

WRONG TIMES TABLE  
THE INCORRECT ANSWERS THAT  
FEEL MOST RIGHT TO ME

	1	2	3	4	5	6	7	8	9	10
1	0	½	4	5	6	7	8	9	10	9
2	½	8	5	6	12	14	12	18	19	22
3	4	5	10	16	13	12	24	32	21	33
4	5	6	16	32	25	25	29	36	28	48
5	6	12	13	25	50	24	40	45	40	60
6	7	14	12	25	24	32	48	50	72	72
7	8	12	24	29	40	48	42	54	60	84
8	9	18	32	36	45	50	54	48	74	56
9	10	19	21	28	40	72	60	74	72	81
10	9	22	33	48	60	72	84	56	81	110

Deep in some corner of my heart, I suspect that real times tables are wrong about  $6 \times 7 = 42$  and  $8 \times 7 = 56$ .

## 10. Rechnen

Mit einer Programmiersprache können wir natürlich auch Berechnungen durchführen. Dies war schliesslich auch die ursprüngliche Motivation für die Konstruktion von Computern<sup>1</sup>. In diesem Kapitel beschäftigen wir uns hauptsächlich mit den grundlegenden Rechenarten, welche wir alle aus dem Mathematikunterricht kennen. Die Lernziele lauten:

- ☐ Sie erstellen Python-Programme, in denen Sie mit Zahlen rechnen.
- ☐ Sie erklären, was ein arithmetischer Operator ist und geben ein Beispiel.
- ☐ Sie erklären, was ein arithmetischer Ausdruck ist und geben ein Beispiel.

### 10.1 Addition, Subtraktion, Multiplikation, Division und Potenzierung

Wie beim Taschenrechner gibt es für jede Rechenart ein Zeichen.

- Addition: +
- Subtraktion: -
- Multiplikation: \*
- Division: /
- ganzzahlige Division: //
- Potenzierung: \*\*

**Definition 9 — Arithmetischer Operator.** Der Fachbegriff für „Rechenzeichen“ lautet in der Programmierung arithmetischer Operator. Mit arithmetischen Operatoren notieren wir mathematische Rechnungen.

Die Arithmetik ist ein Teilgebiet der Mathematik und beschreibt das Rechnen mit Zahlen.

■ **Beispiel 10.1** Listing 10.1 zeigt Beispiele in Kombination mit `print`-Funktionsaufrufen.

```
1 print(5 + 3)
2 print(f"Ergebnis: {5 - 3}")
3 print(f"Erst Punkt: {3.7 + 5 * 3}")
4 print(5 / 3)
5 print(2 ** 16)
```

Listing 10.1: `rechenbeispiel_1.py`

```
8
Ergebnis: 2
Punkt zuerst: 18.7
1.6666666666666667
65536
```

Listing 10.2: Konsolenausgabe

<sup>1</sup> „to compute“ kann mit rechnen oder etwas berechnen übersetzt werden. Im deutschen Sprachraum ist auch Rechner als Bezeichnung für den Computer üblich.

In der Konsole (siehe Listing 10.2) ist zu sehen, dass Zahlen nicht immer exakt berechnet werden. Ausserdem kann man erkennen, dass Python die „Punkt-vor-Strich-Regel“ beachtet (Rechnung in Zeile 3). Formatierter Text ist auch in der Lage, eine Rechnung zu beinhalten. Die Rechnung wird automatisch ausgerechnet.

**Wichtig!** Möchte man „Kommazahlen“ in Rechnungen verwenden, so muss man einen Punkt für das Komma verwenden!

### 10.1.1 Ganzzahlige Division, Klammern und Variablen

Die Division mit dem doppelten Schrägstrich (eng. double forward slash) bewirkt, dass nur der **ganzzahlige Anteil** der Division als Ergebnis herauskommt. Es wird also die gewöhnliche Division berechnet und dann zur **nächsten ganzen Zahl abgerundet** (mathematische Abrundungsfunktion) benutzt. Möchte man die Reihenfolge der Berechnungen beeinflussen, dann kann man wie in der Mathematik **runde** Klammern verwenden. Natürlich kann man bei Rechnungen auch Variablen und Werte kombinieren.

```
1 import random as r
2
3 print(7 // 2)
4 print(1 // 2)
5 print(9.0 // 3)
6 a = r.randrange(1, 11)
7 b = r.randrange(1, 11)
8 ergebnis = (a + b) * (a - b)
```

```
1 import random as r
2
3 a = r.randrange(1, 101)
4 print(f"Seitenlänge: {a}")
5 umfang = a + a + a + a
6 flaeche = a ** 2
7 print(f"Umfang: {umfang}")
8 print(f"Fläche: {flaeche}")
```

Abbildung 10.1: Weitere Beispiele wie man in Python rechnen kann ( `rechenbeispiel_2.py` und `quadrat_berechnungen.py` ). Bei einer Zuweisung wird immer **zuerst** die **rechte Seite** des Zuweisungsoperators ausgewertet. Das Ergebnis wird dann in der Variablen auf der **linken Seite** des Zuweisungsoperators gespeichert.


