INTERNATIONAL AUDIO LABORATORIES ERLANGEN
A joint institution of Fraunhofer IIS and Universität Erlangen-Nürnberg

Multimedia Programmierung mit Python

Nils Werner
Stefan Balke

12. August, 2015

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

Fraunhofer
IIS

Grundlagen

Hallo Welt

```
print(”Hallo Welt”)
```

```
$ python hallo.py
```

Einfache Variablen

```
Ø1. a = 1          # Ganzzahl
Ø2. b = 1.5        # Dezimalzahl
Ø3. s = ”Hallo”    # Text
```

Einfache Operationen

```
01. 1 + 1.5              = 2.5
02. 5 / 2                = 2.5, vorsicht in Python 2.7!
03. 5 // 2               = 2
04. 3 ** 2               = 9
05. "Hallo" + "Talentschool" = "HalloTalentschool"
06. "%d Auto" % a        = "1 Auto"
```

Einfache Operationen

```
01. print(1 + 1.5)
02. print(”Hallo” + ”Talentschool”)
```

```
$ python operationen.py
```

Einfache Aufgabe

Ein Autohändler hat **100** gleiche Autos, die er zu je **5267.6** € verkaufen möchte.

Pro Auto erzielt er einen Gewinn von **17.32%** des Verkaufspreises.

Wieviel Gewinn erzielt er, wenn er **20**, **31** und **67** Autos verkauft?

$$gewinn = 20 * 5267.6 * 17.32/100$$

Einfache Aufgabe

```
01. preis_pro_auto = 5267.6
02. gewinn_marge = 17.32
03. autos_verkauft = 20
04. gewinn = autos_verkauft * preis_pro_auto \
05.          * gewinn_marge / 100
06. print(gewinn)
```

```
$ python auto.py
```

Kommandozeilen Parameter

```
01. import sys
02. preis_pro_auto = 5267.6
03. gewinn_marge = 17.32
04. autos_verkauft = int(sys.argv[1])
05. gewinn = autos_verkauft * preis_pro_auto \
06.          * gewinn_marge / 100
07. print(gewinn)
```

```
$ python auto.py 21
```

Bedingungen und Schleifen

```
01. if gewinn > 2000:
02.     print("Gute Verkaufszahlen diesen Monat")
03. else:
04.     print("Schlechter Monat")
05.
06. print("Abfrage fertig")
```

Bedingungen und Schleifen

```
01. for i in l:
02.     print(i)
03.
04. print("Schleife fertig")
```

Funktionen

```
01. def my_func(a, b):
02.     return a + b
03.
04. print(my_func(1, 3))
```

Bibliotheken

```
01. import math
02.
03. print(math.sin(math.pi))
```

`io`, `os`, `sys`, `math`, `itertools`, `r`, `socket`, `datetime`, `csv`, `numpy`, `scipy` …

Muster

Schleifen

```
01. i = 0
02. while i < 100:
03.     print(i)
04.     i = i+1
```

```
01. for i in range(100): # range(100) = [0,1,2,...99]
02.     print(i)
```

Suche

```
01. for i in [1, 2, 20, 100]:
02.     if i == 3:
03.         print("gefunden")
04.         break
```

```
01. if 3 in [1, 2, 20, 100]:
02.     print("gefunden")
```

Suche

`string`s sind iterierbar:

```
01. if "a" in "Talentschool":
02.     print("gefunden")
```

Slicing

```
01. a = range(20)
02. print(a[4]) # 4
03. print(a[4:8]) # 4, 5, 6, 7
```

```
01. a = "Talentschool"
02. print(a[4]) # "n"
03. print(a[4:8]) # "ntsc"
```

Numerik

Vektorisierung

```
01. import numpy
02.
03. a = numpy.array([0, 1, 2, 3, 4])
04. print(a)          # 0, 1, 2, 3, 4
05. print(a * 2)      # 0, 2, 4, 6, 8
```

