PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

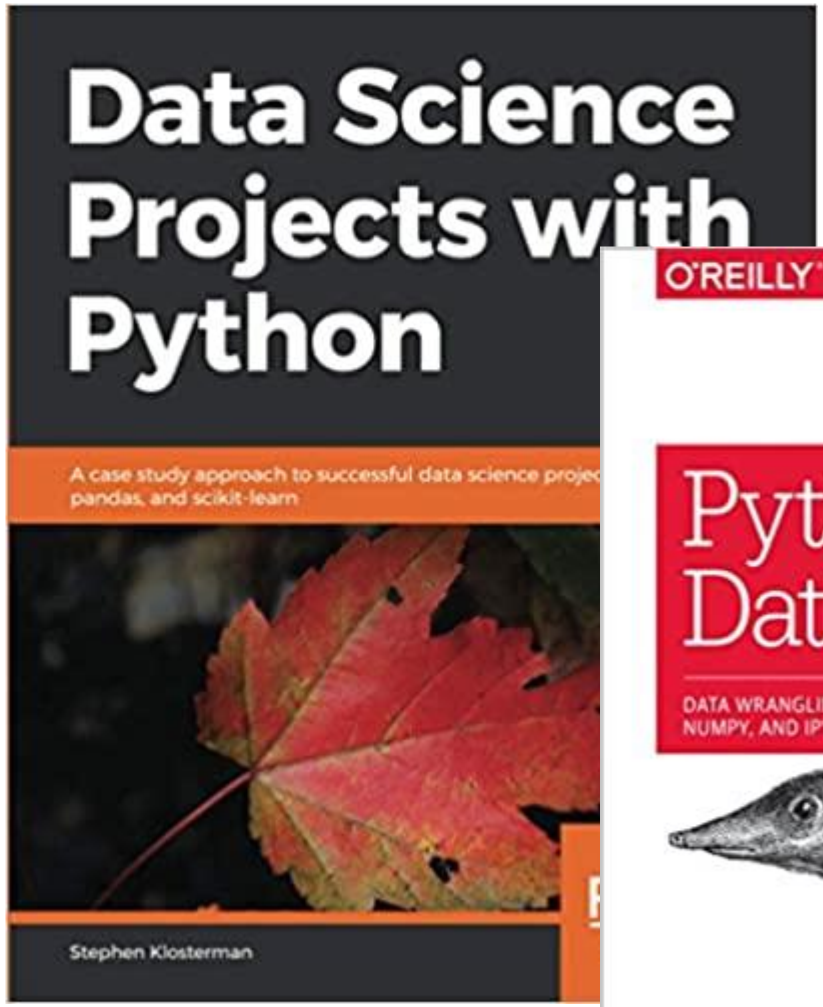
Python Programming
Variables and operations

Robert Haase

April 2021

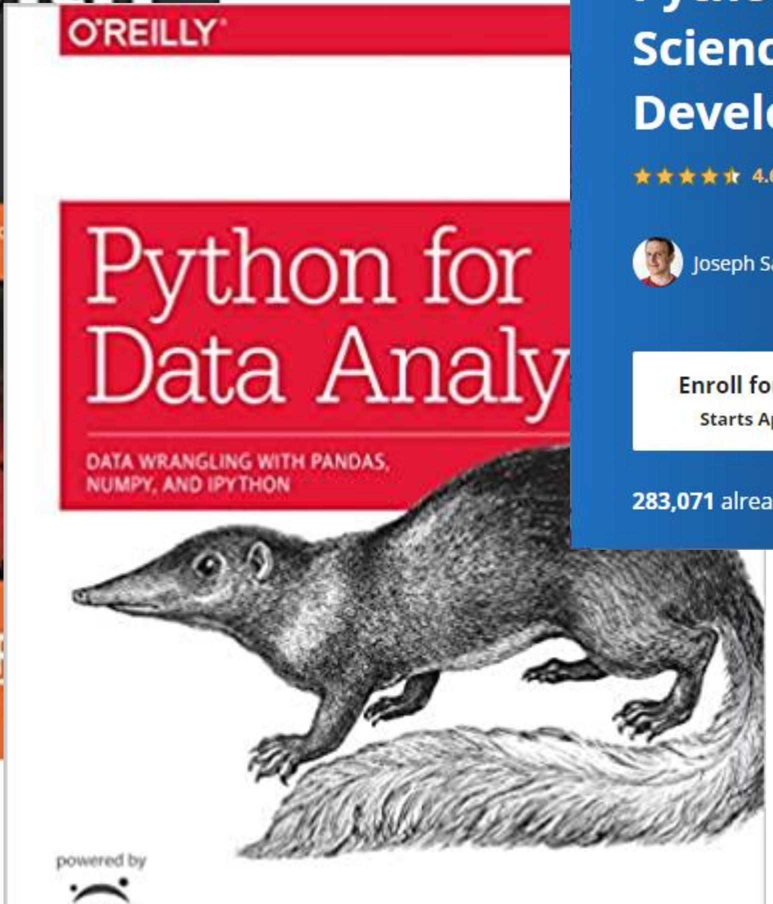
Data science with python

- Why Python?


Data Science Projects with Python

A case study approach to successful data science projec pandas, and scikit-learn

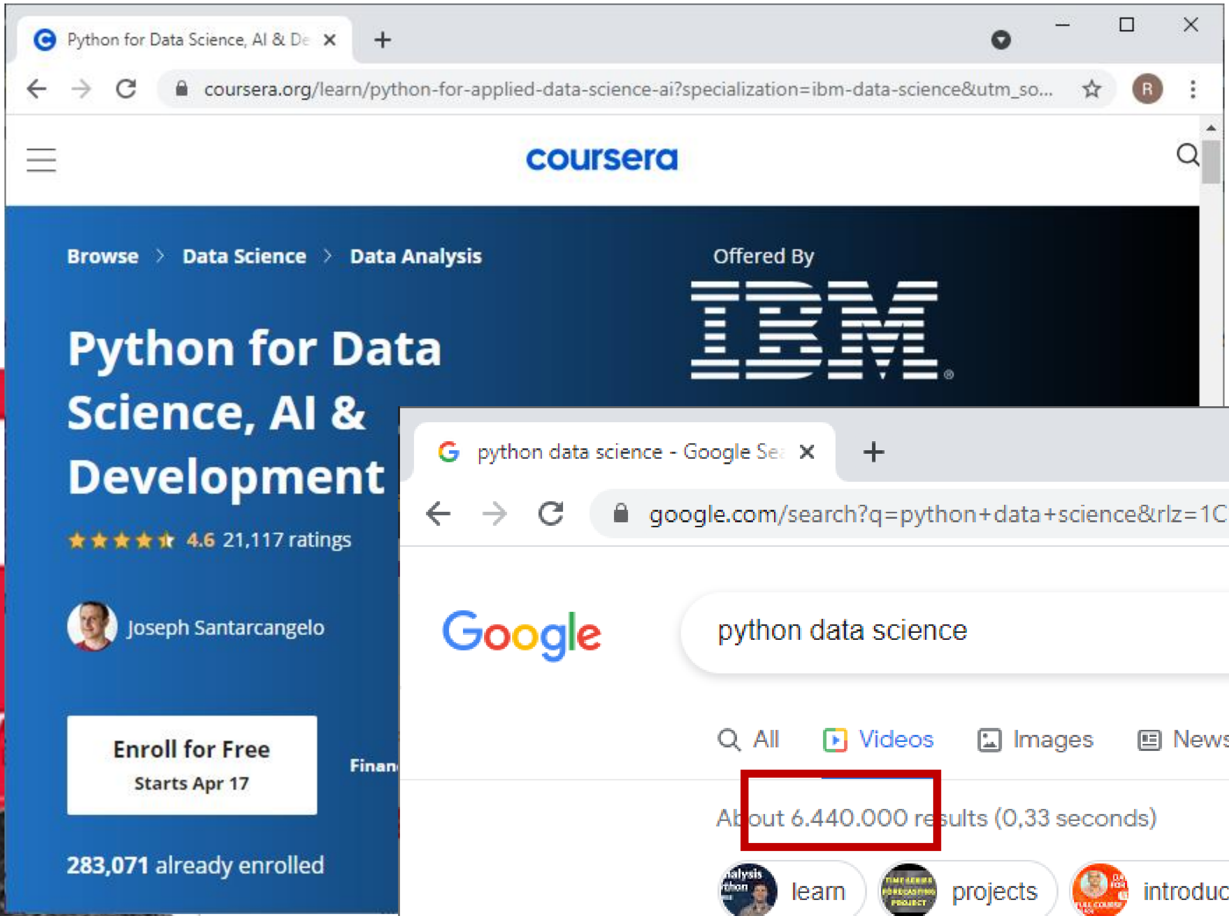
Stephen Klosterman


O'REILLY

Python for Data Analy

DATA WRANGLING WITH PANDAS, NUMPY, AND IPYTHON

powered by


Python for Data Science, AI & De

coursera.org/learn/python-for-applied-data-science-ai?specialization=ibm-data-science&utm_so...

coursera

Browse > Data Science > Data Analysis

Offered By IBM

Python for Data Science, AI & Development

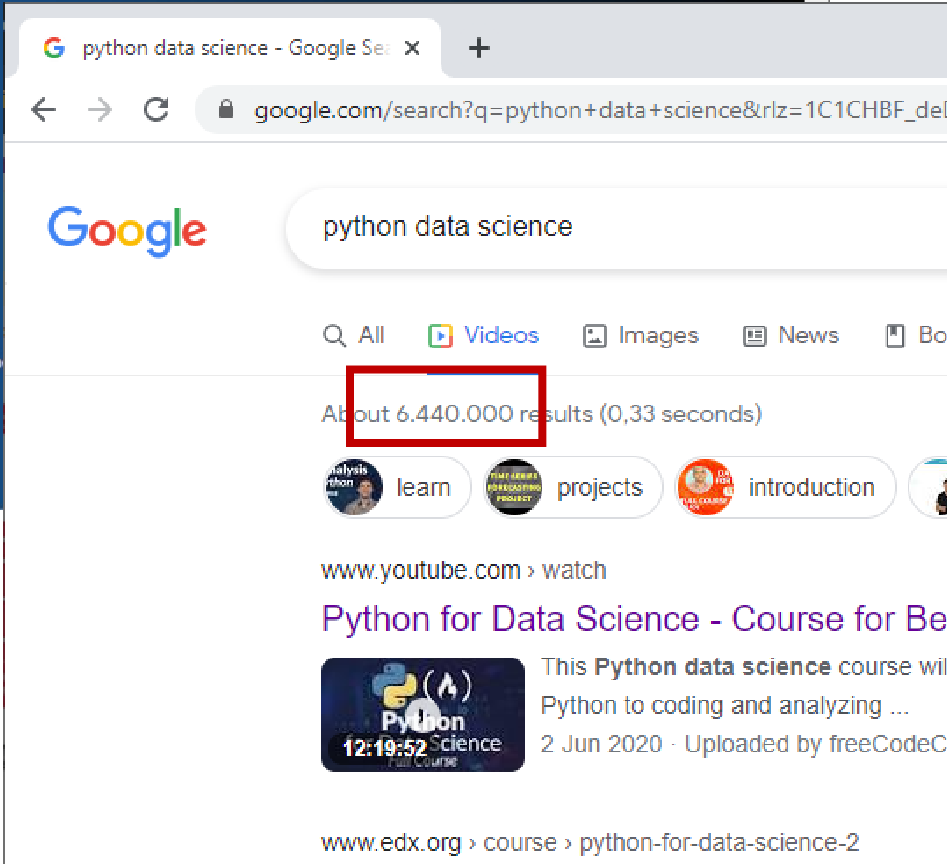
4.6 21,117 ratings

Joseph Santarcangelo

Enroll for Free Starts Apr 17

Finan

283,071 already enrolled


python data science - Google Se

google.com/search?q=python+data+science&rlz=1C1CHBF_de

Google

python data science

All Videos Images News Bo

About 6.440.000 results (0,33 seconds)

learn projects introduction

www.youtube.com › watch

Python for Data Science - Course for Be

This **Python data science** course wi Python to coding and analyzing ...

2 Jun 2020 · Uploaded by freeCodeC

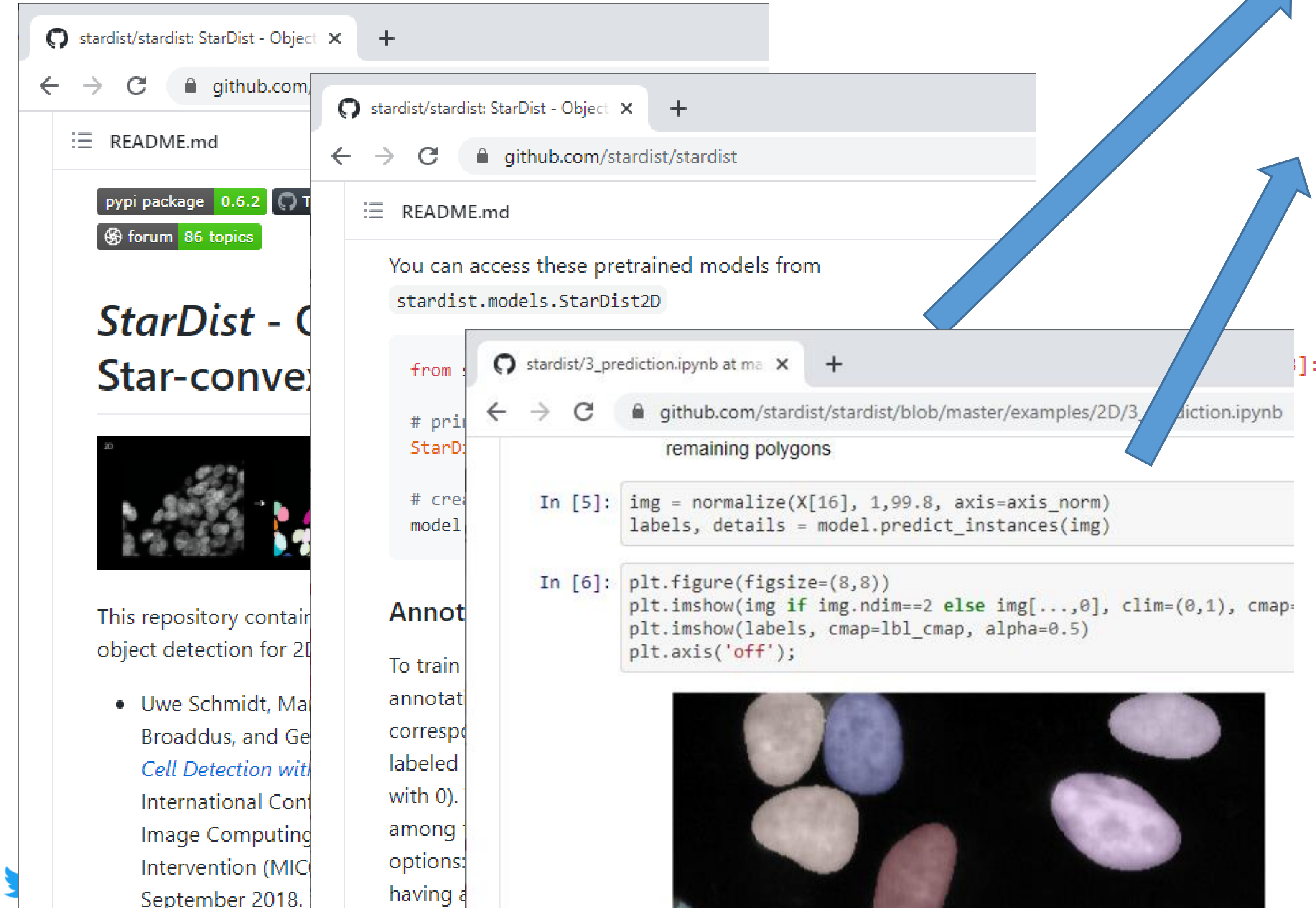
12:19:52

www.edx.org › course › python-for-data-science-2

Data science with python

- Why Python?

