Python für Linguisten

Dozentin: Wiebke Petersen & Co-Dozent: Valentin Heinz

5. Foliensatz
sequentielle Datentypen, Dictionaries

Sequentielle Datentypen

- Tupel, Listen und Strings sind sequentielle Datentypen
- Tupel: `(a,b,c,d)`, Liste: `[a,b,c,d]`, String: `abcd`
- Für alle sequentielle Datentypen funktioniert:
  - Indexing (`seq_data[2]`)
  - Slicing (`seq_data[2:4]`)
  - Konkatenation (`seq_data + seq_data`)
  - Wiederholung (`seq_data * 3`)
  - Längenbestimmung (`len(seq_data)`)
  - Test auf Enthaltensein ((`'a' in seq_data`)).

veränderliche / unveränderliche Datentypen

- Tupel und Strings sind unveränderliche Datentypen
- Listen sind veränderlich, sie können verändert werden durch:
  - Zuweisung neuer Elemente (`my_list[1] = 'x'`
  - Verwendung spezieller Methoden auf Listen (`help(list)`):
    - append: `my_list.append('y')`
    - pop: `my_list.pop()`
    - reverse: `my_list.reverse()`
    - sort: `my_list.sort()`
    - extend, insert, remove
- Vorsicht: Bei der Veränderung einer Liste ändert sich nicht ihr Speicherort, dies hat Auswirkungen auf die Evaluation von Variablen:

```
>>> x = 3
>>> y = x
>>> x = 5
>>> y
3
```

```
>>> x = [1,2,3,4]
>>> y = x
>>> x.reverse()
>>> y
[4, 3, 2, 1]
```

Listen-Abstraktion (list comprehension)

- Elegante Methode zur Erzeugung von Listen:

```
[w for w in 'dies ist ein Satz']
```

- Funktioniert auch mit zusätzlicher Bedingung:

```
[w for w in 'dies ist ein Satz'if w in 'aeiou']
```

- Syntax ähnlich zur impliziten Mengendefinition in der Mathematik:

```
[x**2 for x in range(10) if x % 3 == 0]
```

$\{\ ^2|\quad \in [0, ..., 10]$ and $\quad \mod 3 = 0\}$

Dictionaries

- Dictionaries bestehen aus Schlüssel-Objekt-Paaren:

```
>>> De_Nl = {"Haus": "huis", "Stuhl":"stoel", "Fahrrad":"fiets"}
```

- Die Datentypen der Objekte sind beliebig. Die Schlüssel müssen unveränderlich sein.
- Der Zugriff auf die Elemente der Dictionaries erfolgt über die Schlüssel:

```
>>> De_Nl['Haus']
```

```
for x in De_Nl:
    print x

Stuhl
Tasche
Fahrrad
```

```
for x in De_Nl:
    print De_Nl[x]

stoel
tas
fiets
```

Dictionaries in Listen umwandeln und umgekehrt

- Dictionary in Liste:
  - `De_Nl.items()`
    `[('Stuhl', 'stoel'), ('Tasche', 'tas'), ('Fahrrad', 'fiets')]`
  - `De_Nl.keys()`
    `['Stuhl', 'Tasche', 'Fahrrad']`
  - `De_Nl.values()`
    `['stoel', 'tas', 'fiets']`
- Liste in Dictionary:
  - ```
    frucht = ['Apfel','Orange','Zitrone']
    farbe = ['grün','orange','gelb']
    ff = dict(zip(frucht,farbe))
    ```
    ```