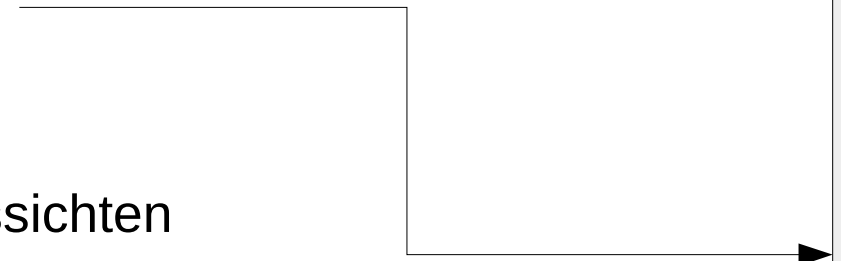
https://www.python.org/community/logos/

Gliederung

1. Prinzipien
2. Geschichtlicher Hintergrund
3. Merkmale
4. Python 3
5. Zukunftsaussichten
6. Fazit
7. Quellen



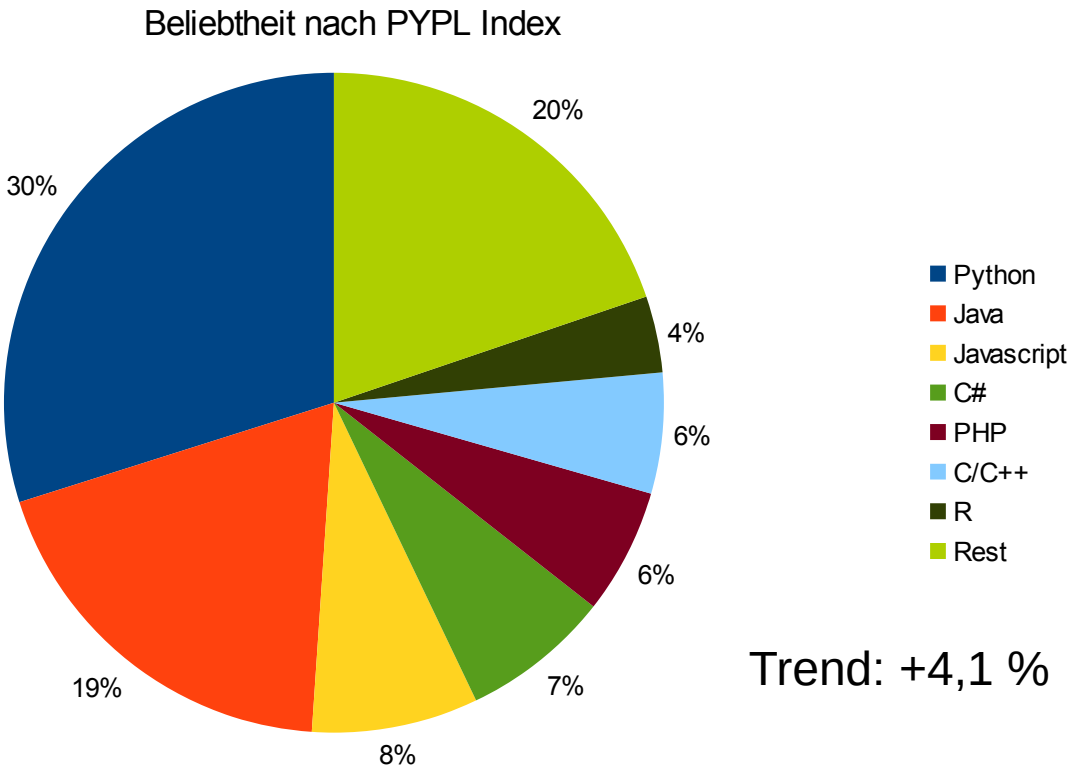
1. Sprachtyp
2. Syntax
3. Kontrollstrukturen
4. Übersetzer
5. Interaktiv
6. Geschwindigkeit
7. Speicherverwaltung
8. Library
9. Erweiterbarkeit
10. Fehler
    - Fehlermeldung
    - Fehlerbehandlung

1. Prinzipien

- The Zen of Python (Teilabschnitt):
  - schön statt hässlich
  - explizit statt implizit
  - einfach statt kompliziert
  - komplex statt undurchschaubar
  - flach statt verschachtelt
  - spärlich statt beschränkt
  - auf die Lesbarkeit kommt es an
  - Spezialfälle sind niemals speziell genug, damit sie Regeln sprengen dürfen