Because copy&paste works so great.



stardist/stardist: StarDist - Object
github.com
README.md
pypi package 0.6.2
forum 86 topics
stardist/stardist: StarDist - Object
github.com/stardist/stardist
README.md
stardist/3_prediction.ipynb at ma
github.com/stardist/stardist/blob/master/examples/2D/3_ diction.ipynb

StarDist - (
Star-conve

This repository contai
object detection for 2

- Uwe Schmidt, Ma
Broaddus, and Ge
Cell Detection wit
International Con
Image Computing
Intervention (MIC
September 2018.

You can access these pretrained models from `stardist.models.StarDist2D`

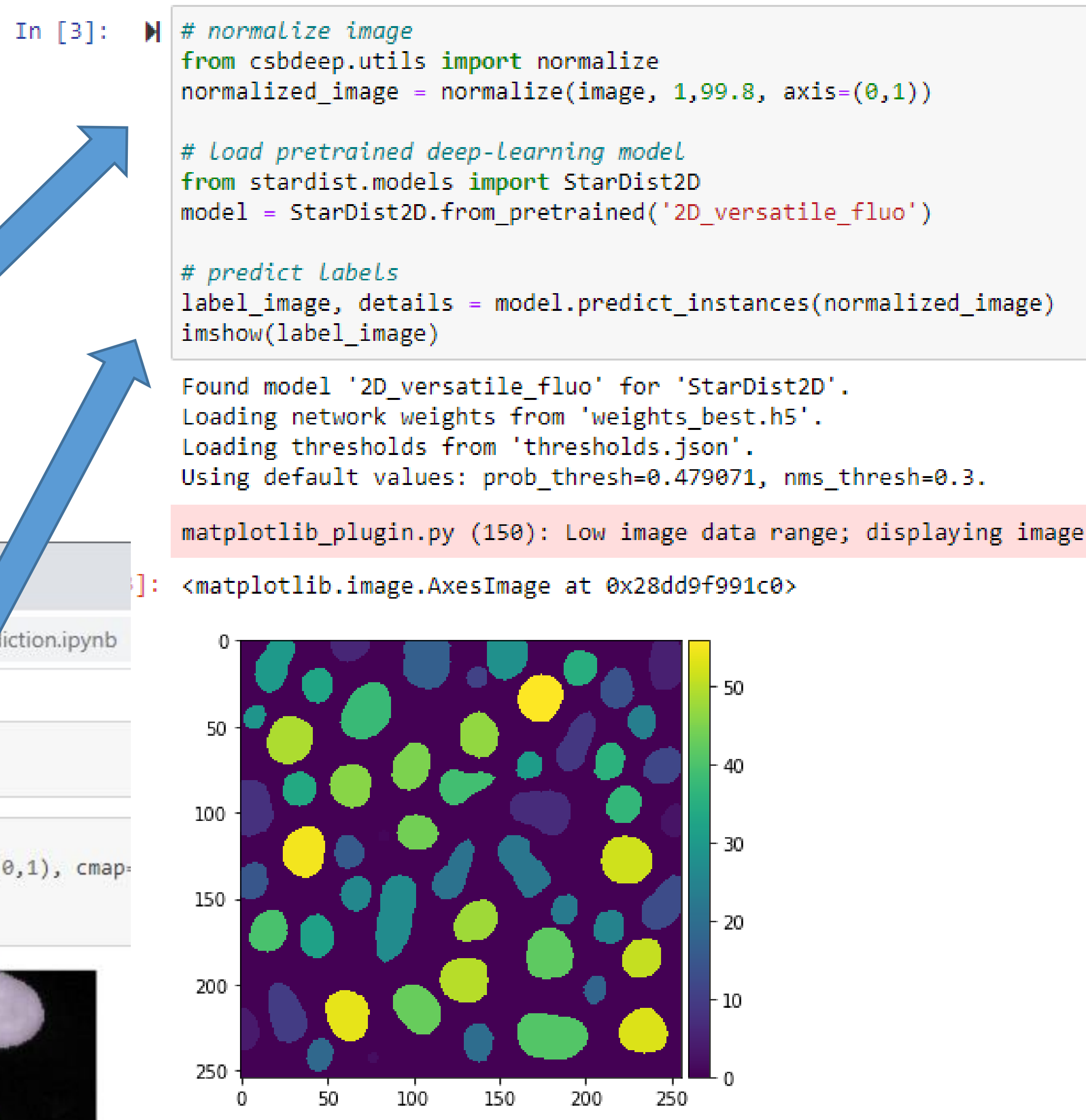
```
from s

# pri
StarD

# cre
model
```

Annot

To train
annotat
corresp
labeled
with 0).
among
options:
having a

remaining polygons

```
In [5]:  img = normalize(X[16], 1,99.8, axis=axis_norm)
         labels, details = model.predict_instances(img)

In [6]:  plt.figure(figsize=(8,8))
         plt.imshow(img if img.ndim==2 else img[...,0], clim=(0,1), cmap=
         plt.imshow(labels, cmap=lbl_cmap, alpha=0.5)
         plt.axis('off');
```

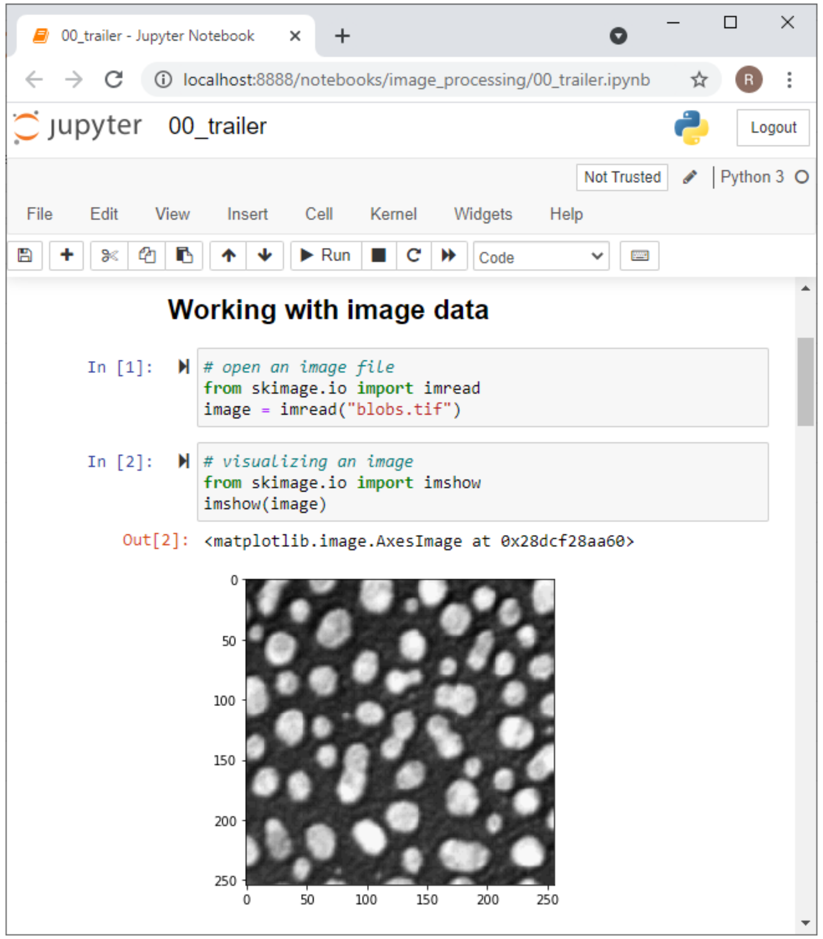


```
In [3]:  # normalize image
         from csbdeep.utils import normalize
         normalized_image = normalize(image, 1,99.8, axis=(0,1))

         # load pretrained deep-learning model
         from stardist.models import StarDist2D
         model = StarDist2D.from_pretrained('2D_versatile_fluo')

         # predict labels
         label_image, details = model.predict_instances(normalized_image)
         imshow(label_image)

Found model '2D_versatile_fluo' for 'StarDist2D'.
Loading network weights from 'weights_best.h5'.
Loading thresholds from 'thresholds.json'.
Using default values: prob_thresh=0.479071, nms_thresh=0.3.

matplotlib_plugin.py (150): Low image data range; displaying image

]:  <matplotlib.image.AxesImage at 0x28dd9f991c0>
```

https://github.com/stardist/stardist

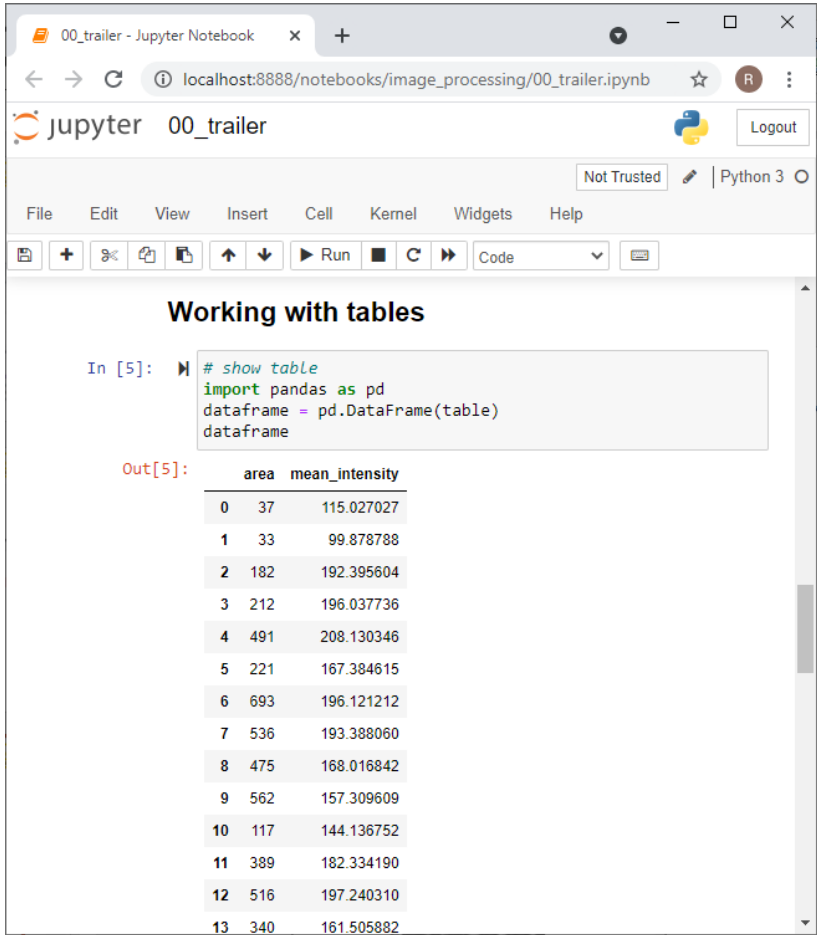
Python

- Major goals of image analysis via scripting:
  - reproducible workflows for processing images (raw data) into quantitative information
  - Sharing knowledge


00_trailer - Jupyter Notebook
localhost:8888/notebooks/image_processing/00_trailer.ipynb
jupyter 00_trailer
Logout
Not Trusted
Python 3
File Edit View Insert Cell Kernel Widgets Help
Run
Code

Working with image data

```
In [1]:  # open an image file
         from skimage.io import imread
         image = imread("blobs.tif")

In [2]:  # visualizing an image
         from skimage.io import imshow
         imshow(image)

Out[2]:  <matplotlib.image.AxesImage at 0x28dcf28aa60>
```


00_trailer - Jupyter Notebook
localhost:8888/notebooks/image_processing/00_trailer.ipynb
jupyter 00_trailer
Logout
Not Trusted
Python 3
File Edit View Insert Cell Kernel Widgets Help
Run
Code

Working with tables

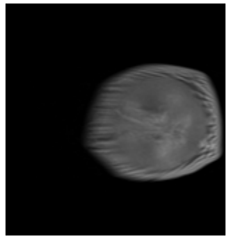
```
In [5]:  # show table
         import pandas as pd
         dataframe = pd.DataFrame(table)
         dataframe
```

Out[5]:

	area	mean_intensity
0	37	115.027027
1	33	99.878788
2	182	192.395604
3	212	196.037736
4	491	208.130346
5	221	167.384615
6	693	196.121212
7	536	193.388060
8	475	168.016842
9	562	157.309609
10	117	144.136752
11	389	182.334190
12	516	197.240310
13	340	161.505882

How to eat a banana the computational way using Python

banana0008.tif
banana0009.tif
banana0010.tif
banana0011.tif
banana0012.tif



- Remove shell
- Repeat until nothing left:
    - Take a bite
    - Chew
    - Swallow
- Digest

- Access folder
- Repeat for all images:
    - Open an image file
    - Segment the banana slice
    - Analyse it
- Save measurements

```
slice_areas = []
for root, dirs, files in os.walk(data_folder):
  for file in files:
    if file.endswith('tif'):

      # Load data
      from skimage.io import imread
      image = imread(root + file)

      # segment it
      from skimage.filters import threshold_otsu
      binary_image = image > threshold_otsu(image)

      from skimage.measure import label
      labels = label(binary_image)

      # measure radius
      from skimage.measure import regionprops
      statistics = regionprops(labels)
      areas = [s.area for s in statistics]

      # store result in array
      import numpy as np
      slice_areas.append(np.max(areas))
```

Image data source: Nasreddin Abolmaali, TU Dresden

The command line

Robert Haase

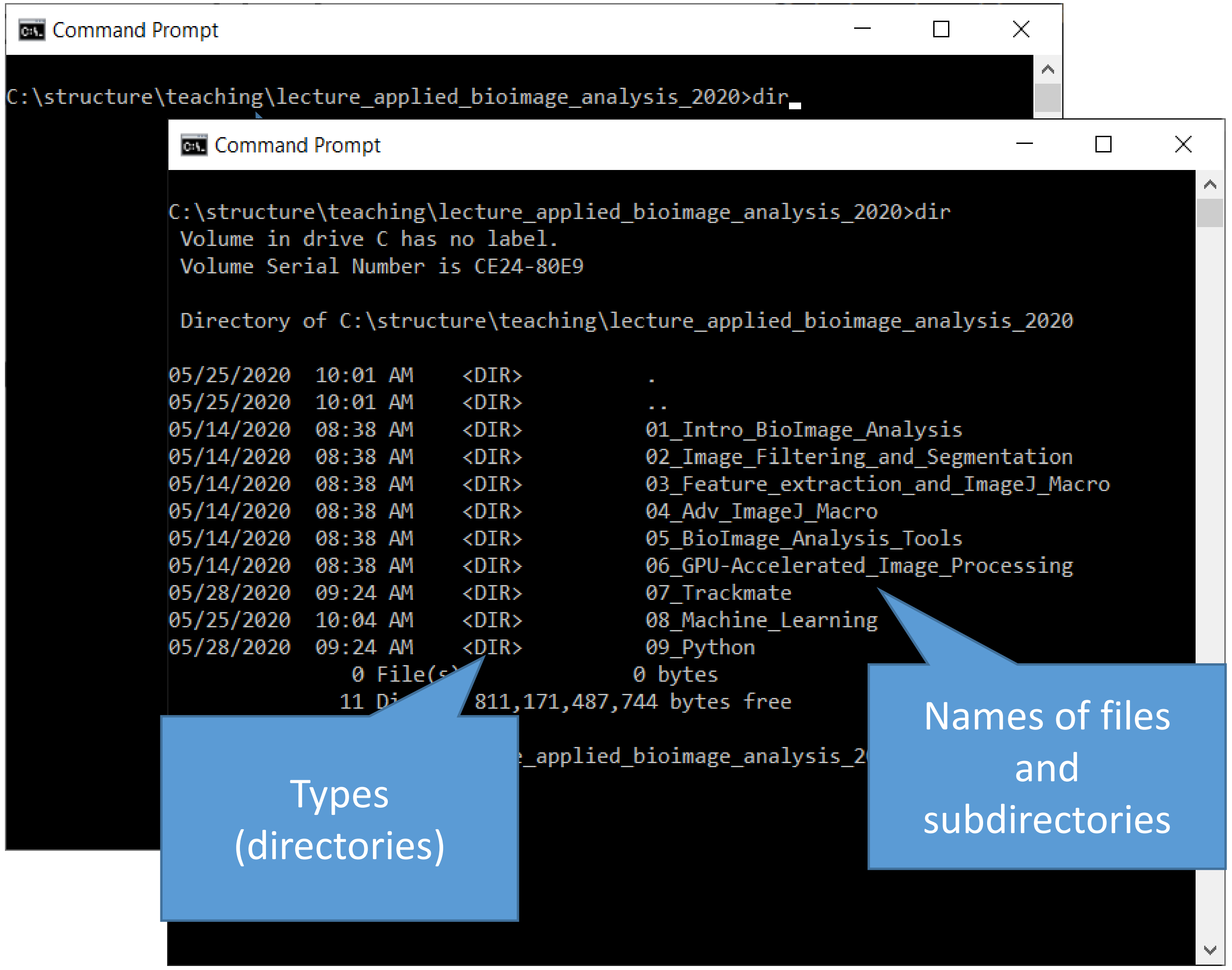
April 2021

The command line

- A.k.a. the Terminal or Eingabeaufforderung: Welcome to the 20th century!

