

The ABC of computational Text Analysis

09: Introduction to Python

Alex Flückiger
30 April 2020

Recap last Lecture

- converting any kind of data to `.txt`
- data is never raw but depends on many decisions

You better think about it!

Outline

- enter the shiny world of Python 🤪

development environment
basic syntax



Python

Programming Language

Python is

- general-purpose
not specific to any domain
- interpreted
no compiling
- standard language in data science

How to learn programming?

three inconvenient truths 🥲

- programming cannot be learnt in a course

I try to make the start as easy as possible!

- frustration is normal

fight your way!

- the Python ecosystem is huge

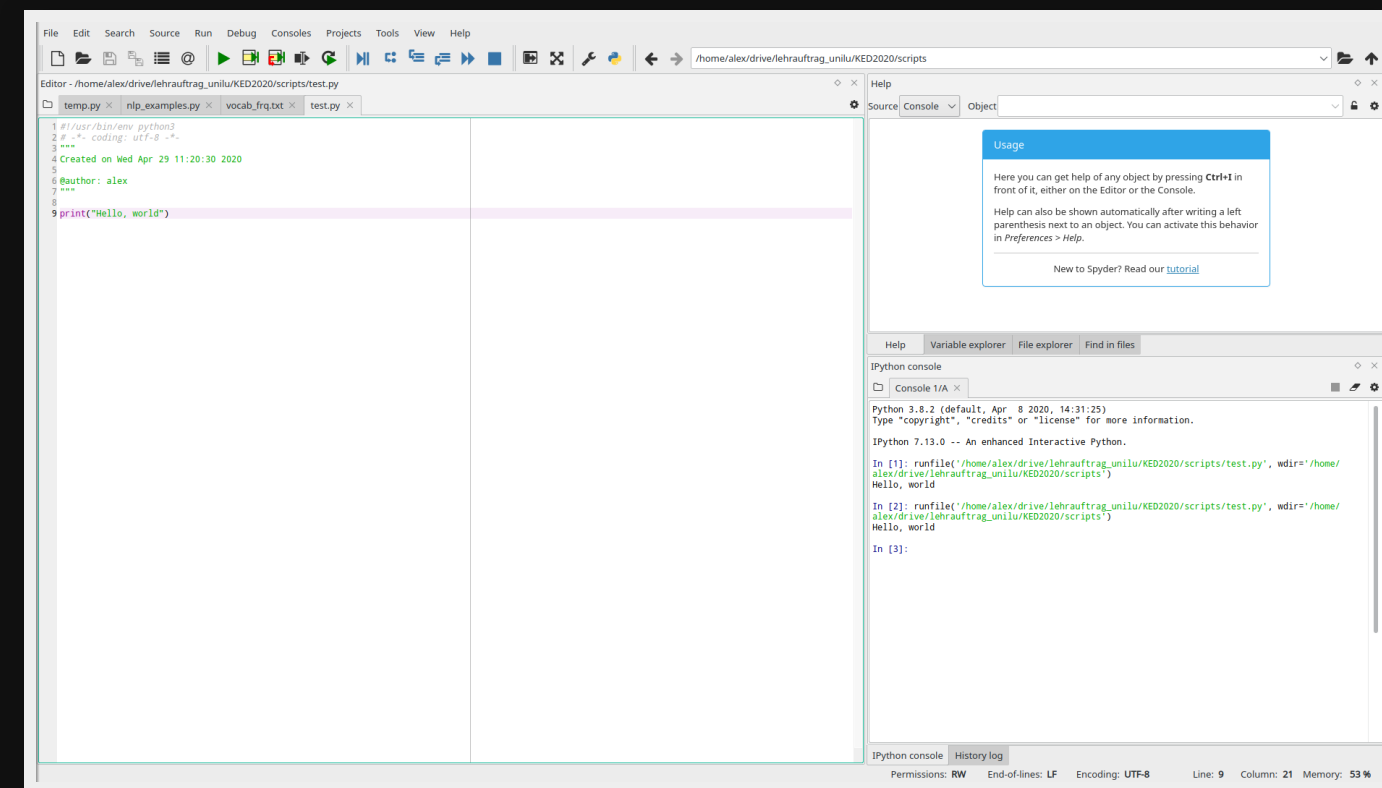
grow skills by step-by-step

Programming can be absolutely captivating! ✌️

Development Editor

Spyder IDE

- integrated development environment (IDE)
 - interactive development*
 - similar to RStudio*
- views
 - scripting*
 - variable explorer*
 - python console*