2. Geschichtlicher Hintergrund

- 1990 Guido van Rossum
- 1994 Python 1.0
- 2000 Python 2.0
- 2008 Python 3.0



Trend: +4,1 %

https://www.jacob.de/page/programmiersprachen-trends-2020-55938/

3.1 Sprachtyp

Funktional

- Lambda, map

Objektorientiert

- Klassen und Objekte
- Methoden
- Vererbung

```
1 class Testklasse:
2       def __init__(self):
3           self.zahl = 0
4           self.text = "Hallo"
5
6       def setZahl(self,zahl=int(input("neue Zahl:"))):
7           self.zahl = zahl
8
9       def gibZahl(self):
10          print("Deine Zahl ist:",self.zahl)
11
12if __name__ == "__main__":
13      objekt = Testklasse()
14      objekt.setZahl()
15      objekt.gibZahl()
16      print(objekt.text)
```

3.2 Syntax

- Einrückungen statt Klammern
- Wenige Schlüsselwörter

```
1   meineZahl = int(input("Suche dir eine Zahl! "))
2   if meineZahl > 1000:
3       print("ziemlich groß")
4   if meineZahl == 1000:
5       print("genau 1000!")
6   elif meineZahl > 0:
7       print("wenigstens etwas")
8   else:
9       print("Die Zahl ist zu klein")
```

ERROR

```
1___meineZahl = int(input("Suche dir eine Zahl! "))
2___if meineZahl > 1000:
3_____print("ziemlich groß")
4___if meineZahl == 1000:
...
```

3.3 Kontrollstrukturen

- For, While,
- If...elif...else,
- Kein switch bzw. go to

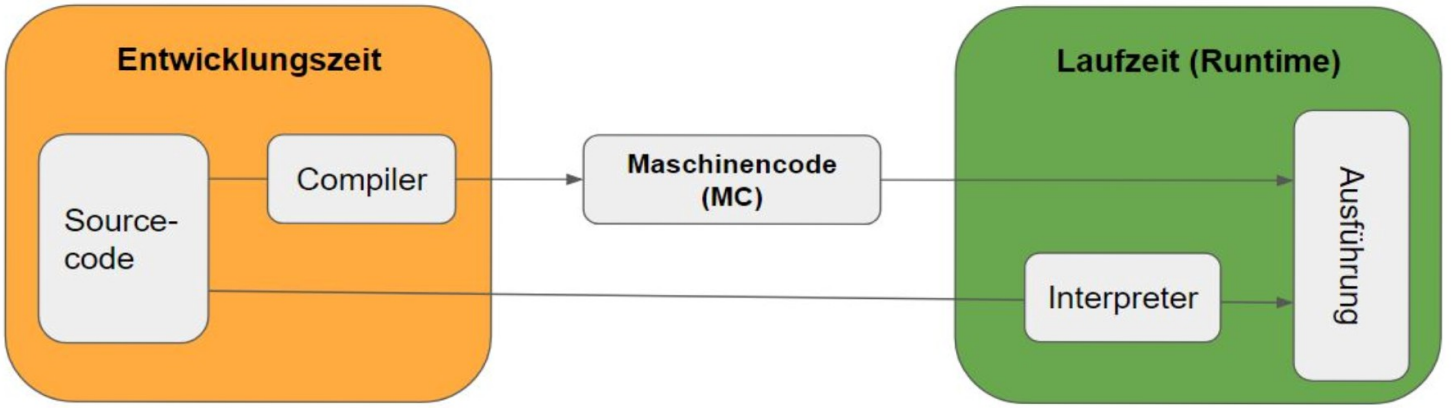
```
1   Liste = [0,2,3,1]
2   for element in Liste[0:]:
3       print(zahl)
```

```
1   for zahl in range(0,11):
2       print(zahl)
```

