Cheatography

Python 3 Cheat Sheet

by Niels Röhrdanz (Niels132) via cheatography.com/44142/cs/13079/

Basisdatentypen		
Integer	int()	2,0,3,-2
Float	float()	2.307, 0.0, -2.34567
String	str()	'Hallo ich bin ein String'
Boolean	bool()	True, False

numerische Operationen		
5+1	>>6	Addition
5-1	>>4	Subtraktion
3/5	>>.6	Division
3//5	>>0	Floor Division
3%5	>>3	Modulo
3*5	>>15	Multiplikation
3**3	>>27	Potenz

sequentielle Datentypen		
Liste	list()	[1,2,3,5,-6,9]
Tupel	tuple()	(1,2,8)
Dictionary	dict()	{'Hallo':'hello','Fünf':5}
String	str()	'Hallo ich bin ein String'

Überprüfen/ändern des Datentypes	
x=5	
type(x)	>>int
float(x)	>>5.0
str(x)	>>'5'
bool(x)	>>True

Operationen auf Strings	
s='hallo du!'	
s.replace('h','H')	>>'Hallo du!'
substrings=s.split[' ']	substrings=['hallo','du']
s.lower()	>>'hallo du!'
s.upper()	>>'HALLO DU!'

indizierung sequentieller Datentypen			
x=[1,3,5,7,-2]	x[0]	>>1	
	x[-1]	>>-2	
	x[:]	>>1,3,5,7,-2	
	x[2:3]	>>5	2 eingeschlossen, 3 ausgeschlossen
	x[::2]	>>1,5,-2	jedes 2. Element

indizierung sequentieller Datentypen (cont)

x[::-1]	-2,7,5,3,1	rückwärts
x[2:]	5,7,-2	alles ab dem 2. Element
x[:2]	1,3	alles bis zum 2. Element (2. Element nicht eingeschlossen)

Die Indizierung sequentieller Datentypen beginnt in Python, wie auch in vielen anderen Programmiersprachen mit der 0. Hier müssen Matlab Nutzer aufpassen!!!

Operationen für sequenzielle Datentypen

x=[2,1,3,4,2]	
2 in x	>>True
x.count(2)	>>2
x.index(2)	>>0
x.remove(2)	>>[1,3,4]
x.sort()	>>[1,2,2,3,4]
x.append(1)	>>[2,1,3,4,2,1]
x+=1	>>[2,1,3,4,2,1]
x*=2	>>[2,1,3,4,2,2,1,3,4,2]
x = [i*2 for i in x if i != 0]	jedes Element mit zwei multiplizieren außer das Element ist 0

Variablen und Zuordnungen

x=1	x referenziert die Instanz 1
x,y,z=1,2,3	
x+=1	x=x+1
x-=1	x=x-1
x*=2	x=x*2
x/=2	x=x/2
x,y=y,x	Werte tauschen
x=y	x und y referenzieren die gleiche Instanz
x is y	überprüfen, ob x und y auf die gleiche instanz zeigen
global x	erstellen einer globalen Variable

built in functions

print(5)	Ausgabe	>>5
x=input('Bitte geben Sie eine Zahl ein: ')	Eingabe	>>Bitte geben Sie eine Zahl ein: _
eval('5+3')	Ausführen von Strings	>>8
enumerate([1,2,3,4])	ermöglicht das iterieren über Liste und Indice bei for loops	

built in functions (cont)

... ...

Es erfolgt in Python standardmäßig keine Ausgabe, eine Unterdrückung mittels ';' ist hier also nicht üblich. Um eine Ausgabe zu erzielen, sollte auf die print() Funktion zurückgegriffen werden

if-Statement

```
#if statement
#auf Einschübe achten
if x==3:
   print(x)
elif x>3:
   print('x ist größer als 3')
else:
   print('x ist zu klein')
#inline if
#auf Übersichtlichkeit achten!!!
if x==3:print(x)
```

for-Schleife

```
# for loop
x=[1,2,3,5,6,1]
for n in x:
   print(n)
>>1
>>2
>>3
>>5
>>...
#erstellen, eines iterierbaren Objektes
x=range(3) #exclusiver Endpunkt
for l in x: print(x)
>>0
>>1
>>2
x=range(2,3) #Start und exclusiver Endpunkt
for l in x: print(x)
>>2
x=range(2,10,2) #Start,exclusiver Endpunkt und Schrittweite
for l in x: print(x)
```

for-Schleife (cont)

```
>>2
>>4
>>6
>>8
#Liste überschreiben
x=[1,3,2,6]
for n in range(len(x)):
   x[n]=x**2 #jeder wert aus der Liste x wird mit seinem Quadrat überschrieben
```

While-Schleife

```
x=0
while x<=10:
   print(x)
   x+=1
>>0
>>1
>>...
>>10
#Endlosschleife
while True:
   print('diese Schleife läuft solange, wie das Programm läuft oder die Schleife abgebrochen wird')
```

