Rundgang in *Mathematica*

Tour en *Mathematica*

(Nach Ideen aus: Handbuch "*Mathematica*" von S. Wolfram)
(Selon les idées prises dans le manuel "*Mathematica*" de S. Wolfram)

Run mit WIN+*Mathematica* Version 5.2
Testé avec *Mathematica* version 5.2+WIN

9. Programmierung
Programmation

Programm zur Erzeugung einer nxn-Hilbert-Matrix
Programme pour la création d'une matrice nxn de Hilbert

Programm
Programme

```
In[1]:= Hilbert[n_]:=Table[1/(i+j-1),{i,n},{j,n}]
```

Aufruf für eine 3x3-Matrix
Appel pour une matrice 3 x 3

```
In[2]:= Hilbert[3]; MatrixForm[%]

Out[2]//MatrixForm=
        Null
```

Programm zur Erzeugung des charakteristischen Polynoms einer Matrix

Programme pour la création du polynome caractéristique d'une matrice

Programm

Programme

```
In[3]:= CharPoly[m_,x_]:=
        Det[m-x IdentityMatrix[Length[m]]]/; MatrixQ[m]
         (* Check auf Matrix bei m *)
```

Wieso /; ?

Pourquoi /; ?

```
In[5]:= ?/;
```

```
patt /; test is a pattern which matches only if the evaluation of test yields True. lhs :>
   rhs /; test represents a rule which applies only if the evaluation of test yields
   True. lhs := rhs /; test is a definition to be used only if test yields True. Mehr...
```

Anwendung

Application

```
In[6]:= m = {{1,2,3},{2,3,4},{3,4,5}};
        Print[MatrixForm[m]];
        CharPoly[m,x]
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 5 \end{pmatrix}$$

```
Out[8]= 6 x + 9 x^2 - x^3
```

```
In[9]:= m = {1,2,3,2,3,4,3,4,5};
        Print[MatrixForm[m]];
        CharPoly[m,x]
```

$$\begin{pmatrix} 1 \\ 2 \\ 3 \\ 2 \\ 3 \\ 4 \\ 3 \\ 4 \\ 5 \end{pmatrix}$$

```
Out[11]= CharPoly[{1, 2, 3, 2, 3, 4, 3, 4, 5}, x]
```

Programm zur Auffindung der nächsten Primzahl

Programme pour trouver le prochain nombre premier

Programm

Programme

```
In[12]:= NextPrime[n_Integer]:=
         Module[{k=n},
                 While[!PrimeQ[k],k++];
                 Return[k]
               ]
```

Anwendung auf n =111

Application pour n = 111

```
In[13]:= NextPrime[111]

Out[13]= 113
```

Programm zur Berechnung statistischer Grössen, ohne Loops zu verwenden (elegant)

▮ Programme pour calculer des valeurs statistiques, sans utiliser des loops (élégant)

Programm
▮ Programme

In[14]:=
```
mean[list_List]    :=Apply[Plus,list] / Length[list];

variance[list_List]:=
Mean[(list-Mean[list])^2] Length[list]/(Length[list]-1);

quantile[list_List,q_]:=
   Part[Sort[list],-Floor[-q Length[list]]]/; 0<q<1;

Alles[list_, q_]:=Module[{},
        Print["Mean = ",mean[list]," = ",
        Mean[list]];
        Print["Variance = ",variance[list]," = ",
        Variance[list]];
        Print["Quantile = ",quantile[list,q]," = ",
        Quantile[list,q]];
                        ]
```

```
General::spell1 :
 Possible spelling error: new symbol name "mean" is similar to existing symbol "Mean". Mehr…

General::spell1 :
 Possible spelling error: new symbol name "variance" is similar to existing symbol "Variance". Mehr…

General::spell1 :
 Possible spelling error: new symbol name "quantile" is similar to existing symbol "Quantile". Mehr…
```

In[19]:=
```
??Module
```

```
Module[{x, y, ... }, expr] specifies that occurrences
   of the symbols x, y, ...  in expr should be treated as local. Module[
   {x = x0, ... }, expr] defines initial values for x, ... . Mehr…

