Einführung in die Computer-Algebra

Teil 1 : Erste Schritte mit *Mathematica*

Informatik I – Mathematik mit dem Computer
Prof. Dr. Alfred Schreiber
Institut für Mathematik und ihre Didaktik · Universität Flensburg

---

Literatur, etc.

Stephen Wolfram:

Das *Mathematica* Buch. Die offizielle Dokumentation.

Addison-Wesley 1997 (und weitere Versionen)

Marie-Luise Herrmann:

Mathematica. Eine beispielorientierte Einführung.

Addison-Wesley 1997

Roman. E. Maeder:

Computer Science with *Mathematica*.

Cambridge University Press 2000

Weitere Quellen:

- Website und Links zu dieser Vorlesung
- Computer im Mathematikunterricht (aus: Grundzüge der Mathematikdidaktik)
- Homepage von Wolfram Research, Inc. (www.wolfram.com)
- *MathReader* (erlaubt das Lesen von Notebooks)

Menüs, Paletten, Notebooks, Einstellungen

File

Save

Save As ...

Save As Special ...

Palettes ‣ Auswahl diverser Paletten

Notebooks

Hierarchischer Aufbau (Zellen)

Typen von Zellen: Text, Input, Output, etc.

Einstellungen (Preferences ... im Menü Edit)

Notebook Options > Display Options > Window Toolbars

Cell Options > Display Options > ShowGroupOpenCloseIcon (True!)

Cell Options > New Cell Defaults >DefaultNewCellStyle (z.B. "Text" oder "Input")

Elementares numerisches Rechnen

■ Termeingabe in einer Input-Zelle

Bestätigen durch SHIFT + ENTER oder ENTER (Ziffernblock)

```
7 + 15
```

```
22
```

■ Rückgriff auf das letzte Ergebnis %

```
% * 3 / 11
```

```
6
```

■ Runde Klammern dienen der Gliederung (Rangfolge)

```
1234 * (35 - 3^10)
```

```
-72823276
```

■ Große Zahlen

```
2^1000
```

```
10715086071862673209484250490600018105614048117055336074437503883703510
 51124936122493198378815695858127594672917553146825187145285692314043591
 84577574698574803934567774824230985421074605062371141877954182153046471
 49835819412673987675591655439460770629145711964776865421676604298316521
 624386837205668069376
```

■ Numerische Näherungswerte

```
N[%]
```

```
1.07151 × 10^301
```

■ Argumentklammern sind eckig [...]

```
Mod[2^32 + 1, 641]
```

```
0
```

■ Zwei wichtige mathematische Konstanten

Die Kreiszahl π

```
Pi
N[Pi, 30]
```

π

```
3.14159265358979323846264338328
```

Die Eulersche Zahl *e*

```
E
N[E, 30]
```

e

```
2.71828182845904523536028747135
```

■ Quadratwurzel

Anfangsbuchstaben der Namen eingebauter Funktionen sind **groß** zu schreiben!

```
Sqrt[174]
```

$\sqrt{174}$

Bestimmte Funktionen lassen sich in nachgestellter Schreibweise (hinter // ausführen)

```
% // N
```

```
13.1909
```

■ Fakultät-Funktion

Beispiel für die Übernahme einer verbreiteten historischen Schreibweise

```
25!
```

```
15511210043330985984000000
```

■ Binomialzahlen

Beispiel einer eingebauten Funktion von zwei Argumenten

```
Binomial[7, 3]
```

```
35
```

Gebrauch von Symbolen (Variablen)

■ "Buchstabenrechnung"

Symbole können undefiniert in Ausdrücken erscheinen

```
Sqrt[a]
(a + b)^2
```

$$\sqrt{a}$$

$$(a + b)^2$$

Auskunft über den sog. Kontext einholen

```
?a b
```

```
Global`a
Global`b
```

■ Wertzuweisung

```
a = 13
```

```
13
```

Nun ist das Symbol a definiert

```
?a
```

```
Global`a
```

```
a = 13
```

Freie Namenswahl

(**keine** Unterstriche verwenden! Am besten nur Buchstaben *A*, *a*, *B*, *b*, ... und Ziffern 0, 1, 2, ...)

```
meinevar = 75
```

```
75
```

■ Gebrauch des Zeichens "="

"=" kann ähnlich wie in der Mathematik verwendet werden ...

```
b = c = a
```

```
13
```

Aber: **WARNUNG –** Eine Wertzuweisung ist **keine Gleichung**.

Die folgende Zeichenkette ist Unsinn!

```
8 = c
```

```
Set::setraw : Cannot assign to raw object 8.
```

```
13
```

■ Namen sollen "sprechen"

Ein Semikolon trennt zwei Befehle voneinander

```
radius = 5;
kreisfläche = Pi * radius^2
```

```
25 π
```

```
% // N
```

```
78.5398
```

■ Inhalt von Symbolen löschen

```
? b
```

```
Global`b
```

```
b = 13
```

Clear löscht den Inhalt von einer oder mehreren Variablen

```
Clear[b, c]
```

Die Symbole sind aber noch bekannt (d.h. hier: im Kontext Global)

```
? b
? c
```

```
Global`b
```

```
Global`c
```

Statt Clear[a] ist auch folgende Wertzuweisung möglich

```
a =.
```

■ Bezug zu Symbolen entfernen

```
Remove[b]
```

Zu b gibt es keinen Bezug mehr

```
? b
? c
```

```
Information::notfound : Symbol b not found.
```

```
Global`c
```

■ Ersetzungsregeln

Auf Ausdrücke Transformationsregeln anwenden: `Ausdruck /. ls -> rs`

Ein gewöhnlicher Ausdruck:

```
x + 2 y + z
```

```
x + 2 y + z
```

x durch a ersetzen:

```
x + 2 y + z /. x → a
```

```
a + 2 y + z
```

a durch a–z ersetzen:

```
% /. a → a - z
```

```
a + 2 y
```

Relationale und logische Operatoren

■ Kleiner(gleich), Größer(gleich)

```
32 < 1 + 6
```

```
False
```

```
45 > 3
```

```
True
```

```
40 ≤ 41
```

```
True
```

■ Gleich, Ungleich

```
2 + 3 == 5
```

```
True
```

Negation von Gleich: **!=**

```
2 + 3 ≠ 5
```

```
False
```

■ Und, Oder

Und-Verknüpfung

```
(Pi > E) && (Sqrt[6] > 2.5)
```

```
False
```

Oder-Verknüpfung (einschließend)

```
(Pi > E) || (Sqrt[6] > 2.5)
```

```
True
```

■ Wenn ... dann ... sonst

```
If[Pi > E, 1, 0]
```

```
1
```

```
If[2 + 2 ≠ 4, "Huh!", "Gottseidank!"]
```

```
Gottseidank!
```

Listen

Was ist eine Liste? - Schreibweise

Listen (geordnete Sammlung beliebiger Elemente)

```
{5, x, -Pi, 5, "Hallo"}
```

```
{5, x, -π, 5, Hallo}
```

Eine Liste ist keine Menge

```
{5, x, -Pi, 5, "Hallo"} == {5, x, -Pi, "Hallo"}
```

```
False
```

Auf die Reihenfolge komm es an

```
{5, x, -Pi, 5, "Hallo"} == {5, 5, x, "Hallo", -Pi}
```

```
False
```

Warum kann *Mathematica* den nachstehenden Test auf Gleichheit nicht entscheiden?

```
{5, x, -Pi, 5, "Hallo"} == {5, 5, -Pi, x, "Hallo"}
```

```
{5, x, -π, 5, Hallo} == {5, 5, -π, x, Hallo}
```

Erzeugung

Eine Liste der ersten 10 Quadratzahlen

```
Table[k^2, {k, 1, 10}]
```

```
{1, 4, 9, 16, 25, 36, 49, 64, 81, 100}
```

■ Wertzuweisungen

Es ist vorteilhaft, Listen einen Namen zu geben

```
qzliste = Table[k^2, {k, 1, 10}];
```

Elementweise Mehrfach-Wertzuweisung

```
{a, b, c} = {7, -1, 23};
```

```
b
```

```
-1
```

■ Listen als Elemente von Listen

Listen können selbst wieder Element einer Liste sein usf.

