

### Grundlagen

<code>python -V**</code>	Version Python 2
<code>python3 -V</code>	Version Python 3
<code>print('Hello World')</code>	Ausgabe Kommandozeile
<code>print(1+2)</code>	Ausgabe Zahlenoperationen
<code>input("Eingeben:")</code>	Eingabe
<code>quit()</code>	Pythonkonsole verlassen
<code>exit()</code>	Pythonkonsole verlassen
<code>"String"</code>	Hochkommata doppelt
<code>'String'</code>	oder einfach (bevorzugt)
<code>\n</code>	New Line; Escape Character
<code>\"</code>	Backslash für verbotene Buchstaben (z.B. Hochkomma)
<code>""" Drei """</code>	Drei Hochkommata = Automatischer Zeilenumbruch
<code>"String" + 'cool'</code>	String Konkatination
<code>print("spam" * 3)</code>	String Multiplikation
<code>int("4")</code>	Typkonversion (selbiges mit <code>float()</code> und <code>str()</code> )

Nach der Installation Zugriff über Kommandozeile, der Python GUI oder IDLE (Integrierte Entwicklungsumgebung). Die Python Konsole ist eine **REPL** (Read-eval-print loop)

### Rechenoperationen

<code>2 [+/*/:] 2</code>	Rechenoperationen
<code>2* ( 3+4 )</code>	Klammern
<code>10.56</code>	Float: Fließkommazahl
<code>4 / 2 = 2.0</code>	Division von Integer ergibt Float
<code>2 ** 5</code>	Exponentiation 2 hoch 5
<code>9 ** (1/2)</code>	Wurzel
<code>20 // 6 = 3</code>	Ganzzahldivision
<code>1.25 &amp; 0.5</code>	Modulo

Rechenoperationen können im Normalfall entsprechend der mathematischen Notation angegeben werden. Division mit der 0 produziert einen Fehler

### [Beispiel] Typenkonversion

```
>>> float(input("Enter a number:
")) + float(input("Enter another
number: "))
Enter a number: 40
Enter another number: 2
42.0
```

Eingabenumwandlung in Fließkommazahlen mit anschließender Rechnung.

### Variablen

<code>x = 7</code>	Variablendeklaration
<code>del variable</code>	Variable löschen
<code>x += 3</code>	Werterhöhung, ebenso bei Strings, mit vielen Operationen möglich

Variablen haben keinen spezifischen Datentyp. Erlaubte Zeichen sind Buchstaben, Nummern und Unterstriche. Variablen dürfen nicht mit Zahlen beginnen.

### Kontrollstrukturen

`b = True|False` Boolean, groß geschrieben

<code>==</code>	Gleich
<code>!=</code>	Ungleich
<code>&lt;   &gt;</code>	Größer / kleiner
<code>&lt;=   &gt;=</code>	Kleiner gleich, größer gleich

`and` Logisches UND  
`or` Logisches ODER  
`not 1==1` Logisches NICHT

`if` If-Abfrage  
 Bedingung:  
 statements

`else:` Else Statement  
 statements

`elif` else-if Abfrage  
 Bedingung:  
 statements

`while` While-Schleife  
 Bedingung:  
 Statement

`while 1==1:` Endlosschleife  
 Statement

`break` Beenden Schleife  
`**continue` Schleife stoppen, Sprung nächste Iteration

Python benutzt Leertzeichen zu Beginn einer Zeile, um die Zugehörigkeit zu einem Codesegment zu markieren (Beispiel if-Abfrage).

### Listen / Arrays

<code>arr = ["a","b","c"]</code>	Array/Liste
<code>print(arr[0])</code>	Ausgabe Listenelement
<code>arr = ["a",5,[3,3.0]]</code>	Mehrere Datentypen und Listen
<code>st = "Hello"</code> <code>print(st[2])</code>	String als Array, Ausgabe entsprechend möglich
<code>nums[2] = 5</code>	Neuzuordnung Feld 2
<code>nums = [1, 2, 3]</code> <code>print(nums + [4, 5, 6])</code>	Addition von Arrays, Multiplikation möglich
<code>print("egg" in words)</code>	Prüft Existenz vom String im Array <i>words</i>
<code>print(not 4 in nums)</code> <code>print(4 not in nums)</code>	Prüfung Element nicht in Liste

Beim Array werden die Elemente durch Kommas separiert. Es ist möglich, bei der Definition ein Komma an letzter Stelle einzufügen, was kein zusätzliches Element einfügt, aber syntaktisch valide ist. Einzelne Felder können neu zugeordnet werden.

### [Beispiel] If & While

```
i = 0
while True:
    i = i + 1
    if i == 2:
        print("Skipping 2")
        continue
    if i == 5:
        print("Breaking")
        break
    print(i)
print("Finished")
```

### Boolean Operatoren

Operator	Description
<code>**</code>	Exponentiation (raise to the power)
<code>~ + -</code>	Complement, unary plus and minus (method names for the last two are <code>+</code> @ and <code>-</code> @)
<code>* / % //</code>	Multiply, divide, modulo and floor division
<code>+ -</code>	Addition and subtraction
<code>&gt;&gt; &lt;&lt;</code>	Right and left bitwise shift
<code>&amp;</code>	Bitwise 'AND'
<code>^  </code>	Bitwise exclusive 'OR' and regular 'OR'
<code>&lt;= &lt; &gt;=</code>	Comparison operators
<code>&lt;&gt; == !=</code>	Equality operators
<code>= %= /= //= -= +=</code> <code>**= +=</code>	Assignment operators
<code>is is not</code>	Identity operators
<code>in not in</code>	Membership operators
<code>not or and</code>	Logical operators



By **aragow**  
[cheatography.com/aragow/](http://cheatography.com/aragow/)

Not published yet.  
Last updated 12th April, 2017.  
Page 2 of 2.

Sponsored by **CrosswordCheats.com**  
Learn to solve cryptic crosswords!  
<http://crosswordcheats.com>