Attributes[Module] = {HoldAll, Protected}
```

Anwendung
▮ Application

In[20]:=
```
l={1,2,2,3,3,3,4,4,4,5,5,6,7,7,8,9}; q=0.1;
Alles[l,q]
```

$$\text{Mean} = \frac{73}{16} = \frac{73}{16}$$

$$\text{Variance} = \frac{1279}{240} = \frac{1279}{240}$$

$$\text{Quantile} = 2 = 2$$

Programm "Zufallsspaziergang" - was macht dieses Programm?

Programme "promenade au hasard" - que fait ce programme?

Programm
Programme

```
In[22]:= zs[n_Integer]:=
           FoldList[Plus,0,Table[Random[]-1/2,{n}]]
```

Anwendung
Application

```
In[23]:=
         zs[100]

Out[23]= {0, 0.397161, 0.774418, 0.552511, 0.608016, 0.882651, 0.783349, 0.931527,
          1.1167, 1.60128, 1.49378, 1.11713, 0.690229, 0.635201, 0.961627, 1.22891,
          1.14827, 0.72646, 0.716078, 0.572787, 0.196796, -0.170223, -0.245881,
          -0.034506, 0.118632, -0.145548, -0.0984632, -0.165181, -0.567548, -0.606363,
          -0.959977, -0.674872, -0.762409, -0.785804, -0.531922, -0.370169, -0.530801,
          -0.999169, -0.571712, -0.177244, 0.242764, 0.696207, 0.634045, 0.671805,
          0.967804, 1.28827, 0.801763, 1.12815, 0.771008, 0.855649, 0.822062, 0.715164,
          0.260393, -0.116152, -0.296126, -0.188128, -0.055363, 0.0914873, 0.157632,
          0.603876, 0.397274, 0.512492, 0.65118, 0.202956, 0.0763466, 0.238122,
          -0.061029, -0.0470129, 0.0303786, 0.371692, 0.0590442, 0.246676, 0.181206,
          -0.0621206, 0.158818, -0.0466518, -0.15735, -0.524132, -0.62322, -0.436688,
          -0.180151, -0.193784, 0.140984, 0.381272, 0.34441, 0.71556, 0.411639,
          0.600152, 0.1899, -0.100726, 0.394504, 0.0690005, 0.081357, -0.0505823,
          0.257296, 0.24416, -0.178013, -0.566626, -0.979687, -1.28735, -1.09883}
```

Programm, das die ersten n Terme in der Kettenbruchentwicklung einer Zahl findet: elegante funktionale Programmierung

Programme qui trouve les n premiers termes d'un nombre

Programm

Programme

```
In[24]:= ?ContinuedFraction
```

```
ContinuedFraction[x, n] generates a list of the first n terms in
   the continued fraction representation of x. ContinuedFraction[x] generates
   a list of all terms that can be obtained given the precision of x. Mehr...
```

```
In[25]:= ContinuedFraction[h, 30]

Out[25]= ContinuedFraction[h, 30]
```

```
In[26]:= ownContinuedFraction[x_Real,n_Integer]:=
           Floor[NestList[Function[{u},1/(u-Floor[u])],
                          x,n-1]]
```

Anwendung: Goldener Schnitt in Kettenbruchentwicklung

Application: Section d'or en évolution de fractions continues

```
In[27]:=
h=N[(Sqrt[5]-1)/2,100];
Print[h]
ownContinuedFraction[h,30]
```

```
0.6180339887498948482045868343656381177203091798057628621354486227052604628189024497072072041
893911375

Out[29]= {0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}
```

Anwendung: e in Kettenbruchentwicklung

Application: e en évolution de fractions continues

```
In[30]:=
e=N[E,100];
Print[e];
ownContinuedFraction[e,30]
```

```
2.7182818284590452353602874713526624977572470936999595749669676277240766303535475945713821785
25166427

Out[32]= {2, 1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, 1, 1,
          10, 1, 1, 12, 1, 1, 14, 1, 1, 16, 1, 1, 18, 1, 1, 20}
```

```
In[33]:= ContinuedFraction[e, 30]

Out[33]= {2, 1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, 1, 1,
          10, 1, 1, 12, 1, 1, 14, 1, 1, 16, 1, 1, 18, 1, 1, 20}
```

Anwendung: Pi in Kettenbruchentwicklung
▮ Application: Pi en évolution de fractions continues

```
In[34]:=
         p=N[Pi,100];
         Print[p]
         ContinuedFraction[p,30]

        3.14159265358979323846264338327950288419716939937510582097494459230781640628620899862803482534
        2117068

