



 python™  
The Language

# A bit of history

- ▶ Python was conceived in 1980 and started being implemented in 1989 by Guido van Rossum at CWI (Netherlands), entitled BDFL (Benevolent Dictator For Life) by the community
- ▶ Focus on code readability, providing a syntax that allows more concepts in fewer lines of code
- ▶ Fully open-source with a extremely active and wide community



# A bit of history – Versions

The first main community backed version:

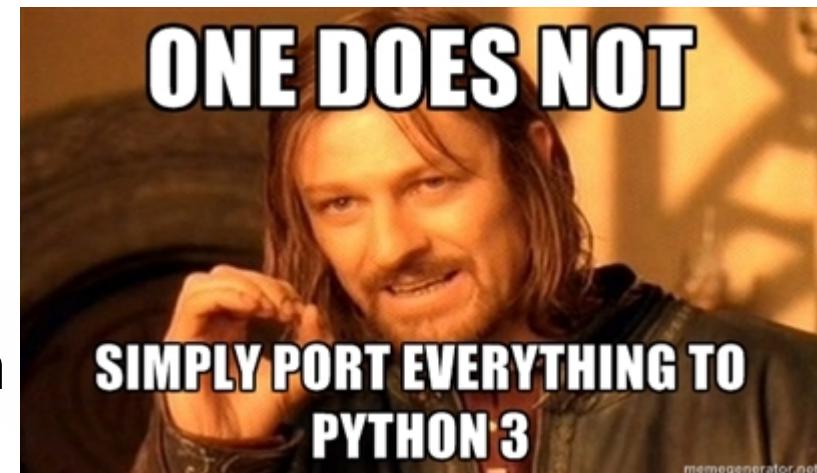
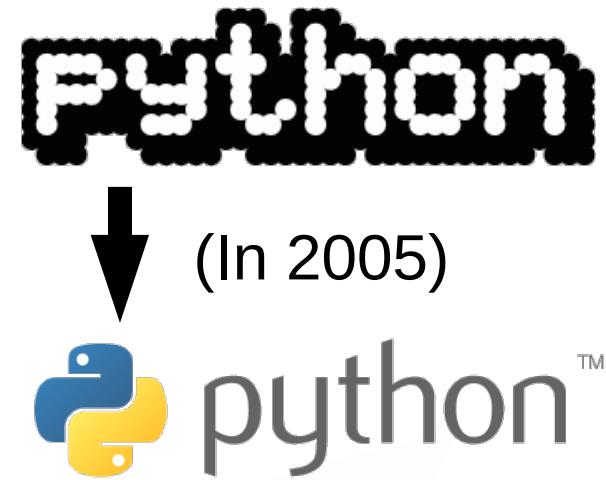
- ▶ Python 2.0, October 16, 2000
  - ▶ Cycle-detecting garbage collector for memory management
  - ▶ Unicode support



# A bit of history – Versions

The first main community backed version:

- ▶ Python 2.0, October 16, 2000
  - ▶ Cycle-detecting garbage collector for memory management
  - ▶ Unicode support
- ▶ Python 3.0, December 3, 2008
  - ▶ Backwards-**incompatible**
  - ▶ Major features backported to python 2.6 & 2.7

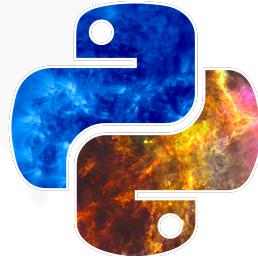


# A bit of history – Use of python

Python is currently being used for “everything”



