                      % (Oder alles wenn keine Variable angegeben)
who                   % Gibt Informationen über aktuelle Variablen
                      % & Funktionen (auch `whos`)
format long/short     % Ändert das Format (Anzahl der Fließkommastellen)

save Dateiname [Variable] % Speichert den aktuellen Zustand (Variablen&Funktionen)
                      % oder eine bestimmte Variable als Datei
load Dateiname        % Lädt einen Zustand aus einer Datei
more on/off           % Aktiviert / Deaktiviert seitenweise Ausgabe
```

#### Variablen & Listen

```
a = Ausdruck          % Weist der Variable a einen Wert zu

[a0, a1, a2]           % Definiert eine Liste (auch [a0 a1 a2])
[a0; a1; a2]           % Definiert einen Vektor
[a00, a01, a02; a10, a11, a12] % Definiert eine Matrix
                      % (Im Beispiel 2 Zeilen à 3 Werten)

n:m                    % Generiert eine Liste ganzer Zahlen von n bis m
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, gleichmäßig
                      % verteilter Zahlen von n bis m

a(n)                   % Gibt ntes Element von Liste zurück
                      % (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 können Konstanten, Variablen, Vektoren oder Matrizen sein. Für 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)       % Natürlicher 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 & sec)

abs(a)       % Betrag (z.B. abs(-7) = 7)
sign(a)      % Vorzeichen (z.B. sign(-7) = -1)
round(a)     % Zur nächsten 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 Identität (√-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 über 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 nächste plot() Befehl wird
                  % in einem neuen Fenster angezeigt
figure(n)         % Der nächste plot() Befehl wird
                  % im nten, bereits existierenden, Fenster angezeigt

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

## Scripte

```
edit    % Startet / Wechselt zu einem Editor
Name    % Führt einen Script mit Dateiname "Name.m" aus,
        %   welcher im aktuellen Verzeichnis liegt.

cd Pfad % Wechselt den den aktuellen Pfad (cd .. -> Einen Ordner zurück)
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 Rückgabewert(en):

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

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 zurück (Variable "ans").

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

### Logik

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

a > b  % Größer
a >= b % Größer/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 Fälle zutrifft

end
```

### Schleifen ("For", "While")

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

end

while e % Solang die Logische Bedingung e erfüllt ist
        %   werden gegebene Operationen ausgeführt

end
```