Einstieg in Python

Operatoren, Variablen, Kontrollstrukturen, Funktionen, Listen

Was ist Python?

- moderne Skriptsprache
- plattformunabhängig
- weit verbreitet und quasi überall im Einsatz

Warum Python lernen?

- Einfach zu lesen und lernen

> *Python is powerful... and fast; plays well with others; runs everywhere; is friendly & easy to learn; is Open.*

- Fokus auf "schnellen" Ergebnissen
- Große Sammlung an Zusatzbibliotheken für alle denkbaren Anwendungsbereiche
- Buchempfehlung: ***Automate the boring Stuff with Python*** (http://automatetheboringstuff.com)

Zum Vergleich: ”Hello World”

In Java:

```
public class HelloWorld {

public static void main(String[] args){
System.out.println("Hello World!");
}
}
```

In Python:

```
print("Hello World!")
```

Zahlen & Operatoren - Python als Taschenrechner

- Python unterteilt in: Ganzzahlen (`integer`) und Fließkommazahlen (`float`)
- klassiche Rechenoperatoren: `+ - * / // %`
- Vergleichsoperatoren: `< <= == >= > !=`
- "Komplexeres" Rechnen über Zusatzbibliotheken etwa `import math` oder `import numpy`

In [3]:
```
3 * 7
```

Out[3]:
```
21
```

Strings - Arbeiten mit Text

- Strings sind Zeichenketten wie etwa `"Hello World!"`
- Python bietet **Funktionen** um etwa:
  - Textinformationen zu erhalten zB. : `isalnum()` , `istitle()` , `count()`
  - Text zu bearbeiten/manipulieren: `lower()` `strip()` `replace("o","i")`

. Tipp: Druck auf TAB zeigt mögliche Methoden

In [4]:
```
"Hello World".count("o")
```

Out[4]:
```
2
```

Funktionen - Bitte mach was!

- ähnlich wie in der Mathematik: Eingabe -> (Ausgabe)
- built-in Funktionen wie `min()` , `max()`, `type()`
- `print()` gibt etwas auf der Konsole aus
- `input()` nimmt Benutzereingabe entgegen
- Strings, Zahlen, spätere Klassen die wir kennenlernen stellen eigene Funktionen

In [5]:
```
max(1,9,2,4,13,8,6)
```

Out[5]:
```
13
```

Zwischenaufgaben

1. Schreiben sie ein Programm, dass:
    - Den Benutzer nach seinem Namen und Alter fragt
    - Den Namen + das Alter in Sekunden wieder ausgibt

Kontrollstrukturen - Programmablauf steuern

- Kontrollstrukturen steuern, in welcher Reihenfolge Code läuft
  - *Wenn der Benutzer einen Knopf drückt, mache ..*

    -> **Bedingungen** `if` `elif` `else`

  - *Wiederhole 100x mal* oder *Für jeden Benutzer mache folgendes..*
    -> **Schleifen** : `for` `while`

In [4]:
```
beispielwort = "Lausbub"
for buchstabe in beispielwort:
    if(buchstabe == 'b'):
        print("x", end='')
    else:
        print(buchstabe, end="")
```

```
Lausxux
```

Listen - Daten strukturiert ablegen

- Listen dienen dazu Datenmengen strukturiert abzulegen ~vgl. Exceltabelle
- Listen werden mit [ ] erzeugt, Elemente werden mit Komma getrennt
  - `freunde = ['anna', 'bill', 'claude']`

In [1]:
```
freunde = ['anna', 'bill', 'claude']
print(freunde)
```

```
['anna', 'bill', 'claude']
```

Listen anlegen

1. Liste manuell
    - `list_numbers = [1,2,3,4,5,6,7,8,9,10]`
2. Liste mit einer Schleife und append() anlegen

```
list_numbers = []
for number in range(1,11):
    list_numbers.append(number)
```

3. Liste mittels List Comprehension anlegen

```
list_numbers = [number for number in range(1,11)]
```

In [8]:
```
list_numbers = [1,2,3,4,5,6,7,8,9,10]
print(numbers)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

In [2]:
```
list_numbers = []
for number in range(1,1001):
    list_numbers.append(number)
print(list_numbers[0:100])
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 2
4, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 4
5, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 6
6, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 8
7, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100]
```

In [6]:
```
quadratzahlen = [i*i for i in range(101)]
print(quadratzahlen)
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 36
1, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900, 961, 1024, 1089, 1156, 122
5, 1296, 1369, 1444, 1521, 1600, 1681, 1764, 1849, 1936, 2025, 2116, 2209, 2304, 240
1, 2500, 2601, 2704, 2809, 2916, 3025, 3136, 3249, 3364, 3481, 3600, 3721, 3844, 396
9, 4096, 4225, 4356, 4489, 4624, 4761, 4900, 5041, 5184, 5329, 5476, 5625, 5776, 592
9, 6084, 6241, 6400, 6561, 6724, 6889, 7056, 7225, 7396, 7569, 7744, 7921, 8100, 828
1, 8464, 8649, 8836, 9025, 9216, 9409, 9604, 9801, 10000]
```

Übungsaufgaben:

1. Erzeugen sie eine Liste aller ungeraden Zahlen von 1-20:
    - manuell
    - per Schleife
    - List Comprehension

In [19]:
```
numbers = []

for number in range(1,101):
    if(number%3 == 0):
        numbers.append(number*number)


print(numbers)
```

```
[9, 36, 81, 144, 225, 324, 441, 576, 729, 900, 1089, 1296, 1521, 1764, 2025, 2304, 26
01, 2916, 3249, 3600, 3969, 4356, 4761, 5184, 5625, 6084, 6561, 7056, 7569, 8100, 864
9, 9216, 9801]
```