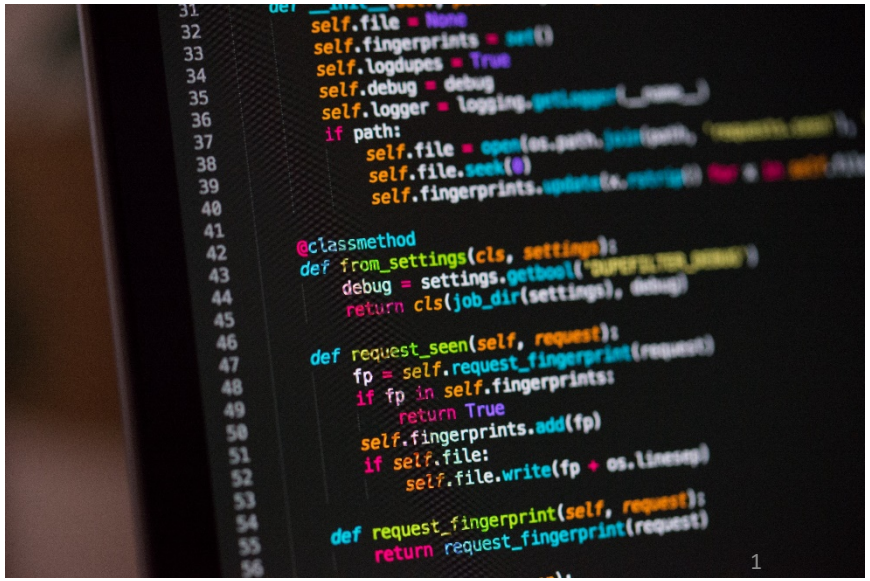
Einführung in Python

Prof. Dr.-Ing. Prof. h.c. Detlef Jensen
Dekan Fachbereich Technik

Tel.: +49 (0) 481 / 85 55 - 300
mobil: +49(0)177 / 87 31 62 1
Fax: +49 (0) 481 / 85 55 - 301
Email: jensen@fh-westkueste.de



FH Westküste
Wirtschaft und Technik

Funktionen und Module

Wozu Funktionen?

FH Westküste
Wirtschaft und Technik

Wo sie schon verwendet wurden:

- Mathematische Funktionen: sqrt(s), sin(s), exp(x), ...
- Methoden von Sequenzen: s.islower(), l.append(x), ...
- Erzeugung von Listen: range(x)
- Typ und Identität: type(x), id(x)
- Alle bisherigen haben keinen oder einen Übergabewert und einen Rückgabewert

Vorteile

- Zerlegung eines großen Problems in viele kleine
- Zusammenfassung von Anweisungsblöcken
- Klare Abgrenzung von Funktionalitäten
- Vermeidung von Code-Duplikation
- Höhere Code-Lesbarkeit (Wenn man statt sin(x) jedesmal den kompletten Code ...)
- ...

Syntax von Funktionen

FH Westküste
Wirtschaft und Technik

- **Funktionen werden definiert mit dem Schlüsselwort def**
- **Nach Funktionsname und Parameterliste kommt ein Doppelpunkt**
- **Der darauf folgende Block (Einrückung!) wird bei Aufruf der Funktion ausgeführt**
- **Ausführung bis zum Ende des Blocks oder bis zu einen return**
- **Rückgabewert kann beliebiger Typ sein (Zahl, Liste, String, ...)**
- **Ohne return oder ohne Wert wird None zurückgegeben**

```
>>> def hi ():
. . .    print " Hello World !„
>>> hi ()
Hello World
>>> a = hi ()
Hello World
>>> type (a)
< class ’ NoneType ’>
```

Funktionen sind Objekte

FH Westküste
Wirtschaft und Technik

```
>>> def faculty (n ):
. . .   fac = 1
. . .   for i in range (2 , n +1):
. . .           fac = fac *I
. . .   return fac
>>>
>>> faculty (3)
6
>>> f = faculty
>>> f (4)
24
>>> type (f)
< class ’ function ’>
```