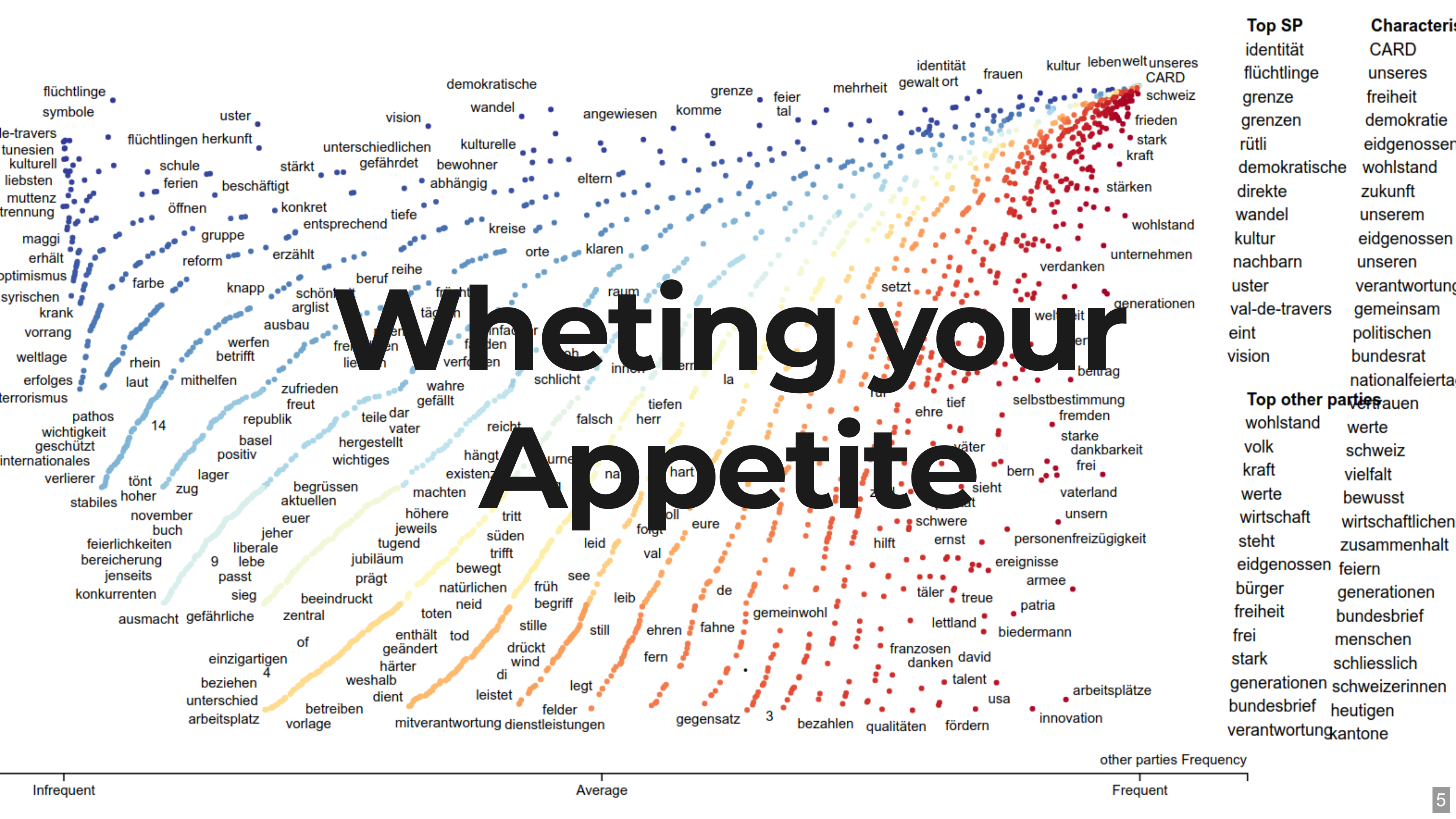


Spyder IDE

First Steps in Python

How to start?

