Spickzettel: Python

12.11.2016

Syntax

- Der Code wird über die Einrückungen strukturiert. Code, der zu einer Kontrollstruktur oder Funktion gehört, muss gleich eingerückt sein
- Kommentare werden mit # eingeleitet
- Gross-/Kleinschreibung muss beachtet werden

Kontrollstrukturen

Schleifen

```
repeat n:
   Anweisungsblock

for variable in range(n):
   Anweisungsblock

for variable in Liste
   Anweisungsblock

while Bedingung:
   Anweisungsblock
```

Eine Schleife kann mit **break** jederzeit abgebrochen werden.
(repeat nur in TigerJython)

Verzweigungen

```
if Bedingung:
    Anweisungsblock
else:
    Anweisungsblock

if Bedingung:
    Anweisungsblock
elif Bedingung:
    Anweisungsblock
else:
    Anweisungsblock
```

Bedingungen verknüpfen:

```
if x < 10  and x != 5:
if x == 2 or x == 5:
```

Funktionen

Definition:

```
def name(Parameter):
    Anweisungsblock
```

Aufruf:

```
name(Parameter)
```

```
def maximum(x, y):
    if x >= y:
        return x
    else:
        return y
```

```
x = 5
def f():
    global x
    x += 1
```

Funktionen können beliebig viele Parameter haben. Die Klammern sind notwendig, auch wenn keine Parameter vorhanden sind.

`return` *Wert* beendet eine Funktion und gibt *Wert* zurück. Eine Funktion muss kein return haben.

Falls man eine globale Variable in einer Funktion verändern will, muss diese mit `global` *Variable* in der Funktion bezeichnet werden.

Datentypen

Variablen sind typenlos und verweisen auf Werte. Jeder Wert hat einen bestimmten Typ.

`bool`	Wahrheitswert	True, False
`int`	Ganze Zahl	234 456
`float`	Fliesskommazahl	6.023e+23
`complex`	Komplexe Zahl	complex(2, 3)
`str`	Zeichenkette/String	"Hallo", 'Antwort'
`list/tuple`	Liste oder Tupel	[1, 2, 3], (5, 6)
`dictionary`	Key-Value-Paare	{3074:"Muri", 6300:"Zug"}

Rechenoperationen

`+ - * /` Addition, Subtraktion, Multiplikation, Division

Bei der Division unterscheidet man zwischen der "normalen" Division `/` und der ganzzahligen Division `//` (6 / 4 = 1.5; 6 // 4 = 1).

Potenzen werden mit ** ausgedrückt (3 **2 = 9, 3 ** 0.5 = 1.732)

Viele mathematische Funktionen sind im Modul *math* enthalten. Dieses kann auf zwei Arten importiert werden:

```
from math import sqrt, pi
print sqrt(3)
print "Pi =", pi
```

```
import math
print math.sqrt(3)
print "Pi =", math.pi
```

Zufallszahlen

Das Modul *random* muss importiert werden: `import` random

`random.random()`	liefert eine Float-Zufallszahl 0 <= z <1
`random.randint(a, b)`	liefert eine Integer-Zufallszahl a <= z <= b

Listen

```
li = [2, 4, 6]
li[0] → 2  (das erste Element)

range(5) → [0, 1, 2, 3, 4]
range(5, 8) → [5, 6, 7]
range(5, 12, 3) → [5, 8, 11]
```

len(liste)
liste.**append(**Element)
liste.**index**(Element)
liste.**insert**(Index, Element)
liste.**remove**(Element)
liste.**sort()**
x **in** liste *True*, falls x in der Liste ist.