Funktionsparameter

FH Westküste
Wirtschaft und Technik

- **Parameter stehen in runden Klammern hinter dem Funktionsnamen**
- **Formalparameter: Bei der Definition der Funktion**
- **Aktualparameter: Folge von Ausdrücken beim Aufruf der Funktion**
- **Beliebig viele Parameter möglich, normalerweise gleich viele Formal- und Aktualparameter**

```
>>> def potenziere ( basis , exponent ):
. . .    return basis ** exponent
>>>
>>> potenziere (2 ,10)
1024
```

FH Westküste
Wirtschaft und Technik

Datei Ein- und Ausgabe

Dateien öffnen

FH Westküste
Wirtschaft und Technik

- **Textuelles Datei-Objekt erstellen**

```
datei = open (" testfile.txt ")          # read
datei = open (" testfile.txt " , 'r ')   # read
datei = open (" testfile.txt " , 'w ')   # write
datei = open (" testfile.txt " , 'a ')   # append
```

- **Binäres Datei-Objekt erstellen**

```
datei = open (" testfile.txt " , 'rb ')  # read
datei = open (" testfile.txt " , 'wb ')  # write
datei = open (" testfile.txt " , 'ab ')  # append
```

- **open Hat noch weitere Optionen (Codierung, Behandlung von newlines, . . .)**

Methoden von Datei-Objekten

FH Westküste
Wirtschaft und Technik

- **Lesen**

```
datei.read ()                    # komplett einlesen
datei.read (n)                   # n Byte einlesen
datei.readline ()                # eine Zeile lesen
datei.readlines ()               # Liste von Zeilen
```

- **Schreiben**

```
datei.write (" Neuer Text \n")
datei.writelines ([ " Erste Zeile \n" , " Zweite Zeile \n" ])
```

- **Nicht vergessen: newlines (\n)**
- **Datei schließen**

```
datei.close ()
```

Textdateien Zeilenweise bearbeiten

FH Westküste
Wirtschaft und Technik

for Schleife über die Zeilen der Datei

```
datei = open (" fulltext.txt ")
for line in datei :                          # Alt .: in datei.readlines ()
        print (line)
# oder :
while True :
        line = f.readline ()
        if not line :
                break
        print (line)
```

Weitere Methoden von Dateiobjekten

- datei.tell() – aktuelle Position ausgeben
- datei.seek(pos) – Zur gegebenen Position gehen
- datei.flush() – Ausgabepuffer leeren (Pufferung wegen Effizienz)

FH Westküste
Wirtschaft und Technik

Wissenschaftliches Rechnen in Python

Nochmal Modul math

FH Westküste
Wirtschaft und Technik

- **Konstanten pi und e**
- **Funktionen für int und float**
- **Alle Rückgabewerte sind float**

```
ceil (x)
floor (x)
exp (x)
fabs (x)                    # wie abs () , nur immer float
ldexp (x , i)               # x * 2** i
log (x [, base ])
log10 (x)                   # == log (x , 10)
modf (x)                    # ( Nachkommateil , Integerteil )
pow (x , y) # x ** y
sqrt (x)
```

Nochmal Modul math

FH Westküste
Wirtschaft und Technik

- **Trigonometrische Funktionen (Bogenmaß)**

```
cos (x ); cosh (x ); acos (x)
sin (x ); . . .
tan (x ); . . .



degrees (x)                    # rad -> deg
radians (x)                    # deg -> rad
```

- **inf/nan**

```
float (" inf ")
float (" - inf ")
float (" nan ")
isinf (x)                      # Überprüfung auf Unendlichkeit
isnan (x)                      # Überprüfung auf NaN
```

- **Complexe Zahlen: cmath**

Und RICHTIGE Mathematik?