- The `dir` command tells you what's in the current directory

- On Mac and Linux the command is called `ls -l`

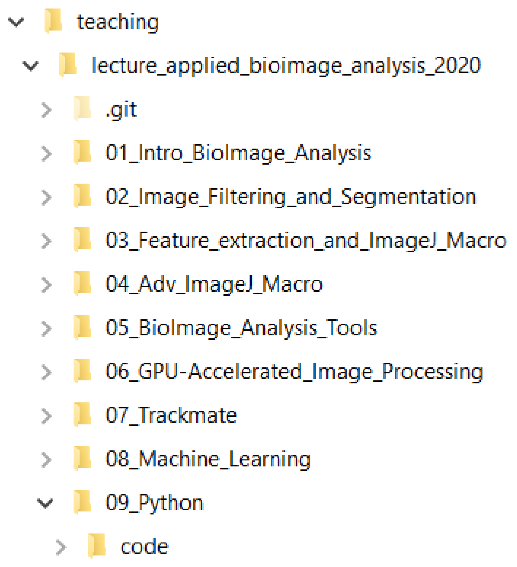


Command Prompt

```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020>dir_
```

Command Prompt

```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020>dir
 Volume in drive C has no label.
 Volume Serial Number is CE24-80E9

 Directory of C:\structure\teaching\lecture_applied_bioimage_analysis_2020

05/25/2020  10:01 AM    <DIR>          .
05/25/2020  10:01 AM    <DIR>          ..
05/14/2020  08:38 AM    <DIR>          01_Intro_BioImage_Analysis
05/14/2020  08:38 AM    <DIR>          02_Image_Filtering_and_Segmentation
05/14/2020  08:38 AM    <DIR>          03_Feature_extraction_and_ImageJ_Macro
05/14/2020  08:38 AM    <DIR>          04_Adv_ImageJ_Macro
05/14/2020  08:38 AM    <DIR>          05_BioImage_Analysis_Tools
05/14/2020  08:38 AM    <DIR>          06_GPU-Accelerated_Image_Processing
05/28/2020  09:24 AM    <DIR>          07_Trackmate
05/25/2020  10:04 AM    <DIR>          08_Machine_Learning
05/28/2020  09:24 AM    <DIR>          09_Python
               0 File(s              0 bytes
              11 Di        811,171,487,744 bytes free

                     _applied_bioimage_analysis_2
```



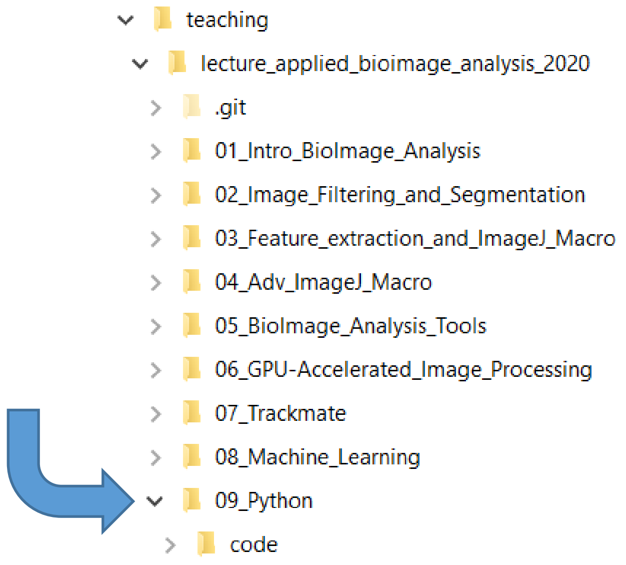
- teaching
  - lecture_applied_bioimage_analysis_2020
    - .git
    - 01_Intro_BioImage_Analysis
    - 02_Image_Filtering_and_Segmentation
    - 03_Feature_extraction_and_ImageJ_Macro
    - 04_Adv_ImageJ_Macro
    - 05_BioImage_Analysis_Tools
    - 06_GPU-Accelerated_Image_Processing
    - 07_Trackmate
    - 08_Machine_Learning
    - 09_Python
      - code

The command line

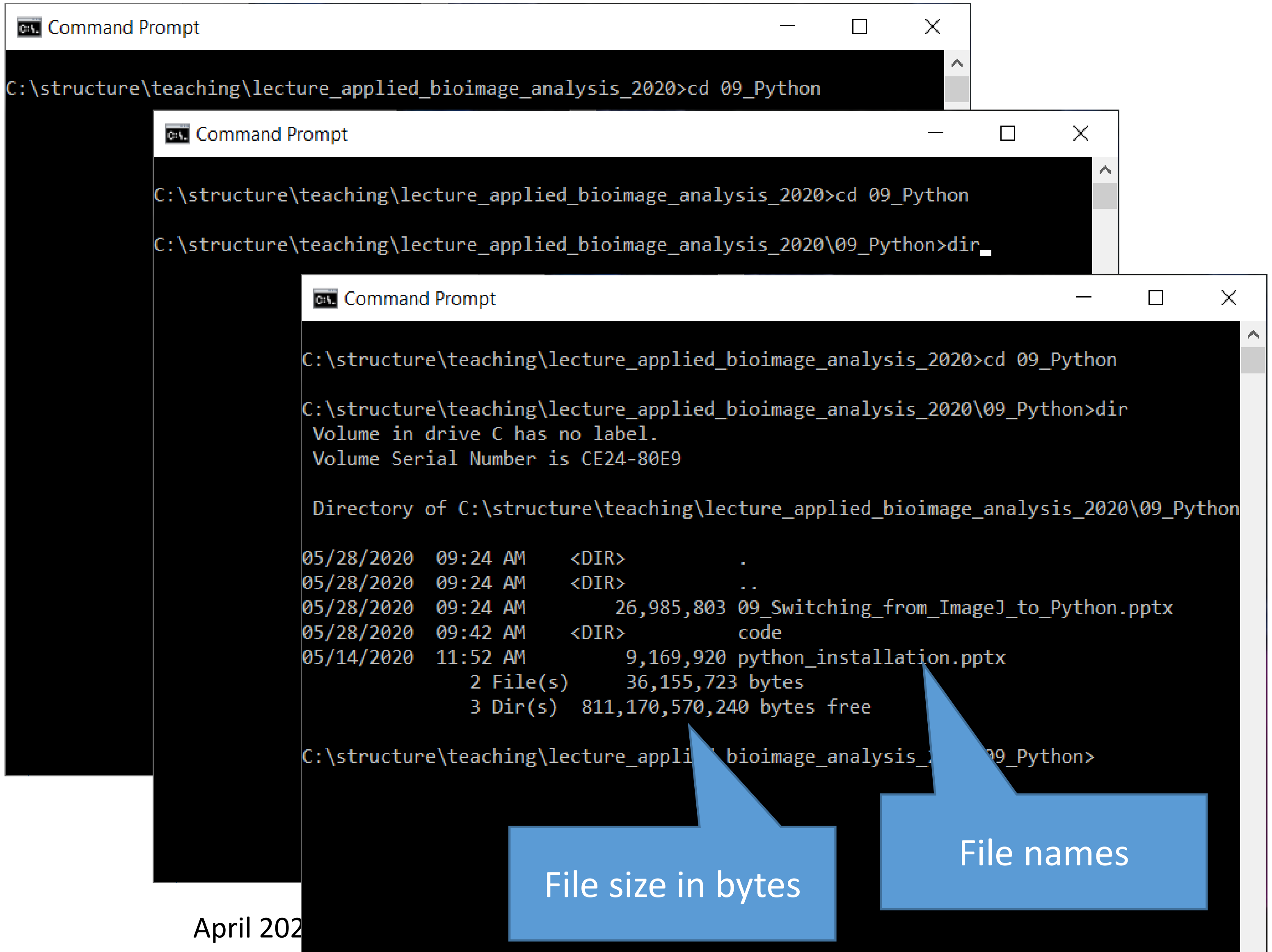
- A.k.a. the Terminal or Eingabeaufforderung: Welcome to the 20th century!

- The `cd` command let's you move between different directories.

- With `cd <pathname>` you go into a sub-directory



```
teaching
  lecture_applied_bioimage_analysis_2020
    .git
    01_Intro_BioImage_Analysis
    02_Image_Filtering_and_Segmentation
    03_Feature_extraction_and_ImageJ_Macro
    04_Adv_ImageJ_Macro
    05_BioImage_Analysis_Tools
    06_GPU-Accelerated_Image_Processing
    07_Trackmate
    08_Machine_Learning
    09_Python
      code
```



```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020>cd 09_Python
```

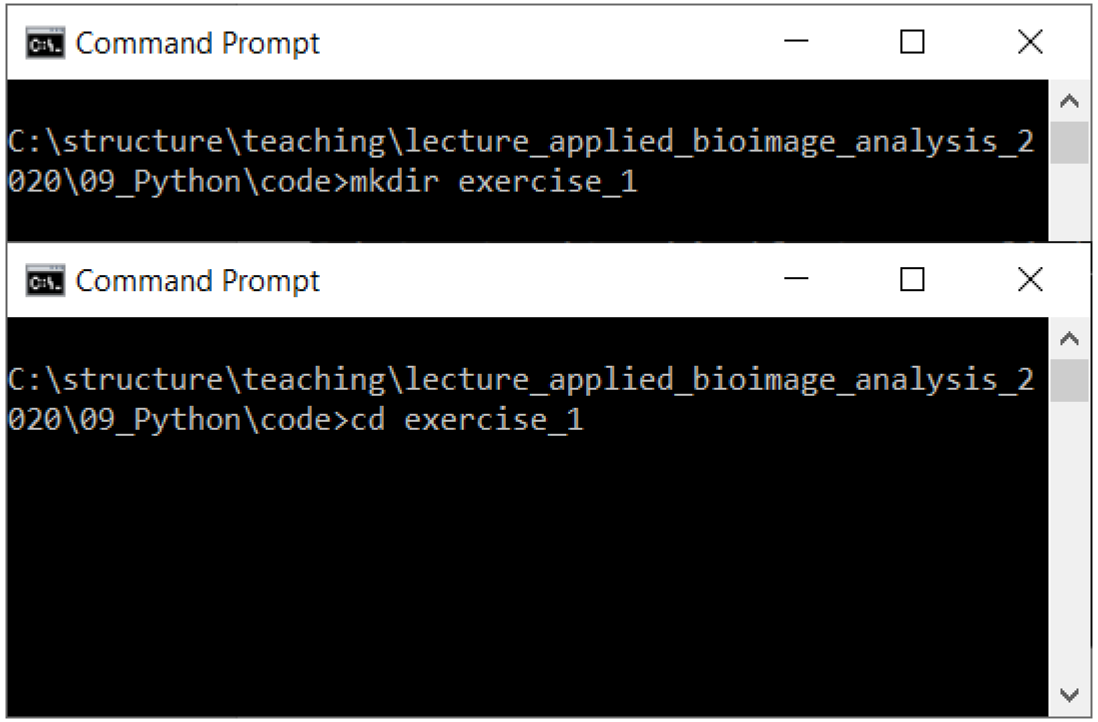
```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020>cd 09_Python

C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python>dir_
```

```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020>cd 09_Python

C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python>dir
 Volume in drive C has no label.
 Volume Serial Number is CE24-80E9

 Directory of C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python

05/28/2020  09:24 AM    <DIR>          .
05/28/2020  09:24 AM    <DIR>          ..
05/28/2020  09:24 AM        26,985,803 09_Switching_from_ImageJ_to_Python.pptx
05/28/2020  09:42 AM    <DIR>          code
05/14/2020  11:52 AM         9,169,920 python_installation.pptx
               2 File(s)     36,155,723 bytes
               3 Dir(s)  811,170,570,240 bytes free

C:\structure\teaching\lecture_appli  bioimage_analysis_  09_Python>
```

The command line

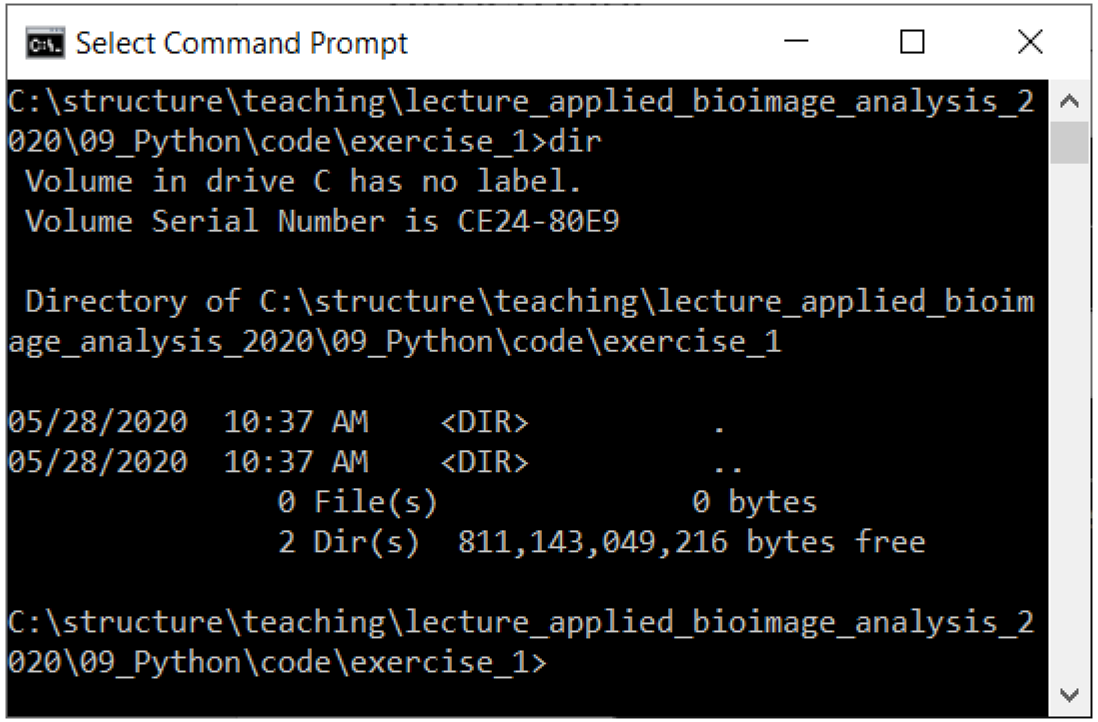
- A.k.a. the Terminal or Eingabeaufforderung: Welcome to the 20th century!

