THU
Technische
Hochschule
Ulm

Vorkurse der Fakultät Informatik
Programmier-Crashkurs – Cheatsheet

September 2025

M. Reiter

Datentypen

Typ	Schlüsselwort	Beispiel
Integer	int	zahl = 10
Float	float	gleitkomma_zahl = 3.14
String	str	text = "Hallo welt!"
Boolean	bool	wahrheitswert_1 = True wahrheitswert_1 = False
Liste	list	liste_zahlen = [1, 2, 3, 4] liste_gemischt = ["hallo", "welt", 1, 2, 3] liste_verschachtelt = [[1, 2], [3, 4, 5]]
Dictionary	dict	d = { "name": "Hans", "age": 12 }

In Python werden Schlüsselwörter der Datentypen nur verwendet, um diese explizit festzulegen, z.B. beim Einlesen einer Benutzereingabe.

Operatoren

Operator	Bedeutung	Beispiel
=	**Zuweisung**	zahl_a = 10 zahl_b = zahl_a + 2
+	**Addition,** **Verbindung von Zeichenketten**	summe = 2 + 3 text = "Hallo" + " " + "welt!"
-	**Subtraktion**	differenz = 7 - 4
*	**Multiplikation**	produkt = 2 * 3
/	**Division**	quotient = 6 / 3
%	**Modulo**	rest = 4 % 3
==	**Gleichheitsoperator**	wahrheitswert = (2==3)
!=	**Ungleichheitsoperator**	wahrheitswert = (2!=3)
<	**Kleiner-als-Operator**	wahrheitswert = (2 < 3)
>	**Größer-als-Operator**	wahrheitswert = (2 > 3)
<=	**Kleiner-gleich-Operator**	wahrheitswert = (2<=3)
>=	**Größer-gleich-Operator**	wahrheitswert = (2>=3)

Kommentar, Eingabe, Ausgabe

```
# Hier steht ein Kommentar! Der wird vom Programm nicht ausgeführt

eingabe = input("Text für Benutzer!") # Erstmal immer eine Zeichenkette!
zahl = int(input("Text für Benutzer!")) # Explizit festgelegter Datentyp

print(2)      # Ausgabe Zahl
print(3.14)   # Ausgabe Fließkommazahl
print("Text") # Ausgabe Zeichenkette
print(2+3)    # Rechnung und Ausgabe

name = "Tom"
alter = 21
print(f"{name} ist {alter} Jahre alt!") # mit f-Strings

print("Hello, \nworld!") # mit Zeilenumbruch durch \n

print("Hello world!", end=" ") # Leerzeichen statt Zeilenumbruch am Ende
der Ausgabe, die nächste Ausgabe bleibt in derselben Zeile!
```

Namenskonventionen

Variablen und Funktionsnamen mit snake_case/lowercase_with_underscores:
`meine_zahl = 10` **oder** `berechne_rabatt()`

Konstanten werden mit UPPERCASE_WITH_UNDERSCORES geschrieben:
`PI = 3.14` **oder** `MAX_GESCHWINDIGKEIT = 10`

Klassen-Namen: CamelCase

```
class MeineKlasse:
      pass
```

Dateinamen und Module: kleine Buchstaben und **Unterstriche:** aufgabe_1_1.py

Bedingte Verzweigungen

```
if x > 10:
    print("x ist größer als 10")
elif x == 10:
    print("x ist 10")
else:
    print("x ist kleiner als 10")
```

Bedingungen benötigen einen Wahrheitswert, z.B. Vergleich mit passendem Operator.

Schleifen

for-Schleife:

```
for i in range(0, 5):
    print(i)  # Gibt 0 bis 4 aus
```

while-Schleife:

```
i = 0
while i < 5:
    print(i)
    i += 1
```

Funktionen

Funktion definieren:

```
def greet(name): #‘name‘ ist Übergabeparameter
      text = f"Hallo, {name}!"
      return text # Rückgabewert
```

Funktion aufrufen:

```
print(greet("Philipp"))
# Funktion über den Funktionsnamen aufrufen
```

Range

```
range(start, stop, step) # Erzeugt ein iterierbares („durchlaufbares“) Zahlen-Objekt („Iterator“)
```

- **start** (optional): Die Zahl, bei der die Sequenz beginnen soll (standardmäßig 0).
- **stop**: Die Zahl, bei der die Sequenz enden soll (diese Zahl wird **nicht** mit einbezogen).
- **step** (optional): Die Schrittgröße, mit der die Sequenz generiert wird (standardmäßig 1).

```
range(5)      # -> Iterator von 0 bis einschließlich 4
range(1,5)    # -> Iterator von 1 bis einschließlich 4
range(2,5,2) # -> Iterator über 2 und 4

liste = list(range(0, 10)) # Wandelt den Iterator explizit zu einer Liste von ints [0, …, 9]
```

Random

```
import random # importiert Modul ‚random‘
zufall = random.randrange(zahl_a, zahl_b) # ruft Funktion von ‚random‘ auf
# Erzeugt Zufallszahl zwischen zahl_a bis zahl_b
# Achtung: zahl_b ist dabei nicht mehr eingeschlossen!
```

Listen

```
liste_a = []                              # leere Liste
liste_b = ["a", "b", "c"]                 # Liste mit 3 Elementen
liste_c = [["a", "b"], ["c", "d", "e"]] # verschachtelte Liste

print(liste_b[1])     # -> ‚b‘ Wert an Index 1 ausgeben
print(liste_c[1][1]) # -> ‚d‘ Wert an Index 1.1 ausgeben

liste_a.append(2).    # ‚2‘ hinten an Liste anhängen
liste_a.insert(0, 3) # ‚3‘ an Index 0 einfügen
liste_a[0] = 1 # Wert an Index 0 mit ‚1‘ überschreiben (nur, wenn Stelle in Liste schon vorhanden!)

len(liste_a) # Anzahl der Elemente in der Liste; hier: 2 Elemente d.h. der letzte Index ist ‚1‘
```