Beispiel: Liste aller Wertepaare (x, x^2)

```
qzWerteTab = Table[{x, x^2}, {x, 5, 9}]
```

```
{{5, 25}, {6, 36}, {7, 49}, {8, 64}, {9, 81}}
```

■ Darstellungsformen

Liste in Form einer Matrix darstellen

```
MatrixForm[qzWerteTab]
```

$$\begin{pmatrix} 5 & 25 \\ 6 & 36 \\ 7 & 49 \\ 8 & 64 \\ 9 & 81 \end{pmatrix}$$

Liste in Form einer Tabelle darstellen (MatrixForm und TableForm lassen sich mittels // anhängen)

```
qzWerteTab // TableForm
```

```
5    25
6    36
7    49
8    64
9    81
```

Sog. Optionen sorgen für eine Beschriftung

```
TableForm[qzWerteTab, TableHeadings → {None, {"x", "x^2"}}]
```

```
x    x^2
5    25
6    36
7    49
8    64
9    81
```

■ Länge einer Liste

```
Length[qzWerteTab]
```

```
5
```

■ Zugriff auf Elemente

Aus der zuvor erzeugten Liste ...

```
qzliste
```

```
{1, 4, 9, 16, 25, 36, 49, 64, 81, 100}
```

wird das Element Nr. 7 herausgeholt:

```
Part[qzliste, 7]
```

```
49
```

Alternative Schreibweise (mit eckigen Doppelklammern)

```
qzliste[[7]]
```

```
49
```

Die Funktion **Part[ ]** bietet mehr Möglichkeiten:

```
Part[qzliste, {7, 9}]
```

```
{49, 81}
```

Symbolisches Rechnen

■ Ausmultiplizieren

```
b = (1 + x)^20
```

$$(1 + x)^{20}$$

```
Expand[b]
```

$$1 + 20\,x + 190\,x^2 + 1140\,x^3 + 4845\,x^4 + 15504\,x^5 + 38760\,x^6 + 77520\,x^7 + 125970\,x^8 + 167960\,x^9 + 184756\,x^{10} + 167960\,x^{11} + 125970\,x^{12} + 77520\,x^{13} + 38760\,x^{14} + 15504\,x^{15} + 4845\,x^{16} + 1140\,x^{17} + 190\,x^{18} + 20\,x^{19} + x^{20}$$

■ Reduzieren

```
Factor[-27 - 54 u - 27 u^2 + 54 x + 162 u x + 108 u^2 x + 45 x^2 - 72 u x^2 - 144 u^2 x^2 - 116 x^3 -
  160 u x^3 + 64 u^2 x^3 - 16 x^4 + 128 u x^4 + 64 x^5 + 162 y^4 + 324 u y^4 + 162 u^2 y^4 - 108 x y^4 -
  540 u x y^4 - 432 u^2 x y^4 - 414 x^2 y^4 - 288 u x^2 y^4 + 288 u^2 x^2 y^4 + 144 x^3 y^4 + 576 u x^3 y^4 +
  288 x^4 y^4 - 324 y^8 - 648 u y^8 - 324 u^2 y^8 - 216 x y^8 + 216 u x y^8 + 432 u^2 x y^8 + 540 x^2 y^8 +
  864 u x^2 y^8 + 432 x^3 y^8 + 216 y^12 + 432 u y^12 + 216 u^2 y^12 + 432 x y^12 + 432 u x y^12 + 216 x^2 y^12]
```

$$(1 + u + x)^2\,(-3 + 4\,x + 6\,y^4)^3$$

Alle Variableninhalte löschen (um im Folgenden keine Überraschungen zu erleben)

```
Clear["Global`*"]
```

■ Vereinfachen (allgemein)

```
Simplify[(a - b)^2 - (a + b)^2]
```

$$-4\,a\,b$$

■ Summen berechnen

```
Sum[k^2, {k, 1, n}]
```

$$\frac{1}{6}\,n\,(1+n)\,(1+2\,n)$$

■ Gleichungen lösen

Lösung wird als Menge von Ersetzungsregeln ausgegeben

```
lsg = Solve[x^2 + a x == 7, x]
```

$$\left\{\left\{x \to \frac{1}{2}\left(-a-\sqrt{28+a^2}\right)\right\},\ \left\{x \to \frac{1}{2}\left(-a+\sqrt{28+a^2}\right)\right\}\right\}$$

Übergang zur Lösungsmenge ...

```
x /. lsg
```

$$\left\{\frac{1}{2}\left(-a-\sqrt{28+a^2}\right),\ \frac{1}{2}\left(-a+\sqrt{28+a^2}\right)\right\}$$

... oder direkt auf eine einzelne Lösung zugreifen:

```
x /. lsg[[1]]
```

$$\frac{1}{2}\left(-a-\sqrt{28+a^2}\right)$$

Beispiel eines linearen Gleichungssystems

```
Solve[{a x + b y == c, d x + e y == f}, {x, y}]
```

$$\left\{\left\{x \to -\frac{-c\,e + b\,f}{-b\,d + a\,e},\ y \to -\frac{-c\,d + a\,f}{b\,d - a\,e}\right\}\right\}$$

An der Grenze algebraischer Lösbarkeit:

```
x /. Solve[x^4 + 3 x - 1 == 0, x][[1]]
```

$$\frac{1}{2}\sqrt{-4\left(\frac{2}{3\left(81+\sqrt{7329}\right)}\right)^{1/3}+\frac{\left(\frac{1}{2}\left(81+\sqrt{7329}\right)\right)^{1/3}}{3^{2/3}}}-\frac{1}{2}\sqrt{\left(4\left(\frac{2}{3\left(81+\sqrt{7329}\right)}\right)^{1/3}-\frac{\left(\frac{1}{2}\left(81+\sqrt{7329}\right)\right)^{1/3}}{3^{2/3}}-\frac{6}{\sqrt{-4\left(\frac{2}{3\left(81+\sqrt{7329}\right)}\right)^{1/3}+\frac{\left(\frac{1}{2}\left(81+\sqrt{7329}\right)\right)^{1/3}}{3^{2/3}}}}\right)}$$

Funktionen

■ Eigene Funktionen definieren

Argument x als Muster x_ (nur auf der linken Seite)

Definitionsoperator :=

```
f[x_] := (x / 2) (x^3 - 5 x^2 + 1)
```

Weitere Handhabung wie in der Mathematik üblich:

```
f[a + 3] // Expand
```

$$-\frac{51}{2} - 13\,a + \frac{9\,a^2}{2} + \frac{7\,a^3}{2} + \frac{a^4}{2}$$

■ Definitionen anzeigen lassen

```
?f
```

```
Global`f
```

$$f[x\_] := \frac{1}{2}\, x\, (x^3 - 5\, x^2 + 1)$$

■ Funktionen mit mehreren Argumenten

Früher definierte Funktionen dürfen (natürlich) benutzt werden

```
g[u_, v_] := u * f[v] + v^2 / u
```

```
g[a, b]
```

$$\frac{b^2}{a} + \frac{1}{2}\, a\, b\, (1 - 5\, b^2 + b^3)$$

■ Nicht-Benötigtes löschen

Unbedingt zu empfehlen!