- The `mkdir` command creates new directories.



```
C:\structure\teaching\lecture_applied_bioimage_analysis_2
020\09_Python\code>mkdir exercise_1
```

```
C:\structure\teaching\lecture_applied_bioimage_analysis_2
020\09_Python\code>cd exercise_1
```



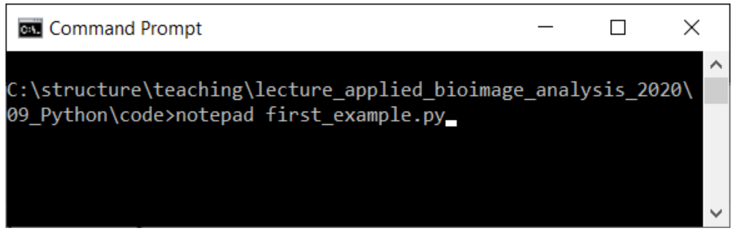
```
C:\structure\teaching\lecture_applied_bioimage_analysis_2
020\09_Python\code\exercise_1>dir
 Volume in drive C has no label.
 Volume Serial Number is CE24-80E9

 Directory of C:\structure\teaching\lecture_applied_bioim
age_analysis_2020\09_Python\code\exercise_1

05/28/2020  10:37 AM    <DIR>          .
05/28/2020  10:37 AM    <DIR>          ..
               0 File(s)              0 bytes
               2 Dir(s)  811,143,049,216 bytes free

C:\structure\teaching\lecture_applied_bioimage_analysis_2
020\09_Python\code\exercise_1>
```

The command line

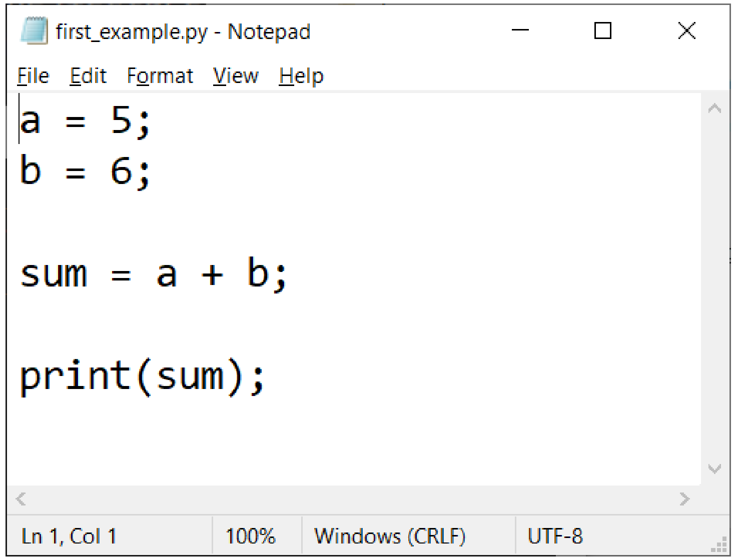
- Windows specific

- Notepad text editor

`notepad <filename>`



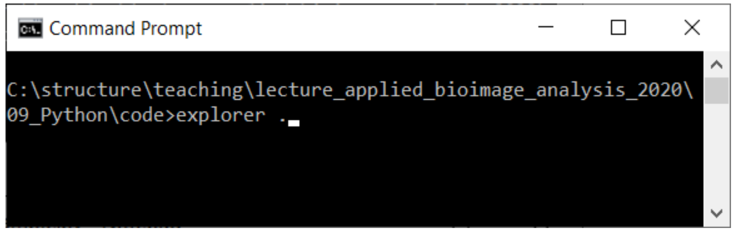
```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\
09_Python\code>notepad first_example.py
```



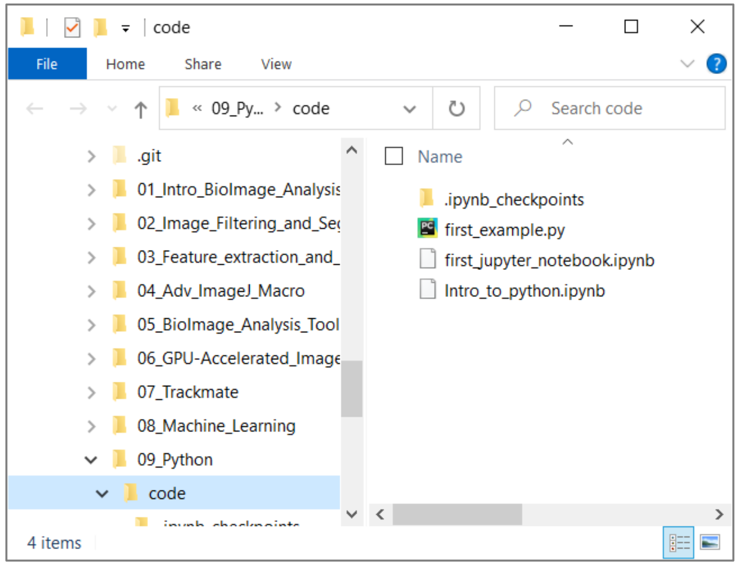
```
a = 5;
b = 6;

sum = a + b;

print(sum);
```

- Windows Explorer

`explorer .`

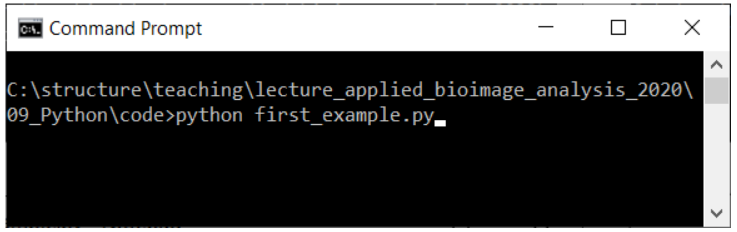


```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\
09_Python\code>explorer .
```

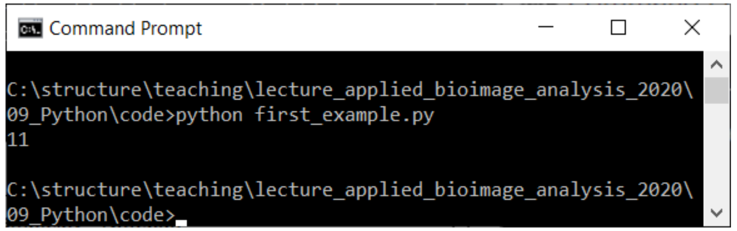


- Execute Python script

`python <filename>`



```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\
09_Python\code>python first_example.py
```



```
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\
09_Python\code>python first_example.py
11

C:\structure\teaching\lecture_applied_bioimage_analysis_2020\
09_Python\code>
```

The command line

- Mac OS specific

- Text editor

`touch <filename>` — Create a new file

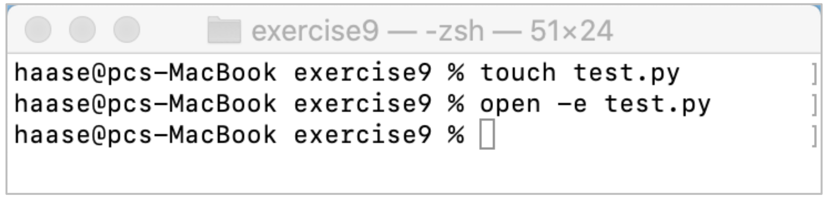
`open -e <filename>`

- Finder

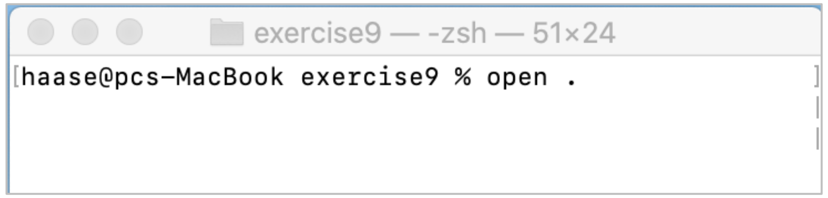
`open .`

- Execute Python script

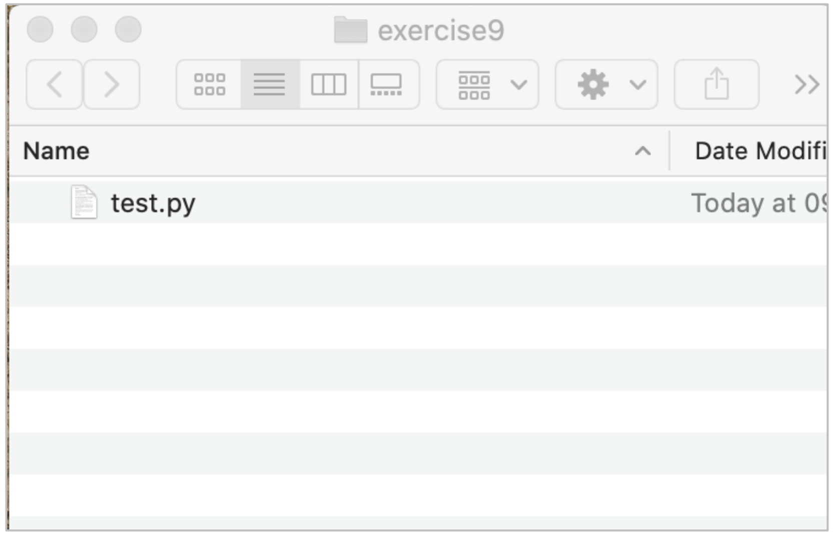
`python <filename>`



```
haase@pcs-MacBook exercise9 % touch test.py
haase@pcs-MacBook exercise9 % open -e test.py
haase@pcs-MacBook exercise9 %
```



```
haase@pcs-MacBook exercise9 % open .
```



The command line

- Linux specific

- Nano text editor

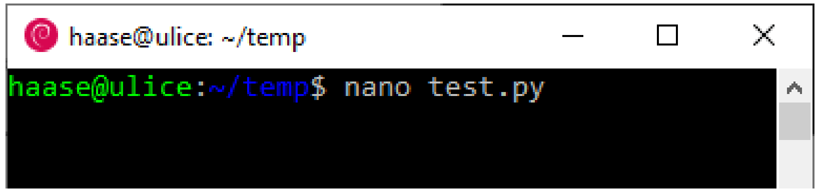
`nano <filename>`

- vim

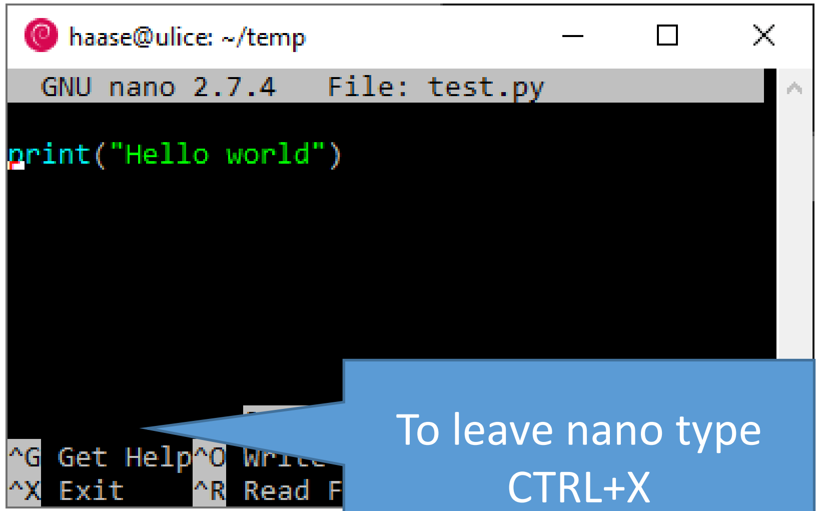
`vim <filename>`

- Execute Python script

`python <filename>`

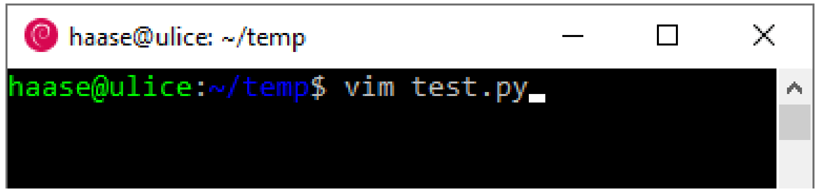

haase@ulice: ~/temp

```
haase@ulice:~/temp$ nano test.py
```

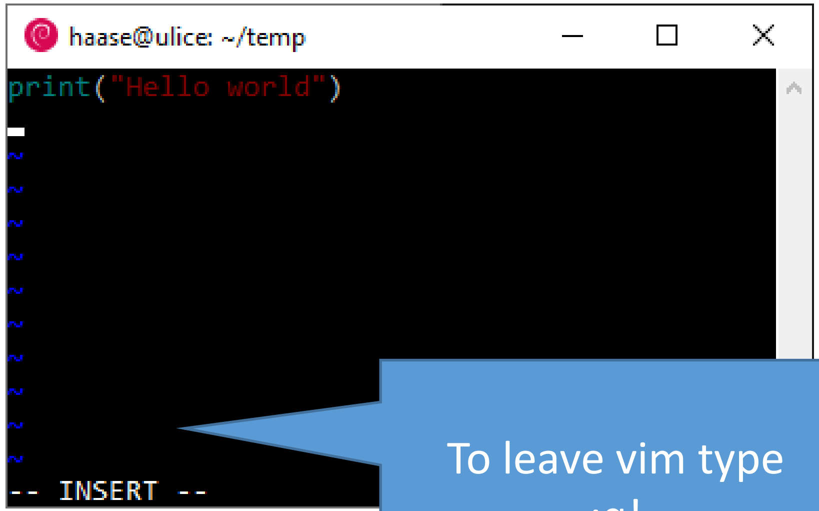

haase@ulice: ~/temp

```
  GNU nano 2.7.4    File: test.py

print("Hello world")




^G Get Help^O Write
^X Exit    ^R Read F
```

To leave nano type
CTRL+X
N

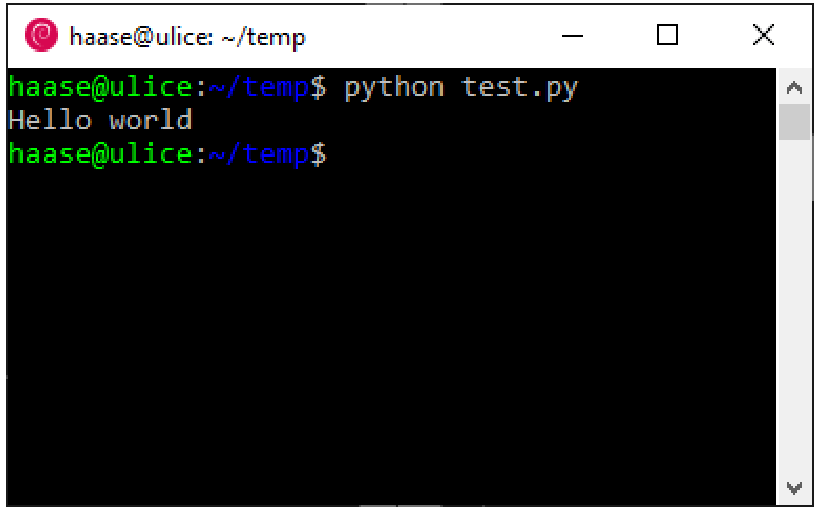

haase@ulice: ~/temp

```
haase@ulice:~/temp$ vim test.py_
```


haase@ulice: ~/temp

```
print("Hello world")

~
~
~
~
~
~
~
~
~
~
-- INSERT --
```

