Dictionaries

> Beim Dictionary handelt es sich um ein assoziatives Feld. Assoziative Felder werden in verschiedenen Programmiersprachen mit verschiedenen Namen versehen. So spricht man in Java und C++ von einer Map, in Perl und Ruby von einem Hash und wie in Python bezeichnen auch Smalltalk, Objective-C und C# assoziative Arrays als Dictionaries.

Von: http://www.python-kurs.eu/python3_dictionaries.php

Hier weden die Daten (Value) mit einem Schlüssel (Key) verknüpft und angesprochen.

```
myCat = {'size': 'fat', 'color': 'gray', 'disposition': 'loud'}
```

```
type(myCat)
```

```
dict
```

```
myCat
```

```
{'color': 'gray', 'disposition': 'loud', 'size': 'fat'}
```

```
myCat['size']
```

```
'fat'
```

```
'My cat has ' + myCat['color'] + ' fur.'
```

```
'My cat has gray fur.'
```

Als Schlüssel können auch Integer angegeben werden, allerdings können diese frei gewählt werden.

```
spam = {12345: 'Luggage Combination', 42: 'The Answer'}
```

```
spam
```

```
{42: 'The Answer', 12345: 'Luggage Combination'}
```

Listen vs. Dictionaries

Bei Dictionaries gibt es keine Reihenfolge. Bei einem Vergleich mit Listen ist die Reihenfolge wichtig. Bei einem Vergleich von Dictionaries ist die Reihenfolge unerheblich.

```
spam = ['cats', 'dogs', 'moose']
bacon = ['cats', 'dogs', 'moose']
```

```
bacon == spam
```

```
True
```

```
bacon = ['dogs', 'moose', 'cats']
```

```
bacon == spam
```

```
False
```

```
eggs = {'name': 'Zophie', 'species': 'cat', 'age': '8'}
ham = {'species': 'cat', 'age': '8', 'name': 'Zophie'}
```

```
eggs == ham
```

```
True
```

```
spam = {'name': 'Zophie', 'age': 7}
spam['color']
```

```
---------------------------------------------------------------------------

KeyError                                  Traceback (most recent call last)

<ipython-input-14-1619663394c3> in <module>()
      1 spam = {'name': 'Zophie', 'age': 7}
----> 2 spam['color']


KeyError: 'color'
```

```
birthdays = {'Alice': 'Apr 1', 'Bob': 'Dec 12', 'Carol': 'Mar 4'}

while True:
   print('Enter a name: (blank to quit)')
   name = input()
   if name == '':
       break

   if name in birthdays:
      print(birthdays[name] + ' is the birthday of ' + name)
   else:
       print('I do not have birthday information for ' + name)
       print('What is their birthday?')
       bday = input()
       birthdays[name] = bday
       print('Birthday database updated.')
```

```
Enter a name: (blank to quit)
```

Die Methoden keys(), values() und items()

```
spam = {'color': 'red', 'age': 42}
for v in spam.values():
    print(v)
```

```
red
42
```

```
for k in spam.keys():
    print(k)
```

```
color
age
```

```
for i in spam.items():
    print(i)
```

```
('color', 'red')
('age', 42)
```

```
spam.keys()
```

```
dict_keys(['color', 'age'])
```

Zu `dict_keys` siehe: https://docs.python.org/3/library/stdtypes.html#dictionary-view-objects

Oder auch: https://stackoverflow.com/questions/18552001/accessing-dict-keys-element-by-index-in-python3

```
list(spam.keys())
```

```
['color', 'age']
```

Mit der Mehrfachzuweisung wird die Ausgabe sehr einfach:

```
spam = {'color': 'red', 'age': 42}
for k, v in spam.items():
    print('Key: ' + k + '\tValue: ' + str(v))
```

```
Key: color  Value: red
Key: age    Value: 42
```

Das Vorhandensein eines Schlüssels oder Werts im Dictionary ermitteln

```
spam = {'name': 'Zophie', 'age': 7}
'name' in spam.keys()
```

```
True
```

```
'Zophie' in spam.values()
```

```
True
```

```
'color' in spam.keys()
```

```
False
```

```
'color' not in spam.keys()
```

```
True
```

```
'color' in spam
```

```
False
```

Die Methode get()

```
picnicItems = {'apples': 5, 'cups': 2}
'I am bringing ' + str(picnicItems.get('cups', 0)) + ' cups.'
```

```
'I am bringing 2 cups.'
```

```
'I am bringing ' + str(picnicItems.get('eggs', 0)) + ' eggs.'
```

```
'I am bringing 0 eggs.'
```