Large-scale servers  
working 24/7



Astronomy

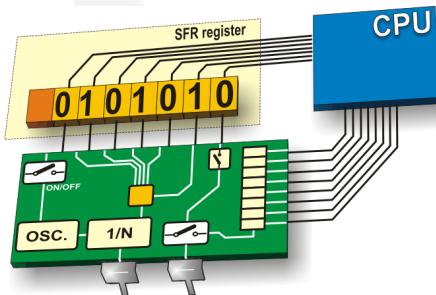


Python 1

Teaching Kids



Throw-away  
scripts



Microcontrollers



Databases



# A bit of history – Use of python

(Some) Companies currently massively using python

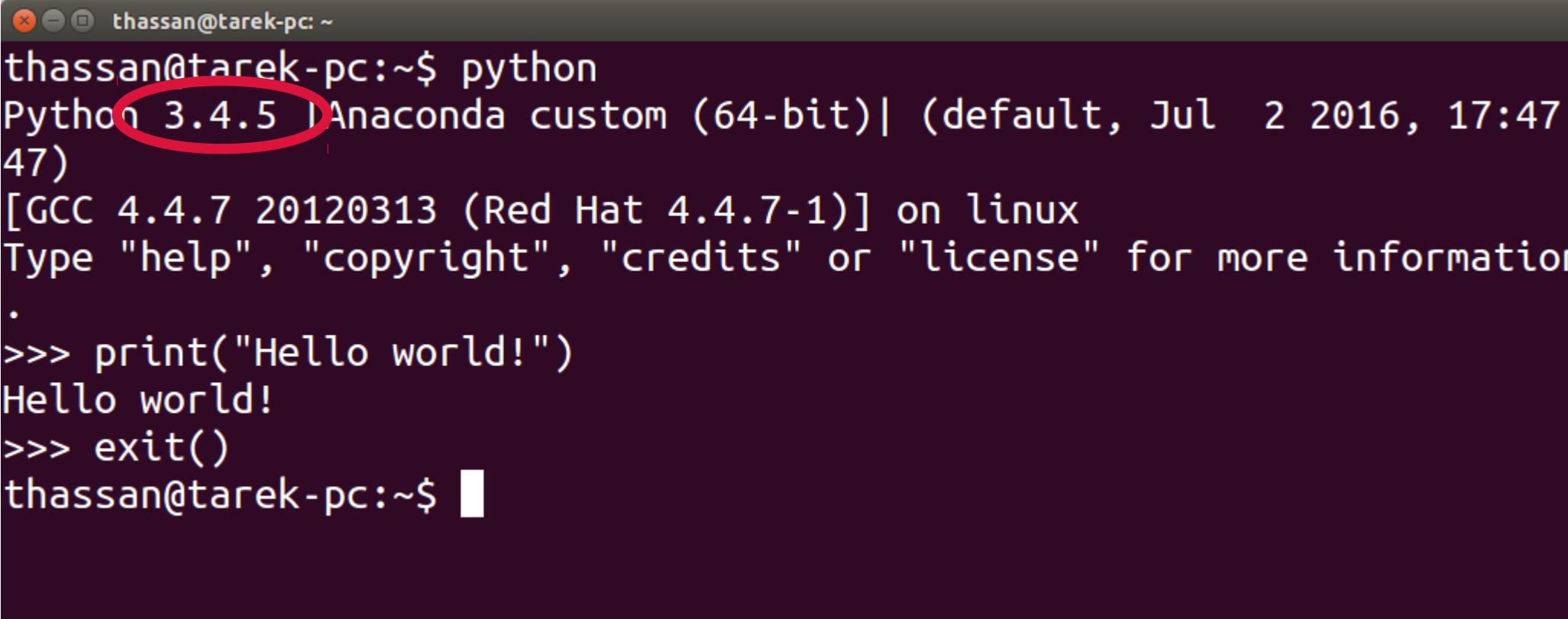


# A bit of history – Python 3

- ▶ Python 3 was designed to correct some fundamental design flaws  
(see [https://en.wikipedia.org/wiki/History\\_of\\_Python#Features](https://en.wikipedia.org/wiki/History_of_Python#Features))
  - ▶ Not possible to retain full backwards compatibility
- ▶ What does this mean?
  - ▶ If you start now with python 3, you will learn the “good way”
  - ▶ Some packages may still be not fully functional in python 3  
(give them some time!)
  - ▶ You can check the version of python you are using simply by:  
**import sys; print(sys.version)**