To leave vim type
:q!


haase@ulice: ~/temp

```
haase@ulice:~/temp$ python test.py
Hello world
haase@ulice:~/temp$
```

PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

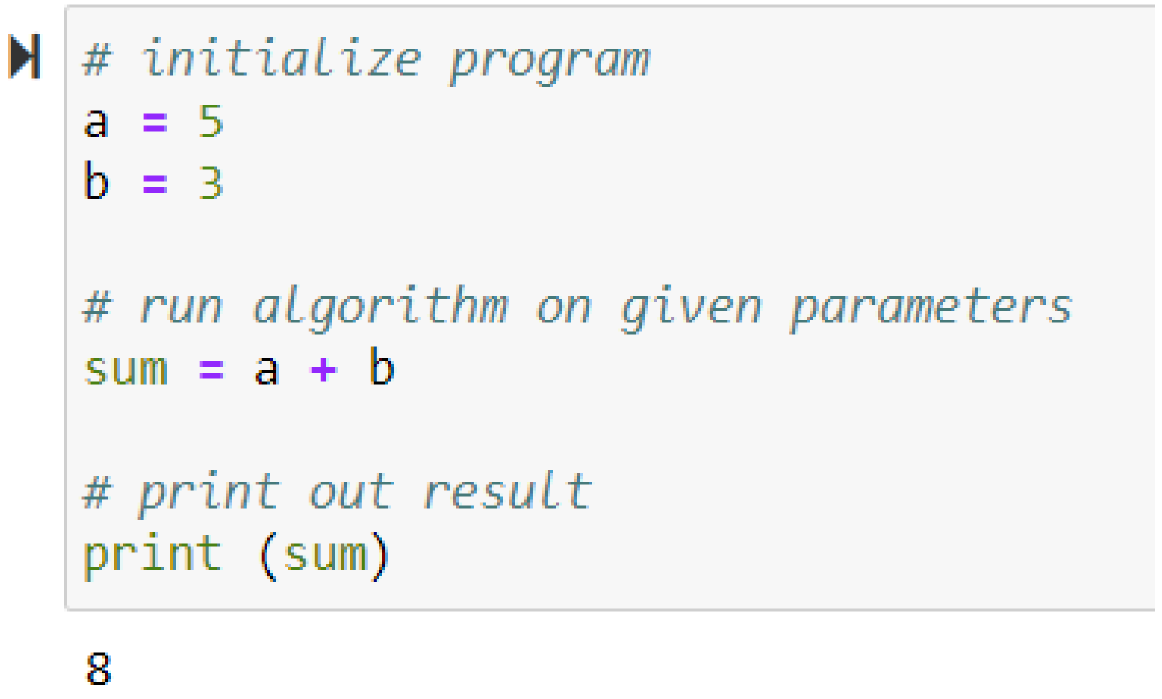
Python programming basics

Robert Haase

April 2021

Working with variables

- Variables can hold numeric values and you can do math with them



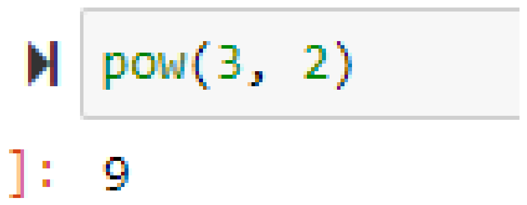
```
# initialize program
a = 5
b = 3

# run algorithm on given parameters
sum = a + b

# print out result
print (sum)
```

```
8
```

Mathematical operations

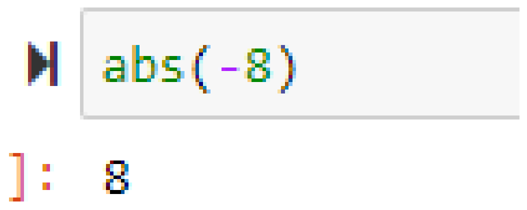
- Math commands supplement operators to be able to implement any form of calculations

- Power



```
pow(3, 2)
]: 9
```

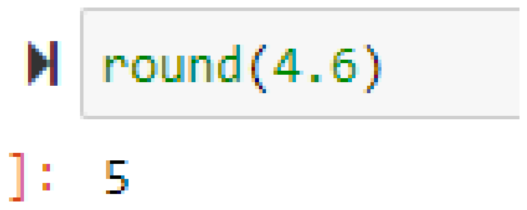
- Absolute



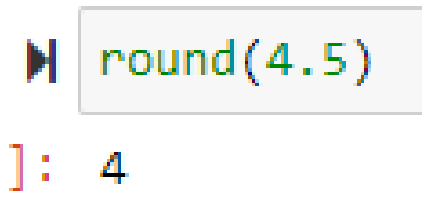
```
abs(-8)
]: 8
```

Be careful with some of them!

- Rounding



```
round(4.6)
]: 5
```



```
round(4.5)
]: 4
```

https://en.wikipedia.org/wiki/Rounding#Round_half_to_even

# Comments

Comments should contain additional information such as

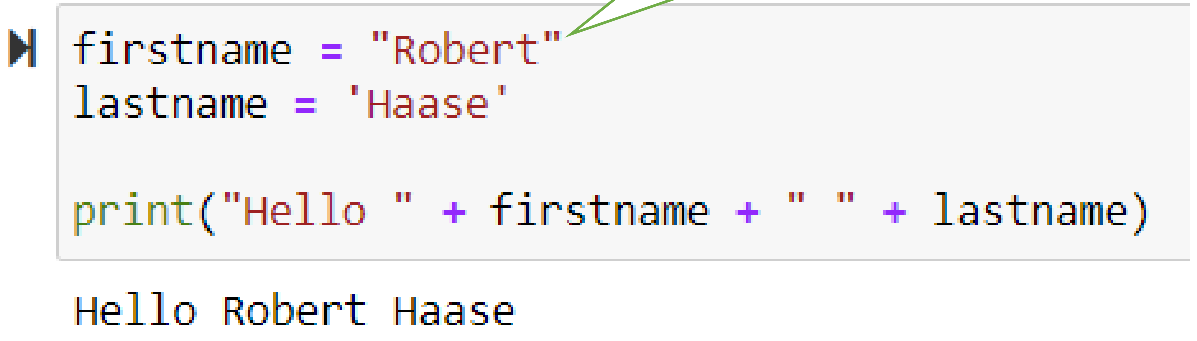
- User documentation
  - What does the program do?
  - How can this program be used?
- Your name / institute in case a reader has a question
- Comment why things are done.
- Do not comment what is written in the code already!

```
#
# This program sums up two numbers.
#
# Usage:
# * Run it in Python 3.8
#
# Author: Robert Haase, PoL TUD
#         Robert.haase@tu-dresden.de
# April 2021

# initialise program
a = 1
b = 2.5

# run complicated algorithm
final_result = a + b

#~~ print the final result~~
print( final_result )
```

Working with variables and string values

- Also strings as values for variables are supported

Single and double quotes allowed

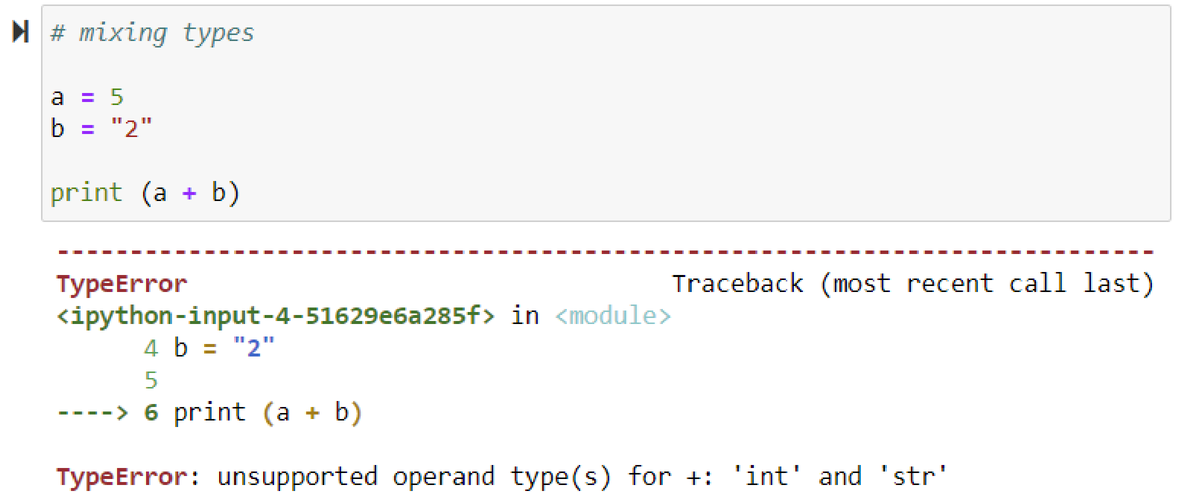


```
firstname = "Robert"
lastname = 'Haase'

print("Hello " + firstname + " " + lastname)
```

```
Hello Robert Haase
```

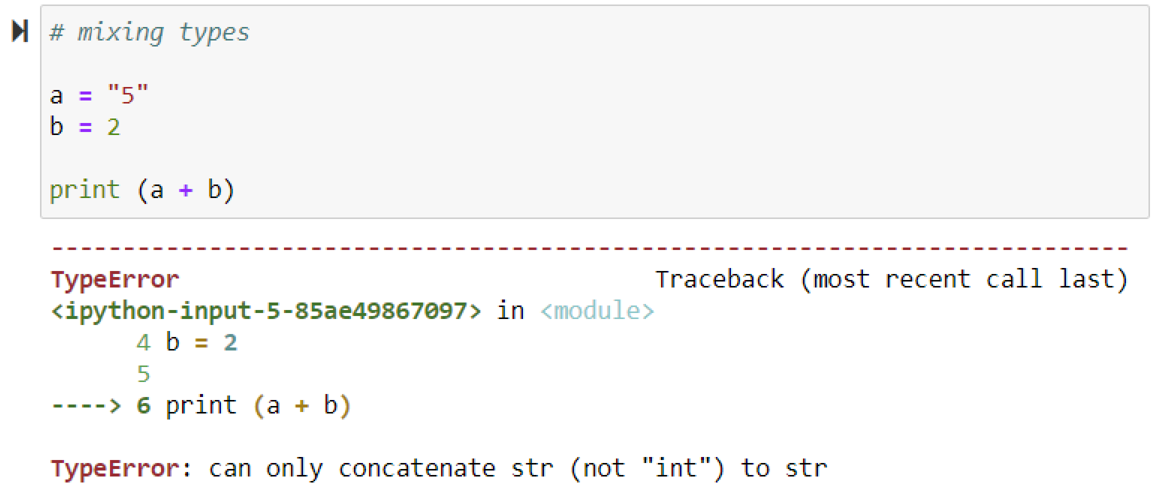
Working with variables and string values

- Also strings as values for variables are supported
- When combining strings and numbers, you need to explicitly define what you want to do.



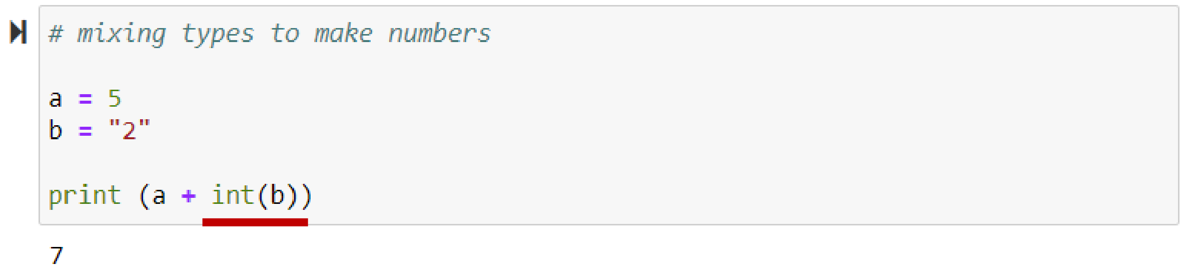
```
# mixing types

a = 5
b = "2"

print (a + b)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-4-51629e6a285f> in <module>
      4 b = "2"
      5
----> 6 print (a + b)

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```



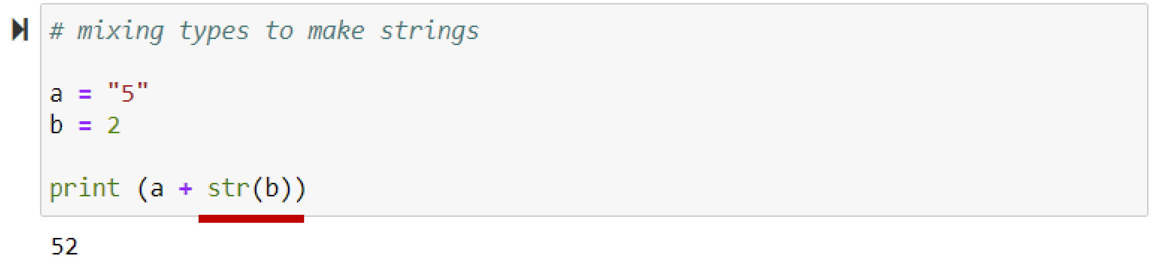
```
# mixing types

a = "5"
b = 2

print (a + b)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-5-85ae49867097> in <module>
      4 b = 2
      5
----> 6 print (a + b)

TypeError: can only concatenate str (not "int") to str
```



```
# mixing types to make numbers

a = 5
b = "2"

print (a + int(b))
```

```
7
```



```
# mixing types to make strings

a = "5"
b = 2

print (a + str(b))
```

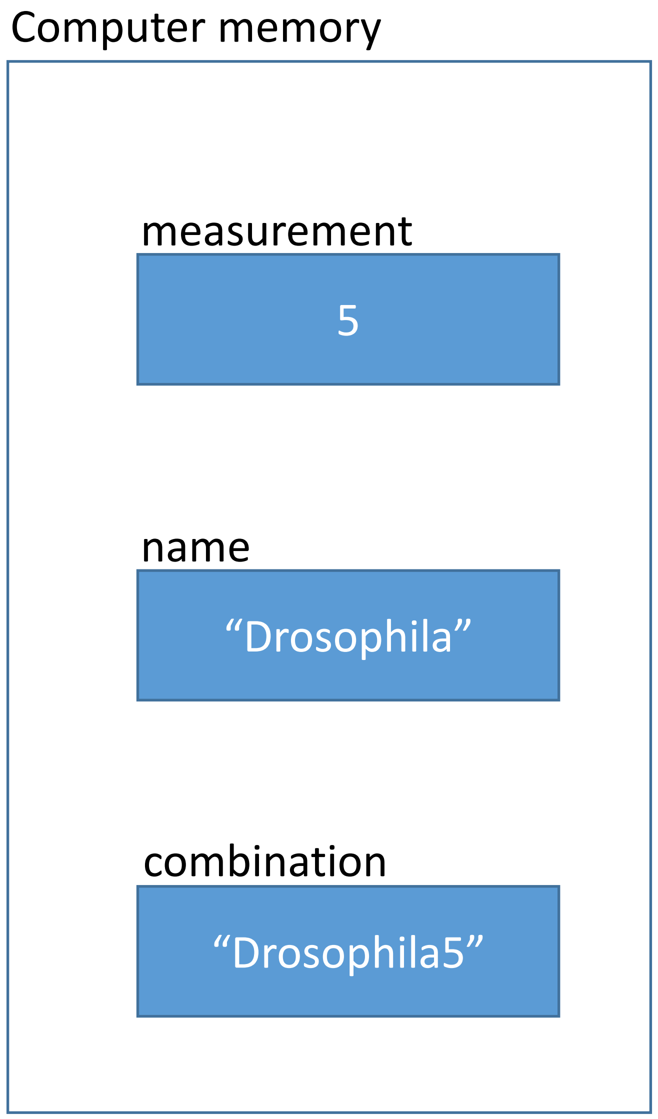
```
52
```

- Conversion to a floating point number: float()

Recap: Variables

- Variables are memory blocks where you can store stuff

