



# Programmieren für Einsteiger

## Python cheat sheet - Tag 1

### Basics

#### Text ausgeben

```
print('Hallo Welt')
```

#### Variablen belegen

```
x = 5
x = 5 + 14
x = x + 1
x = 'Eine String Variable.'
x, y = 19, 'Python'
```

#### Variablentypen

9	type(x)	-> int
9.0		-> float
'String'		-> str
True		-> bool
('Berlin', 12109, 'Musterstr.', 1)		-> tuple
[('Laser', 13.587), [4,5,6]]		-> list
{4, 7, 3, 9, 1}		-> set
{'Eins':1, 'Liste':[1,2,3], 'Monthy':'Python'}		-> dict

#### Rechenoperationen

```
9 + 4 -> 13
9 - 4 -> 5
9 * 4 -> 36
9 / 4 -> 2.25
9 // 4 -> 2
9 % 4 -> 1
9 ** 4 -> 6561
9 < 4 -> False
9 == 9 -> True
```

### Arbeiten mit Sequenzen

**Strings** dienen zur Speicherung und Verarbeitung von Text der Backslash (\) markiert Sonderzeichen

```
'Das Leben des Brian'
"Die Ritter der Kokosnuss"
"Dieser Satz enthält einen \n Zeilenumbruch und ein \t Tabstop."
```

**Tupel** können Daten mit unterschiedlichen Datentypen enthalten. Es handelt sich dabei um einen unveränderlichen Datentypen

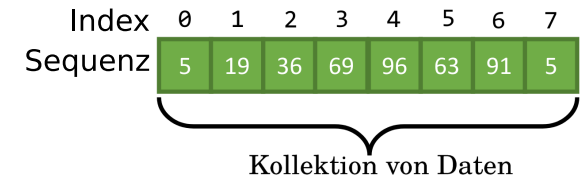
```
person = ('Maximilia', 'Musterfrau', '01.01.1990')
adresse = ('Berlin', 12109, 'Musterstr.', 1, person)
```

**Listen** können Daten mit unterschiedlichen Datentypen enthalten. Der Inhalt von Listen kann einfach verändert werden.

```
liste1 = [1, 2, 3, 4, 5, 'Python', ('A', 'B')]
farben = ['grün', 'gelb', 'blau', 'rot', 'orange', 'mauve']
schachtel = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

#### Grundoperationen für Sequenzen

```
liste1 = [1, 2, 3, 4]
liste2 = [4, 5, 6, 7]
liste1[0] -> 1
liste1[2:4] -> [3,4]
liste = liste1 + liste2 -> [1, 2, 3, 4, 4, 5, 6, 7]
min(liste) -> 1
liste[4] = 8 -> liste = [1, 2, 3, 4, 8, 5, 6, 7]
```



### Typumwandlungen

Variablen können durch Typumwandlungen (= Casting) ineinander umgewandelt werden:

```
5 + int(14) -> 19
str(5) + '14' -> '514'
int(1.7) -> 1
bool(1) -> True
bool(0) -> False
float(12) -> 12.0
tuple([1,2,3]) -> (1,2,3)
list('12345') -> ['1', '2', '3', '4', '5']
dict([('e',19), ('f', 36)]) -> {'e': 19, 'f': 36}
```

### Mengen und Mengenoperationen

Jedes Element in einer Menge darf genau einmal auftauchen. Die Elemente der Menge haben keinen Index.

```
menge = {4, 7, 3, 9, 1}
menge.union({2, 3}) -> {1, 2, 3, 4, 7, 9}
8 in menge -> False
9 in menge -> True
{3,4}.issubset(menge) -> True
set([19,19,5,90,36]) -> {5, 19, 36, 90}
set('Mississippi') -> {'M', 'i', 'p', 's'}
```

### Dictionaries

Ein Dictionary stellt eine Kollektion von Daten dar deren einzelne Elemente so genannte Schlüssel-Wert-Paare sind

```
d = {'Eins': 1, 'Liste': [1,2,3], 'Monty': 'Python'}
d['Eins'] -> 1
d['Zwei'] -> KeyError: 'Zwei'
d['Zwei'] = 2
'Monty' not in d -> False
```