Out[36]= {3, 7, 15, 1, 292, 1, 1, 1, 2, 1, 3, 1, 14,
          2, 1, 1, 2, 2, 2, 2, 1, 84, 2, 1, 1, 15, 3, 13, 1, 4}
```

Programm mit Mix aus symbolischer Konstruktion und mathematischen Operationen: Ableiten der Funktionen einer Liste nach allen Variablen einer andern Liste

▮ Programme avec mélange de construction symbolique et opérations mathématiques: Dériver les fonctions d'une liste d'après toutes les variables d'une autre liste

Programm
▮ Programme

```
In[37]:= Jac[funs_List,vars_List]:=Outer[D, funs, vars]
```

Anwendung
▮ Application

```
In[38]:= f={x^2+x y-1, Cos[y]-z, a x z};
         v={x,y,z};
         Jac[f,v]

Out[40]= {{2 x + y, x, 0}, {0, -Sin[y], -1}, {a z, 0, a x}}
```

Liste von Regeln für Laplace-Transformationen
▮ Liste de règles pour des transformations de

Laplace

Programm (Regeln)
▮ Programme (règles)

```
In[41]:= Laplace[c_,t_,s_]:=c/s/; FreeQ[c,t];
         Laplace[a_+b_,t_,s_]:=
                     Laplace[a,t,s]+Laplace[b,t,s];
         Laplace[c_ a_,t_,s_]:=
                     c Laplace[a,t,s]/; FreeQ[c,t];
         Laplace[t_^n_.,t_,s_]:=
                     n!/s^(n+1)/; (FreeQ[n,t]&&n>0);
         Laplace[a_. Exp[b_.+c_. t_],t_,s_]:=
                 Laplace[a Exp[b],t,s-c]/; FreeQ[{b,c},t];
```

```
In[46]:= ?_.

         p:v is a pattern object which represents an expression
            of the form p, which, if omitted, should be replaced by v. Mehr...
```

```
In[47]:= ?FreeQ

         FreeQ[expr, form] yields True if no subexpression in expr
            matches form, and yields False otherwise. FreeQ[expr, form, levelspec]
            tests only those parts of expr on levels specified by levelspec. Mehr...
```

Anwendung
▮ Application

```
In[48]:= Laplace[E^t+4t^2-2t+1,t,s]
```

$$\text{Out[48]}= \frac{1}{-1+s}+\frac{8}{s^3}-\frac{2}{s^2}+\frac{1}{s}$$

Programm, das pattern matching benutzt um die Anzahl gleicher Elemente einer Liste auszuzählen - kurz, elegant, effizient!
▮ Programme qui utilise le "pattern matching" pour compter le nombre d'éléments égaux d'une liste - bref, élégant, efficace!

Programm
▮ Programme

```
In[49]:= RunEncode[{rest___Integer,same:(n_Integer)..}]:=
             Append[RunEncode[{rest}],{n,Length[{same}]}];
         RunEncode[{}]:={}
```

Anwendung
▮ Application

```
In[51]:= r={1,2,2,2,3,3,3,3,3,3,4,4,5,6,6,6,7,8,9,9,9,10};
         RunEncode[r]