Eventuell nur die Definition löschen ...

```
Clear[g]
```

... oder nicht mehr benötigte Funktionssymbole vollständig entfernen:

```
Remove[f, g]
```

```
f[4]
```

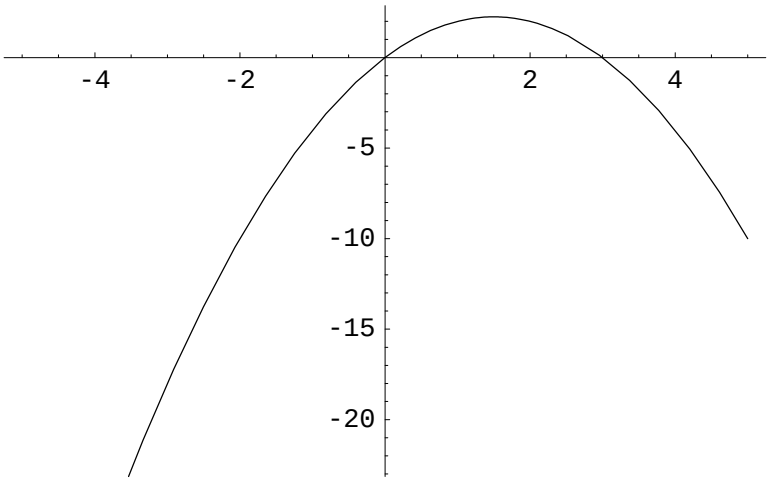
```
f[4]
```

Zeichnen ebener Schaubilder

■ Einfacher Funktionsgraph

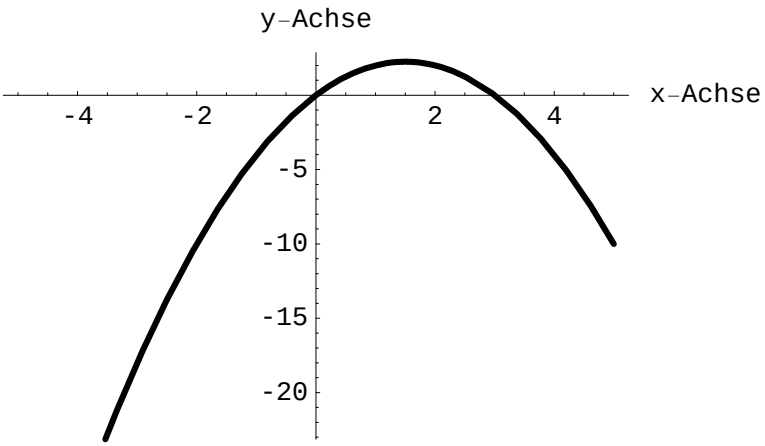
Der Plot-Befehl in der Grundform:

```
Plot[3 x - x^2, {x, -5, 5}];
```



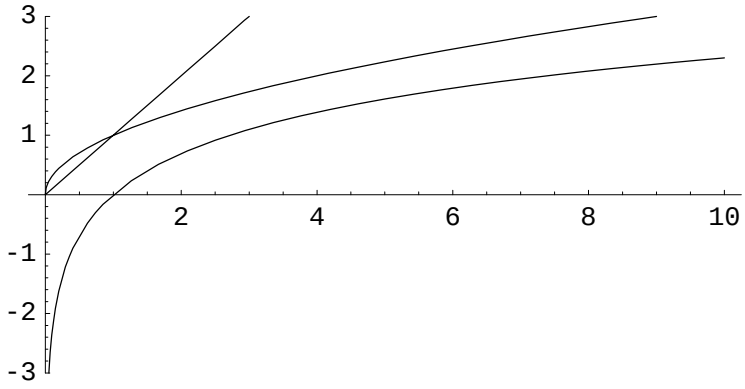
Mit Optionen lässt sich ein Schaubild verfeinern, z.B. Achsenbeschriftung und Strichdicke:

```
Plot[3 x - x^2, {x, -5, 5},
  AxesLabel → {"x-Achse", "y-Achse"}, PlotStyle → {Thickness[0.01]}];
```



■ Mehrere Kurven in einem Schaubild

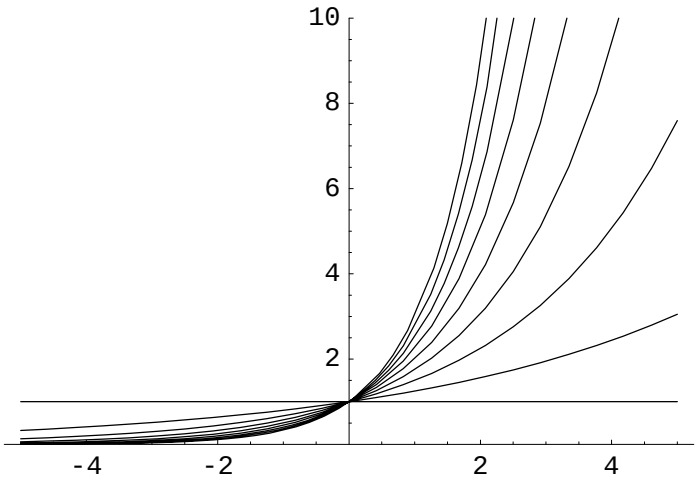
```
Plot[{x, Sqrt[x], Log[x]}, {x, 0, 10}, AspectRatio -> 0.5, PlotRange -> {-3, 3}];
```



■ Funktionsscharen

Die Schar der Exponentialfunktionen

```
schar = Table[a^x, {a, 1, 3, 0.25}];
Plot[Evaluate[schar], {x, -5, 5}, PlotRange → {0, 10}];
```



■ Optionen des Plot-Befehls

So informiert man sich über die (voreingestellten) Optionen eines Befehls:

```
Options[Plot]
```

```
{AspectRatio → 1/GoldenRatio, Axes → Automatic, AxesLabel → None,
 AxesOrigin → Automatic, AxesStyle → Automatic, Background → Automatic,
 ColorOutput → Automatic, Compiled → True, DefaultColor → Automatic,
 Epilog → {}, Frame → False, FrameLabel → None, FrameStyle → Automatic,
 FrameTicks → Automatic, GridLines → None, ImageSize → Automatic,
 MaxBend → 10., PlotDivision → 30., PlotLabel → None, PlotPoints → 25,
 PlotRange → Automatic, PlotRegion → Automatic, PlotStyle → Automatic,
 Prolog → {}, RotateLabel → True, Ticks → Automatic,
 DefaultFont :→ $DefaultFont, DisplayFunction :→ $DisplayFunction,
 FormatType :→ $FormatType, TextStyle :→ $TextStyle}
```

Die Bedeutung der Optionen erhält man über die Hilfe.

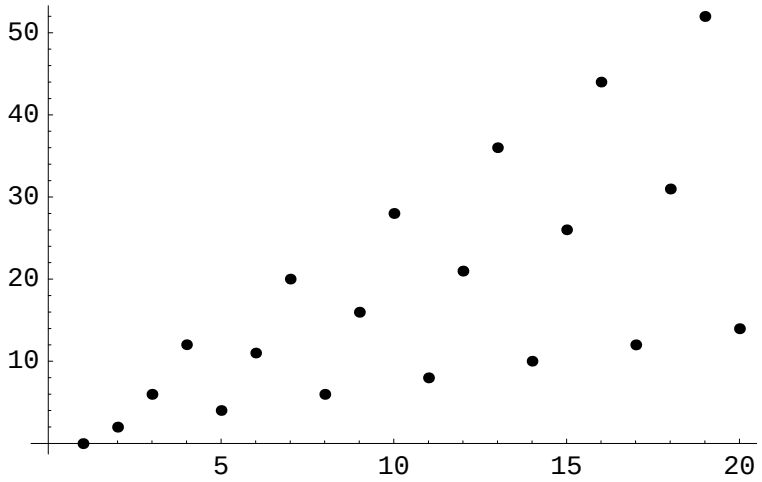
(Im Output die Schreibmarke auf das Optionswort setzen und anschließend F1-Taste drücken).