Ohne die get-Methode

```
'I am bringing ' + str(picnicItems['eggs']) + ' eggs.'
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)

<ipython-input-29-5e4230329398> in <module>()
----> 1 'I am bringing ' + str(picnicItems['eggs']) + ' eggs.'


KeyError: 'eggs'
```

Die Methode setdefault()

```
spam = {'name': 'Pooka', 'age': 5}
spam.setdefault('color', 'black')
```

```
'black'
```

```
spam
```

```
{'age': 5, 'color': 'black', 'name': 'Pooka'}
```

```
spam.setdefault('color', 'white')
```

```
'black'
```

```
spam
```

```
{'age': 5, 'color': 'black', 'name': 'Pooka'}
```

```
spam['color'] = 'white'
spam
```

```
{'age': 5, 'color': 'white', 'name': 'Pooka'}
```

Mit setdefault() kann überprüft werden, ob ein key existiert und belegt wurde. Somit kann nach einem key gesucht werden und wenn dieser noch nicht belegt wurde, wird dieser belegt.

Aufgabe Es sollen die Anzahl von Buchstaben eines Stringes gezählt werden.

Aufgabe Zur Ausgabe von den Buchstaben soll das Modul pprint verwendet werden.

```
theBoard = {'top-L': ' ', 'top-M': ' ', 'top-R': ' ',
            'mid-L': ' ', 'mid-M': ' ', 'mid-R': ' ',
            'low-L': ' ', 'low-M': ' ', 'low-R': ' '}

def printBoard(board):
    print(board['top-L'] + '|' + board['top-M'] + '|' + board['top-R'])
    print('-+-+-')
    print(board['mid-L'] + '|' + board['mid-M'] + '|' + board['mid-R'])
    print('-+-+-')
    print(board['low-L'] + '|' + board['low-M'] + '|' + board['low-R'])
printBoard(theBoard)
```

```
 | |
-+-+-
 | |
-+-+-
 | |
```

```
theBoard = {'top-L': 'O', 'top-M': 'O', 'top-R': 'O',
            'mid-L': 'X', 'mid-M': 'X', 'mid-R': ' ',
            'low-L': ' ', 'low-M': ' ', 'low-R': 'X'}

printBoard(theBoard)
```

```
O|O|O
-+-+-
X|X|
-+-+-
 | |X
```

```
theBoard = {'top-L': ' ', 'top-M': ' ', 'top-R': ' ',
            'mid-L': ' ', 'mid-M': ' ', 'mid-R': ' ',
            'low-L': ' ', 'low-M': ' ', 'low-R': ' '}
turn = 'X'
for i in range(9):
    printBoard(theBoard)
    print('Turn for ' + turn + '. Move on which space?')
    move = input()
    theBoard[move] = turn
    if turn == 'X':
        turn = 'O'
```

```
    else:
        turn = 'X'
```

```
 | |
-+-+-
 | |
-+-+-
 | |
Turn for X. Move on which space?
mid-L
 | |
-+-+-
X| |
-+-+-
 | |
Turn for O. Move on which space?
top-R
 | |O
-+-+-
X| |
-+-+-
 | |
Turn for X. Move on which space?
mid-M
 | |O
-+-+-
X|X|
-+-+-
 | |
Turn for O. Move on which space?
mid-R
 | |O
-+-+-
X|X|O
-+-+-
 | |
Turn for X. Move on which space?
low-R
 | |O
-+-+-
X|X|O
-+-+-
 | |X
Turn for O. Move on which space?
top-L
O| |O
```

```
-+-+-
X|X|O
-+-+-
 | |X
Turn for X. Move on which space?
top-M
O|X|O
-+-+-
X|X|O
-+-+-
 | |X
Turn for O. Move on which space?
low-M
O|X|O
-+-+-
X|X|O
-+-+-
 |O|X
Turn for X. Move on which space?
low-L
```

```
allGuests = {'Alice': {'apples': 5, 'pretzels': 12},
             'Bob': {'ham sandwiches': 3, 'apples': 2},
             'Carol': {'cups': 3, 'apple pies': 1}}

def totalBrought(guests, item):
    numBrought = 0
    for k, v in guests.items():
        numBrought = numBrought + v.get(item, 0)
    return numBrought

print('Number of things being brought:')
print(' - Apples         ' + str(totalBrought(allGuests, 'apples')))
print(' - Cups           ' + str(totalBrought(allGuests, 'cups')))
print(' - Cakes          ' + str(totalBrought(allGuests, 'cakes')))
print(' - Ham Sandwiches ' + str(totalBrought(allGuests, 'ham sandwiches')))
print(' - Apple Pies     ' + str(totalBrought(allGuests, 'apple pies')))
```

```
Number of things being brought:
 - Apples         7
 - Cups           3
 - Cakes          0
 - Ham Sandwiches 3
 - Apple Pies     1
```