# Datentypen und Operatoren

Im Folgenden wird in die Datentypen und arithmetischen Operatoren der Programmiersprache Python eingeführt.

# Zahlen

Bisher haben wir nur mit sogenannten Strings gearbeitet, also Zeichenketten. Python nennt diesen Datentyp `str`. Um jedoch komplexere Programme zu schreiben, muss man mit Zahlen rechnen können. Python stellt uns dazu zwei andere Datentypen zur Verfügung.

## Ganze Zahlen

Eine einfache Zahl als Folge der Ziffern `0` bis `9` im Programm liest Python diese als eine ganze Zahl, auf Englisch 'integer' und in Python `int`.

```
some_int = 0
some_other_int = 100000000000000000000000000000
yet_another_int = 3
```

Zu beachten ist, dass anders als die Programmiersprache C (welches wohl die erste war, die 'ganze Zahlen' `int` nannte),

- Python keine Zahl-Datentypen mit verschiedenen 'Größen' hat
- `int` Zahlen in einem unglaublich großen Bereich darstellen kann (mehrere Millionen Ziffern im Dezimalsystem).

Der Unterstrich `_` kann in einer Zahl in jeder Stelle (außer ganz am Anfang) eingefügt werden, und wird ignoriert. Beginnt eine Zahl mit `0x`, wird sie als Hexadezimalzahl gelesen (Ziffern `0`-`9` und `a`-`f` bzw. `A`-`F`). Beginnt sie mit `0o` oder `0b`, ist sie zur Basis 8 (oktal) oder 2 (binär) geschrieben.

```
some_int = 100
the_same_int = 1_0_0
the_same_again = 0x64
```

## Gleitkommazahlen

Es ist dem Computer nicht möglich, Reelle Zahlen zu speichern. Rationale Zahlen sind als Paar zweier `int`-Werte möglich, jedoch ist dies langsam und speicherintensiv (und noch langsamer, wenn man versucht, jeden Bruch auf den keinsten Nenner zu bringen). Eine Alternative dazu sind fixed-point-Zahlen, wobei mit ganzen Zahlen gerechnet wird, die jeweils den Zähler eines Bruches mit festem Nenner beschreiben. Eine nicht ganz offensichtliche (uhd häufig nicht ideale, aber gut funktionierende) Methode ist die Verwendung von Gleitkommazahlen, welche in Python durch den Datentyp `float` beschrieben werden. Diese haben intern die Form `s * m * 2^e`, wobei

- `s`, das Vorzeichen, -1 oder 1 ist
- `m`, die Mantissa, als Binärzahl in der Form von `1.xyz` gespeichert wird (mit 52 Bits als Nachkommastellen)
- `e`, der Exponent, einen Wert zwischen -1022 und 1023 hat und als Binärzahl mit Offset 1023 gespeichert wird

Fast alle Operationen mit Gleitkommazahlen sind ungenau, insbesondere das Umwandeln zwischen ihnen und Zahlen im Dezimalsystem. Ein bekanntes Beispiel dafür ist auf

0.30000000000000004.com zu finden.

---

Version #4
Erstellt: 2026-05-10 08:34:13 UTC von Fabian Schiener
Zuletzt aktualisiert: 2026-05-10 21:02:40 UTC von Fabian Schiener