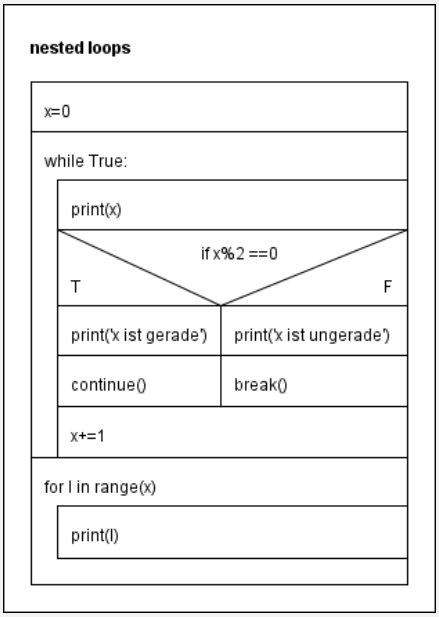
Abbruch von Schleifen

```
while True:
   break() # sorgt für Abbruch der Schleife
```

nested loops

```
x=5
while True:
   if x<=20:
      x+=1
   elif x==6:
      for z in range(x):
         if z<=3:
            print(x)
         else:
            break()
   else:
      break()
```

Struktogramme



Oftmals helfen Struktogramme bei der Planung von Funktionen. Im Bsp. eine nested Loop ohne jeden Sinn, sie dient lediglich der Darstellung des Struktogramms

Funktionen

```
def my_first_function(arg1):
   print(arg1)
my_first_function(3)
>>3
```

Arbeiten mit Dateien

```
#Einlesen von Dateien
f=open(Hallo.txt,'r') #öffnen des Textes
text=f.read() #liest den ganzen Text ein
f.close() #schließen des Textes
#zeilenweises Einlesen
f=open(Hallo.txt,'r') #öffnen des Textes
for line in f:
  x=f.readline()
  print(x) # ausgabe der Zeile
f.close() #schließen des Textes
#Zeilen- und Spaltenweises einlesen einer csv-Datei
'''
 Datei sieht z.B. wie folgt aus
1,4
2,3
```

Arbeiten mit Dateien (cont)

```
1,9
'''
f=open(Bsp.csv,'r') #öffnen des Textes
spalte_1 = [ ]
spalte_2 = [ ]
for line in f:
   x = f.readline()
   spalte_1.append(int(x.split(',')[0]))
   spalte_2.append(int(x.split(',')[1]))
f.close()
#Dateien in eine Datei schreiben
#Fall1 Datei noch nicht vorhanden oder falls sie vorhanden ist soll sie überschreiben werden
f=open('fall1.txt','w') #mode 'w' für write
f.write('Hier steht später irgendwas')
f.close()
#Fall2 in einer Datei weiterschreiben oder erstellen, falls sie nicht vorhanden ist
f=open('fall2.txt','w') #mode 'a' für append
f.write('Hier steht später irgendwas')
f.close()
```

falls man sich nicht sicher ist sollte man auf write verzichten, da hier im schlimmsten Fall alle Daten der Datei verloren gehen!!!

Module

math	Konstante und Funktionen	https://docs.python.org/2/library/math.html
numpy	numerisches Python	https://docs.scipy.org/doc/numpy-1.9.3/reference/
matplotlib	graphische Darstellung	http://matplotlib.org/
mayavi	3D Plotting	http://docs.enthought.com/mayavi/mayavi/mlab.html#d-plotting-functions-for-numpy-arrays
tkinter	GUI	http://infohost.nmt.edu/tcc/help/pubs/tkinter/web/index.html
pygame	Spiele	http://www.pygame.org/docs/
random	Zufallszahlen	https://docs.python.org/2/library/random.html
os	Operating System	https://docs.python.org/2/library/os.html
time	Zeitmessung	https://docs.python.org/2/library/time.html

Importieren von Modulen

import math	Importieren des gesamten Math Modules	math.sin(3)	Modulname muss genannt werden
from math import pi	Importieren einer einzelnen Funktion aus dem Math Modul	pi	nur Funktionsname muss genannt werden
from math import *	Importieren aller Bestandteile des Math Modules	sin(pi)	da nicht das Modul, sondern alle Einzelbestandteile importiert worden sind, reicht es die Funktionsnamen anzugeben
import math as m	Importieren des Math Modules als m	m.sin(m.pi)	das Modul wird al m benannt und wird auch so aufgerufen, gleiches ist für Bestandteile von Funktionen möglich

Es sollte immer erkenntlich sein, welche Funktionen zu welchen Modulen gehören, daher sollte from ... import * nur dann genutzt werden, wenn erkenntlich ist, welche Funktionen zu diesem Modul gehören