**Wichtig!** Im Gegensatz zur Mathematik müssen immer alle **arithmetischen Operatoren** notiert werden. Ein fehlender arithmetischer Operator, wie in folgendem Code,

```
1 import random as r
2
3 x = r.randrange(1, 11)
4 ergebnis = 4x
```

erzeugt einen Fehler.

### 10.1.2 Clean Code

Eine weitere Regel vereinheitlicht die Darstellung einer Rechnung.

 **Clean Code 6 — Leerzeichen 3.** Links und rechts eines arithmetischen Operators fügen wir je ein Leerzeichen ein.

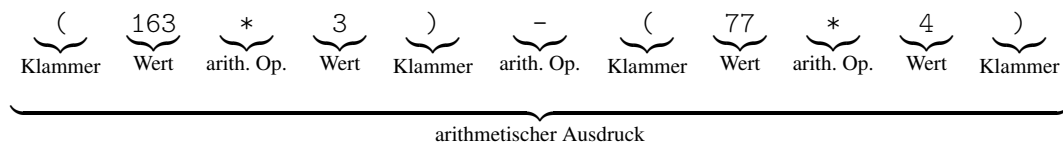
■ **Beispiel 10.2** Wir notieren die Addition von 5 und 3 somit wie folgt: `5 + 3`. Falsch, im Sinne von Clean Code, wäre `5+3`.

### 10.1.3 Arithmetischer Ausdruck

Mathematische Rechnungen sind Programmierbefehle, die in Python eine eigene Kategorie darstellen.

**Definition 10 — Arithmetischer Ausdruck.** Rechnungen mit **arithmetischen Operatoren** werden arithmetische Ausdrücke (eng. arithmetic expressions) genannt. Bei der Programmausführung werden arithmetische Ausdrücke stets von Python direkt **ausgewertet**. Dies bedeutet, Python ermittelt für den arithmetischen Ausdruck einen **Wert** („das Ergebnis der Rechnung“). Bei arithmetischen Ausdrücken ist der Wert immer eine Zahl (entweder eine ganze Zahl oder eine Fließkommazahl).

■ **Beispiel 10.3** Schauen wir uns  $(163 * 3) - (77 * 4)$  im Detail an:



Mit arith. Op. ist der Begriff arithmetischer Operator gemeint. Würden wir den arithmetischen Ausdruck mit einem `print`-Funktionsaufruf kombinieren, dann würde Python in der Konsole die Zahl 181 ausgeben.

```
1 print((163 * 3) - (77 * 4))
```

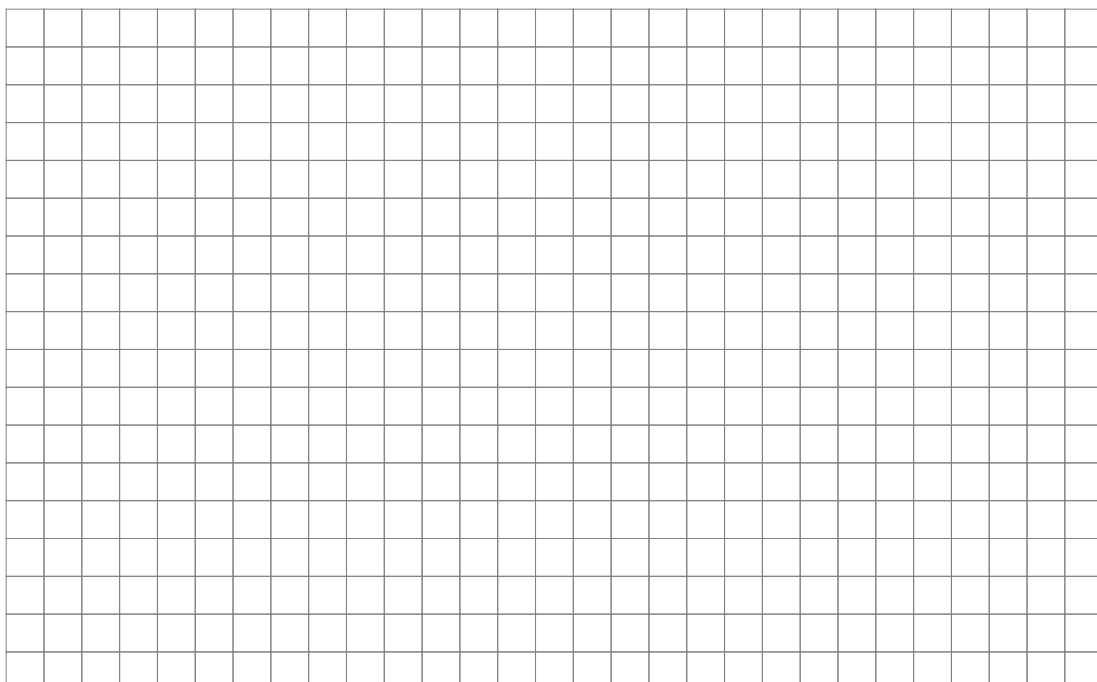
■

## 10.2 Aufgaben

In den folgenden Aufgaben setzen Sie sich mit den arithmetischen Operatoren auseinander.

