An introduction to Python

Thibaut Lunet, Aitor Pérez

March 19, 2018

How to get Python (+ useful packages ...)

We are going to use the **miniconda** installer, which is cross-platform and provides package management, together with the **spyder** IDE.

1. Go to https://conda.io/miniconda.html
   (or Google search : "miniconda download")
2. Depending on the operating system, download installer (Python 2.7)
3. Install Python and required packages
   - Mac OS X or Unix:
     3.1 Open a terminal
     3.2 Run "bash Miniconda[...].sh", and yes for all ...
     3.3 Open a new terminal, or run "source ∼/.bashrc"
     3.4 Run "**conda install spyder numpy scipy matplotlib sympy**"
   - Windows:
     3.1 Double-click on the .exe file, and yes for all ...
     3.2 Open "conda prompt" terminal (installed with miniconda)
     3.3 Run "**conda install spyder numpy scipy matplotlib sympy**"

History

XXX slkdjlksj

Python vs. Others (Matlab, Fortran, C/C++, ...)

- License-free and open-source ($\neq$ Matlab)
- Huge users community, many (free) packages for many applications
- Extremely easy of use for non-I-love-programming people ($\neq$ Fortran, C/C++)
- Easy interface with other (more-efficient) programming languages $\Rightarrow$ computation can be accelerated using Fortran or C/C++ library ...
- Can scale to very large problems (parallel computing, ...)
- Structured and friendly ways for developing library ($\neq$ Matlab)

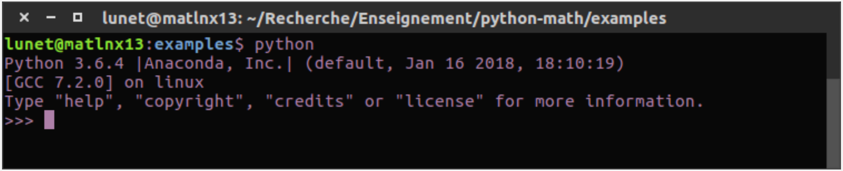
Python = many advantages, with very few drawbacks !

Functioning principles

XXX dsdskldj

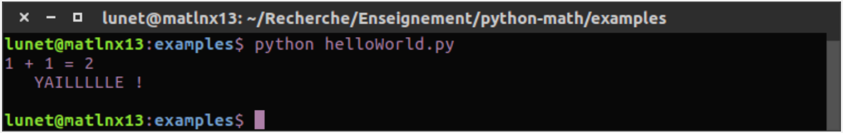
Practical tools

Using python console


lunet@matlnx13: ~/Recherche/Enseignement/python-math/examples

```
lunet@matlnx13:examples$ python
Python 3.6.4 |Anaconda, Inc.| (default, Jan 16 2018, 18:10:19)
[GCC 7.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Running a script


lunet@matlnx13: ~/Recherche/Enseignement/python-math/examples

```
lunet@matlnx13:examples$ python helloWorld.py
1 + 1 = 2
   YAILLLLLE !

lunet@matlnx13:examples$ 
```

All-in-One solution ⇒ **Spyder** !

Hello World!

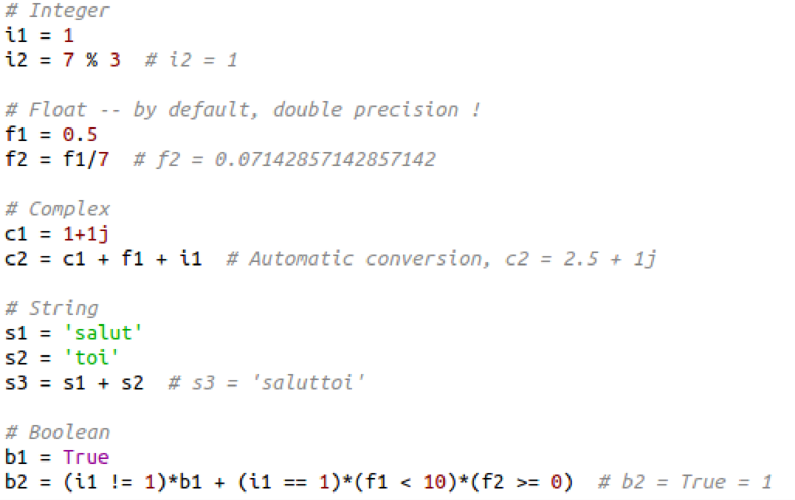
All the examples and python files are available at:
https://gitlab.unige.ch/Thibaut.Lunet/python-math

A first easy step ...

1. Launch Spyder
    - Windows : double click on an icon somewhere ...
    - Mac OS X or Unix : run "spyder" in terminal
2. Discover a wonderful environment #woaaah
3. Go to lower right corner → IPython console
    - write "1+1"
    - press enter ...
4. Go to text editor (middle)
    - write "print('hello world')
    - save and run the file ...

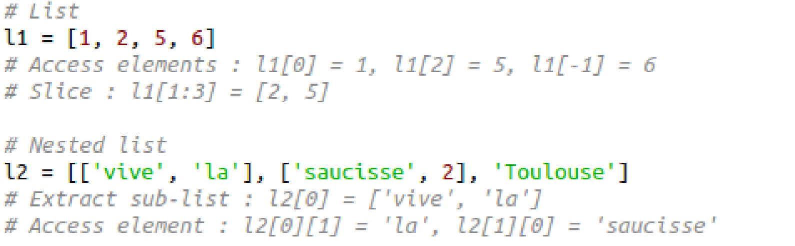
Basic variables types and operations

Slide codes at: python-math/presentation/code_examples.py



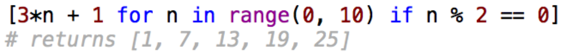
```
# Integer
i1 = 1
i2 = 7 % 3  # i2 = 1

