

Alexander Jenke, Theodor Straube

# Übung2: PEP8, Basics & String Methods

Programmierkurs Python // Mittwoch, 06. November 2019

# ToC

- Fibonacci
  - If
  - While
  - For
- PEP 8
- Funktionsaufrufe
- Variablentypen
  - Type Conversion
- String Methods

# Fibonacci Reihe

- Das If-Statement führt abhängig von Bedingungen unterschiedlichen Code aus

```
if bar == "Hello World":  
    print("Hallo Welt")  
  
elif bar > 0:  
    print(bar - 1)  
  
else:  
    foo(bar)
```

- Logische Ausdrücke:  
== gleich  
!= ungleich  
> größer  
< kleiner  
and logisches UND  
or logisches ODER  
( ) Ausdrücke gruppieren

- While-Schleifen wiederholen Code, solange eine Bedingung erfüllt ist

```
i = 0  
while i < 5:  
    print(i)  
    i += 1
```

- For-Schleifen führen Code n-mal aus

```
for i in [0, 1, 2, 3, 4, 5]:  
    print(i)
```

```
for i in range(5):  
    print(i)
```

# PEP 8

## Style Guide for Python Code

<https://www.python.org/dev/peps/pep-0008/>

- Einrückungen: 4 Leerzeichen
- Maximale Zeilenlänge: 79 Zeichen
- Leerzeilen:
  - 2 Zeilen um Top-Level Funktionen und Klassen
  - 1 Zeile um Funktionen innerhalb von Klassen
- ASCII oder UTF-8 Encoding
- Nur ein Import pro Zeile
- Kommentare bilden vollständige Sätze
- Kommentare auf englisch
- Benennungen:
  - Module alllowercase
  - Klassen CamelCase
  - Variablen lowercase\_with\_underscores
  - Funktionen lowercase\_with\_underscores
  - Globale Variablen UPPER\_CASE\_WITH\_UNDERSCORES
- Leerzeichen:
  - nach Kommas `x, y = foo(lorem, ipsum,)`
  - Um Gleichheitszeichen
  - Um mathematische Operatoren

# Funktionsaufrufe

## mit Argumenten

- Argumente
  - ohne Keyword
  - mit Keyword
  - Arbitrary ( \* & \*\*)
- Keywords legen Default-Werte fest, wenn das Keyword nicht im Aufruf übergeben wird.
- Zuweisung der Argumente ohne Keyword geschieht über die Reihenfolge
- Die Reihenfolge der Argumente mit Keyword ist egal
- Keine Keywords vor non-keyword Argumenten
- \*args sammelt alle überflüssigen non-keyword Argumente in einer Liste
- \*\*kwargs sammelt alle unbekannten Keyword-Argumente in einer Liste

```
def print(self, *args, sep=' ', end='\n', file=None):...
```

# Funktionen

## sind auch nur Variablen

- Funktionsnamen sind Variablen, welche aufrufbar sind
- Aufruf erfolgt über das Hinzufügen der Klammern
- Inhalt der Variablen kann ersetzt werden
  - Funktionalität der Funktion kann zur Laufzeit ausgetauscht werden
- Schnittmenge der Variablen- & Funktionsnamen muss eindeutig sein

```
def greet():  
    print("Hallo!")  
  
def greet_politely():  
    print("Guten Tag.")  
  
greet = greet_politely  
greet()
```

# Variablentypen

- int                   Ganzzahlen
- float                Fließkommazahlen
- complex            Komplexe Zahl
- bool                Wahrheitswert
- str                  Text
- Umformen:
  - int(3.5) => 3
  - float(3) -> 3.0
  - ...

- Variablen sind typ-ungebunden

```
var = None  
var = 3.14  
var = "Lorem Ipsum"
```

- Typ kann definiert werden

```
var: int = 1
```

```
def foo(bar: str):  
    print(bar)
```

Diese Definition ist jedoch nicht bindend

```
var: int = "Lorem ipsum"
```

# String Methods

## the basics

- **Strings formatieren:**

```
name = "Alex"  
time = 12  
"Hallo %s, willkommen in Python. Es ist %i Uhr." % (name, time)  
"Hallo {}, willkommen in Python. Es ist {} Uhr.".format(name, time)
```

- **Strings manipulieren:**

- `.lower()` *Return a copy of the string converted to lowercase*
- `.split(sep)` *Return a list of the words in the string, using sep as the delimiter string*
- `.isalpha()` *Return True if the string is an alpha-numeric string*
- `.replace(old, new)` *Return a copy with all occurrences of substring old replaced by new*
- `.find(sub)` *Return the lowest index where substring sub is found*
- `.count(sub)` *Return the number of non-overlapping occurrences of substring sub*
- `len(str)` *Return the length of str*
- ...



# String Methods

many more ...

- capitalize *Return a capitalized version of the string*
- casefold *Return a version of the string suitable for caseless comparisons*
- center *Return a centered string of length width*
- encode *Encode the string using the codec registered for encoding*
- endswith *Return True if S ends with the specified suffix*
- expandtabs *Return a copy where all tab characters are expanded using spaces*
- isascii *Return True if all characters in the string are ASCII*
- isidentifier *Return True if the string is a valid Python identifier*
- islower *Return True if the string is a lowercase string*
- istitle *Return True if the string is a title-cased string*
- join *Concatenate any number of strings.*
- partition *Partition the string into three parts using the given separator [substr, separator, substr]*
- rfind *Return the highest index in S where substring sub is found*
- strip *Return a copy of the string with leading and trailing whitespace removed*
- swapcase *Convert uppercase characters to lowercase and lowercase characters to uppercase*
- translate *Replace each character in the string using the given translation table*
- zfill *Pad a numeric string with zeros on the left, to fill a field of the given width*

# nächste Woche

- Mehr Zeit für Aufgaben
- Zeit für Fragen
- Lösungen