Out[52]= {{1, 1}, {2, 3}, {3, 7}, {4, 2}, {5, 1}, {6, 3}, {7, 1}, {8, 1}, {9, 3}, {10, 1}}
```

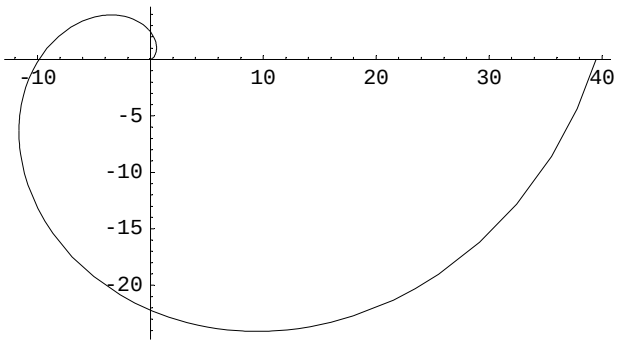
Programm, um Graphik zu erzeugen
▮ Programme pour créer des graphiques

Programm
▮ Programme

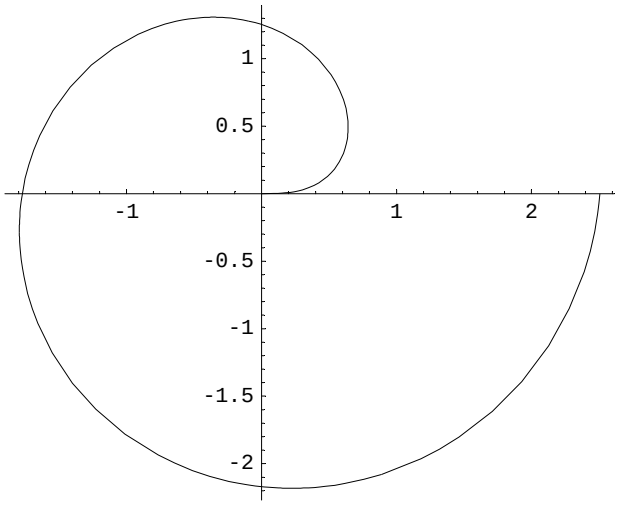
```
In[53]:= PolarPlot[r_,{t_,tmin_,tmax_}]:=
             ParametricPlot[{r Cos[t], r Sin[t]},
                  {t,tmin,tmax},AspectRatio->Automatic]
```

Anwendungen
▮ Applications

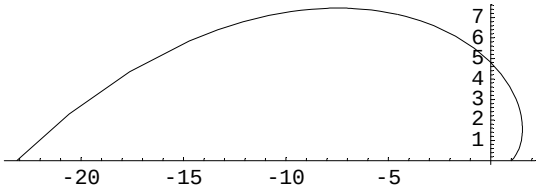
```
In[54]:= PolarPlot[t^2,{t,0,2 Pi}];
```



In[55]:= **PolarPlot[t^(1/2),{t,0,2 Pi}];**



In[56]:= **PolarPlot[E^t,{t,0,Pi}];**



Programm, das die Lösungen einer Polynom-Gleichung als Punkte der komplexen Ebene darstellt

Programme qui représente les solutions d'une équation polynomiale comme points d'un plan complexe

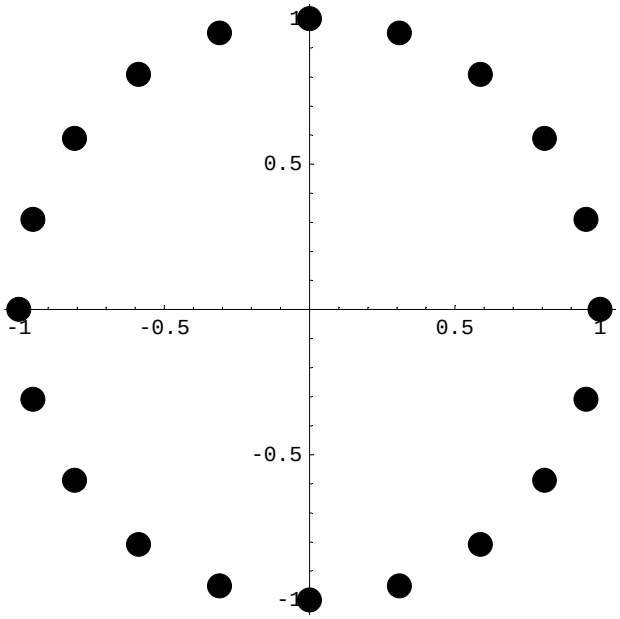
Programm

Programme

In[57]:=
```
RootPlot[poly_,z_]:=
    ListPlot[{Re[z],Im[z]}/. NSolve[poly==0,z],
    AspectRatio->Automatic,
        PlotStyle->{PointSize[0.04]}]/;
                PolynomialQ[poly,z]
```

Anwendung: Einheitswurzeln
Application: Racines unifiées (de l'unité)

In[58]:= **RootPlot[z^20 -1,z];**



Programm, das Zahlen in Matrixform aus einem File liest und als 3D-Graphik darstellt
Programme qui lit des nombres en forme de matrice dans un fichier et les représente en graphique 3D

Programm
Programme

In[59]:=
```
myfile[myfile_String]:=
    ReadList[myfile,Number,RecordLists->True]
```

In[60]:=
```
FilePlot3D[myfile_String]:=
    ListPlot3D[ReadList[myfile,Number,
                        RecordLists->True]]
```

Anwendung
Application

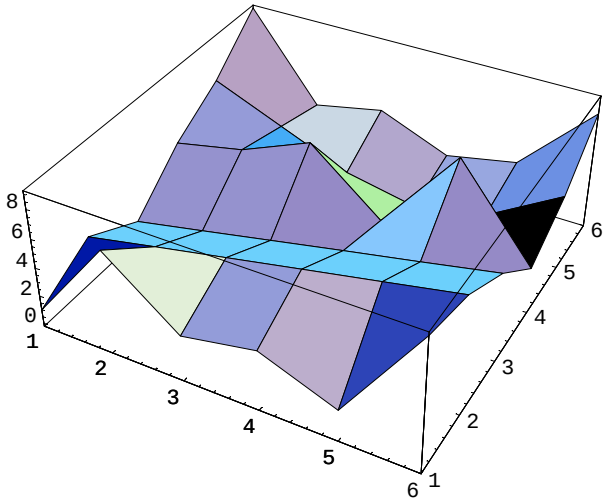
In[61]:= **myFilePath = "C:\work\MathematicaData\AADATEN"**

Out[61]= C:\work\MathematicaData\AADATEN

Hallo!!!! >>> Eigenes Directory/ Verzeichnis/ Dossier setzen!!!!!
● Hallo!!!! >>> Remplacer par le propre directoire!!!!!

In[62]:=
```
Print[myfile[myFilePath]];
FilePlot3D[myFilePath]
```

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



Out[63]= - SurfaceGraphics -

Anhang
▮ Annexe

In[64]:=
```
?RecordLists
```