# Getting started with python

- ▼ Open a terminal and type “python”



```
thassan@tarek-pc:~$ python
Python 3.4.5 |Anaconda custom (64-bit)| (default, Jul  2 2016, 17:47:47)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "help", "copyright", "credits" or "license" for more information
.
>>> print("Hello world!")
Hello world!
>>> exit()
thassan@tarek-pc:~$
```

# Getting started with python

- ▼ To use an external library, you need to “import” it:

```
thassan@tarek-pc:~$ python
Python 3.4.5 |Anaconda custom (64-bit)| (default, Jul  2 2016, 17:47:47)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import math
>>> print (math.pi)
3.141592653589793
>>> █
```

- ▼ Python standard library: <https://docs.python.org/3/library/index.html>

# Getting started with python

- ▼ You have more interactive options, like “ipython”:

```
IPython: home/thassan
thassan@tarek-pc:~$ ipython
Python 3.4.5 |Anaconda custom (64-bit)| (default, Jul  2 2016, 17:47:47)
Type "copyright", "credits" or "license" for more information.

IPython 5.1.0 -- An enhanced Interactive Python.
?          -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help       -> Python's own help system.
object?    -> Details about 'object', use 'object??' for extra details.

In [1]: import math

In [2]: math.p[  
        math.pi  
        math.pow
```

