Scripting 1

TSBE Frühlingssemester 2018

http://smlz.github.io/tsbe-2018fs/scri/

Marco Schmalz

marco.schmalz@gibb.ch

Programm

1. Formalitäten
2. Setup
3. Erstes Beispiel: let's print!

   Pause ☕

4. Guido van Rossum
5. Einfache Datentypen (Zahlen, Text, Wahrheitswerte)
6. Ausgaben formatieren

   Pause ☕

7. Crazy Code: `f"{f'{}'}"`
8. Benutzereingabe abfragen
9. Verzweigungen: `if`

Formales

Proben

- 27. März 2018
- 28. April 2018

Diplomprüfung

- 15. Juni 2018

Setup

Python 3.6: https://www.python.org/downloads

Texteditor:

- Atom: https://atom.io
- Visual Studio Code: https://code.visualstudio.com

Interaktiver Python Modus

Den interaktiven Modus erreicht man, indem man Python ohne weitere Parameter aufruft. Die drei grösser-als-Zeichen (`>>>`) zeigen an, dass wir uns im interaktiven Modus befinden.

```
$ python3.6
>>> 1 + 1
2
>>> "Hallo" + " " + "Welt" + "!"
'Hallo Welt!'
>>> True or False
True
>>> "Birnen" == "Äpfel"
False
>>> quit()
```

Der interaktive Modus eignet sich sehr gut, um ein kurzes Codestück zu testen, oder um kurz auszuprobieren, ob etwas in Python so funktioniert.

Ein Python-Skript schreiben

1. Eine Datei mit der Endung **.py** erstellen.
2. Python-Code in die Datei schreiben. Bsp.: `mein_programm.py`

```
print("Die Antwort zum Leben, dem Universum und allem:")
print(42)
```

3. Die Datei ausführen, indem man Python mit dem Dateinamen als Parameter aufruft:

```
$ python3.6 mein_programm.py
Die Antwort zum Leben, dem Universum und allem:
42
```

Ein erstes Beispiel

```
print("Hallo miteinander")  # Freundlich sein hilft immer!
```

Rechnen in Python

```
>>> 3 + 4
7
>>> 2 * 5
10
>>> 2 ** 10    # Hoch
1024
>>> 5 / 2
2.5
>>> 1.5 - 0.5
1.0
>>> 7 // 3     # Ganzzahlige Division
2
>>> 7 % 2      # Rest der Division (Modulo)
1
>>> (3 + 5) * 2 - 1
15
```

Datentypen

1. **Ganzzahlen** `int`:
    - Können beliebig gross werden
    - Bsp.: `-1`, `42`, `29979245800`
2. **Fliesskommazahlen** `float`:
    - Bsp.: `-1.2`, `2.718281828459045`, `8.2e6`
3. **Wahrheitswerte** `bool`:
    - `True`, `False`
4. **Tex** `str`:
    - Bsp.: `"Hallo"`, `'Guten Tag'`, `'Der CEO war "bescheiden"'`

Dingen einen Namen geben

```
duo = 2

trio = 3

pi = 3.14159

e = 2.718281828459045

vorname = "Frieda"

nachname = "Müller"

geschlecht = "w"

hat_ein_auto = True

mag_hunde = False
```

In Python sind keine Typendeklarationen nötig.

Guido van Rossum

Community

Mehr Text

```
vorname = 'Franz'
nachname = "Klammer"
brille ="O'Neil"

# Mehrzeilige Strings
fahrstil = """
  elegant
  geschmeidig
  schnell
"""
```

Länge von Strings:

```
>>> len('abc')
3
```

Mehr Printing

Mehrer Werte auf einmal ausgeben

```
>>> print("Hallo", "Welt!")
Hallo Welt!

>>> print("Eins", 2, 'drei', 4.0)
Eins 2 drei 4.0

>>> print(1, 2, 3, sep=";")
1;2;3

>>> print("Hallo", end="-")        # keine neue Zeile beginnen
Hallo->>>
```

`f-String`: Formatierte Strings

Beispiel:

```
alter = 18
name = 'Peter'
print(f"{name} ist {alter} Jahre alt.")
```

Übersicht: https://pyformat.info/

Spezifikation: https://docs.python.org/3.6/library/string.html#format-string-syntax

Benutzereingaben: Die `input`-Funktion

`input()` gibt eine Benutzereingabe zurück

Beispiel:

```
firstname = input('Dein Vorname: ')
nachname = input('Dein Nachname: ')

print("Hallo", firstname, lastname)
```

Text zu Zahl konvertieren

- Einen String in eine Ganzzahl konvertieren: `int`

```
>>> zahl = int("42")
>>> zahl
42
```

- Einen String in eine Fliesskommazahl konvertieren: `float`

```
>>> zahl = float("-2.7")
>>> zahl
-2.7
```

- Gibt einen Fehler zurück, fall die Konvertierung nicht möglich ist:

```
>>> int("zwei")
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 10: 'zwei'
```

Verzweigungen: if

```
alter = 24

if alter < 16:
    # Achtung, einrücken!

    print("Du musst noch viel lernen!")

    # So, das reicht für diesen Fall. Wieder an den Anfang der Zeile!

elif alter > 65:
    print("Nie wieder arbeiten!")
else:
    print("An die Arbeit!")
```

Vergleiche und logische Operatoren

- `==`: gleich
- `!=`: ungleich
- `is`: identisch
- `>`: grösser
- `<`: kleiner
- `>=`: grösser-gleich
- `<=`: kleiner-gleich

- `and`: Und-Verknüpfung (beide Bedingungen müssen erfüllt sein)
- `or`: Oder-Verknüpfung (mindestens eine Bedingung muss erfüllt sein)
- `not`: Negierung

```
if alter >= 18 or alter < 65:
    print("Leider kein Rabatt")

if 16 < alter <= 65:  # Abkürzung
    print("An die Arbeit!")
```