Sammlungen

by Woche 2

In Python gibt es vier grundlegende Arten von Sammlungen (Collections):

- Listen
- Tuples
- Sets
- Dictionaries

Alle haben gemeinsam, dass sie mehrere Werte in einer einzigen Variablen speichern können. Einfach ausgedrückt kann man also beispielsweise von einer Spalte in einer Tabelle sprechen. Die Unterschiede liegen in der Art und Weise, wie die Werte gespeichert und abgerufen werden können. Listen sind die am häufigsten verwendete Art von Sammlungen in Python. Die anderen drei haben spezielle Eigenschaften, die sie für bestimmte Anwendungsfälle nützlich machen. In den folgenden Kapiteln werden wir uns jede Art von Sammlung genauer ansehen.

> **i** Hinweis
>
> Die Grundlagen bis hierhin und vor allem diese Kapitel zu den verschiedenen Sammlungen wirken ggf. etwas zu theoretisch und abstrakt. Es ist klar, dass für eine bessere Motivation möglichst bald Kapitel kommen sollten, in denen wir mit echten Daten arbeiten und der praktische Bezug klarer wird. Nichtsdestotrotz ist es wichtig, diese Grundlagen zu kennen, um später effizient und effektiv mit Python arbeiten zu können. Es muss nicht alles was nun folgt sofort auswendig gelernt werden, aber viele der Funktionalitäten oder zumindest ihrer Konzepte werden wir auch später noch regelmäßig brauchen, sodass es sich lohnt, sich damit auseinanderzusetzen.

Hier ist eine tabellarische Übersicht über die Unterschiede und Anwendungen der vier Arten von Sammlungen:

Homogenität

Alle vier Arten von Sammlungen müssen nicht homogen sein, was bedeutet, dass sie Werte unterschiedlicher Datentypen enthalten können. In Python ist es also möglich, eine Liste zu erstellen, die z.B. sowohl Zahlen als auch Strings enthält. Das gleiche gilt für Tuples, Sets und Dictionaries und natürlich noch mehr Datentypen.

```
# heterogene Sammlungen
liste_het = [1, 'zwei', 3.21]
tuple_het = (1, 'zwei', 3.21)
set_het = {1, 'zwei', 3.21}
dict_het = {1: 1, 'zwei': 2, 'drei': 'drei'}
```

```
# homogen ist natürlich auch möglich
list_hom = [1, 2, 3]
tuple_hom = (1, 2, 3)
set_hom = {1, 2, 3}
dict_hom = {'eins': 1, 'zwei': 2, 'drei': 3}
```

Verschachtelung

Alle vier Arten von Sammlungen können auch verschachtelt werden, d.h. eine Liste kann auch eine Liste enthalten. Da sie auch nicht homogen sein müssen, können sie auch eine Sammlung enthalten, die eine andere Sammlung enthält, die eine andere Sammlung enthält, und so weiter.

```
liste1 = [1, 2, 3]
liste2 = [1, [2, 3], 4]
liste3 = [1, [2, 3], 4, [[5, 6], 7]]
tuple1 = ({1}, (2, 3), liste3)

print(tuple1)
```

```
({1}, (2, 3), [1, [2, 3], 4, [[5, 6], 7]])
```

Es gibt allerdings eine Einschränkung bzgl. der Verschachtelung von Sammlungstypen. Listen und Tuples können problemlos andere Sammlungen enthalten, einschließlich anderer Listen, Tuples, Sets und Dictionaries. Sets und Dictionaries hingegen können nur immutable Datentypen enthalten, d.h. keine Listen, Sets oder Dictionaries.

Bei den anderen in der Tabelle aufgeführten Aspekten unterscheidet sich immer mindestens ein Typ Sammlung von den anderen, sodass wir dies in den folgenden Unterkapiteln genauer betrachten werden.