■ Datenlisten zeichnen

Beispiel-Daten:

```
daten = Table[Mod[k (k - 1), 3 k + 1], {k, 1, 20}];
```

```
ListPlot[daten, PlotStyle → {PointSize[0.015]}];
```

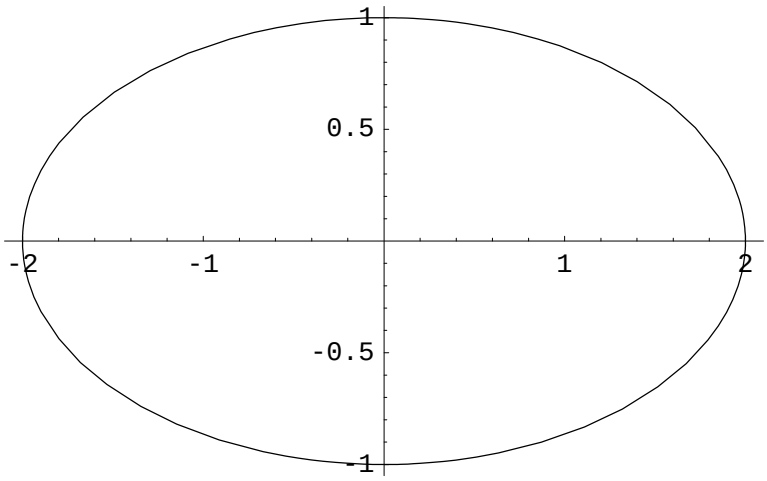


Mit der Option `PlotJoined -> True` können die Punkte verbunden werden.

■ Kurven in Parameterdarstellung

Eine Ellipse (deren Schaubild ist nicht der Graph einer Funktion):

```
ParametricPlot[{2 Cos[t], Sin[t]}, {t, 0, 2 * Pi}];
```

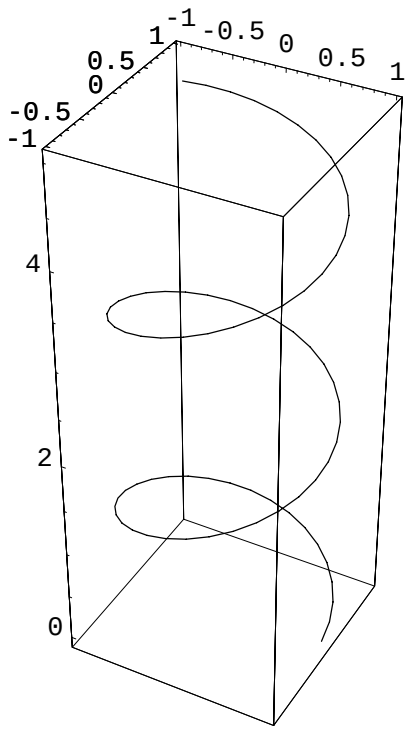


3D-Grafik

■ Dreidimensionales Linien-Zeichnen

Eine Schraubenlinie in Parameterdarstellung

```
ParametricPlot3D[{Cos[t], Sin[t], t / 3}, {t, 0, 15}];
```

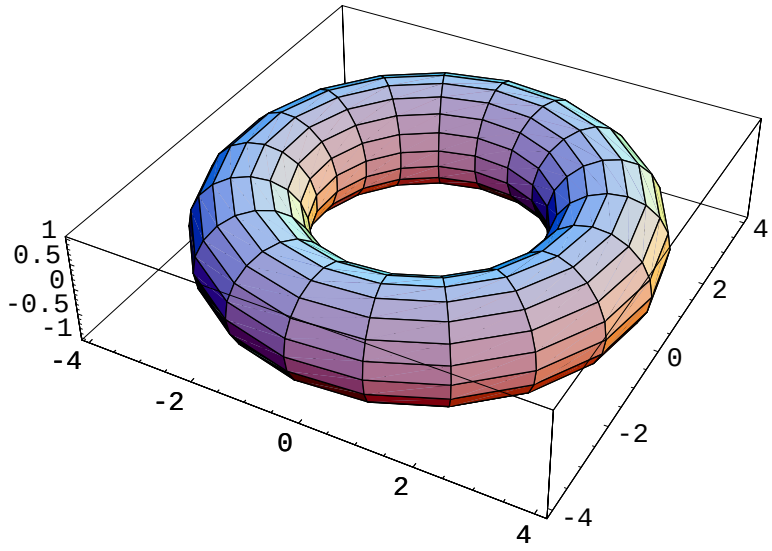


■ Flächen im Raum

Eine Fläche benötigt im Raum 2 Parameter.

Der folgende Torus wird dadurch erzeugt, dass mit der Variation von *u* ein Kreis entsteht und dieser zusätzlich mit der Variation von *t* auf einer Kreisbahn um die z-Achse rotiert:

```
ParametricPlot3D[
  {Cos[t] * (3 + Cos[u]), Sin[t] * (3 + Cos[u]), Sin[u]}, {t, 0, 2 Pi}, {u, 0, 2 Pi}];
```



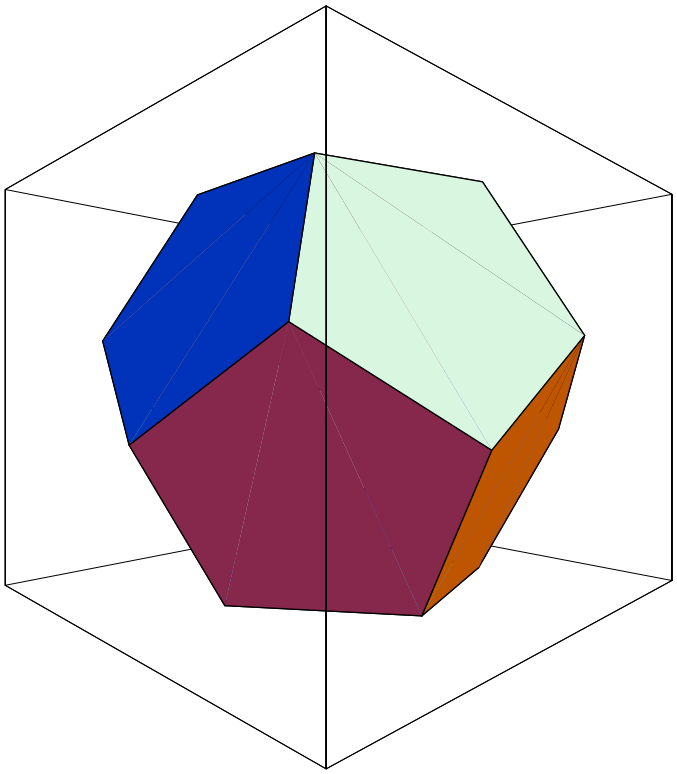
■ Polyeder

Zunächst ist ein sog. *Mathematica*-Paket zu laden:

```
<< Graphics`Polyhedra`
```