Math

import math	
math.sin(), math.cos(), math.tan()	Sinus, Cosinus, Tangens
math.asin(), math.acos(), math.atan()	Arcus Sinus, Cosinus, Tangens
math.e	e
math.log10()	10er Logarithmus
math.log()	Logarithmus
math.sqrt()	Wurzel
math.pi	Pi
math.radians()	umwandlung in Radialmaß
math.degree()	Umwandlung in Bogenmaß

weitere Funktionen:
https://docs.python.org/2/library/math.html

Numpy

import numpy as np	
x=np.array([[1,23,3],-[1]])	erstellen eines 2D-Arrays, der NumPy-Array bietet als n-dimensionaler Array das Analogon zu einem Vektor in Matlab
x[:][0]	>>[1, 23, 3]
x[0][1]	>>23
x[:][:]	>> array([[1, 23, 3], [1]], dtype=object)
np.append(x[0],x[0])	array([1, 23, 3, 1, 23, 3])
np.pi	Pi
np.sqrt()	Wurzel

Cheatography

Python 3 Cheat Sheet

by Niels Röhrdanz (Niels132) via cheatography.com/44142/cs/13079/

Numpy (cont)

np.sin()	Sinus
...	...

Dokumentation:
https://docs.scipy.org/doc/numpy-1.9.3/user/

matplotlib

import matplotlib as mpl	
import mpl.pyplot as plt	
fig=plt.figure()	erstellen einer Figure
ax=fig.gca()	Erstellen der Achsen, wobei gca für 'get current axis' steht
ax.set_xlabel('X-Achse')	Achsenbeschriftung (äquivalent für die Y-Achse)
ax.plot([1,1,2],[2,1,3])	plotten von Punkte (verbunden zu Linie)
plt.ylim((0, 110))	Minimum und Maximum der Y-Achse festlegen analog für X-Acchse
ax.invert_yaxis()	Invertieren der Y-Achse
ax.grid()	Gittermuster erstellen
plt.show()	Anzeigen der Graphik

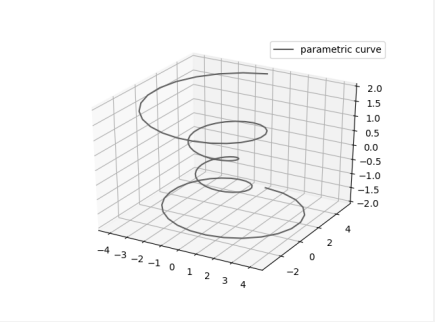
Dokumentation:
http://matplotlib.org/

mayavi

from mayavi import mlab	
figure = mlab.gcf()	Figure
mlab.mesh(x, y, z)	
mlab.plot3d()	plotet eine Linie mit gegebenen Koordinaten X,Y,Z
mlab.show()	zeigen der aktuellen Figure
mlab.points3d()	
mlab.surf()	
mlab.imshow()	
mlab.contour_surf()	
mlab.quiver3d()	
mlab.flow()	

Dokumentation:
http://docs.enthought.com/mayavi/mayavi/mlab.html#d-plotting-functions-for-numpy-arrays

Bsp. Matplotlib 3d



```
import matplotlib as mpl
from mpl_toolkits.mplot3d import Axes3D
import numpy as np
import matplotlib.pyplot as plt
mpl.rcParams['legend.fontsize'] = 10
fig = plt.figure()
ax = fig.gca(projection='3d')
theta = np.linspace(-4 np.pi, 4 np.pi, 100)
z = np.linspace(-2, 2, 100)
r = z**2 + 1
x = r * np.sin(theta)
y = r * np.cos(theta)
ax.plot(x, y, z, label='parametric curve')
ax.legend()
ax.set_xlabel('X Label')
ax.set_ylabel('Y Label')
ax.set_zlabel('Z Label')
plt.show()
```

os

import os	
os.listdir(path)	liefert eine Liste mit allen Dateien in einem bestimmten Pfad
os.getcwd()	liefert aktuelles Arbeitsverzeichnis
os.chdir(path)	ändern des aktuellen Arbeitsverzeichnis (path ist ein String)

weitere Funktionen
https://docs.python.org/2/library/os.html

random

import random		
x=[1,4,5,2,7,8]		
random.sample(x,2)	>>[2,5]	Ziehen ohne zurücklegen
random.shuffle(x)	>>[8,7,1,4,5,2]	Mischen
random.choice(x)	>>5	Zufallsauswahl
random.randint(0,9)	>>3	Generiert einen Integer in dem Intervall [0,9]

Mehr Funktionen auf
https://docs.python.org/2/library/random.html

time

import time	
t=time.time()	Zeitpunkt
delta_t=time.time-t	vergangene Zeit in sec
time.strftime('%A %d %m %Y')	>>'Saturday 14 10 2017'
time.sleep(sec)	Pausieren für sec

Dokumentation:
https://docs.python.org/2/library/time.html