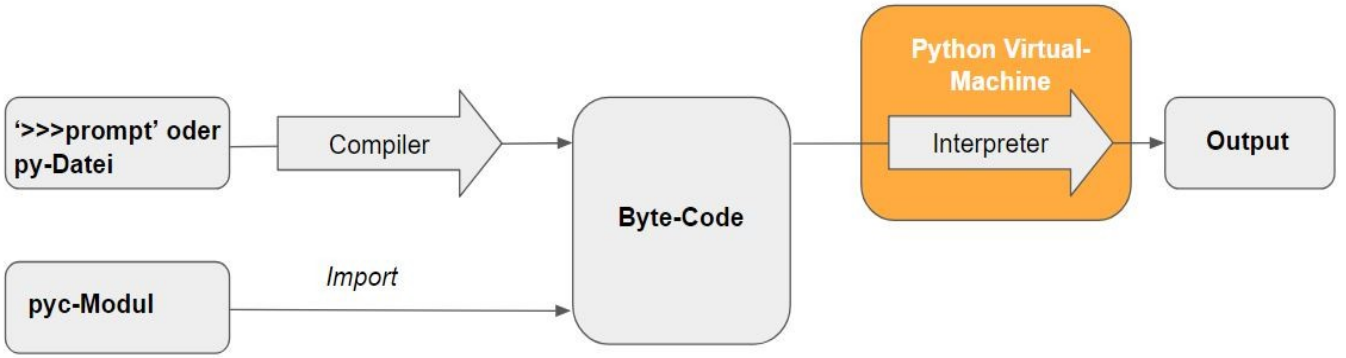
```
1   meineZahl = int(input("Suche dir eine Zahl! "))
2   if meineZahl > 1000:
3       print("ziemlich groß")
4       if meineZahl == 1000:
5           print("genau 1000!")
6   elif meineZahl > 0:
7       print("wenigstens etwas")
8   else:
9       print("Die Zahl ist zu klein")
```

3.4 Übersetzer

Interpreter oder Compiler? Beides!