1. open the program `Spyder`
2. set working directory
3. save your script
4. write code `print("Hello, World!")`
5. run code + debug
6. run saved script in shell `python your_script.py`



Syntax

Variables

variables are kind of storage boxes

```
# define variables
x = "at your service"
y = 2
z = ", most of the time."

# combine variables
int_combo = y * y           # for numbers any mathematical operation
str_combo = x + z           # for text only concatenation with +

# show content of variable
print(str_combo)
```

Data Types

type is implicit (dynamic)

Name	What for?	Type	Examples
String	Text	str	"Hi!"
Integer, Float	Numbers	int, float	20, 4.5
List	Lists (ordered, mutable)	list	["Good", "Afternoon", "Everybody"]
Boolean	Truth values	bool	True, False
⋮	⋮	⋮	⋮
Tuple	Lists (ordered, immutable)	tuple	(1, 2)
Dictionary	Relations (unordered, mutable)	dict	{"a":1, "b": 2, "c": 3}

Data Type Conversion

combine variables of the same type only

```
# check the type
type(YOUR_VARIABLE)

# convert types (similar for other types)
int('100')    # convert to integer
str(100)      # convert to string

# combine two types
x = 3
mixed = "x has the value: " + str(x)
print(mixed)
```


Equal-Sign: = vs. ==

■ contradicts the intuition

```
# assign a value to a variable  
x = 1  
word = "Test"  
  
# compare two values if they are identical  
1 == 2           # False  
word == "Test"   # True
```

Comments

- comments ~ lines ignored by Python
- do it, it helps you ...
to learn initially
to understand later

```
# single line comment
```

```
"""  
comment across  
multiple  
lines  
"""
```

Iterations

for-loop

do something with each element of a collection

```
sentence = ['This', 'is', 'a', 'sentence']  
  
# iterate over each element  
for token in sentence:  
  
    # do something with the element  
    print(token)
```

Conditionals

if-else statement

condition action on variable content

```
sentence = ['This', 'is', 'a', 'sentence']

if len(sentence) < 3:
    print('This sentence is shorter than 3 tokens')
elif len(sentence) == 3:
    print('This sentence has 3 tokens')
else:
    print('This sentence is longer than 3 tokens')
```

Indentation

indentation matters!

- intend code within code blocks
loops, if-statements etc.
- press tab to intend



```
if 5 > 2:  
    print('5 is greater than 2')
```



```
if 5 > 2:  
print('5 is greater than 2')
```


Methods

```
# split at whitespace
tokens = 'This is a sentence'.split(' ')

# check the variable
print(tokens, type(tokens))

# add something to a list
tokens.append('.')

# join elements to string
tokens = ' '.join(tokens)
print(tokens, type(tokens))
```

Functions and Arguments

- functions have the form

`function_name(arg1, arg2)`

- functions may have arguments

```
# define a new function
def word_properties(word):
    """
    My first function to print word properties.
    It takes any string as argument (variable word).
    """

    # print(), len() and sorted() work also as functions
    length = len(word)
    sorted_letters = sorted(word, reverse=True)
    print(word, 'length:', length, 'letters:', sorted_letters)

word_properties('computer') # call function with any word
```