```
measurement = 5
```



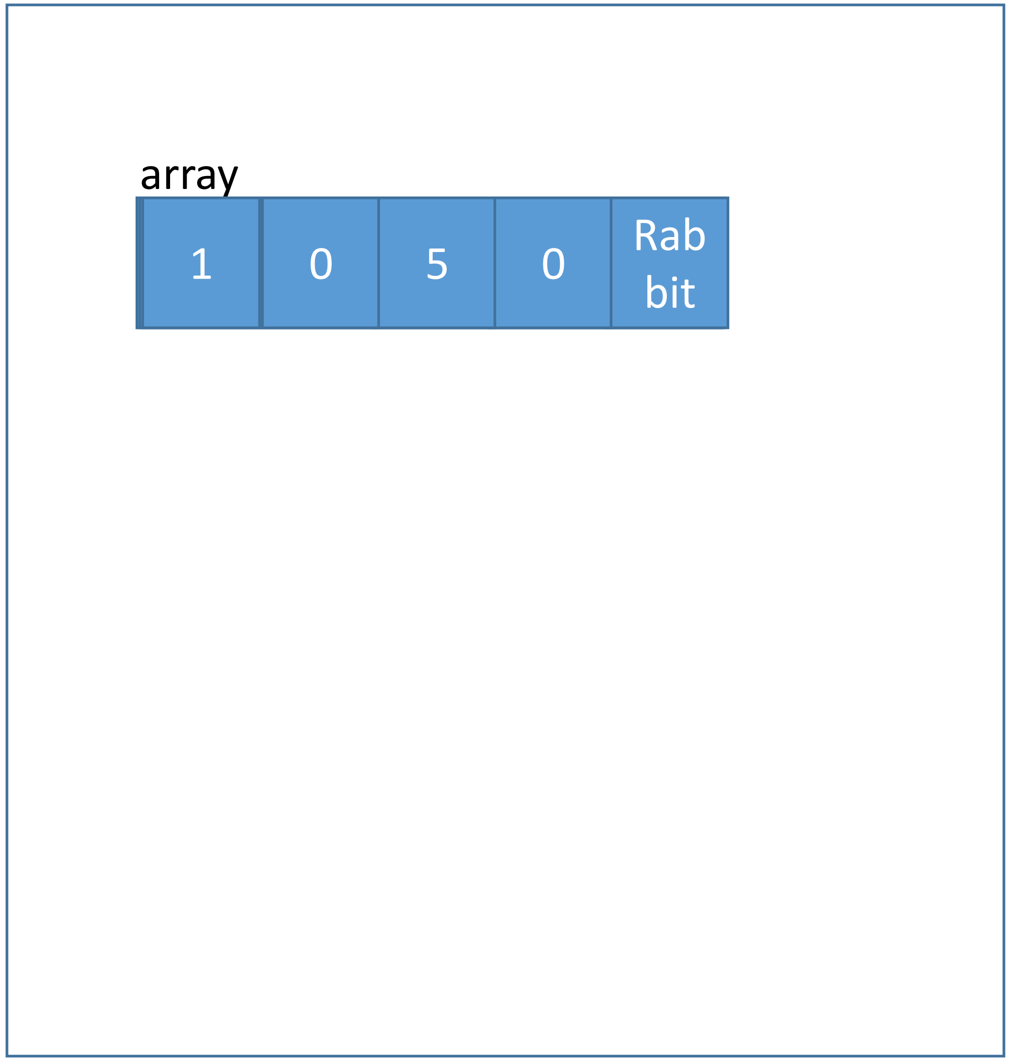
Arrays

- Arrays are variables, where you can store multiple values

Give me a "0", five times!

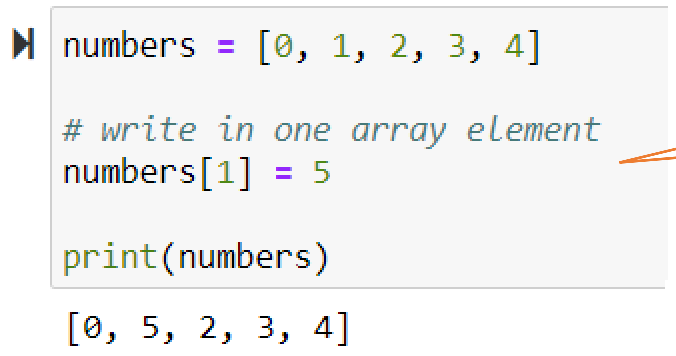
```
array = [0] * 5
```

Computer memory



Arrays in Python

- Accessing array elements



```
numbers = [0, 1, 2, 3, 4]

# write in one array element
numbers[1] = 5

print(numbers)
```

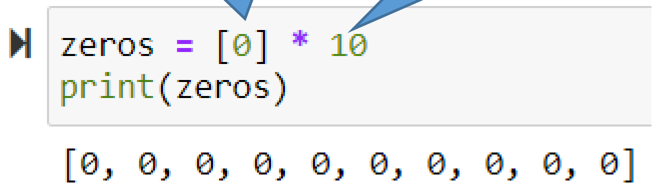
```
[0, 5, 2, 3, 4]
```

Note: The first element has index 0!

- Creating arrays of defined size

Value of all elements

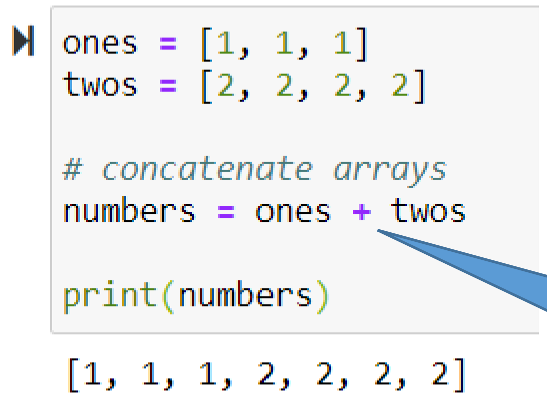
Number of elements



```
zeros = [0] * 10
print(zeros)
```

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

- Concatenating arrays



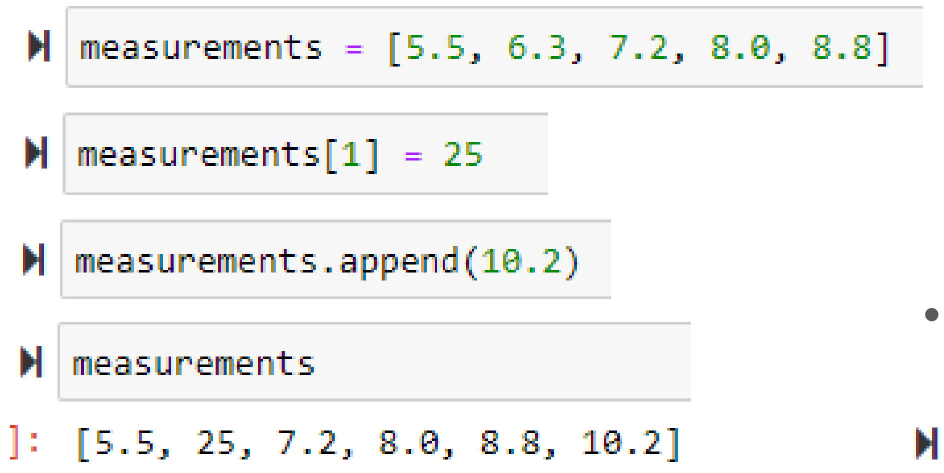
```
ones = [1, 1, 1]
twos = [2, 2, 2, 2]

# concatenate arrays
numbers = ones + twos

print(numbers)
```

```
[1, 1, 1, 2, 2, 2, 2]
```

+ means appending

Arrays: Lists versus Tuples

- Lists can be edited



```
measurements = [5.5, 6.3, 7.2, 8.0, 8.8]
```

```
measurements[1] = 25
```

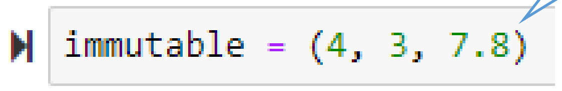
```
measurements.append(10.2)
```

```
measurements
```

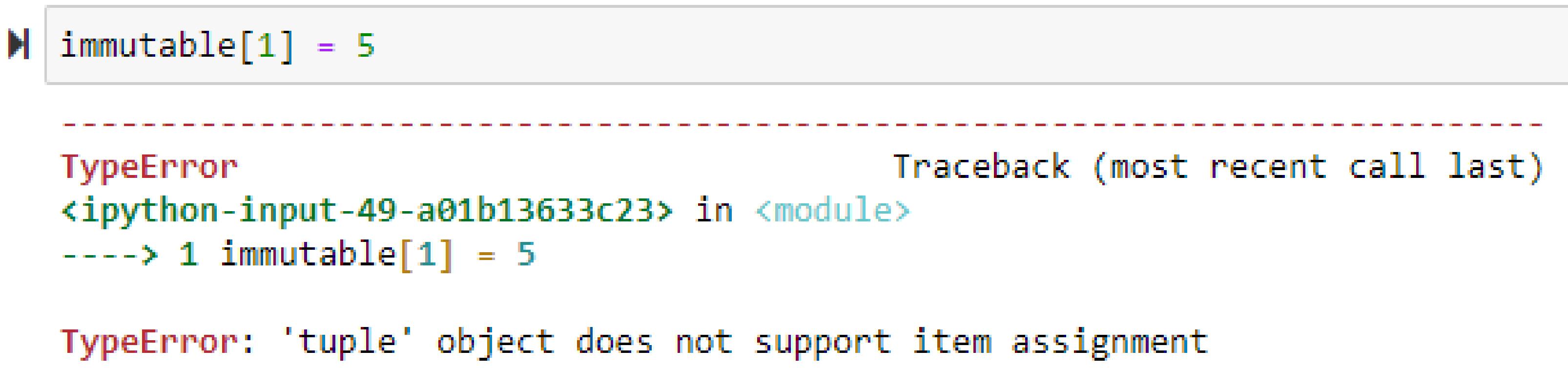
```
]: [5.5, 25, 7.2, 8.0, 8.8, 10.2]
```

- Tuples not

Note: round brackets



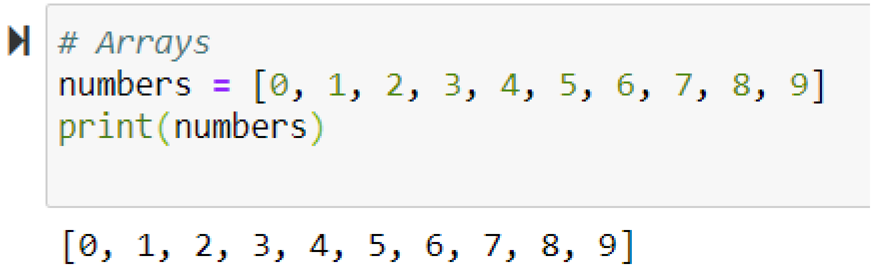
```
immutable = (4, 3, 7.8)
```



```
immutable[1] = 5
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-49-a01b13633c23> in <module>
----> 1 immutable[1] = 5

TypeError: 'tuple' object does not support item assignment
```

Subsets



```
# Arrays
numbers = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
print(numbers)
```

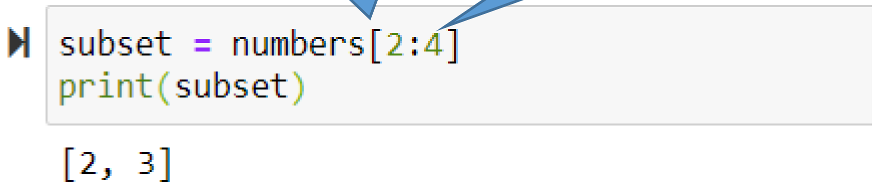
```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

- Creating subsets of arrays

Starting at (including)

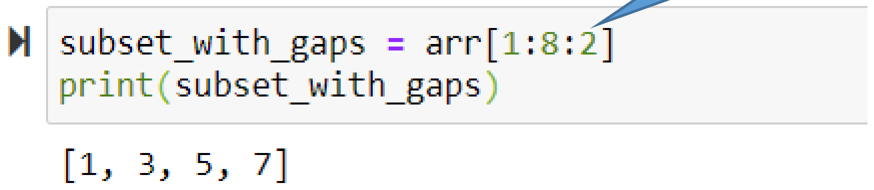
Ending at (excluding)

Step size



```
subset = numbers[2:4]
print(subset)
```

```
[2, 3]
```



```
subset_with_gaps = arr[1:8:2]
print(subset_with_gaps)
```

```
[1, 3, 5, 7]
```

Arrays in Python

- Arrays can contain anything – including strings

```
string_array = ["A", "B", "C", "D", "E", "F"]
print(string_array)
```

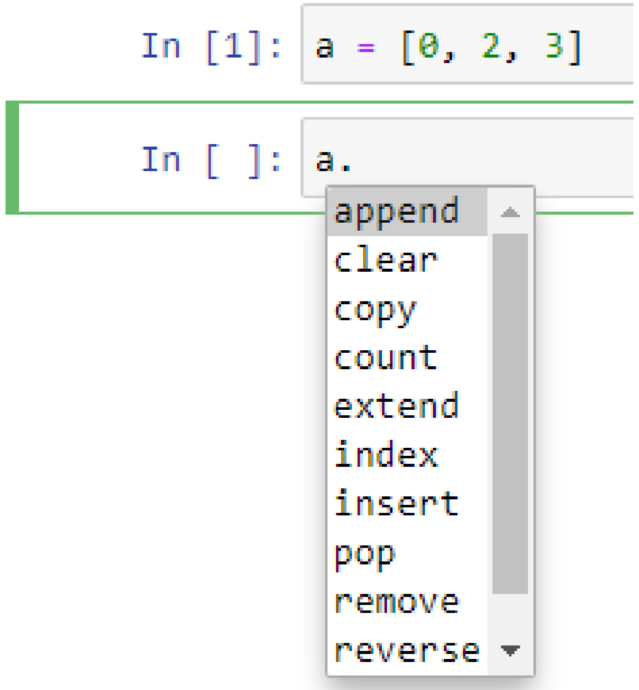
```
['A', 'B', 'C', 'D', 'E', 'F']
```

- And work then exactly the same as numeric arrays

```
string_subset = string_array[1:4]
print(string_subset)
```

```
['B', 'C', 'D']
```