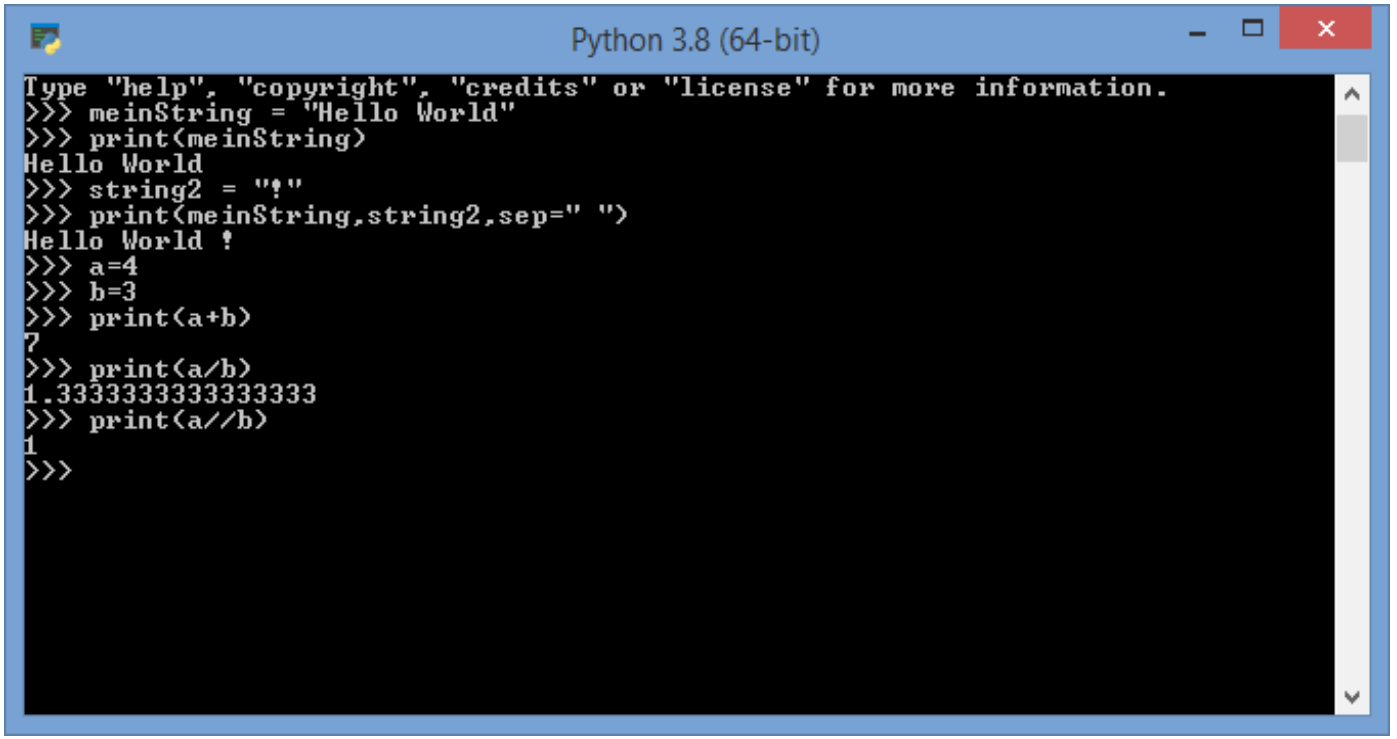


https://www.data-science-architect.de/python-compiliert-interpretiert/



3.5 Interaktiv

- Python Shell
- CMD/Terminal
- Testen



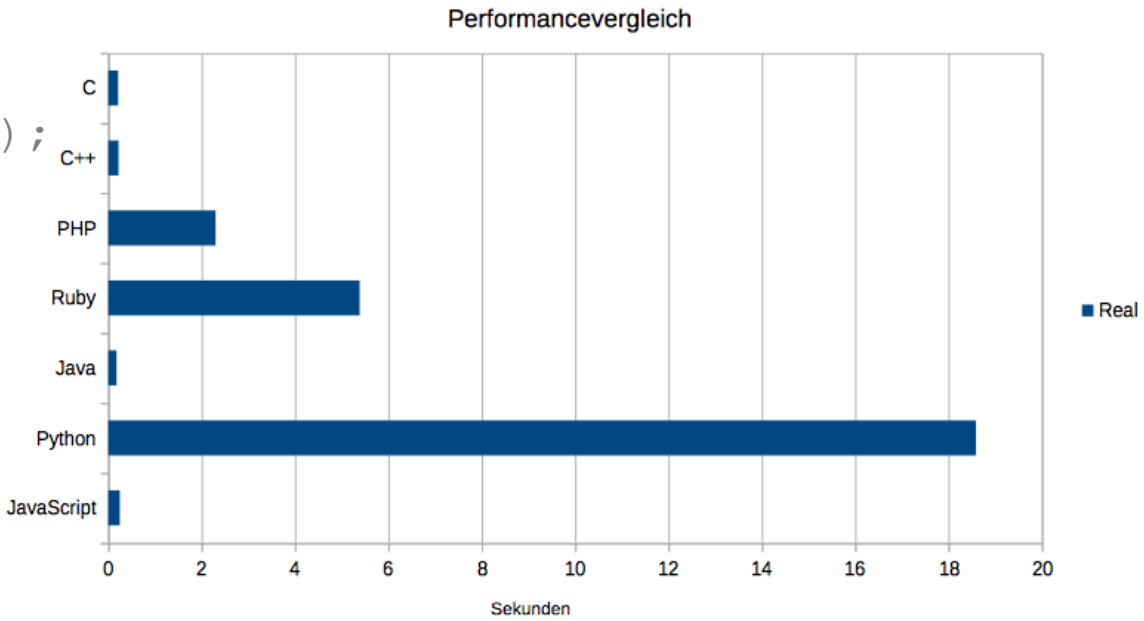
3.6 Geschwindigkeit

Java ~0,2s

```
1 public class Time {
2    public static void main(String[]args)
3    {
4       long sum = 0;
5       for(long i = 0; i < 100000000; i++)
6       {
7           sum += i;
8       }
9       System.out.println(sum);
10   }
11}
```

Python ~18,5s

```
1   sum = 0
2   for i in range(100000000):
3     sum += i
4   print(sum)
```



https://devopsworld.de/1133

3.7 Speicherverwaltung

- dynamisch

→ Objekte

- Garbagecollector
- Datentypen
  - Typzuweisung möglich
- Generator

```
x = 42
y = "Hi"
z,a = True,False

x = "nicht mehr 42"
```

```
zahl = 7.4
print(int(zahl))
```