Vektorisierung

```
01. import numpy
02.
03. a = numpy.arange(5)
04. a = numpy.array([8, 13, 21, 34, 55])
05. print(a)          # 0, 1, 2, 3, 4
06. print(a + b)      # 8, 14, 23, 37, 59
```

Matrizen

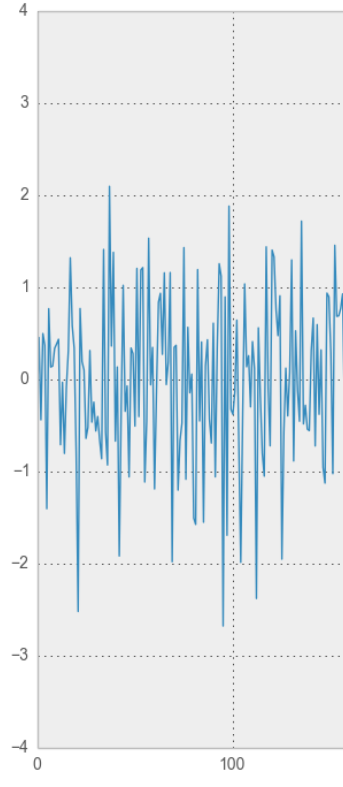
```
01. import numpy
02.
03. a = numpy.random.randn(5, 5) # 5x5 Matrix
04. print(a[2:4, 1])             # Zeilen 2-3, Spalte 1
05. print(a[:, 1])               # Alle Zeilen, Spalte 1
```

Matrizen

```
01. import numpy
02.
03. a = numpy.random.randn(5, 5) # 5x5 Matrix
04. print(numpy.sum(a))          # Summe der Matrix
05. print(numpy.sum(a, axis=0)) # Spaltenweise Summe
06. print(numpy.sum(a, axis=1)) # Zeilenweise Summe
```

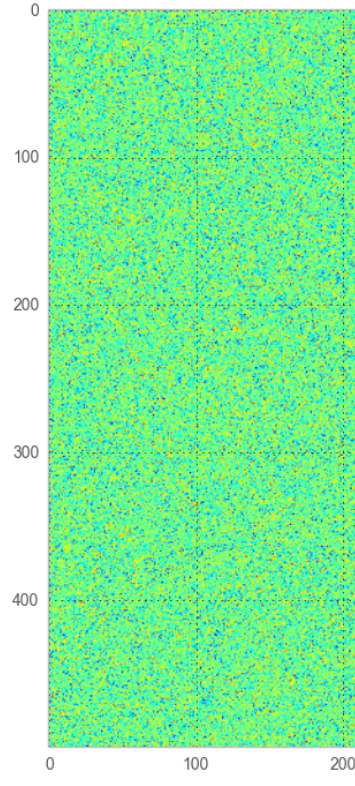
Funktionsgraphen

```
01. import numpy
02. import matplotlib.pyplot as plt
03.
04. a = numpy.random.randn(500) # 500 Werte
05. plt.plot(a)
06. plt.show()
```



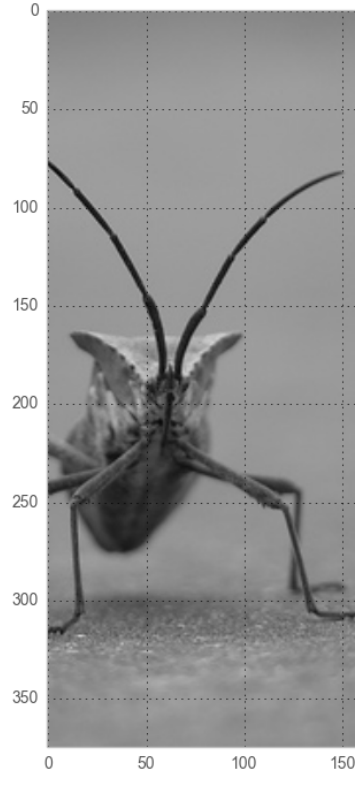
2D Graphen

```
01. import numpy
02. import matplotlib.pyplot as plt
03.
04. a = numpy.random.randn(500, 500) # 500x500 Werte
05. plt.imshow(a)
06. plt.show()
```



