Tuples

by Woche 2

Der zweite Typ von Sammlungen in Python sind Tuples. Tuples sind Listen sehr ähnlich: Sie sind nicht homogen, Verschachtelungen sind möglich, sie sind geordnet und können Duplikate enthalten. Tuples unterscheiden sich aber auch von Listen: Sie sind immutable und werden mit runden Klammern `()` erstellt.

Arbeiten mit Tuples

Tuples können prinzipiell auf die gleiche Weise wie Listen erstellt und via indices behandelt werden. Ebenso können bestimmte Funktionen wie `len()` und `sum()` genutzt werden.

```
mein_tuple = ('Etwas', 42, 'Text', 42, 12.6, True)
mein_tuple
```

```
('Etwas', 42, 'Text', 42, 12.6, True)
```

```
zahlen_tuple = (1, 6, 3, 2)
sum(zahlen_tuple[2:])
```

```
5
```

Listen können auch in Tuples umgewandelt werden und umgekehrt. Dabei wird die ursprüngliche Sammlung nicht verändert, sondern eine neue Sammlung erstellt.

```
x = [1, 2, 3]
y = tuple(x)
print(x)
```

```
[1, 2, 3]
```

```
print(y)
```

```
(1, 2, 3)
```

```
x = (1, 2, 3)
y = list(x)
print(x)
```

```
(1, 2, 3)
```

```
print(y)
```

```
[1, 2, 3]
```

Immutable

Der markante Unterschied zu Listen ist, dass Tuples immutable sind. Demzufolge haben Tuples auch keine Methoden wie `append()`, `remove()`, `del()` oder `pop()`, da diese Methoden die Sammlung verändern würden.

```
# Füge ein Element hinzu
x = (1, 2, 3)
x.append(4)
```

```
Error: 'tuple' object has no attribute 'append'
```

```
# Lösche das erste Element
x = (1, 2, 3)
del(x[0])
```

```
Error: 'tuple' object doesn't support item deletion
```

Wenn man aber doch einen Tuple um Werte erweitern möchte oder bestimmte Werte löschen möchte, kann man das tun, indem man eine veränderte Kopie des Tuples erstellt. Man kann dann auch die alte Version mit der neuen überschreiben. Der `+` Operator funktioniert in diesem Fall genauso wie bei Listen und mittels Indizierung kann auch ein Slice des Tuples erstellt werden.

```
# Füge ein Element hinzu
x = (1, 2, 3)
x = x + (4,)
x
```

```
(1, 2, 3, 4)
```

```
# Lösche das erste Element
x = (1, 2, 3)
x = x[1:]
x
```

```
(2, 3)
```

> **Hinweis**
>
> Um einen Tupel mit nur einem Element zu erstellen, muss ein Komma hinter dem Element stehen. Andernfalls wird das Element als normaler Wert - also nicht als Tupel - interpretiert. Deshalb steht im Code oben `(4,)` anstatt `(4)`. Um es noch verwirrender zu machen: In diesem Fall würde sogar `4,`, also ohne Klammern, ausreichen. Meines Erachtens ist das aber nicht so intuitiv und sollte deshalb vermieden werden.

Vorteil zu Listen

Ein Vorteil von Tuples gegenüber Listen ist, dass sie weniger Speicherplatz einnehmen und schneller erzeugbar sind. Grund dafür ist eben, dass sie immutable sind. Hier zum Beweis ein kleiner Performance-Test: Mittels der Funktion `timeit()` aus dem gleichnamigen Modul wird die Zeit in Sekunden gemessen, die benötigt wird, um 10.000.000 Mal eine Liste bzw. ein Tuple zu erstellen.

```
import timeit

timeit.timeit(
  stmt = 'x = [1, "two", True]',
  number = 10000000
)
```

```
0.16691379999974743
```

```
#

timeit.timeit(
  stmt = 'x = (1, "two", True)',
```

```
  number = 10000000
)
```

```
0.04501359997084364
```

Weitere Ressourcen

- why are TUPLES even a thing?
- Python Tutorial #21 (deutsch) - Tuple
- 5.3. Tuples and Sequences

Übungen

Tuples sind

- (A) Mutable
- (B) Immutable

Öffne dein Jupyter Notebook aus den Übungen im letzten Kapitel zu den Listen und speichere es als Kopie des ursprünglichen ab. Erzeuge in dieser zweiten Version neben den Listen ‘a’, ‘b’ und ‘c’ auch Tuple ‘x’, ‘y’ und ‘z’ mit gleichem Inhalt. Versuche dann die gleichen Operationen auf den Tuples durchzuführen, die du auf den Listen durchgeführt hast. Wie oben erklärt, wird dies in einigen Fällen nicht direkt möglich sein, da gewisse Funktionen nicht zur Verfügung stehen. Versuche aber trotzdem - mithilfe von entsprechenden alternativen Vorgehensweisen - die gleichen Ergebnisse zu erzielen. Strukturiere das Jupyter Notebook mithilfe von Überschriften und kurzen Erläuterungen so, dass es für jeden Schritt eine Gegenüberstellung der Vorgehensweisen zwischen Listen und Tuple gibt. Ziel ist es, dass du dieses Dokument später als Referenz nutzen kannst, falls du dich mal nicht mehr an die Unterschiede erinnern solltest.

- (A) Geschafft