PYTHON GRUNDLAGEN – CHEAT SHEET

GRUNDTYPEN

int	70	0	-300
float	9.45	0.0	
bool	True	False	
str	„Hello World!“		

BOOLEAN

`a and b` – muss beides gelten
`a or b` – eins der beiden muss gelten
`not a` – nicht a muss gelten
→ Vergleiche:
`< > <= >= == !=`

INDEXING in Sequenzen

```
lst = [10,20,30,40,50]
lst[0] → 10
lst[1:3] → [20,30,40]
lst[:] → [10,20,30,40,50]
lst[:3] → [10,20,30]
lst[3:] → [40,50]
```

LISTEN, TUPEL, DICT's, SET

```
list=[1,2,3]
tuple=(1,2,3)
dict={1:„eins“,2:„zwei“}
set = {1,2}
```

DISPLAY

`print(„v“,x)` sep=“ “ trennt die Elemente; end=“\n“ Ende des Prints

LISTEN – geordnet, änderbar, erlauben Duplikate

1. `lst.append(val)` – fügt den Wert am Ende ein
2. `lst.extend(seq)` – fügt eine Sequenz dem Ende zu
3. `lst.insert(idx,val)` – fügt einen Wert an Position idx
4. `lst.remove(val)` – löscht den Wert
5. `lst.pop([idx])` - löscht den Wert & gibt den Wert an Position idx aus (default: letzter Wert)
6. `lst.sort()` – sortiert die Liste

TUPEL– geordnet, unveränderbar, erlauben Duplikate

1. `tupel.count(val)` – zählt val im Tupel
2. `tupel.index(val)` – liefert den Index von

VARIABLEN ZUWEISEN

`x=2+10+cos(y)` – normale Zuweisung
`a=b=c=0` – gleiche Zahl mehreren Variablen zuweisen
`y,z,r = 9,10,11` – mehrere Zuweisung gleichzeitig
`x+=3` – Inkrementieren <-> `x=x+3`
`x-=2` – Dekrementieren <-> `x=x-2`

SEQUENZ OPERATIONEN

`len(c)` – Länge von c
`min(c)max(c)`
`sum(c)`
`sorted(c)` – sortiert c
`val in c` – schaut, ob v in c (Gegenstück: not in)

TYPKONVERSION

Allgemein gilt: type(Inhalt)

Beispiele:

1. `int(„15“)` → 15 von String zu Int
2. `float(„11.5“)` → 11.5 von String zu float
3. `round(15.56,1)` → 15.6 Runden der Zahl
   Die Eins steht dabei für die Nachkommastellen
4. `list(„abc“)` von String zur Liste
   → `[ 'a' , 'b', 'c' ]`
5. `dict([(3,“drei“),(2, „zwei“)])` von String zum Dictionary
   → `{2:'zwei', 3:'drei'}`
6. `set([„eins“,„zwei“])` von String zum Set
   → `{'eins', 'zwei'}`
7. `':'.join(['toto','12','pswd'])` Liste von Strings in einen ganzen String packen
   →`'toto:12:pswd'`
8. `list of str "words with spaces".split()`
   String in Liste abspeichern
   → `['words','with','spaces']`
9. `list of str "1,4,8,2".split(",")`
   → `['1','4','8','2']`
10. `[int(x) for x in ('1','29','-3')]`
    → `[1,29,-3]`

EINLESEN

```
s = input(„Alter:“)
```
→ Input ist immer String!

F STRINGS

```
f"Hello, {name}. You are {age}."
```

DICTIONARIES – geordnet, änderbar, Duplikate nicht erlaubt

1. `d[key]= value` dem Schlüssel wird ein Wert zugwiesen
2. `d[key]` der Wert wird ausgespuckt
3. `d.keys()` alle Schlüssel des Dictionaries werden ausgegeben
4. `d.values()` alle Werte des Dictionaries werden ausgegeben
5. `d.items()` gibt einen Tupel mit Schlüssel – Wert Paar zurück
6. `d.clear()` löscht alle Elemente des Dictionaries
7. `del d[key]` löscht den Eintrag mit dem Schlüssel key
8. `d.pop(key)` löscht den Eintrag mit dem Schlüssel key
9. `d.popitem()` löscht das zuletzt hinzugefügte Schlüssel – Wert paar
10. `d.get(key)` gibt den Wert des Schlüssels aus

SETS – ungeordnet, unveränderbar, Duplikate nicht erlaubt

1. `s.update(s2)` – aktualisiert ein Set mit einem anderen
2. `s.add(key)` – ein neues Element wird zu dem Set hinzugefügt
3. `s.remove(key)` – ein Element wird gelöscht
4. `s.clear()` – alle Elemente des Sets werden gelöscht
5. `s.pop()` – ein Element wird gelöscht
6. `s.union(s2)` – gibt ein Set mit der Vereinigung beider Sets, ohne Wiederholung

FUNKTIONEN

```
def fkt(x,y,z):
    """Doku"""
    Code
    return res
```

DATEIEN

`f.write()` – schreibt in die Datei
`f.read()` – liest von der Datei

```
with open (…) as f:
    for line in f:
        Anweisungen
```

IF-ELSE ANWEISUNGEN

eine Anweisung wird nur dann ausgeführt, wenn eine Kondition stimmt:

```
if logical condition:
    Anweisung

if alter < 18:
    state = „Kind“
elif alter < 25:
    state = „Jugendlich“
else:
    state = „Erwachsen“
```

WHILE SCHLEIFE

wird so lange die Bedingung wahr ist ausgeführt

```
while condition:
    Anweisung

while i < = 100:
    s = s+i**2
    i = i+1
print(„sum:“, s)
```

→ Variable i muss aktualisiert werden, sonst Endlosschleife

FEHLERBEHANDLUNG

```
try:
    Anweisung
except Exception:
    Fehlerbehandlung
```

finally Block wird immer ausgeführt!

FOR SCHLEIFE

Für jedes Element in einer Liste /Set/Dictionary/… wird eine Anweisung ausgeführt

```
for var in sequene:
    Anweisung
```

```
lst = [11,18,9,12,23,4,17]
lost = []
for idx in range(len(lst)):
    val = lst[idx]
    if val > 15:
        lost.append(val)
        lst[idx] = 15
print("modif:",lst,"-
lost:",lost)
```

RANGE

```
range(start,end,step)   range(2,10,3)- 2,5,8
range(3)  - 1,2
range(3,7)  - 3,4,5,6
```

SCHLEIFEN KONTROLLE

break – es wird aus der Schleife gesprungen
continue – springt zur nächsten Iteration