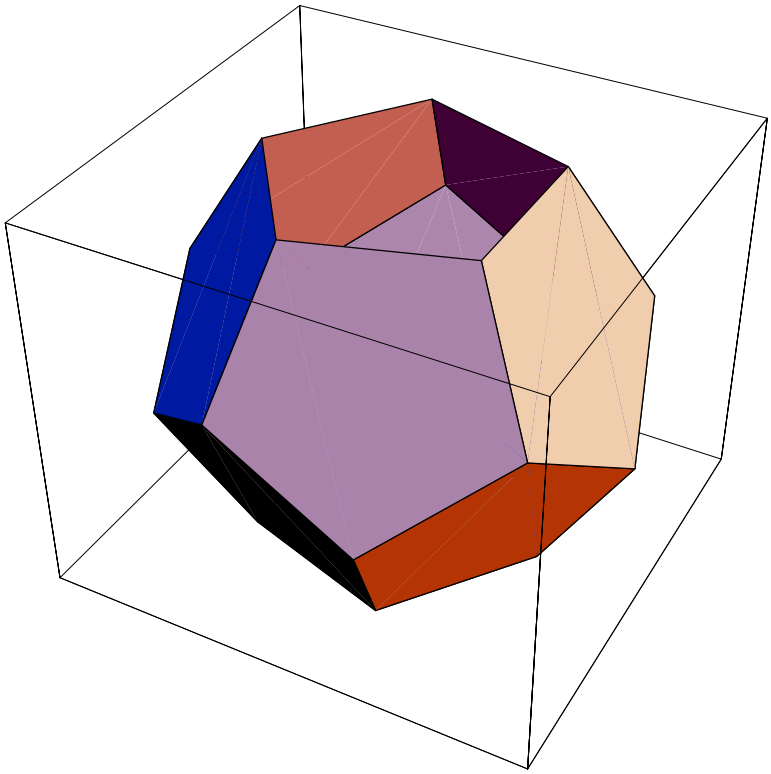
Darstellung eines Dodekaeders:

```
Show[Graphics3D[Dodecahedron[], ViewPoint -> {1, 1, 0}]];
```



Wir entfernen die obenliegende Seitenfläche:

```
dd = Drop[Dodecahedron[], {1}];
Show[Graphics3D[dd]];
```



Animation

■ *Mathematica*-Package für Animation

Folgender Befehl lädt das Animationspaket:

```
<< Graphics`Animation`
```

■ Ebene Grundform

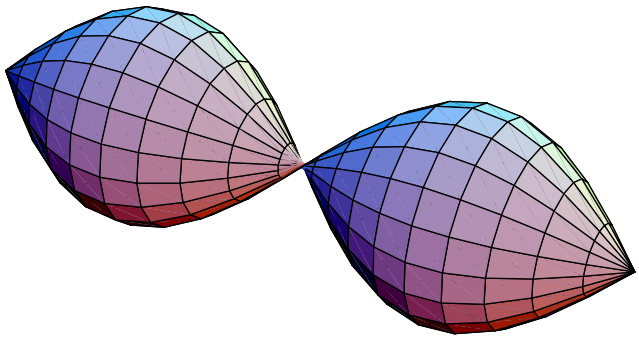
`Animate[ ]` übernimmt einen Plot-Befehl, der eine Schar von Objekten erzeugt. Die Anweisungsliste für den Scharparameter wird hinzugefügt:

```
Animate[Plot[Sin[n x], {x, 0, 2 Pi}, Axes -> False], {n, 1, 6, 0.5}];
```

■ Animation einer räumlichen Figur

Zunächst wird ein Objekt g erzeugt:

```
g = ParametricPlot3D[{x, Cos[t] Sin[x],Sin[t] Sin[x]},
      {x, -Pi, Pi}, {t, 0, 2Pi}, Axes -> False, Boxed -> False];
```



Nun wird g mittels **SpinShow** animiert (10 Phasen). Aufgrund der Symmetrie ist nur eine Halbdrehung erforderlich:

```
SpinShow[g,Frames -> 10, SpinRange -> {0 Degree, 180 Degree}];
```

Einführung in die Computer-Algebra

Teil 2 : Fortgeschrittenere Möglichkeiten mit *Mathematica*

Informatik I – Mathematik mit dem Computer
Prof. Dr. Alfred Schreiber
Institut für Mathematik und ihre Didaktik · Universität Flensburg

---

Numerisches Lösen von Gleichungen

```
Remove["Global`*"]
```

■ Vorbereitung

Linke Seite einer Gleichung zweckmäßgerweise als Funktion definieren:

```
g[x_] := x^3 - 2 x + 1
```

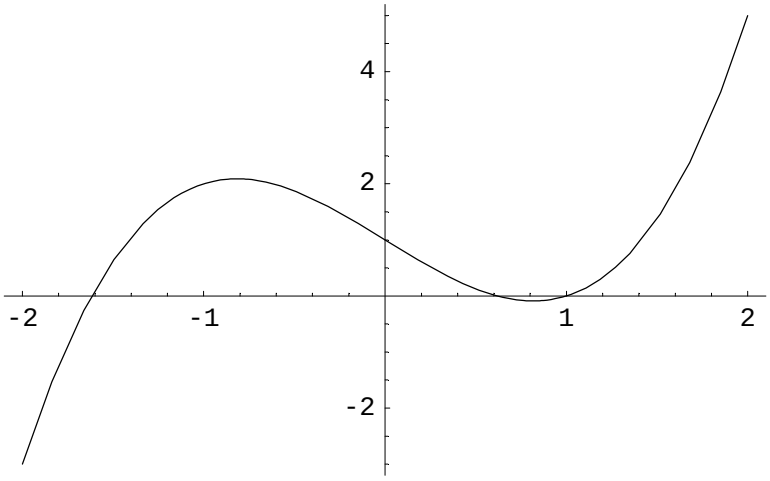
Gleichung durch Symbol (glg) wiedergeben:

```
glg = g[x] == 0
```

$$1 - 2\,x + x^3 == 0$$

■ Reeller Funktionsgraph

```
Plot[g[x], {x, -2, 2}];
```



Grafik markieren (Mausklick) und anschließend mittels [Strg]-Taste + Mauscursor die Lage der Nullstellen angenähert messen.

In diesem Fall lassen sich die 3 reellen Nullstellen von $g(x)$ auch algebraisch berechnen:

```
x /. Solve[glg, x]
```

$$\left\{1, \frac{1}{2}\left(-1-\sqrt{5}\right), \frac{1}{2}\left(-1+\sqrt{5}\right)\right\}$$

■ Numerische Lösungen

Erste Möglichkeit (nur für Polynome!):

```
x /. NSolve[glg, x]
```

```
{-1.61803, 0.618034, 1.}
```

Die zweite (allgemeinere) Möglichkeit verlangt Angabe eines Startwerts (für das eingebaute Newtonsche Verfahren):

```
FindRoot[glg, {x, 0}]
FindRoot[glg, {x, 2}]
FindRoot[glg, {x, -1}, WorkingPrecision → 30, AccuracyGoal → 30]
```

```
{x → 0.618033}
```

```
{x → 1.}
```

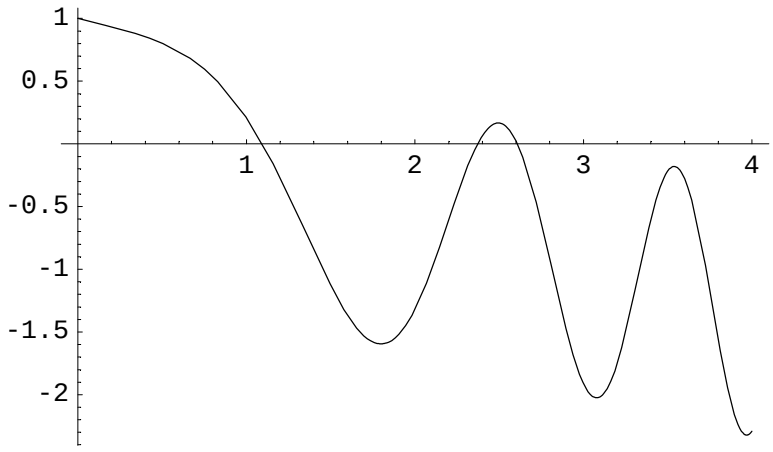
```
{x → -1.61803398874989484820458683437}
```