Indexing

python starts counting from zero!

```
sentence = ['This', 'is', 'a', 'sentence']

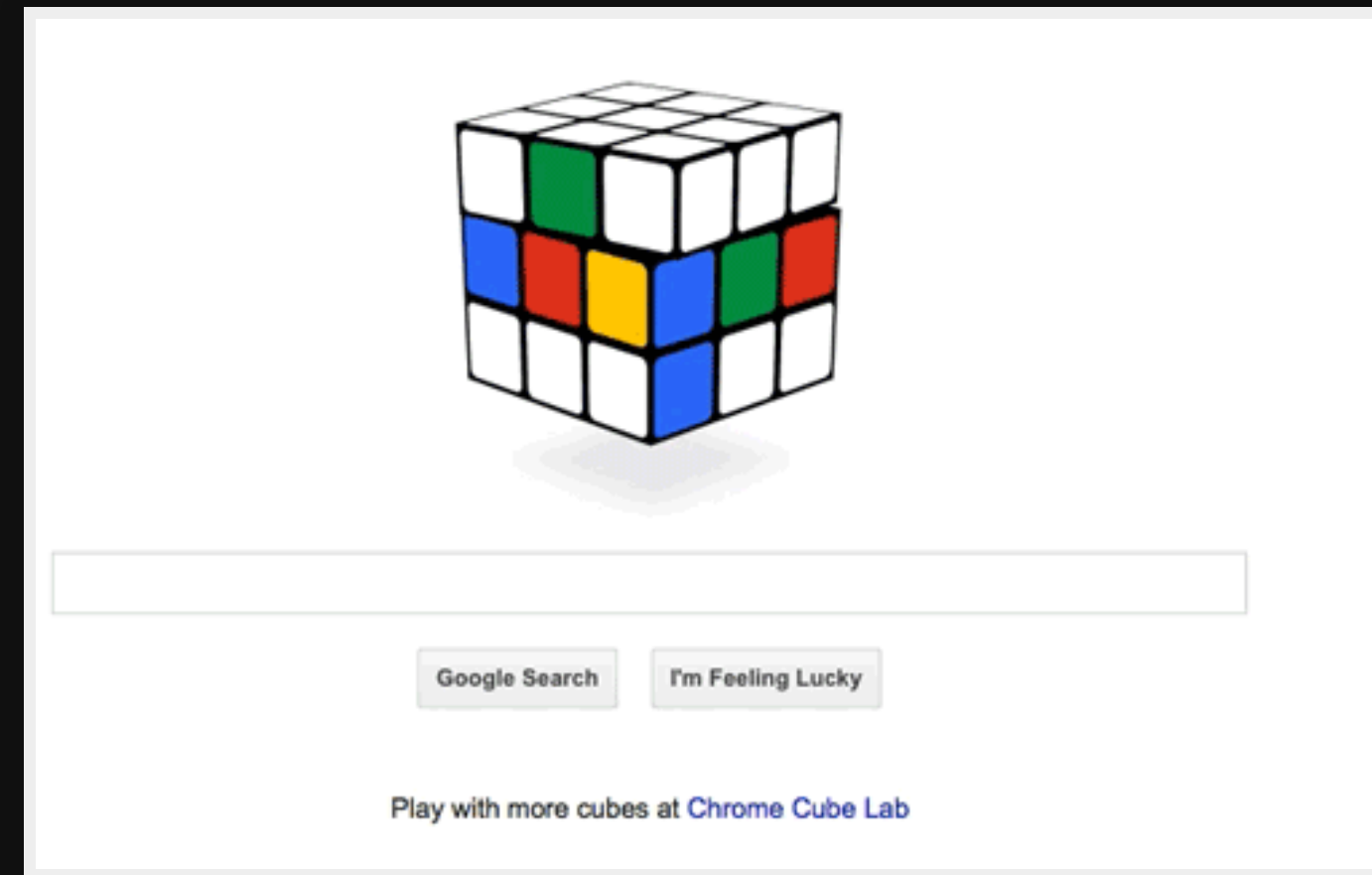
# element at position X
first_tok = sentence[0]      # 'This'

# elements of subsequence [start:end]
sub_seq = sentence[0:3]      # ['This', 'is', 'a']

# elements of subsequence backwards
sub_seq_back = sentence[-2:] # ['a', 'sentence']
```

Syntax Errors

1. read the message
2. find the source of the error
script name + line number
3. paste message into Google



Learning by doing, doing by googling

Modules/Packages

- modules provide functionalities
- no programming from scratch 🎉

NLP Packages

- spaCy
industrial-strength Natural Language Processing (NLP)
- textaCy
NLP, before and after spaCy
- scattertext
beautiful visualizations of how language differs across corpora

In-class: Install Packages for next week

```
# Windows users  
# open a Anaconda Prompt and install the following  
pip install spacy  
conda install -c conda-forge pyemd  
pip install textacy  
pip install scattertext  
  
# Mac users  
# open a Terminal and install after replacing the username  
/Users/<Your username>/anaconda3/bin/python -m pip install spacy  
/Users/<Your username>/anaconda3/bin/python -m pip install textacy  
/Users/<Your username>/anaconda3/bin/python -m pip install scattertext  
  
# All users: install language specific models  
python -m spacy download de_core_news_sm  
python -m spacy download en_core_web_sm
```

In-class: Exercises I

1. Make sure that your local copy of the Github repository KED2020 is up-to-date with `git pull`. Check out the script with the basics of Python:
`scripts/python_basics.py`.
2. Try to understand and run the commands line-wise. Modify them to see how the output changes. Initially, the try-and-error is good strategy to learn.

In-class: Exercises II

1. Write a Python script that

takes text (a string)

splits it into words (a list)

iterates over all the tokens and print all tokens that are longer than 5 characters

Bonus: wrap your code in a function.

2. Go to the next slide. Start with some of the great interactive exercises out there in the web.

Resources

learn basics interactively

- [Python Principles](#)
- [LearnPython](#)

official Python introduction

- [Python introduction](#)