```
RecordLists is an option for ReadList which specifies whether objects
   from separate records should be returned in separate sublists. Mehr...
```

In[65]:=
```
myFilePath = "C:\work\MathematicaData\AADATEN"
```

Out[65]= C:\work\MathematicaData\AADATEN

Hallo!!!! >>> Eigenes Directory/ Verzeichnis/ Dossier setzen!!!!!
● Hallo!!!! >>> Remplacer par le propre directoire!!!!!

In[66]:=
```
!! C:\work\MathematicaData\AADATEN
```

```
1 7 3 4 2 9
3 4 5 6 7 8
1 2 3 4 5 6
4 5 7 3 9 3
6 4 2 1 2 5
9 3 4 2 3 8
```

In[67]:=
```
<< C:\work\MathematicaData\AADATEN
```

Out[67]= 5184

In[68]:=
```
Get["C:\work\MathematicaData\AADATEN"]
```

Out[68]= 5184

Was ist jetzt passiert? ● Qu'est-ce qui est arrivé?

```
In[69]:= ??<<
```

```
<<name reads in a file, evaluating each expression in it, and returning the last one. Mehr...

Attributes[Get] = {Protected}

Options[Get] = {CharacterEncoding:↦ $CharacterEncoding, Path:↦ $Path}
```

Programm, das externe Daten manipuliert - hier: das Files sucht mit gegebenen Teilstrings

▮ Programme qui manipule des données externes - Ici: le fichier cherche par des stings partiels donnés

Programm

▮ Programme

```
In[70]:= ?Select
```

```
Select[list, crit] picks out all elements ei of list for which crit[ei] is True. Select[
   list, crit, n] picks out the first n elements for which crit[ei] is True. Mehr...
```

```
In[71]:= myNewFilePath = "C:\work\MathematicaData\AADATEN"

Out[71]= C:\work\MathematicaData\AADATEN
```

Hallo!!!! >>> Eigenes Directory/ Verzeichnis/ Dossier setzen!!!!!
● Hallo!!!! >>> Remplacer par le propre directoire!!!!!

```
In[72]:= fn=FileNames[]>> C:\work\MathematicaData\fnxxx;
         fnxxx=Flatten[ReadList[myNewFilePath,
                         Expression,RecordLists->True,
                         WordSeparators->{" "}]];
         Print[fnxxx];
         Print[fnxxx[[1]]];
         Print[fnxxx[[3]]];
         Clear[Where];
         Where[s_String_]:=
            Select[fnxxx,(Length[FindList[#,s,1]]>0)&]
```

```
{1512, 20160, 720, 11340, 480, 5184}

1512

720
```

Anwendung: Suche nach *e* im momentanen directory
▮ Application: Cherche *e* dans le directoire momentané

In[79]:= **Where["e"]**

Out[79]= Where[e]

Wo liegt also das Problem hier? ▮ Où est le problème ici?

"Putzmaschine" einsetzen
▮ Employer la "machine de nettoyage"

In[80]:= **(* Old Form: Remove["Global`@*"] *)**

In[81]:= **Remove["Global`*"]**