■ Beispiel: Eine transzendente Gleichung

```
f[x_] := Cos[x^2] - x/3;
```

Schaubild zur Lokalisierung reeller Nullstellen:

```
Plot[f[x], {x, 0, 4}];
```




```
glg = f[x] == 0;
```

Die Startwerte können nun so gewählt werden, dass *Mathematica* die im Schaubild gesichteten reellen Lösungen findet:

```
FindRoot[glg, {x, 1}]
```

```
{x → 1.09427}
```

```
FindRoot[glg, {x, 2}]
```

```
{x → 2.3715}
```

Durch den Startwert 3 wird keine neue Nullstelle gefunden:

```
FindRoot[glg, {x, 3}]
```

```
{x → 2.3715}
```

Man braucht einen Startwert zwischen den Nullstellen:

```
FindRoot[glg, {x, 2.4, 3}]
```

```
{x → 2.60774}
```

Operationen mit Listen

```
Remove["Global`*"]
```

■ Erweiterte Zugriffsverfahren

Eine Beispiel-Liste

(inhaltlich sinnlos, aber zweckmäßig für die folgende Demonstration):

```
mliste = {0, -1, "dummy", Sqrt[5], Pi, 2 / 3, -1, {1, 2, 3}, {}}
```

$$\left\{0, -1, \text{dummy}, \sqrt{5}, \pi, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}\right\}$$

Die ersten 3 Elemente:

```
Take[mliste, 3]
```

```
{0, -1, dummy}
```

Die letzten 3 Elemente:

```
Take[mliste, -3]
```

```
{-1, {1, 2, 3}, {}}
```

Erstes und letztes Element:

```
First[mliste]
Last[mliste]
```

```
0
```

```
{}
```

Elemente nach "Eigenschaften" auswählen (reine Funktionen mit Wahrheitswert; siehe weiter unten).

$\sqrt{5}$ und π sind (ohne numerische Auswertung) Symbole und werden nicht als Zahlen erkannt:

```
Select[mliste, NumberQ]
```

$$\left\{0, -1, \frac{2}{3}, -1\right\}$$

```
Select[mliste, IntegerQ]
```

```
{0, -1, -1}
```

```
Select[mliste, ListQ]
```

```
{{1, 2, 3}, {}}
```

■ Zugehörigkeit und Position

Ist -1 Element der Liste mliste?

```
MemberQ[mliste, -1]
```

```
True
```

An welcher Position kommt das Element -1 vor?

Mathematica gibt eine Liste aus, die die Positionen in Listenform enthält:

```
Position[mliste, -1]
```

```
{{2}, {7}}
```

■ Anhängen eines Elements

Element a anhängen:

```
AppendTo[mliste, a]
```

$$\left\{0, -1, \text{dummy}, \sqrt{5}, \pi, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}, a\right\}$$

mliste enthält das neue Element a:

```
mliste
```

$$\left\{0, -1, \text{dummy}, \sqrt{5}, \pi, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}, a\right\}$$

Der folgende Befehl lässt mliste unverändert:

```
mliste2 = Append[mliste, 704]
```

$$\left\{0, -1, \text{dummy}, \sqrt{5}, \pi, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}, a, 704\right\}$$

```
mliste
```

$$\left\{0, -1, \text{dummy}, \sqrt{5}, \pi, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}, a\right\}$$

■ Element am Listenanfang einfügen

```
PrependTo[mliste, 13]
```

$$\left\{13, 0, -1, \text{dummy}, \sqrt{5}, \pi, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}, a\right\}$$

Alternativ (ohne Änderungen von mlist):

```
Prepend[mliste, b]
```

$$\{b, 13, 0, -1, \text{dummy}, \sqrt{5}, \pi, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}, a\}$$

```
mliste
```

$$\{13, 0, -1, \text{dummy}, \sqrt{5}, \pi, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}, a\}$$

■ Einfügen und Löschen

Element 1000 an Position 7 einfügen:

```
Insert[mliste, 1000, 7]
```

$$\{13, 0, -1, \text{dummy}, \sqrt{5}, \pi, 1000, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}, a\}$$

Das eingefügte Element befindet sich **nur** in der von Insert **ausgegebenen** Liste, aber **nicht** in der als Parameter übergebenen Liste mliste:

```
mliste
```

$$\{13, 0, -1, \text{dummy}, \sqrt{5}, \pi, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}, a\}$$

Das Element an Position 5 von mliste löschen (und das Ergebnis dieser Operation in neueliste speichern):

```
neueliste = Delete[mliste, 5]
```

$$\{13, 0, -1, \text{dummy}, \pi, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}, a\}$$

Zum Vergleich die alte und die neue Liste:

```
mliste
neueliste
```

$$\{13, 0, -1, \text{dummy}, \sqrt{5}, \pi, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}, a\}$$

$$\{13, 0, -1, \text{dummy}, \pi, \frac{2}{3}, -1, \{1, 2, 3\}, \{\}, a\}$$

■ Listen als Mengen

Tabula rasa ...

```
Remove["Global`*"]
```

Eine Liste mit mehrfachen Element-Vorkommen:

```
aliste = {5, 2, 4, 6, 5, 2, 3, 4, 4, 1};
```

Die Menge der Elemente von aliste:

```
amenge = Union[aliste]
```

```
{1, 2, 3, 4, 5, 6}
```

■ Einebnen einer Liste

Eine Liste, der Elemente z.T. Listen sind, usw.

```
mliste = {1, {1, 4}, {3, {1, 2}}};
```

Einebnen heißt: Die Objekte auf der untersten Ebene sammeln

```
Flatten[mliste]
```

```
{1, 1, 4, 3, 1, 2}
```

■ Vereinigung

```
amenge
```

```
{1, 2, 3, 4, 5, 6}
```

```
bmenge = Table[k, {k, 4, 10}]
```

```
{4, 5, 6, 7, 8, 9, 10}
```

Vereinigung als Liste!

```
cmenge = Join[amenge, bmenge]
```

```
{1, 2, 3, 4, 5, 6, 4, 5, 6, 7, 8, 9, 10}
```

■ Durchschnitt

```
Intersection[amenge, bmenge]
```

```
{4, 5, 6}
```

```
Intersection[bmenge, cmenge]
```

```
{4, 5, 6, 7, 8, 9, 10}
```

■ Komplement

```
Complement[bmenge, amenge]
```

```
{7, 8, 9, 10}
```

■ Kartesisches Produkt

Zwei Ausgangslisten:

```
aliste = {a, b, c};
bliste = {1, 2, 3, 4, 5};
```

```
akreuzb = Outer[List, aliste, bliste]
```

```
{{{a, 1}, {a, 2}, {a, 3}, {a, 4}, {a, 5}},
 {{b, 1}, {b, 2}, {b, 3}, {b, 4}, {b, 5}},
 {{c, 1}, {c, 2}, {c, 3}, {c, 4}, {c, 5}}}
```

Achtung: Es handelt sich nicht um die Menge aller Paare!

```
Length[akreuzb]
```

```
3
```

Erst die folgende Operation (Einebnen in der 1. Stufe) liefert das übliche kartesische Mengenprodukt:

```
Flatten[akreuzb, 1]
```

```
{{a, 1}, {a, 2}, {a, 3}, {a, 4}, {a, 5}, {b, 1}, {b, 2},
 {b, 3}, {b, 4}, {b, 5}, {c, 1}, {c, 2}, {c, 3}, {c, 4}, {c, 5}}
```

Schleifen

```
Remove["Global`*"]
```

■ Do-Schleife

Wiederholung einer Operation mittels **Do** (nach dem Vorbild von Table, Sum etc.):

```
Do[a[i] = i^2, {i, 1, 10, 2}]
```

Das Ergebnis sichbar machen (der Befehl **Array[a,n]** erzeugt eine Liste von *n* Elementen der Form **a[i]**):

```
Array[a, 10]
```

```
{1, a[2], 9, a[4], 25, a[6], 49, a[8], 81, a[10]}
```

■ For-Schleife

Ein (aus der herkömmlichen Programmierung bekannter) Schleifentyp, bei dem die Anzahl der Schleifendurchläufe (wie bei der Do-Schleife) vorgegeben ist.