FH Westküste
Wirtschaft und Technik

Bisherige Sequenztypen (list, tuple, . . .)

- **Können als Arrays verwendet werden**
- **Können verschiedene beliebige Typen enthalten**
  - Sehr flexibel, aber auch langsam
  - Schleifen sind nicht sehr effizient
- **Vektoren/Matrizen und Operationen auf diesen sind extrem umständlich**
- **Für effizientes wissenschaftliches Rechnen sind andere Datentypen und Methoden nötig**

Module

- **NumPy**
- **Matplotlib**
- **SciPy**

Modul numpy

FH Westküste
Wirtschaft und Technik

Homogene Arrays

- **NumPy stellt beliebig-dimensionale Arrays bereit**
- **Alle Elemente des Arrays haben den gleichen Typ**
- **Beispiel:**

```
from numpy import *
a = array ([[1 ,2 ,3] ,[4 ,5 ,6]])
print (a)
type (a)
a.shape
print (a [0 ,2])
a [0 ,2] = -1
b = a *2
print (b)
```

Homogene Arrays (2)

FH Westküste
Wirtschaft und Technik

- **Arrays können aus (verschachtelten) Sequenztypen erstellt werden**
- **Direkter Zugriff mit []**

```
a = array ([1 ,2 ,3 ,4 ,5 ,6 ,7 ,8])
a [1]
a = array ([[1 ,2 ,3 ,4] ,[5 ,6 ,7 ,8]])
a [1 ,1]
a = array ([[[1 ,2] ,[3 ,4]] ,[[5 ,6] ,[7 ,8]]])
a [1 ,1 ,1]
```

- **Eigenschaften von Arrays**

```
a.ndim                  # Anzahl Dimensionen
a.shape                 # Anzahl Einträge in jeder Dimension
a.size                  # Anzahl Elemente
a.dtype                 # Datentyp der Elemente
a.itemsize              # Nötige Anzahl Bytes für den Datentyp
```

Untermodule

FH Westküste
Wirtschaft und Technik

Modul numpy.random

- **Zufallszahlen aus vielen unterschiedlichen Verteilungen**
- **Mächtiger als das Standard-Modul random**
- **Arbeitet auf Arrays und gibt Arrays zurück**

```
from numpy . random import *
binomial (10 , 0.5)                 # 10 Versuche , Erfolg 50%
binomial (10 , 0.5 , 15)            # 15 Mal voriges
binomial (10 , 0.5 , (3 ,4))        # (3 x4 ) Array
randint (0 , 10 , 15)               # [0 ,10) , int

rand ()                             # [0 ,1) , float
rand (3 ,4)                         #(3 x4 ) Array
```

Untermodule (2)

FH Westküste
Wirtschaft und Technik

Modul numpy.linalg

```
>>> from numpy . linalg import *
>>> a = arange (12); a. resize ((3 ,4))
>>> norm (a ); norm (x)          # Matrix - und Vektornorm
>>> inv (a)                      # Inverse der Matrix
>>> solve (a , b)                # LAPACK LU Zerlegung
>>> det (a)                      # Determinante
>>> eig (a)                      # Eigenwerte und Eigenvektoren
>>> cholesky (a)                 # Cholesky - Zerlegung
```

Modul numpy.fft

- **Fourier-Transformation**

Und weitere . . .

FH Westküste
Wirtschaft und Technik

Matplotlib

Matplotlib

FH Westküste
Wirtschaft und Technik

Wozu Matplotlib?