3.8 Library

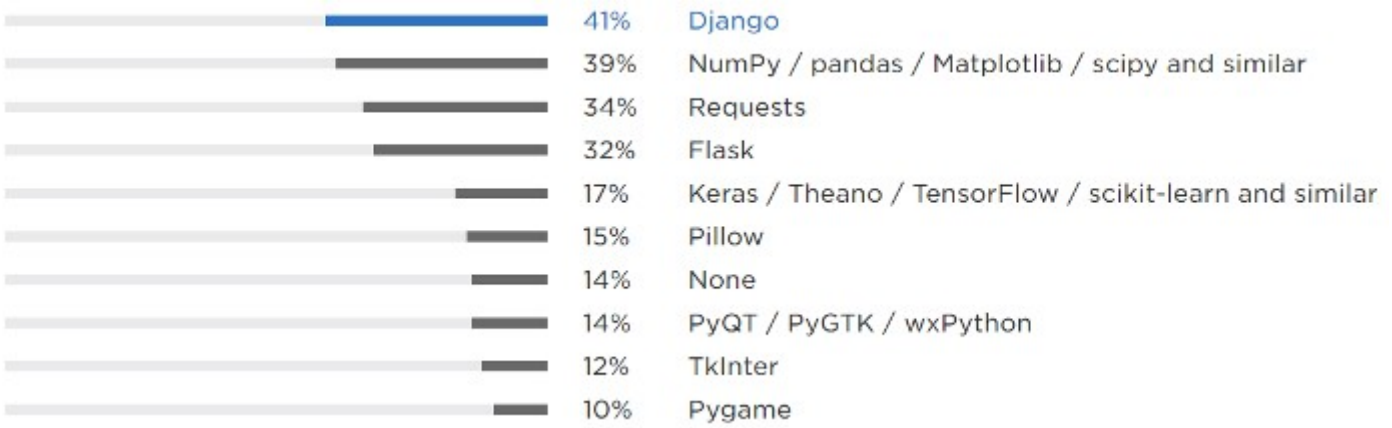
- Umfangreiche Standardbibliothek
- Viele Bibliotheken
- Import
- Eigene Module

```
import math
```

```
From WO import Was
```