Bem.: Die For-Schleife ist entbehrlich.

```
For[i = 1, i ≤ 5, i++, Print[a[i]]]
```

```
1
a[2]
9
a[4]
25
```

■ While-Schleife

`While[bedingung, aktion]` bedeutet:

Solange ***bedingung*** wahr ist, führe ***aktion*** aus.

```
i = 1;
While[Mod[i, 5] ≠ 0, (i = i + 3; Print[i])];
```

```
4
7
10
```

N.B.: Wenn bei Eintritt in die Schleife die Bedingung nicht wahr ist, wird die Schleife nicht durchlaufen (sog. abweisende Schleife).

Man setze etwa im obigen Beispiel $i = 5$.

Definition von Funktionen

```
Remove["Global`*"]
```

■ Reine Funktionen

Eine Funktion wie $f(x) = 5\,x^2 - x + 1$ lässt sich wie folgt definieren:

```
f[x_] := 5 x^2 - x + 1
```

```
f[a]
```

```
1 - a + 5 a^2
```

Eine alternative und häufig benutzte Definitionsform ist die einer sog. reinen Funktion.

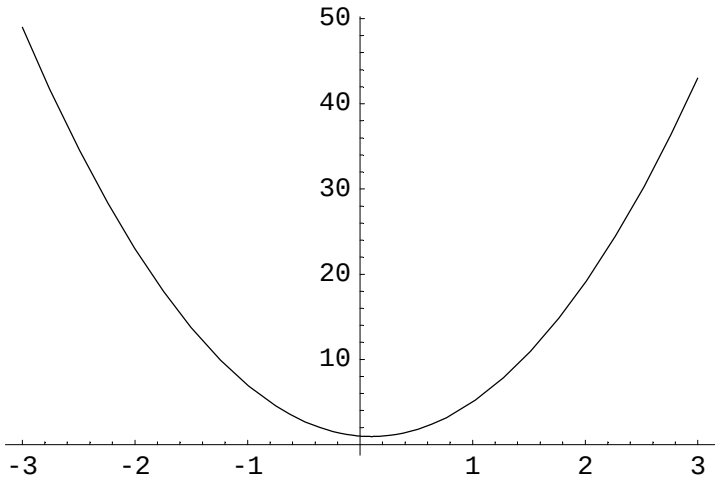
Idee dazu: f = die Funktion, welche x den Wert $5x^2 - x + 1$ zuordnet

```
g = Function[x, 5 x^2 - x + 1]
```

```
Function[x, 5 x^2 - x + 1]
```

g kann in der üblichen Weise benutzt werden:

```
Plot[g[u], {u, -3, 3}];
```



Eine andere Form der Definition reiner Funktionen

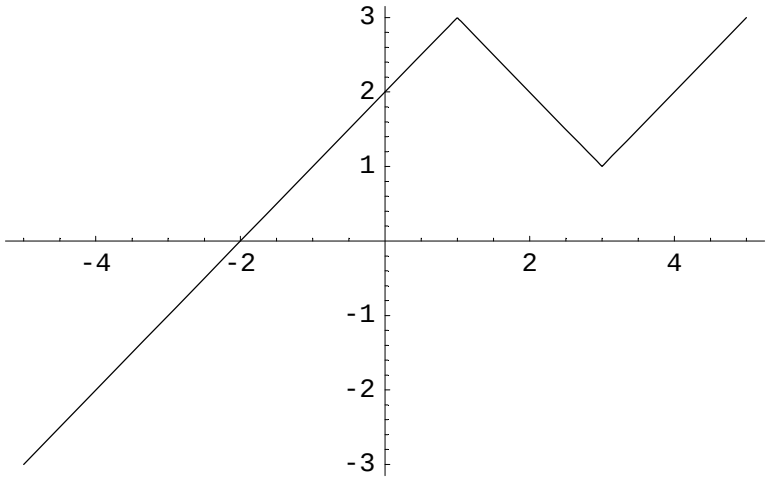
(# ist der Platzhalter für das Argument, & markiert das Ende der Definition):

```
h = # + Abs[3 - #] - Abs[1 - #] &
```

```
#1 + Abs[3 - #1] - Abs[1 - #1] &
```

Abs[.] ist der Absolutbetrag.

```
Plot[h[x], {x, -5, 5}];
```



■ Kompilierte Funktionen

Kompilation bedeutet: Übersetzung in maschinennahen Code. Sie dient der Beschleunigung von Berechnungen.

Kompilierte Funktionen werden wie reine Funktionen definiert (mittels Compile statt Function):

```
fc = Compile[x, x^3 Cos[x]];
f = Function[x, x^3 Cos[x]];
```

Vergleich der Rechenzeit:

```
f[4.8] // Timing
fc[4.8] // Timing
```

```
{0. Second, 9.67669}
```

```
{0. Second, 9.67669}
```

```
Clear[f, fc, g]
```

Unterschiede zeigen sich erst, wenn zahlreiche Operationen wiederholt ausgeführt werden sollen:

Beispiel einer unkompilierten Funktion:

```
g[m_] := Sum[Sum[Mod[i, k], {i, 1, m}], {k, 1, m}]
```

```
g[1000] // Timing
```

```
{15.82 Second, 225771449}
```

Das zugehörige Kompilat erweist sich als deutlich schneller:

```
gc = Compile[n, Sum[Sum[Mod[i, k], {i, 1, n}], {k, 1, n}]];
```

```
gc[1000] // Timing
```

```
{1.15 Second, 225771449}
```

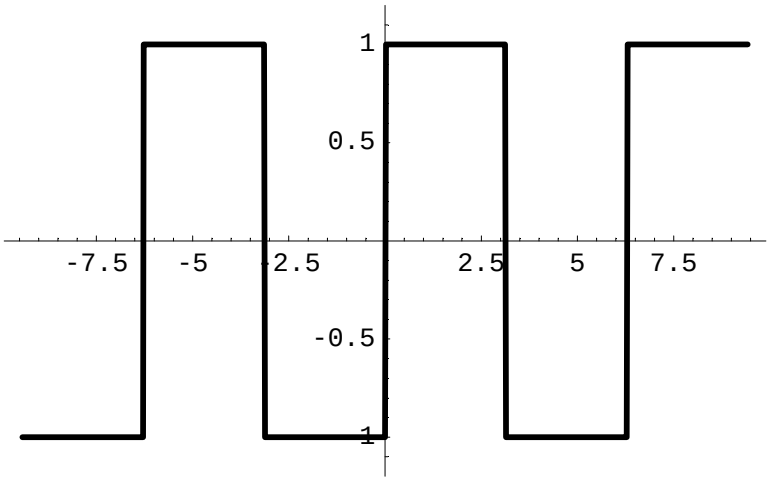
■ Bedingte Definitionen

Fallunterscheidung mittels **If[_,_,_]**

(geeignet nur für 2 Fälle):

```
f[x_] := If[Sin[x] > 0, 1, -1]
```

```
Plot[f[x], {x, -3 Pi, 3 Pi}, PlotRange → {-1.2, 1.2}, PlotStyle → {Thickness[0.008]}];
```