- **Objektorientierte Bibliothek für zweidimensionale plots**
- **Entworfen mit den Ziel einer ähnlichen Funktionalität wie Matlab plots**
- **Insbesondere zum Darstellen wissenschaftlicher Daten, baut auf numpy-Datenstrukturen auf**

Version-Durcheinander

- **Numpy, scipy, ipython und matplotlib werden oft gleichzeitig benutzt**
- **Die Pakete hängen voneinander ab (matplotlib benutzt z.B. numpy-Arrays**
- **Pylab ist ein (inoffizielles) Paket, das alle vier enthält**
- **Je nach Betriebssystem(sversion) müssen evtl. unterschiedliche Pakete installiert werden (D.h. der Modulname im import-Befehl kann auch unterschiedlich sein)**

Möglichkeiten zum Importieren

FH Westküste
Wirtschaft und Technik

- **python interactive**

```
import scipy, matplotlib.pylab
x = scipy.randn (10000)
matplotlib.pylab.hist (x , 100)
```

```
import numpy.random , matplotlib.pylab
x = numpy.random.randn (10000)
matplotlib.pylab.hist (x , 100)
```

Beispiel - Erster Plot

FH Westküste
Wirtschaft und Technik

- http://matplotlib.sourceforge.net/users/screenshots.html

```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np

# Data for plotting
t = np.arange(0.0, 2.0, 0.01)
s = 1 + np.sin(2 * np.pi * t)

fig, ax = plt.subplots()
ax.plot(t, s)

ax.set(xlabel='time (s)', ylabel='voltage (mV)',
       title='About as simple as it gets, folks')
ax.grid()

fig.savefig("test.png")
plt.show()
```

Beispiel - Histogramm

FH Westküste
Wirtschaft und Technik

```
import matplotlib
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(19680801)

# example data
mu = 100  # mean of distribution
sigma = 15  # standard deviation of distribution
x = mu + sigma * np.random.randn(437)

num_bins = 50

fig, ax = plt.subplots()

# the histogram of the data
n, bins, patches = ax.hist(x, num_bins, density=1)

# add a 'best fit' line
y = ((1 / (np.sqrt(2 * np.pi) * sigma)) *
     np.exp(-0.5 * (1 / sigma * (bins - mu))**2))
ax.plot(bins, y, '--')
ax.set_xlabel('Smarts')
ax.set_ylabel('Probability density')
ax.set_title(r'Histogram of IQ: $\mu=100$, $\sigma=15$')

# Tweak spacing to prevent clipping of ylabel
fig.tight_layout()
plt.show()
```

Date handling

FH Westküste
Wirtschaft und Technik

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import matplotlib.cbook as cbook

years = mdates.YearLocator()   # every year
months = mdates.MonthLocator()  # every month
years_fmt = mdates.DateFormatter('%Y')

# Load a numpy structured array from yahoo csv data with fields date, open,
# close, volume, adj_close from the mpl-data/example directory.  This array
# stores the date as an np.datetime64 with a day unit ('D') in the 'date'
# column.
with cbook.get_sample_data('goog.npz') as datafile:
    data = np.load(datafile)['price_data‘]
fig, ax = plt.subplots()
ax.plot('date', 'adj_close', data=data)
```

Date handling

FH Westküste
Wirtschaft und Technik

```
# format the ticks
ax.xaxis.set_major_locator(years)
ax.xaxis.set_major_formatter(years_fmt)
ax.xaxis.set_minor_locator(months)

# round to nearest years.
datemin = np.datetime64(data['date'][0], 'Y')
datemax = np.datetime64(data['date'][-1], 'Y') + np.timedelta64(1, 'Y')
ax.set_xlim(datemin, datemax)

# format the coords message box
ax.format_xdata = mdates.DateFormatter('%Y-%m-%d')
ax.format_ydata = lambda x: '$%1.2f' % x  # format the price.
ax.grid(True)