testtest.py | test.py

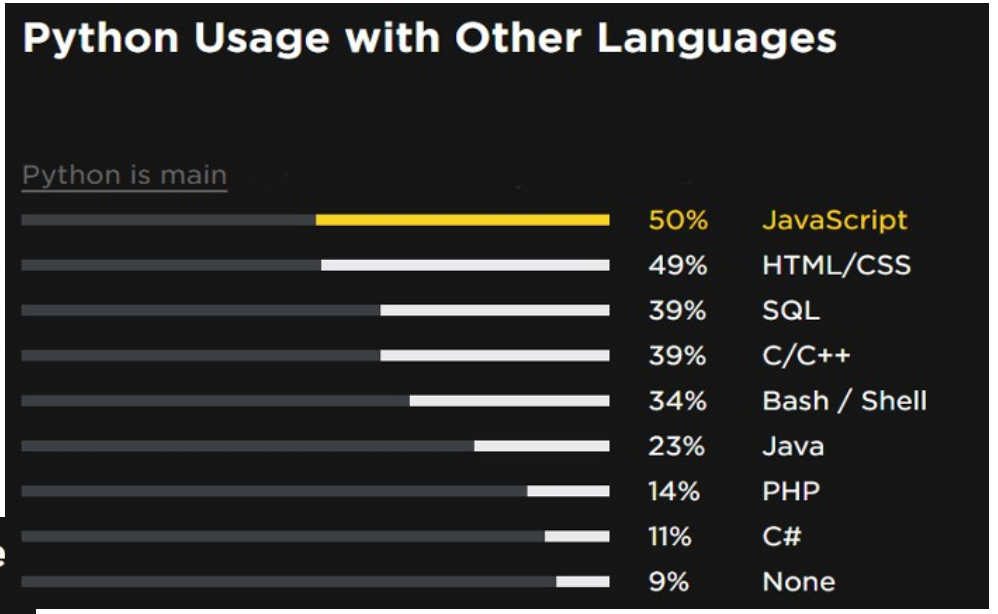
```
1  from test import Testklasse
```



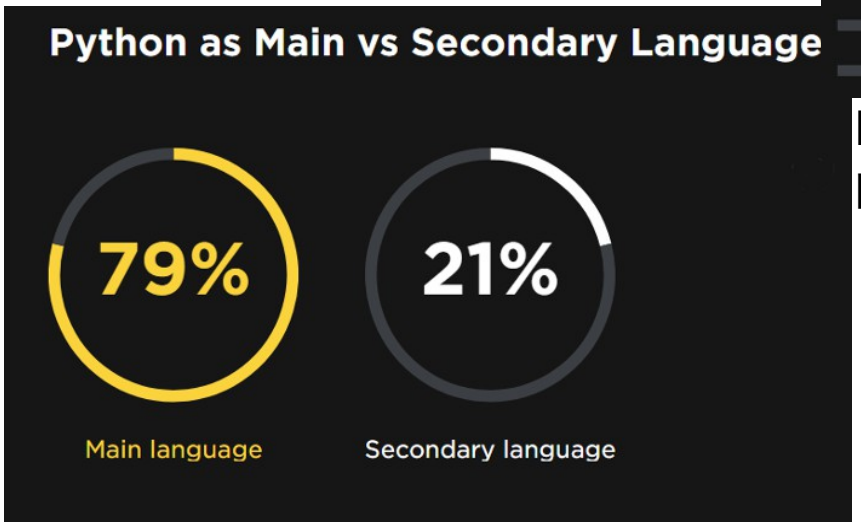
https://opensource.com/article/18/5/numbers-python-community-trends

3.9 Erweiterbarkeit

- Add-ons und Pakete
- Vorteile
  - Zeilen
  - Performance
  - Vor- und Nachteile nutzen



https://opensource.com/article/18/5/numbers-python-community-trends



3.10.1 Fehlermeldung

- Arten
  - Syntax
  - Logical
  - Built-in
    - ZeroDivision
    - NameError
    - TypeError
- assert
- Try und Except

```
Traceback (most recent call last):
  File "Hier steht der Dateipfad", line 5, in <module>
    objekt.hallo()
AttributeError: 'Testklasse' object has no attribute 'hallo'
```

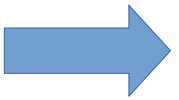
```
Traceback (most recent call last):
  Dateipfad, Zeile, (Modul)
    Fehlercode
Art des Fehlers
```

```
Traceback (most recent call last):
  File "Hier steht der Dateipfad", line 2, in <module>
    objekt = Vergleicher(aa)
NameError: name 'aa' is not defined
```

3.10.2 Fehlerbehandlung

- Try: eine Aktion ausführen
- Except: Fehlermeldung zurückwerfen
- Else-Zweig falls es zu keiner Exception kommt

```
1   Liste =[0,0,2,1,0,3]
2   i=0
3
4   for element in Liste[0:]:
5       try:
6           zahl = 1 / element
7       except ZeroDivisionError:
8           print("Element",i,"ist eine Null")
9           i += 1
10      else:
11          print("Element",i,"ist keine Null")
12          i += 1
```



```
Element 0 ist eine Null
Element 1 ist eine Null
Element 2 ist keine Null
Element 3 ist keine Null
Element 4 ist eine Null
Element 5 ist keine Null

Process finished with exit code 0
```

4.1 Änderungen

Beispiel	Python 2	Python 3
Bibliothek und Module	Redundanz	Aufteilung in Module
Print()	Print	Print()
Integer	Long/int/float	Kein long | floor und true Division
String	ASCII	Unicode und Byte
Vergleiche	Liste > String	Kein Vergleich
Input	Objekt	String
Type Hints	/	Typzuweisung
Schleifen	Global	Lokal
assert	failUnlessEqual(a,b)	assertEqual(a,b)
Exceptions		präziser
….		

4.2 Auswirkungen

- Keine Rückkompatibilität
- Parallele Versionen
- Neue Inhalte
- Nutzungsfreundlicher
- Performance
- PEP

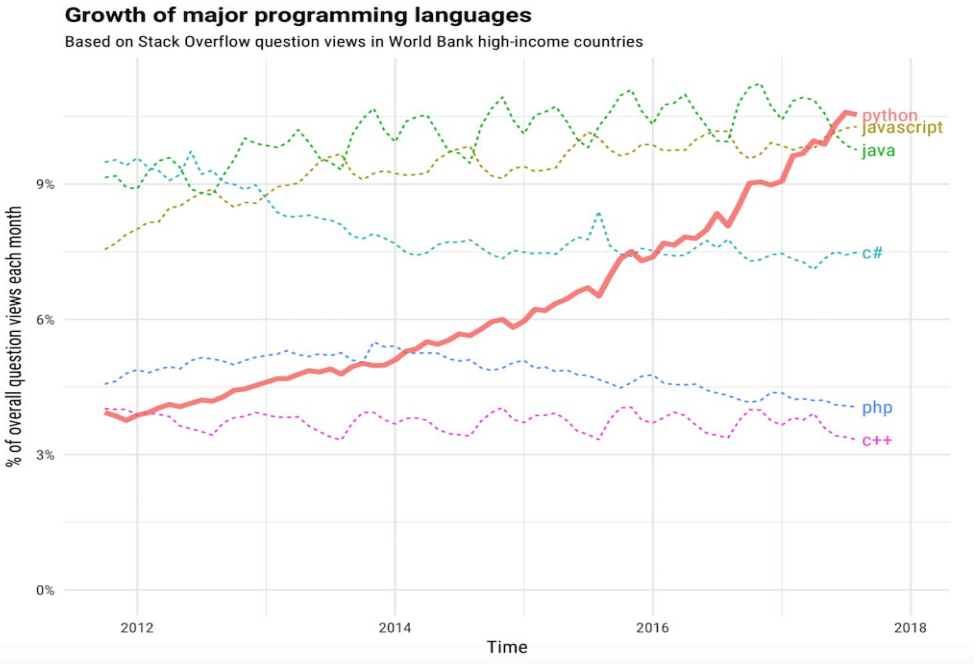
Python 3	Python 2	
75%	25%	All Python
70%	30%	Web developers
80%	20%	Data scientists