# Float -- by default, double precision !
f1 = 0.5
f2 = f1/7  # f2 = 0.07142857142857142

# Complex
c1 = 1+1j
c2 = c1 + f1 + i1  # Automatic conversion, c2 = 2.5 + 1j

# String
s1 = 'salut'
s2 = 'toi'
s3 = s1 + s2  # s3 = 'saluttoi'

# Boolean
b1 = True
b2 = (i1 != 1)*b1 + (i1 == 1)*(f1 < 10)*(f2 >= 0)  # b2 = True = 1
```

Lists



```
# List
l1 = [1, 2, 5, 6]
# Access elements : l1[0] = 1, l1[2] = 5, l1[-1] = 6
# Slice : l1[1:3] = [2, 5]

# Nested list
l2 = [['vive', 'la'], ['saucisse', 2], 'Toulouse']
# Extract sub-list : l2[0] = ['vive', 'la']
# Access element : l2[0][1] = 'la', l2[1][0] = 'saucisse'
```

Python allows to use list comprehension:



```
[3*n + 1 for n in range(0, 10) if n % 2 == 0]
# returns [1, 7, 13, 19, 25]
```

Exercise 1

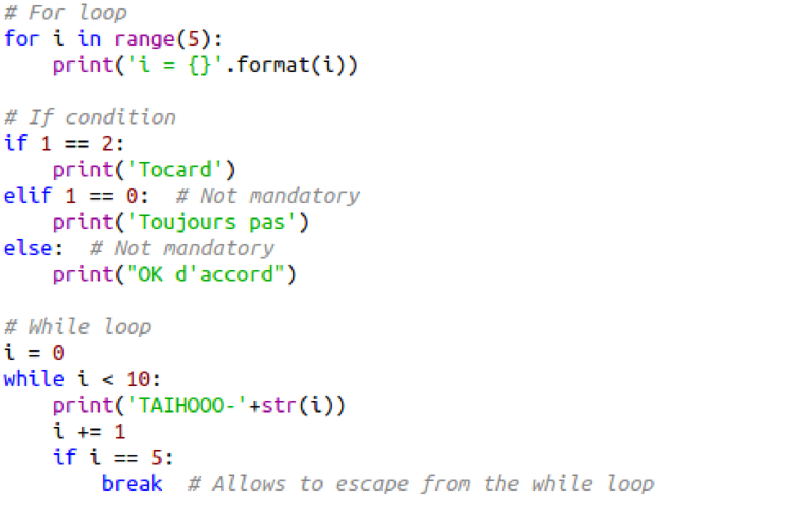
Rewrite the `filter_positive` function and reduce it to one line.

Dictionary

TODO

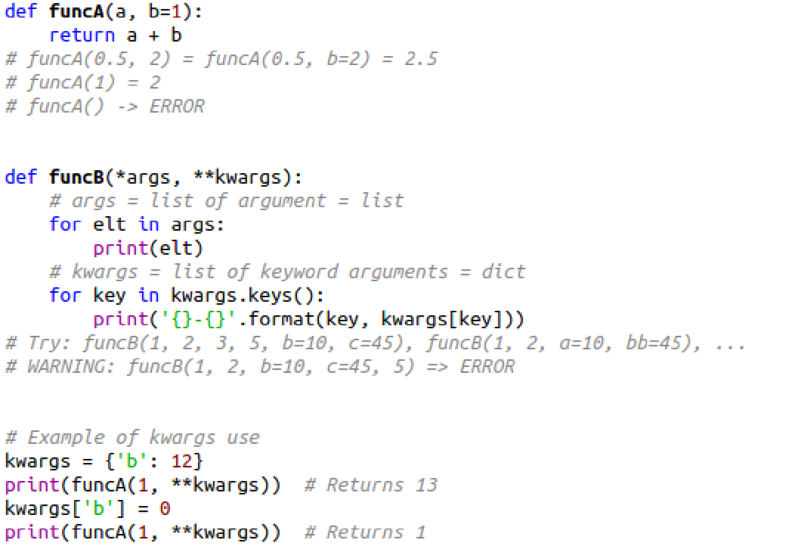
Basic structures

Tabs matter !



```
# For loop
for i in range(5):
    print('i = {}'.format(i))

# If condition
if 1 == 2:
    print('Tocard')
elif 1 == 0:  # Not mandatory
    print('Toujours pas')
else:  # Not mandatory
    print("OK d'accord")

# While loop
i = 0
while i < 10:
    print('TAIHOOO-'+str(i))
    i += 1
    if i == 5:
        break  # Allows to escape from the while loop
```

Function definition



```
def funcA(a, b=1):
    return a + b
# funcA(0.5, 2) = funcA(0.5, b=2) = 2.5
# funcA(1) = 2
# funcA() -> ERROR


def funcB(*args, **kwargs):
    # args = list of argument = list
    for elt in args:
        print(elt)
    # kwargs = list of keyword arguments = dict
    for key in kwargs.keys():
        print('{}-{}'.format(key, kwargs[key]))
# Try: funcB(1, 2, 3, 5, b=10, c=45), funcB(1, 2, a=10, bb=45), ...
# WARNING: funcB(1, 2, b=10, c=45, 5) => ERROR


# Example of kwargs use
kwargs = {'b': 12}
print(funcA(1, **kwargs))  # Returns 13
kwargs['b'] = 0
print(funcA(1, **kwargs))  # Returns 1
```

File I/O

XXX sdjhj

Using Numpy to manipulate arrays

XXX TODO

Overview of Scipy functionalities

XXX TODO

Data visualization with Matplotlib

XXX TODO

Symbolic computation with Sympy

XXX TODO