# rotates and right aligns the x labels, and moves the bottom of the
# axes up to make room for them
fig.autofmt_xdate()

plt.show()
```

Mehr als NumPy?

FH Westküste
Wirtschaft und Technik

- **SciPy hängt von NumPy ab**
- **SciPy arbeiten mit NumPy arrays**
- **Stellt Funktionalität für Mathematik, Naturwissenschaft und Ingeneursanwendungen zur Verfügung**
- **Noch in Entwicklung**
- **Bei NumPy geht es im Wesentlichen um (N-dimensionale) Arrays**
- **SciPy stellt eine große Zahl an Werkzeugen zur Verfügung, die diese Arrays verwenden**
- **SciPy beinhaltet die NumPy Funktionalität (Nur ein import nötig)**
- **Es gibt noch einige weitere Bibliothen für wissenschaftliches Rechnen (teils aufbauend auf NumPy und SciPy)**
- **Hier daher nur eine kurze Übersicht**
- **Weitere Informationen auf www.scipy.org**

SciPy - Unterpakete

FH Westküste
Wirtschaft und Technik

cluster	Clustering algorithms
constants	Physical and mathematical constants
fftpack	Fast Fourier Transform routines
integrate	Integration and ordinary differential equation solvers
interpolate	Interpolation and smoothing splines
io	Input and Output
linalg	Linear algebra
ndimage	N-dimensional image processing
odr	Orthogonal distance regression
optimize	Optimization and root-finding routines
signal	Signal processing
sparse	Sparse matrices and associated routines
spatial	Spatial data structures and algorithms
special	Special functions
stats	Statistical distributions and functions

Spezielle Funktionen

FH Westküste
Wirtschaft und Technik

- **Airy-Funktion**
- **Elliptische Integralfunktion**
- **Bessel-Funktionen (+ Nullstellen, Integrale, Ableitungen,...)**
- **Struve-Funktion**
- **Jede Menge statistische Funktionen**
- **Gamma-Funktion**
- **Hypergeometrische Funktionen**
- **Orthogonale Polynome (Legendre, Chebyshev, Jacobi,...)**
- **Parabolische Zylinderfunktion**
- **Mathieu-Funktion**
- **Kelvin-Funktion**
- **...**

Beispiel: Interpolation - Linear

FH Westküste
Wirtschaft und Technik

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import interpolate

x = np.linspace(0, 10, num=11, endpoint=True)
y = np.cos(-x**2/9.0)
f = interpolate.interp1d(x, y)
f2 = interpolate.interp1d(x, y, kind='cubic')

xnew = np.linspace(0, 10, num=41, endpoint=True)

plt.plot(x, y, 'o', xnew, f(xnew), '-', xnew, f2(xnew), '--')
plt.legend(['data', 'linear', 'cubic'], loc='best')
plt.show()
```

Beispiel: Interpolation - Kubische Splines

FH Westküste
Wirtschaft und Technik

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import interpolate

x = np.arange(0, 2*np.pi+np.pi/4, 2*np.pi/8)
y = np.sin (x)
spline = interpolate.splrep (x ,y ,s =0)
xnew = np.arange(0, 2*np.pi, np.pi/50)
ynew = interpolate.splev ( xnew , spline )

plt.figure()
plt.plot(x, y, 'x', xnew, ynew, xnew, np.sin(xnew), x, y, 'b')
plt.legend(['Linear', 'Cubic Spline', 'True'])
plt.axis ([ -0.05 ,6.33 , -1.05 ,1.05])
plt.title('Cubic-spline interpolation')
plt.show ()
```

Datenanalyse mit Python

FH Westküste
Wirtschaft und Technik

- **Es existieren verschiedene Bibliotheken:**
  - Pandas
  - PySpark
  - SciKit-Learn

Zugriff auf Finanzdaten

FH Westküste
Wirtschaft und Technik

```
from pandas_datareader import data as web
import datetime as dt
import matplotlib.pyplot as plt
#from matplotlib import style

start =dt.datetime(2017,1,1)
ende = dt.datetime(2019,1,1)

df = web.DataReader('AAPL','yahoo',start,ende)

print(df.head)
df.to_csv("apple.csv")

#style.use('ggplot')
df['Adj Close'].plot()
#plt.grid(True)

plt.show()
```