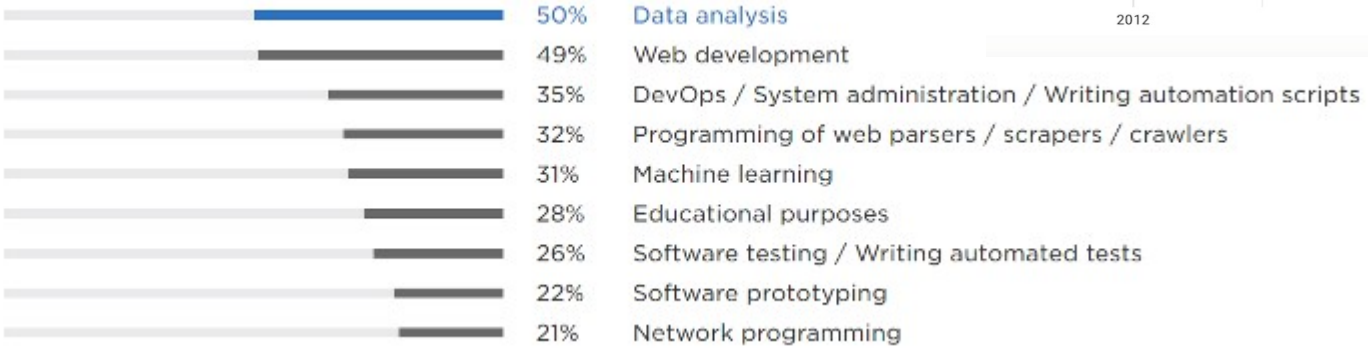
https://devopedia.org/python-2-vs-3

5. Zukunftsaussichten

Vorteile

- Leicht zu lesen und schreiben
- Plattformunabhängig
- Wachsende Community
- Weiterentwicklung





https://opensource.com/article/18/5/numbers-python-community-trends

6. Fazit

- Pro
  - Anfängerfreundlich
  - Python 3.0:
    - Fehlerbehebung
    - Sicherheit
    - Präzision
  - Weiterführende Nutzung
    - Community
    - Arbeitsmarkt
  - Wachsender IT-Bereich

- Contra
  - 2 Versionen
  - Langsam
    - Interpreter
  - Kann nicht alles ersetzen
    - Funktional: Lisp
    - Performance: C