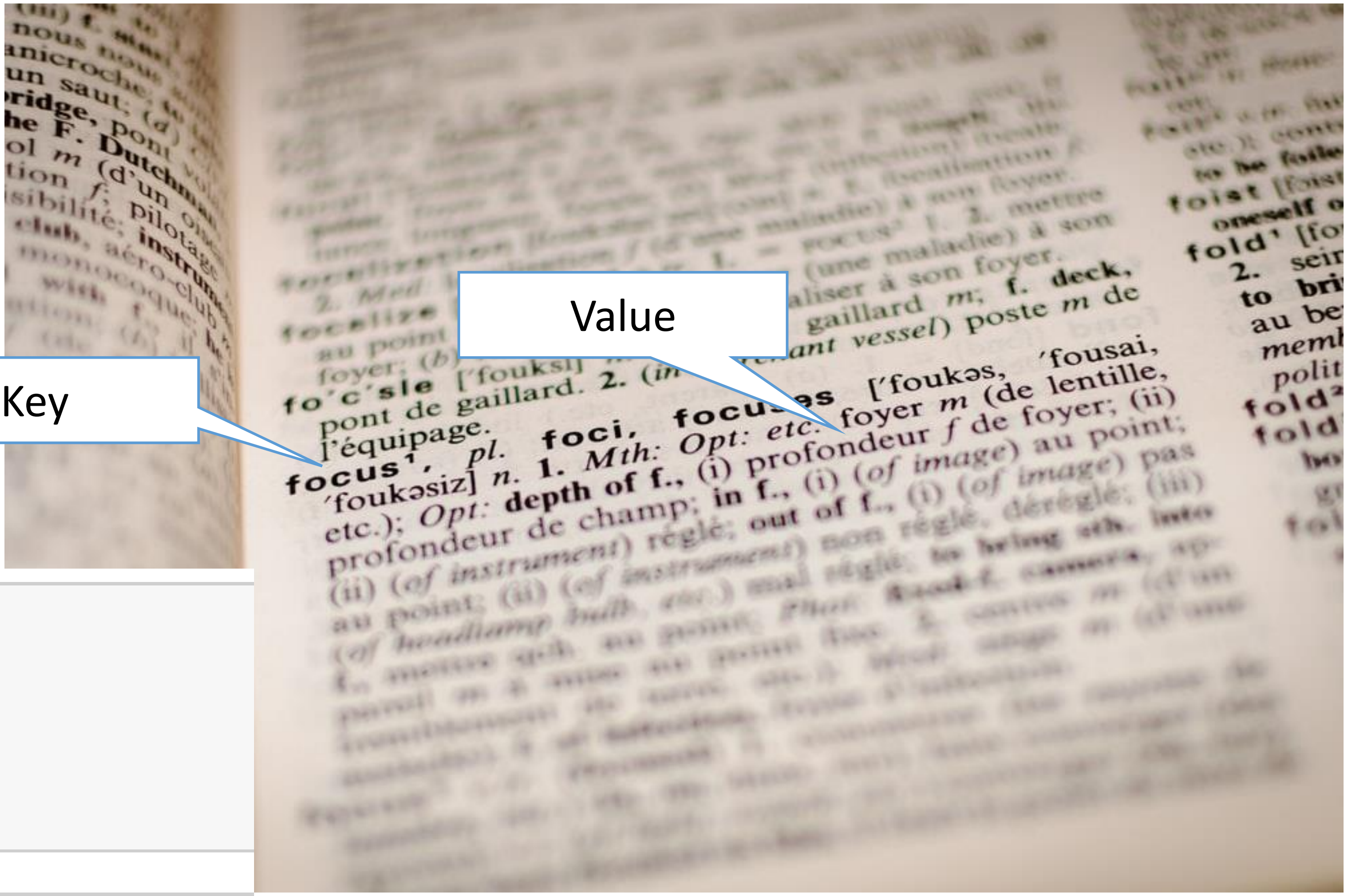
- Find out more with <TAB>:



```
In [1]: a = [0, 2, 3]

In [ ]: a.
```

Dictionaries

- Dictionary: a list of key-value pairs



Key

Value

```
german_english_dictionary = {
    'Vorlesung':'Lecture',
    'Gleichung':'Equation'
}
```

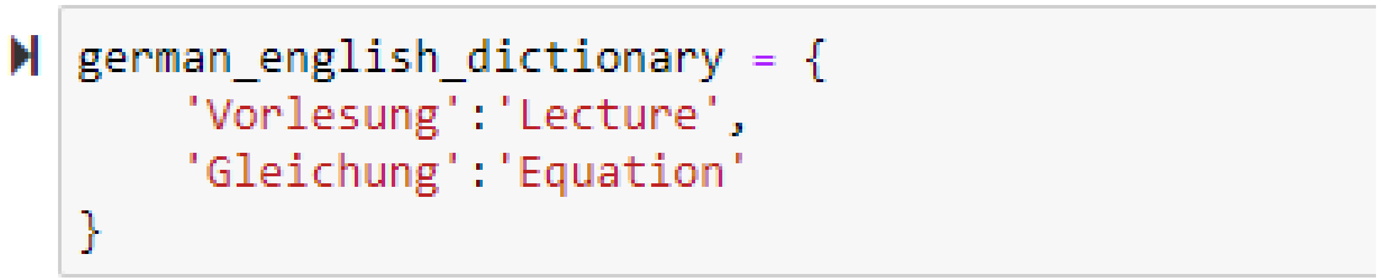
```
german_english_dictionary
```

```
]: {'Vorlesung': 'Lecture', 'Gleichung': 'Equation'}
```

Image source: Romain Vignes

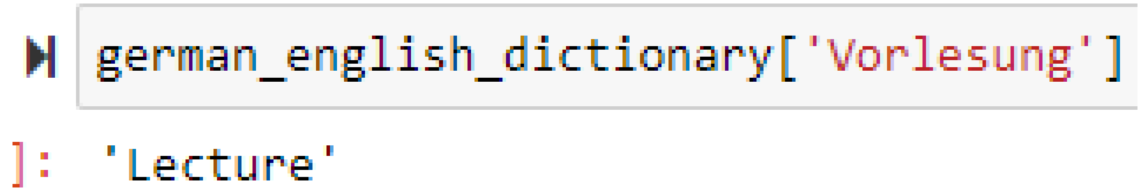
Dictionaries

- Dictionary: a list of key-value pairs



```
german_english_dictionary = {
    'Vorlesung':'Lecture',
    'Gleichung':'Equation'
}
```

- Look up something in the dictionary

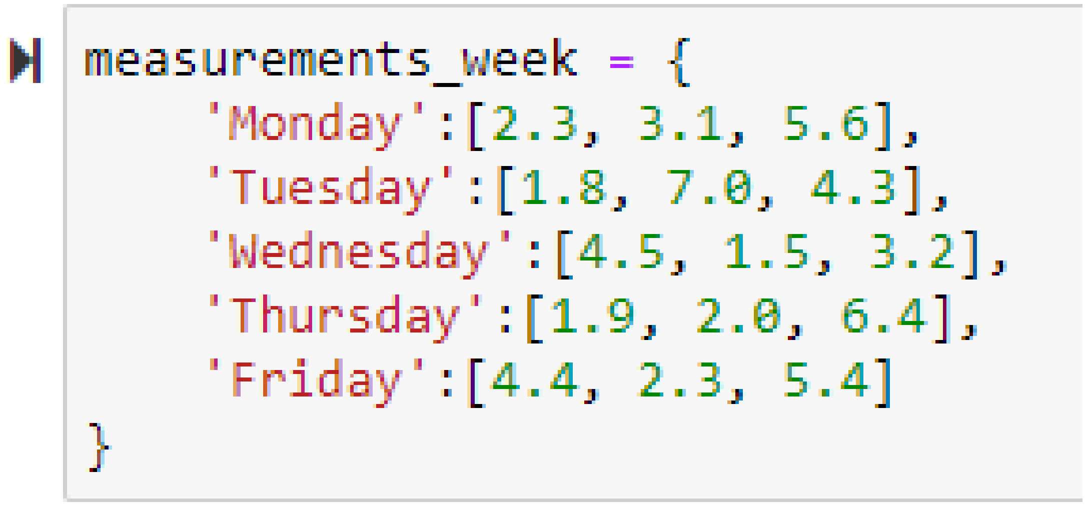


```
german_english_dictionary['Vorlesung']
```

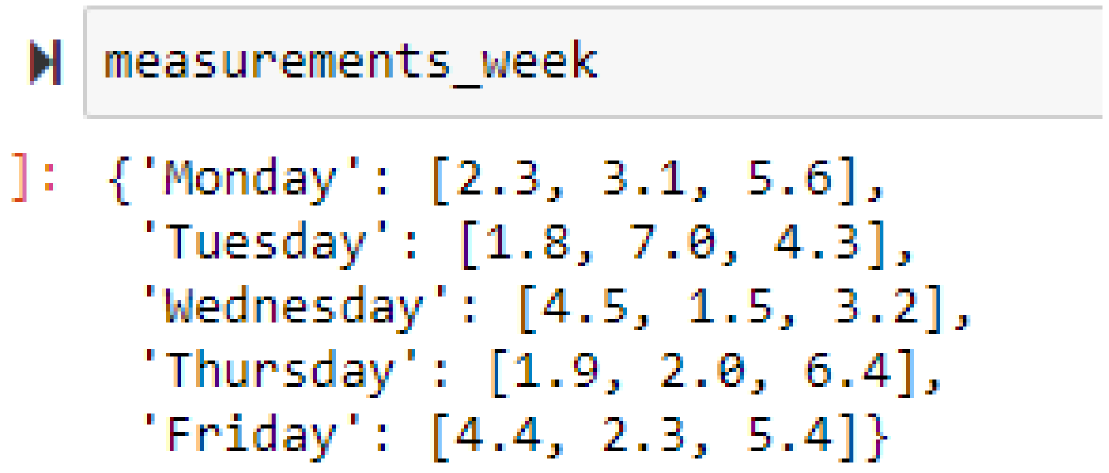
```
]:  'Lecture'
```

Tables

- Tables can be dictionaries with lists as values



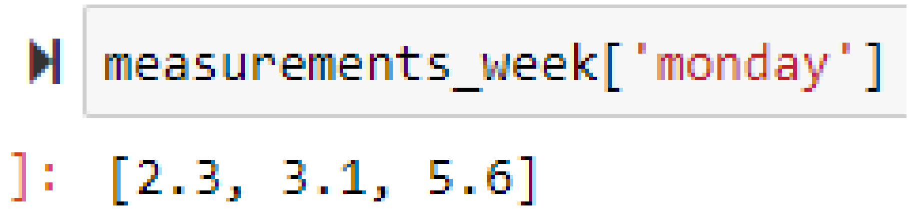
```
measurements_week = {
        'Monday':[2.3, 3.1, 5.6],
        'Tuesday':[1.8, 7.0, 4.3],
        'Wednesday':[4.5, 1.5, 3.2],
        'Thursday':[1.9, 2.0, 6.4],
        'Friday':[4.4, 2.3, 5.4]
}
```



```
measurements_week
```

```
]:  {'Monday': [2.3, 3.1, 5.6],
     'Tuesday': [1.8, 7.0, 4.3],
     'Wednesday': [4.5, 1.5, 3.2],
     'Thursday': [1.9, 2.0, 6.4],
     'Friday': [4.4, 2.3, 5.4]}
```

- Retrieve a column

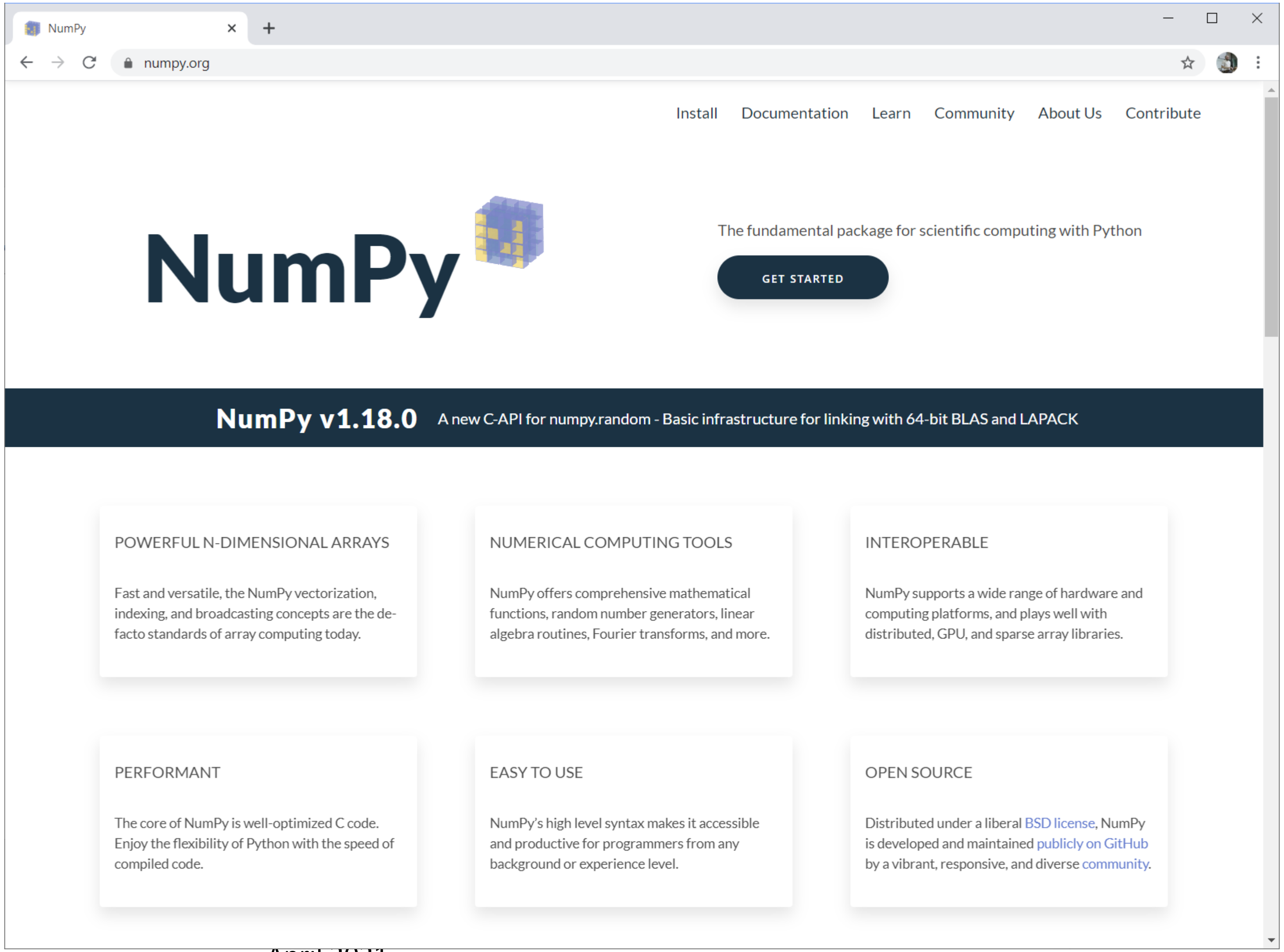


```
measurements_week['monday']
```

```
]:  [2.3, 3.1, 5.6]
```

numpy

- The fundamental package for scientific computing with python.
- `conda install numpy`



NumPy

numpy.org

Install Documentation Learn Community About Us Contribute

NumPy

The fundamental package for scientific computing with Python

GET STARTED

NumPy v1.18.0 A new C-API for numpy.random - Basic infrastructure for linking with 64-bit BLAS and LAPACK

POWERFUL N-DIMENSIONAL ARRAYS

Fast and versatile, the NumPy vectorization, indexing, and broadcasting concepts are the de-facto standards of array computing today.

NUMERICAL COMPUTING TOOLS

NumPy offers comprehensive mathematical functions, random number generators, linear algebra routines, Fourier transforms, and more.

INTEROPERABLE

NumPy supports a wide range of hardware and computing platforms, and plays well with distributed, GPU, and sparse array libraries.

PERFORMANT

The core of NumPy is well-optimized C code. Enjoy the flexibility of Python with the speed of compiled code.

EASY TO USE

NumPy's high level syntax makes it accessible and productive for programmers from any background or experience level.

