Cheatography

Python für Beginner Cheat Sheet

by max3l via cheatography.com/121235/cs/22126/

Variablen

name = Objekt	Zuweisung von Objekten
name = Ausdruck	Ausdruck wird ausgewertet und dessen Ergebnis als "name" gespeichert

Variablen sind **Namen für Objekte**. Sie werden oft verwendet um Werte zu speichern, auf die man später zurückgreifen möchte.

Objekte sind **alles mögliche**. In der Objektorientierung sind beispielsweise **Exemplare von Datentypen** und **Instanzen von Klassen** Objekte.

Variablen können **global** oder **lokal** sein. **Globale** Variablen werden außerhalb von Funktionen oder Klassen definiert und sind **überall** gültig. **Lokale** Variablen sind nur **innerhalb** der Funktion oder Klasse gültig.

Ausdrücke

Berechnung (3*2)
Funktionsaufruf (func())
Benennungen (*variablenName*)
Diese können auch beliebig kombiniert werden.

Datentypen

Integer	**Ganze Zahlen**	2 oder -5
String	**Text** in Hochkommas	"Hallo" oder "2"
Float	(Fließ-) **Kommazahlen**	1.23453 oder 2.0

List	Listen (beliebige Länge und Elemente)
[1, "Hallo", 3.0]	Listen werden von **[ ]** umgeben und die Elemente durch ein Komma getrennt.
Dictionary	Weist Werten (**Keys**) andere Werte (**Values**) zu.
{"Key1":1, "-Key2":"W-ert2"}	Ähnlich zu Listen. Anstelle von [ ] wird **{ }** verwendet und der Wert eines Schlüssel wird mit **:** zugewiesen.
None	Nichts `None`

Nutzereingaben

input()	Gibt Eingabe des Nutzers als **String** zurück
input(<Text>)	Gibt vor der Nutzereingabe Text aus

Bedingte Verzweigung

if Bedingung:	Wenn die **True** ergibt wird der eingerückte Block ausgeführt.
else:	Else folgt immer auf eine if-Bedingung und ist optional. Der eingerückte Block nach else wird ausgeführt, wenn der if-Block nicht ausgeführt wurde,
elif Bedingung:	Steht vor else und nach if. Elif verhält sich wie if, wird aber nur ausgeführt, wenn vorherige if oder elif Blöcke nicht ausgeführt wurden.

Bedingte Verzweigung Beispiel

```
def test(a):
    if a == 2:
        print("a ist 2)
    elif a%2 == 0:
        print("a ist durch 2 teilbar, aber nicht genau 2.")
    elif a%3 == 0:
        print("a ist durch 3 teilbar, aber nicht durch 2.")
    else:
        print("a ist nicht durch 2 oder durch 3 teilbar.")
```

Schleifen

for variable in range(start, ende, schrittgröße):

Die Varible wird immer bei jedem Durchlauf der Schleife aktualisiert. Sie startet bei dem Startwert und bei jedem Durchlauf um Schrittweite erhöht. Dabei wird der eingerückte Block jedesmal ausgeführt. Es ist zu beachten, dass wenn die Variable auf den Wert Ende gesetzt wird, der Block nicht nochmal ausgeführt wird.

range(ende)	= range(0, ende, 1)
range(start, ende)	= range(start, ende, 1)

while <Bedingung>:

While-Schleifen führen den eingerückten Block aus, wenn die Bedingung zu True berechnet wird. Dies wird vor jedem Durchlauf überprüft.

Cheatography

Python für Beginner Cheat Sheet

by max3l via cheatography.com/121235/cs/22126/

Funktionen und Rückgabewerte

`def func():`	Definition der Funktion func ohne Parameter
`def func(arg1):`	Definition der Funktion func mit einem Parameter
`def func(arg1, arg2):`	Es können beliebig viele Parameter verwendet werden

Parameter sind **Variablen**, welche zur Laufzeit an übergebene **Argumente** gebunden werden und nur innerhalb der Funktion gültig sind.

`func()`	Aufruf der Funktion func ohne Argumente
`func(2)`	Aufruf der Funktion func mit dem Integer 2 als Argument

Funktionen werden durch **( )** ausgeführt. Wenn **Argumente**, also Werte an die Funktion übergeben werden soll, stehen diese in der Klammer.

`return`	Beendet eine Funktion ohne etwas zurück-zugeben
`return 2`	Beendet eine Funktion und gibt den Wert 2 zurück
`return name`	Beendet eine Funktion und löst name auf

`return` beendet eine Funktion und gibt den ausgewerteten Wert des anschließenden Ausdrucks zurück. Falls eine Primitive (Integer, String, Float, ...) dort steht wird diese zurückgegeben.

`var = func()`	Der Rückgabewert (Wert von return) wird in "var" gespeichert

Ausgeben - Zurückgeben

Ausgeben	Etwas dem Nutzer anzeigen (**print**)
Zurück-geben	Etwas beim Beenden an den aufrufenden übergeben (**return**)

Vergleichsoperatoren

==	Testet auf Wertegleichheit
!=	Testet auf Werteungleichheit
<=	Testet ob der Wert links kleiner gleich dem Wert rechts ist
>=	Testet ob der Wert links größer gleich dem Wert rechts ist
<	Testet ob der Wert links kleiner als der Wert rechts ist
>	Testet ob der Wert links größer als der Wert rechts ist

Vergleichsoperatoren (cont)

== und != funktionieren auch bei anderen Datentypen als bei Integers.