→ Import

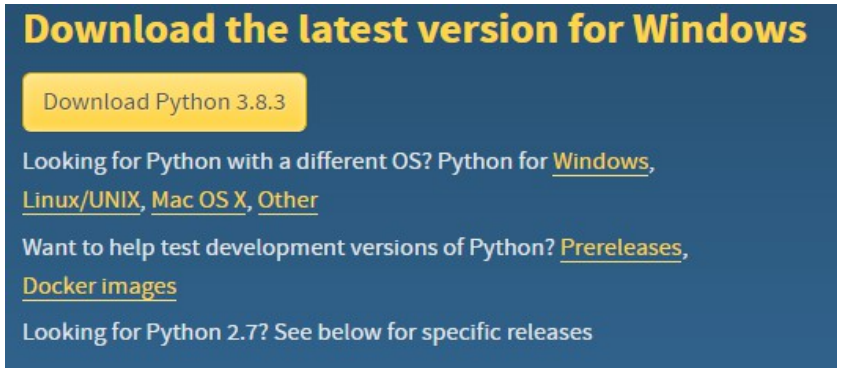
7. Quellen

- https://de.wikipedia.org/wiki/Python_(Programmiersprache) Unbekannt 21.05.20
- https://docs.python.org/3.0/whatsnew/3.0.html Guido van Rossum 12.05.20
- https://lerneprogrammieren.de/python-2-vs-3/ Unbekannt 12.05.20
- https://www.embedded-software-engineering.de/migration-auf-python-3-a-808804/Rainer Grimm 14.05.20
- https://www.bigdata-insider.de/was-ist-python-a-730480/ Stefan Luber 14.05.20
- https://www.dev-insider.de/was-ist-python-a-843060/ Alex Buerkle 16.05.20
- https://www.data-science-architect.de/python-compiliert-interpretiert/ Andreas Wygrabek 21.05.20
- https://www.computerwoche.de/a/python-lernen-leicht-gemacht,3331244,2 Serdar Yegulalp 16.05.20
- https://cito.github.io/byte_of_python/read/features-of-python.html Bernd Hengelein 16.05.20
- https://entwickler.de/online/python/python-tutorial-einfuehrung-579865097.html Tam Hanna 19.05.20
- https://www.kite.com/blog/python/functional-programming/ Amandine Lee 19.05.20
- https://www.python-kurs.eu/python3_kurs.php Bernd Klein 19.05.20
- https://sebastianraschka.com/Articles/2014_python_2_3_key_diff.html Sebastian Raschka 20.05.20
- https://www.python.org/dev/peps/pep-0001/#what-is-a-pep Barry Warsaw 20.05.20

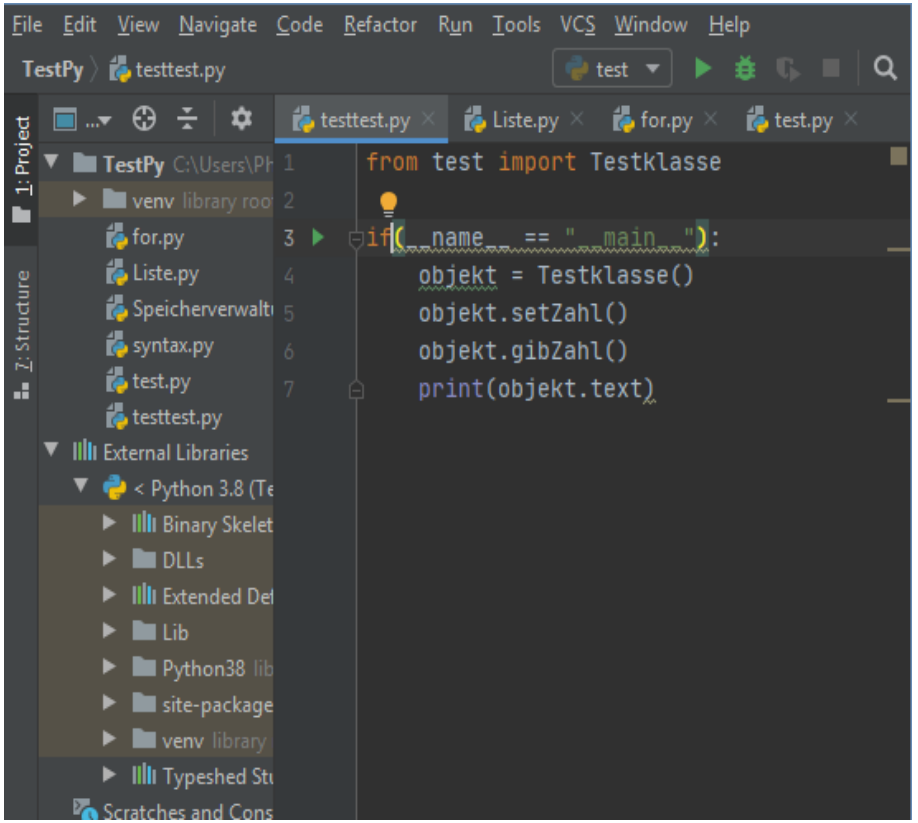
(8.) Einstieg zu Python

1) https://www.python.org/downloads/
2) IDE
3) Youtube Videos / Guides

1)



2)



PyCharm Community Edition

3)
- https://www.python-kurs.eu/
- https://www.youtube.com/watch?v=rfscVS0vtbw