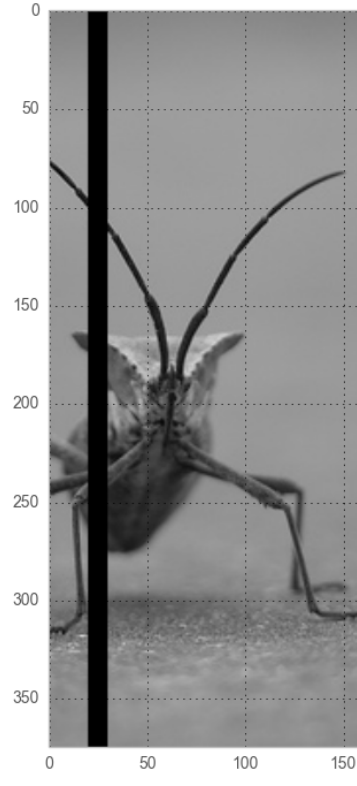
Bilder

```
01. import numpy
02. import matplotlib.pyplot as plt
03. import matplotlib.image as mpimg
04.
05. a = mpimg.imread('stinkbug.png')
06. plt.imshow(a)
07. plt.show()
```



Datenmanipulation

```
01. import numpy
02. import matplotlib.pyplot as plt
03. import matplotlib.image as mpimg
04.
05. a = mpimg.imread('stinkbug.png')
06. a[:, 20:30] = 0
07. plt.imshow(a)
08. plt.show()
```



Audio

Audio Eingang

```
01. import audio
02.
03. with audio.audio(
04.     input=True, rate=44100,
05.     buffer_length=1024, channels=2
06. ) as input:
07.
08.     while True:
09.         frame = input.read(1024)
```

Audio Ausgang

```
01. import audio
02.
03. with audio.audio(
04.     output=True, rate=44100,
05.     buffer_length=1024, channels=2
06. ) as input:
07.
08.     while True:
09.         frame = input.write(numpy.array(...))
```

Audio Bypass

```
01. import audio
02.
03. with audio.audio(
04.     output=True, input=True, rate=44100,
05.     buffer_length=1024, channels=2
06. ) as input:
07.
08.     while True:
09.         input.write(input.read(1024))
```

Video

Video

```
01. import numpy as np
02. import cv2
03.
04. cap = cv2.VideoCapture(0)
05.
06. while True:
07.     ret, frame = cap.read()
08.     cv2.imshow('frame', frame)
```

Videomanipulation

```
01. import numpy as np
02. import cv2
03.
04. cap = cv2.VideoCapture(0)
05.
06. while True:
07.     ret, frame = cap.read()
08.     frame[:, 20:30, 0:2] = 0
09.     cv2.imshow('frame', frame)
```

Videomanipulation

```
01. import scipy.ndimage
02. import cv2
03. cap = cv2.VideoCapture(0)
04.
05. while True:
06.     ret, frame = cap.read()
07.     frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY
08.     frame = scipy.ndimage.filters.gaussian_filter(frame, 3)
09.     cv2.imshow('frame', frame)
```

Referenzen

Referenzen

- https://docs.python.org/2/
- http://docs.scipy.org/doc/numpy/reference/
- http://docs.scipy.org/doc/scipy/reference/
- http://matplotlib.org/
- http://people.csail.mit.edu/hubert/pyaudio/
- http://docs.opencv.org/

Broadcasting

```
01. import numpy
02.
03. a = numpy.random.randn(3, 5) # 3x5 Matrix
04. b = numpy.arange(5)          # 5-Element Vektor
05. a * b                        # Alle Zeilen werden mit
                                 # Vektor b multipliziert
```