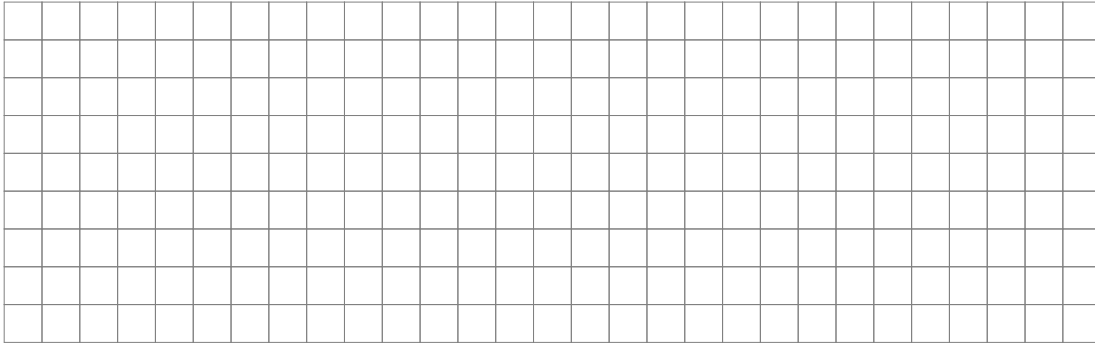
### 10.2.1 Aufgabe 1

Notieren Sie ein Programm, welches das **Volumen** und den **Oberflächeninhalt** eines **Würfels** berechnet. Verwenden Sie für die Kantenlänge  $a$  eine zufällige Zahl zwischen 1 und 100. Das Volumen berechnet sich durch  $a^3$  und der Oberflächeninhalt mit  $6 \cdot a^2$ . Verwenden Sie Variablen und eine saubere Konsolenausgabe (siehe Abbildung 10.1).



### 10.2.2 Aufgabe 2

Notieren Sie ein Programm, welches ein **regelmässiges Siebeneck** zeichnet. Die Seitenlänge 150 soll in einer Variablen gespeichert sein. Der Drehwinkel der Turtle soll durch einen arithmetischen Operator berechnet und in einer Variablen gespeichert werden. Verwenden Sie dann eine **for**-Schleife mit einem **range**-Funktionsaufruf, um das 7-Eck zu zeichnen.

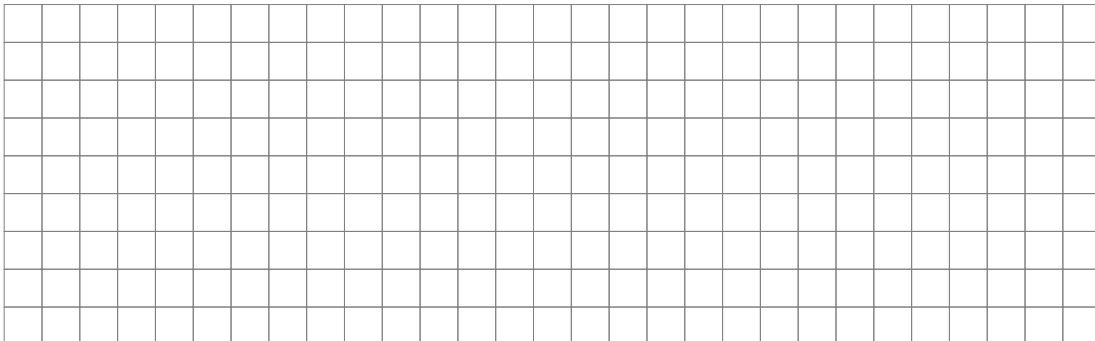


### 10.2.3 Aufgabe 3

Notieren Sie ein Programm, welches von drei zufälligen Zahlen das arithmetische Mittel („Durchschnitt“) berechnet und in der Konsole ausgibt. Sie müssen dazu die drei Zahlen addieren und durch 3 teilen. Die Ausgabe in der Konsole sollte dann wie folgt lauten:

Das arithmetische Mittel von 34, 31 und 52 ist 39.0.

Die Zahlen sind nur ein Beispiel. Das Programm muss für drei beliebige Zahlen funktionieren.



## 10.3 Der Modulooperator

Beim Umrechnen einer Dezimalzahl zu einer Dualzahl gibt es ein Verfahren, welches wiederholt die ganzzahlige Division mit Rest durchführt. Bei diesem Verfahren ist man insbesondere am Rest der Division interessiert, da man dadurch die Dualzahl erhält (siehe Skript über „Digitalisierung“). Wir können das Verfahren auch programmieren. Mit dem Modulooperator können wir unkompliziert den Rest einer Division bestimmen. Listing 10.3 zeigt vier Beispiele. Die dazugehörige Ausgabe ist in Listing 10.4 dargestellt.

```
1 print(5 % 2)
2 print(19 % 2)
3 print(18 % 6)
4 print(20 % 9)
```

Listing 10.3: Das Prozentzeichen ist der Modulooperator.

```
1
1
0
2
```

Listing 10.4: Reste der ganzzahligen Divisionen.