Fallunterscheidung mittels Which

(Vorteil: mehr als 2 Fälle möglich):

```
vorz[x_] := Which[
   x < 0, -1,
   x == 0, 0,
   x > 0, 1];
```

```
vorz[-7]
```

```
-1
```

Eine flexible und bequeme Form der bedingten Definition durch den Operator **/;**

```
Clear[f];
f[n_] := 0 /; n < 0;
f[n_] := 1 /; Not[IntegerQ[n]];
f[n_] := 2 /; EvenQ[n];
f[n_] := Sqrt[n]
```

```
{f[-3], f[Sqrt[5]], f[14], f[7]}
```

$$\{0, 1, 2, \sqrt{7}\}$$

■ Rekursive Definitionen

Rekursive Definition der Fibonacci-Folge:

```
Clear[f]
f[n_] := f[n - 1] + f[n - 2]
f[0] = f[1] = 1;
```

```
Table[f[i], {i, 10}]
```

```
{1, 2, 3, 5, 8, 13, 21, 34, 55, 89}
```

```
f[25] // Timing
```

```
{3.63 Second, 121393}
```

Neudefinition als Funktion "mit Gedächtnis":

```
Clear[f]
f[n_] := f[n] = f[n - 1] + f[n - 2]
f[0] = f[1] = 1;
```

```
f[25] // Timing
```

```
{0. Second, 121393}
```

```
$RecursionLimit = 2000;
f[1000] // Timing

(* Berechnung erfordert Löschen von f und Neudefinition *)
```

```
{0.11 Second,
 70330367711422815821835254877183549770181269836358732742604905087154533
  71181969335797422494945626117334877504492417659910881863632654502236433
  71060120533741212738673391111981393731255987676900919022452453234035033
  1}
```

■ Module

Bei vielen Berechnungen genügt es nicht, einen Formelausdruck auszuwerten; oft sind mehrere (von Bedingungen abhängige) Rechenschritte nacheinander auszuführen.

In *Mathematica* lassen sich dynamische Berechnungen dieser Art durch das **`Module[.]`**-Konstrukt wiedergeben.

Beispiel: Zentralwert einer Stichprobe

```
s = {2, 4, 4, 3, 1, 5, 4, 3, 2, 5, 6, 2, 3};
```

Die Liste wird nach Rangordnung (hier: Größe) sortiert:

```
sord = Sort[s]
```

```
{1, 2, 2, 2, 3, 3, 3, 4, 4, 4, 5, 5, 6}
```

Länge ermitteln:

```
slen = Length[s]
```

```
13
```

Element "in der Mitte" (Position 7) suchen (Zentralwert oder Median):

```
Part[sord, Quotient[sl, 2] + 1]
```

```
3
```

Diese Rechenschritte sollen a) zusammengefasst und b) allgemein dargestellt werden:

```
Zentralwert[s_] := Module[{sord, slen, mpos, zwert},
  sord = Sort[s];
  slen = Length[s];
  mpos = Quotient[slen, 2];
  zwert = If[OddQ[slen], sord[[mpos + 1]], (sord[[mpos]] + sord[[mpos + 1]]) / 2];
  zwert
 ]
```

Die Probe auf's Exempel:

```
Zentralwert[s]
```

```
3
```

Eine Stichprobe von gerader Anzahl:

```
Zentralwert[{2, 1, 6, 1, 3, 3}]
```

$$\frac{5}{2}$$

Wichtige Bemerkung:

Innerhalb von Module erklärte Synmbole (Variable) haben nur lokale Gültigkeit (d.h. sind außerhalb des Moduls nicht bekannt).

Packages

```
Remove["Global`*"]
```

■ Vorgefertigte Pakete nutzen

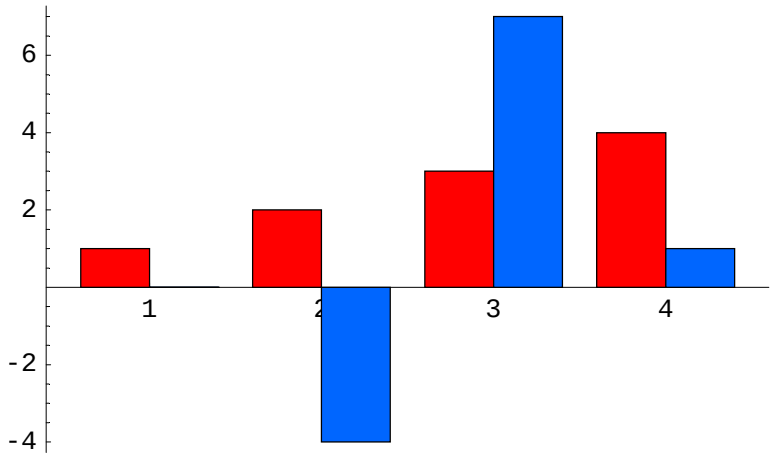
Nicht alle Funktionen von *Mathematica* sind bereits nach dem Start des Kerns verfügbar. Sie lassen sich aber durch Laden sog. Packages nachträglich verfügbar machen.

Ein Beispiel aus dem **Standard-Paket** *Graphics*

Lade-Befehl (**`Needs["...`...`"]`** oder kürzer **`<<...`...`...`**)

```
Needs["Graphics`Graphics`"]
```

```
BarChart[{1, 2, 3, 4}, {0, -4, 7, 1}];
```



■ Eigene Pakete entwickeln

Ein Package ist eine spezielle Datei mit der Endung **.m**

Sie kann automatisch erzeugt werden, wenn man die Befehle (Funktionen), die das Paket enthalten soll, in Initialisierungzellen schreibt.

Als Demonstrationsbeispiel diene ein Mini-Package mit Funktionen zur Mengenalgebra. Es knüpft an den Abschnitt "Operationen mit Listen" an.

Mengenoperator:

```
(* Init *)
Menge[a_List] := Union[a]
```

```
Menge[{9, 9, 4, 4, 3, 2, 0}]
```

```
{0, 2, 3, 4, 9}
```

Ist das Argument keine Liste, bewirkt Menge nichts:

```
Menge[2]
```

```
Menge[2]
```

Vereinigungsmenge:

```
(* Init *)
Vereinigungsmenge[a_List, b_List] := Menge[Join[a, b]]
```

```
Vereinigungsmenge[{1, 4, 2, 5}, {3, 4, 5, 6, 1}]
```

```
{1, 2, 3, 4, 5, 6}
```

Schnittmenge:

```
(* Init *)
Schnittmenge[a_List, b_List] := Menge[Intersection[a, b]]
```

```
Schnittmenge[{1, 4, 2, 5}, {3, 4, 5, 6, 1}]
```

```
{1, 4, 5}
```

Differenzmenge:

```
(* Init *)
Differenzmenge[a_List, b_List] := Menge[Complement[a, b]]
```

Produktmenge:

```
(* Init *)
Produktmenge[a_List, b_List] := Flatten[Outer[List, Menge[a], Menge[b]], 1]
```

```
Produktmenge[{1, 2}, {a, b, c}]
```

```
{{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}}
```

Die mit (* Init *) gekennzeichneten Zellen können nun in eine neue Notebook-Datei **`Mengenalgebra.nb`** kopiert und dort als Initialisierungszellen ausgezeichnet werden. Beim Abspeichern kann man entscheiden, ob das zugehörige Paket **`Mengenalgebra.m`** automatisch erzeugt werden soll.

■ Dokumentation, Bereitstellung

Selbstentwickelte Packages sollten dokumentiert werden. Sie werden dann durch bestimmte technische Maßnahmen bereitgestellt (vgl. dazu die *Mathematica*-Hilfe sowie die angegebene Literatur R. Maeder sowie M.-L. Herrmann).