

Operatoren - Übersicht

- Arithmetrische Operatoren
- Vergleichsoperatoren
- Binäre Operatoren
- Logische Operatoren
- Unäre Operatoren

Arithmetrische Operatoren

+ Addition

```
>>> 2 + 3
```

```
5
```

- Subtraktion

```
>>> 2 - 3
```

```
-1
```

* Multiplikation

```
>>> 2 * 3
```

```
6
```

/ Division

```
>>> 3 / 2
```

```
1.5
```

// Division ganzzahlig

```
>>> 3 // 2
```

```
1
```

% Rest

```
>>> 3 % 2
```

```
1
```

** Exponentiation

```
>>> 3 ** 2
```

```
9
```

```
>>>
```

Vergleichsoperatoren

`==` gleich

```
>>> 2 == 3  
False
```

`!=` ungleich

```
>>> 2 != 3  
True
```

`<` kleiner

```
>>> 2 < 3  
True
```

`<=` kleiner oder gleich

```
>>> 2 <= 3  
True
```

`>` größer

```
>>> 2 > 3  
False
```

`>=` größer oder gleich

```
>>> 2 >= 3  
False
```

Bit-Operatoren

& AND

| OR

^ XOR

~ Komplement (Einerkomplement)
funktioniert in Python nicht wegen
unbegrenztem Zahlenraum.

« links Verschieben

» rechts Verschieben

```
>>> bin(0b10101 & 0b01011)
'0b00001'
```

```
>>> bin(0b10101 | 0b01010)
'0b11111'
```

```
>>> bin(0b10101 ^ 0b01011)
'0b11110'
```

```
>>> bin(~0b10101)
'-0b10110'
```

```
>>> bin(0b101<<3)
'0b101000'
```

```
>>> bin(0b101>>2)
'0b1'
```

Logische Operatoren

Mit logischen Operatoren lassen
sich boolsche Werte
vergleichen:

not

and

or

```
>>> not False
True
>>> not True
False
>>> True and True
True
>>> True and False
False
>>> False and False
False
>>> True or False
True
>>> True or True
True
>>> False or False
False
>>>
```

Unäre Operatoren

Ein unärer Operator hat nur einen Ausdruck

+ positive Zahl

- negative Zahl

abs() absoluter Wert

~ Komplement

```
>>> + 3  
3
```

```
>>> - 3  
-3
```

```
>>> abs (-3)  
3
```

```
>>> ~3  
-4
```

Zusammengesetzte Operatoren

Einige Operatoren lassen sich zusammensetzen. So kann man Tipparbeit sparen:

```
>>> x = 2
```

```
>>> x += 3
```

```
>>> x
```

```
5
```

```
>>> x -= 2
```

```
>>> x
```

```
3
```

Probiert aus bei welchen es geht und bei welchen nicht.

```
>>> x *= 2
```

```
>>> x
```

```
6
```

```
>>> x /= 3
```

```
>>> x
```

```
2.0
```

Operatoren auf Sequenzen

in prüft ob ein Wert in einer Sequenz enthalten ist

```
>>> x = [1, 2, 3]
```

```
>>> 2 in x  
True
```

```
>>> 5 in x  
False
```

not in prüft ob ein Wert nicht in einer Sequenz enthalten ist

```
>>> 2 not in x  
False
```

```
>>> 5 not in x  
True
```


Verketteten von Sequenzen

Die mathematischen Operatoren
+ und **+=** lassen sich auf
Sequenzen anwenden:

```
>>> 'Hallo' + 'Micropython'  
'HalloMicropython'
```

```
>>> 'Hallo' + ' ' + 'Micropython'  
'Hallo Micropython'  
>>>
```

```
>>> s = 'Hallo'  
>>> s += ' '  
>>> s += 'Micropython'  
>>> s  
'Hallo Micropython'
```

```
>>> l1 = [1, 2, 3]  
>>> l2 = [4, 5, 6]  
>>> l1 += l2  
>>> l1  
[1, 2, 3, 4, 5, 6]
```