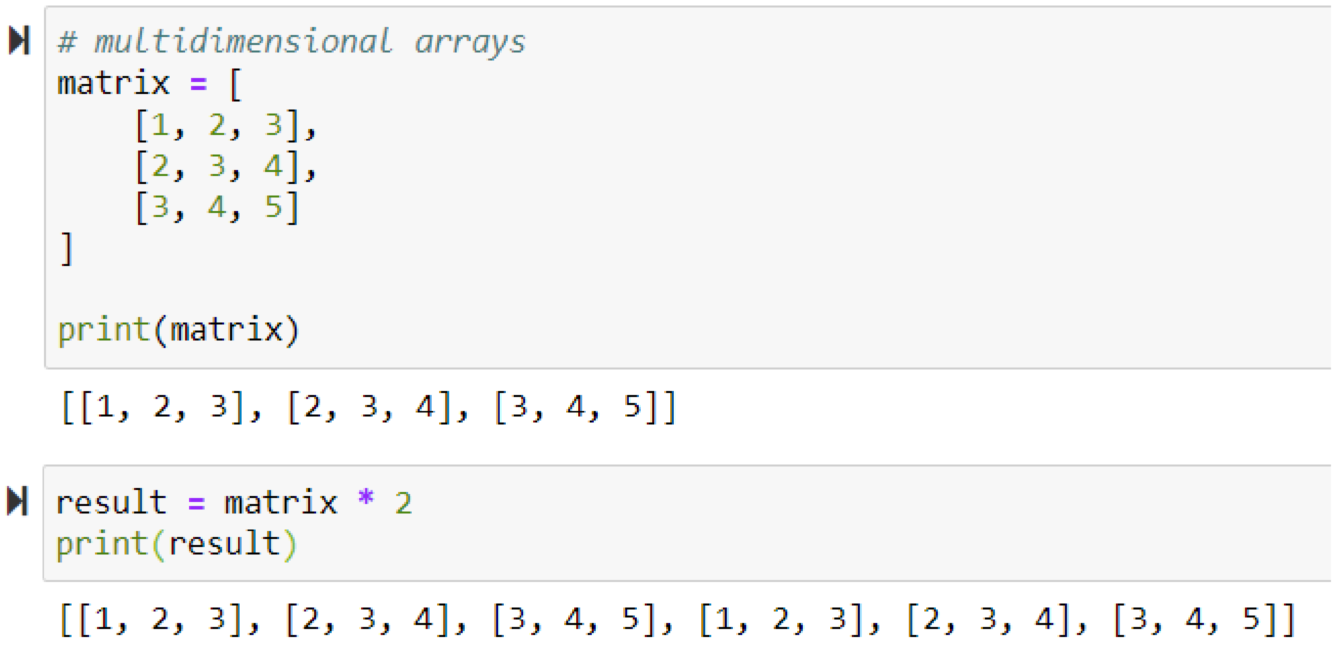
OPEN SOURCE

Distributed under a liberal BSD license, NumPy is developed and maintained publicly on GitHub by a vibrant, responsive, and diverse community.

numpy

- Simplifying mathematical operations on n-dimensional arrays

- Python arrays



```
# multidimensional arrays
matrix = [
    [1, 2, 3],
    [2, 3, 4],
    [3, 4, 5]
]

print(matrix)
```

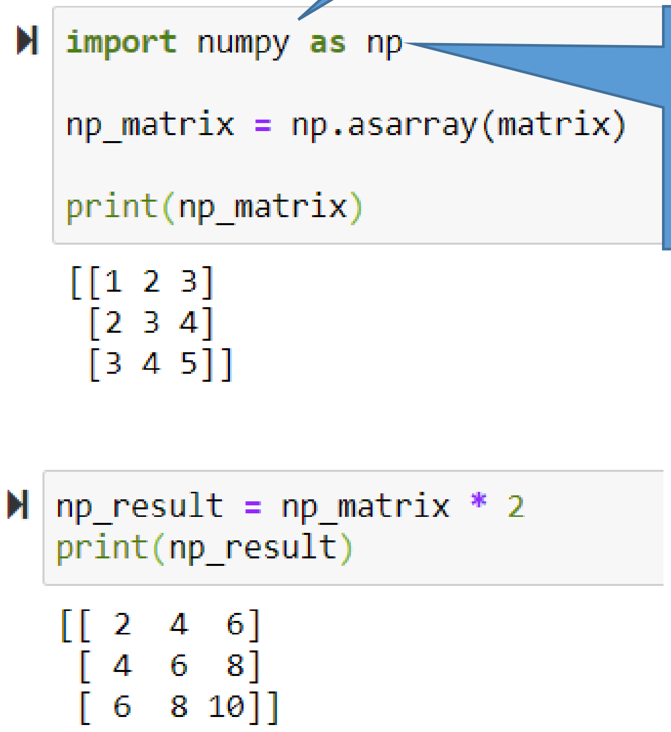
```
[[1, 2, 3], [2, 3, 4], [3, 4, 5]]
```

```
result = matrix * 2
print(result)
```

```
[[1, 2, 3], [2, 3, 4], [3, 4, 5], [1, 2, 3], [2, 3, 4], [3, 4, 5]]
```

- numpy arrays

Tell python that you want to use a library called numpy



```
import numpy as np

np_matrix = np.asarray(matrix)

print(np_matrix)
```

If “numpy” is to long, you can give an alias “np”

```
[[1 2 3]
 [2 3 4]
 [3 4 5]]
```

```
np_result = np_matrix * 2
print(np_result)
```

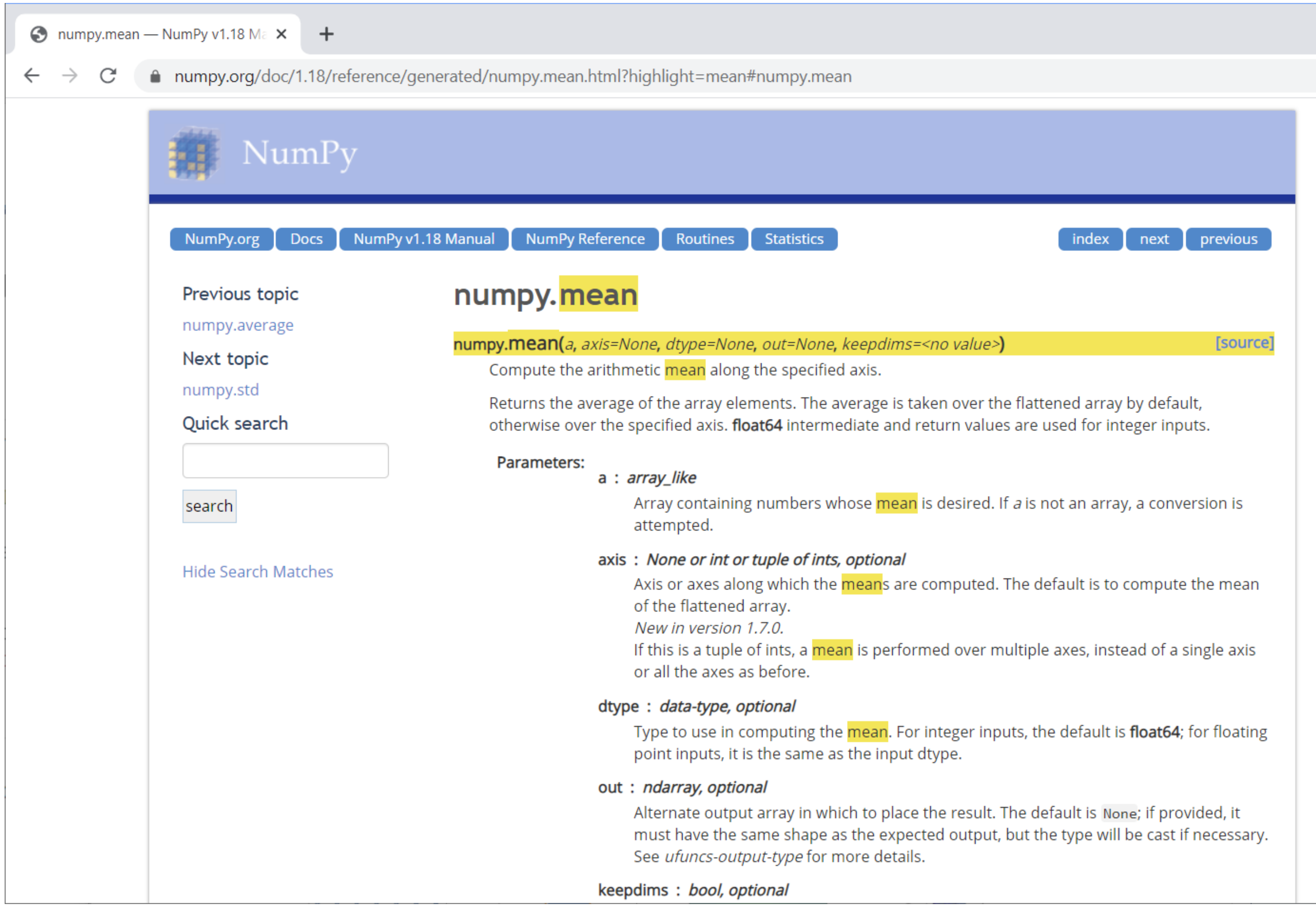
```
[[ 2  4  6]
 [ 4  6  8]
 [ 6  8 10]]
```

Basic descriptive statistics with numpy

- Basic descriptive statistics

```
import numpy as np

measurements = [1, 4, 6, 7, 2]

mean = np.mean(measurements)
print("Mean: " + str(mean))
```

```
Mean: 4.0
```



numpy.mean — NumPy v1.18 Ma
numpy.org/doc/1.18/reference/generated/numpy.mean.html?highlight=mean#numpy.mean
NumPy
NumPy.org | Docs | NumPy v1.18 Manual | NumPy Reference | Routines | Statistics
index | next | previous
Previous topic
numpy.average
Next topic
numpy.std
Quick search
search
Hide Search Matches

numpy.mean

numpy.mean(*a, axis=None, dtype=None, out=None, keepdims=<no value>*) [source]

Compute the arithmetic mean along the specified axis.

Returns the average of the array elements. The average is taken over the flattened array by default, otherwise over the specified axis. **float64** intermediate and return values are used for integer inputs.

Parameters:

a : *array_like*
Array containing numbers whose mean is desired. If *a* is not an array, a conversion is attempted.

axis : *None or int or tuple of ints, optional*
Axis or axes along which the means are computed. The default is to compute the mean of the flattened array.
New in version 1.7.0.
If this is a tuple of ints, a mean is performed over multiple axes, instead of a single axis or all the axes as before.

dtype : *data-type, optional*
Type to use in computing the mean. For integer inputs, the default is **float64**; for floating point inputs, it is the same as the input dtype.

out : *ndarray, optional*
Alternate output array in which to place the result. The default is `None`; if provided, it must have the same shape as the expected output, but the type will be cast if necessary. See *ufuncs-output-type* for more details.

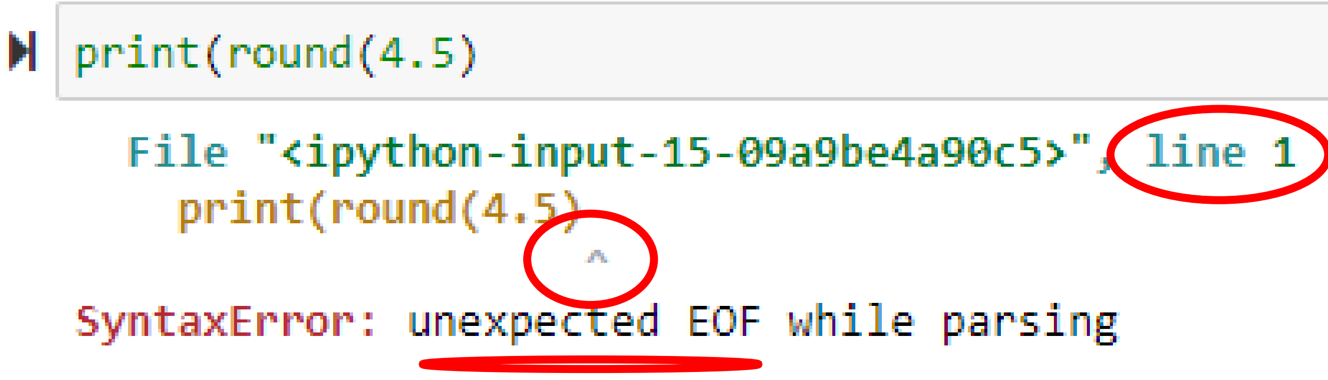
keepdims : *bool, optional*

Troubleshooting

If your program throws error messages:

- Don’t panic.
- *“There are two ways to write error-free programs; only the third one works.”*

  Alan J. Perlis, Yale University
- Read where the error happened.
  - You may see your fault immediately, when looking at the right point.
- Read what appears to be wrong.
  - If you know, what’s missing, you may see it, even if it’s missing in a slightly different place.
  - Sometimes, something related is missing



```
print(round(4.5)
```

```
  File "<ipython-input-15-09a9be4a90c5>", line 1
    print(round(4.5)
                    ^
SyntaxError: unexpected EOF while parsing
```

Test yourself: Type conversion

- Play with the python language. Could you predict what's the output? If not, try!

```
# initialise program
a = "1"
b = 2

# do some calculations
print(a + b)
print(b + a)
```

```
# initialise program
a = "1"
b = 2

# do some calculations
print(a / b)
print(b / a)
```

```
# initialise program
a = "1"
b = 2

# do some calculations
print(0 + a + b)
print("" + b + a)
```

```
# initialise program
a = "1";
b = 2;

# do some calculations
print(a / b)
print(b / a)
```

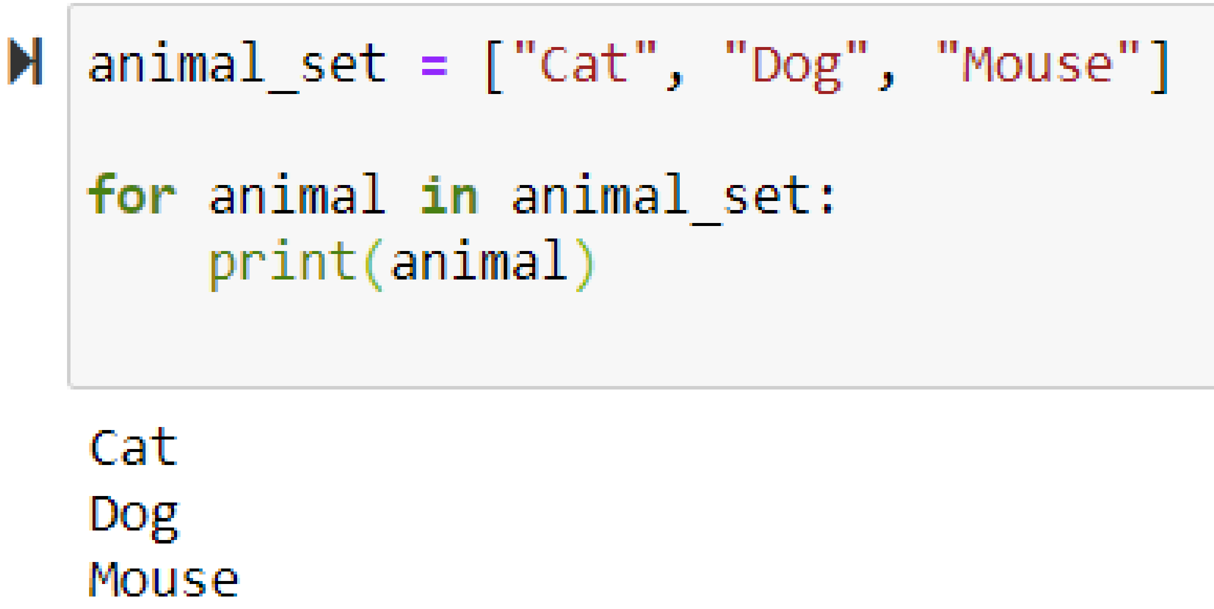
Summary

Today, you learned

- Jupyter notebooks
- Data structures
  - Variable
  - Arrays: lists / tuples
  - dictionaries
  - tables
- Operators and operations

Coming up next

- Loops
- Conditions
- Functions
- Libraries



```
animal_set = ["Cat", "Dog", "Mouse"]

for animal in animal_set:
    print(animal)
```

```
Cat
Dog
Mouse
```