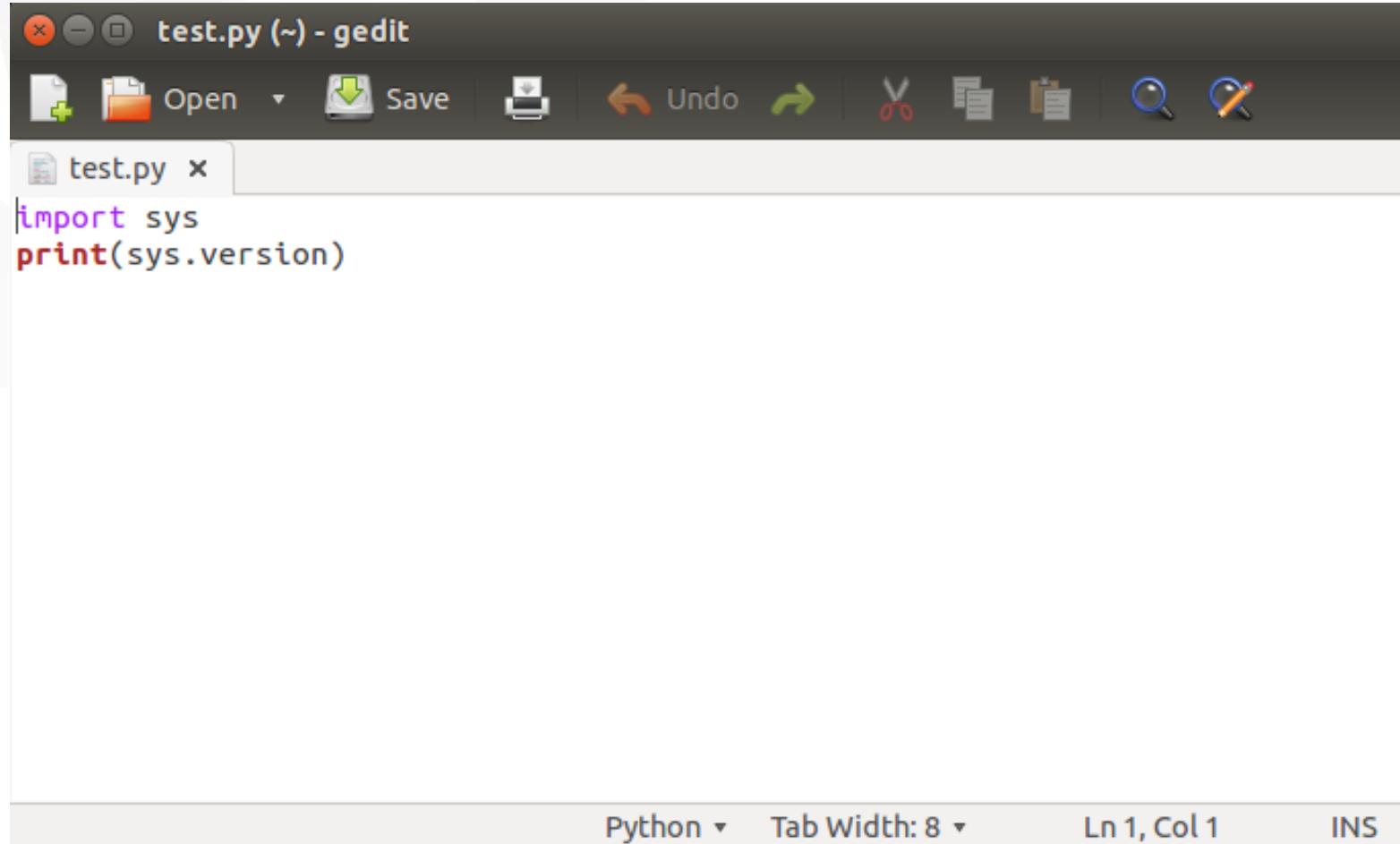
# Getting started with python

- ▼ For writing scripts, you can use any plain text editor:

```
thassan@tarek-pc:~$ vim test.py
thassan@tarek-pc:~$ cat test.py
import sys
print(sys.version)
thassan@tarek-pc:~$ python test.py
3.4.5 |Anaconda custom (64-bit)| (default, Jul  2 2016, 17:47:47)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)]
thassan@tarek-pc:~$ █
```

# Getting started with python

- For writing scripts, you can use any “plain” text editor:



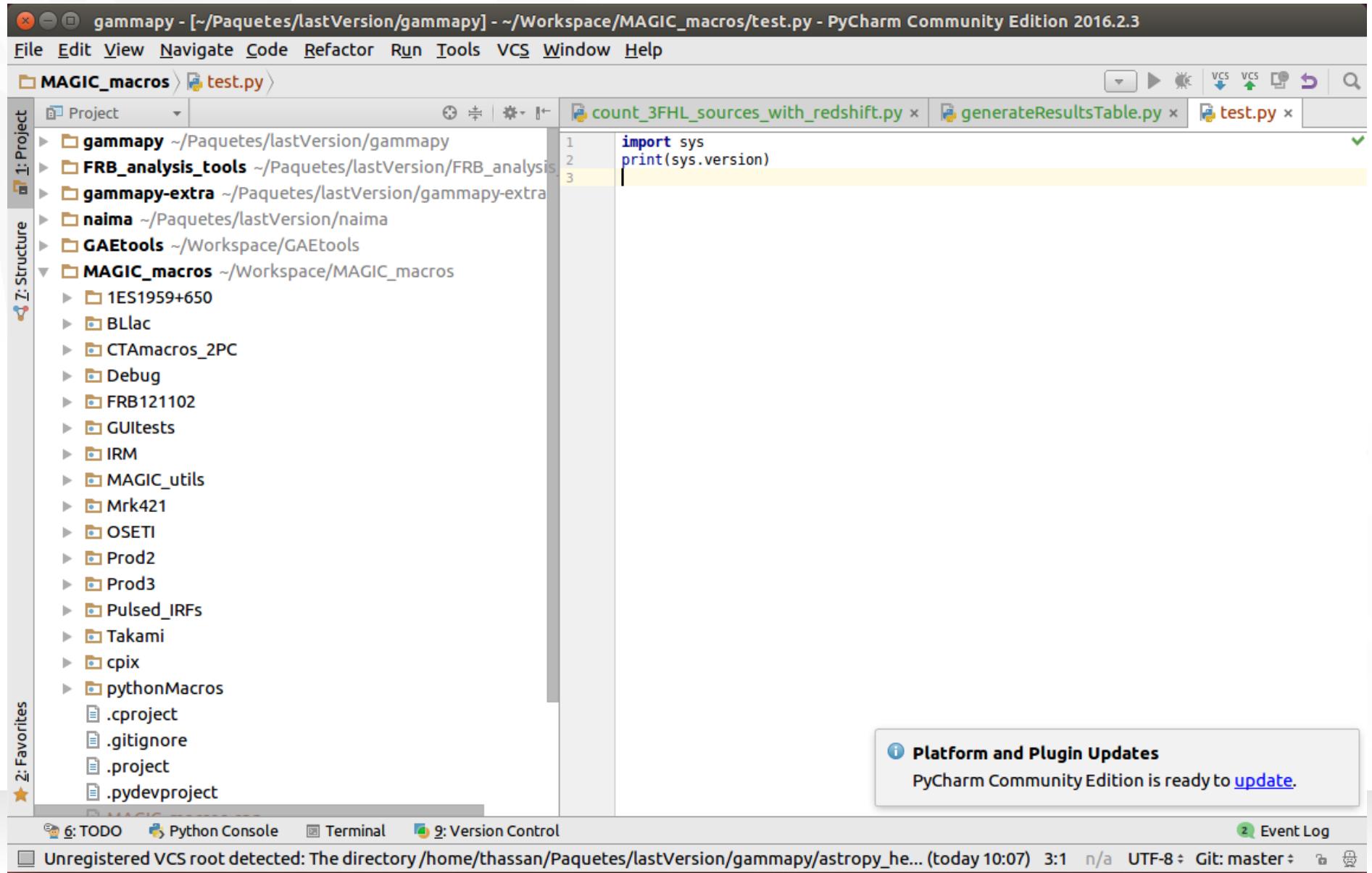
The screenshot shows the gedit text editor interface. The title bar reads "test.py (~) - gedit". The toolbar includes standard icons for file operations (Open, Save, Undo, Redo, Cut, Copy, Paste, Find, Replace). The main window displays the code for a Python script:

```
import sys
print(sys.version)
```

The status bar at the bottom shows "Python" and "Tab Width: 8". The cursor position is "Ln 1, Col 1" and the insertion mode is "INS".

# Getting started with python

- Or use an IDE: (e. g. pyCharm, eclipse, vim, wing, spyder...)



# How python works: binaries

- As it is possible (and usual) to have several “pythons” installed, some clarifications are useful
  - When you enter the python console, you execute a python binary (you may have several installed!)

```
thassan@tarek-pc:~$ which python
/home/thassan/Paquetes/anaconda3/bin/python
thassan@tarek-pc:~$ python
Python 3.4.5 |Anaconda custom (64-bit)| (default, Jul  2 2016, 17:47:47)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()
thassan@tarek-pc:~$ /usr/bin/python
Python 2.7.6 (default, Oct 26 2016, 20:30:19)
[GCC 4.8.4] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

# How python works: binaries & libraries

- ▼ Some libraries may be installed only in some python environments:

```
thassan@tarek-pc:~$ /home/thassan/Paquetes/anaconda3/envs/cta/bin/python
Python 3.5.1 |Continuum Analytics, Inc.| (default, Jun 15 2016, 15:32:45)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import gammipy
>>> 
```

```
thassan@tarek-pc:~$ /usr/bin/python
Python 2.7.6 (default, Oct 26 2016, 20:30:19)
[GCC 4.8.4] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import gammipy
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named gammipy
>>> 
```

# How python works: binaries & libraries

- ▶ To install a library:
  - ▶ You may do it manually (usually not encouraged):
    - ▶ Download package, “compile” and install

# How python works: binaries & libraries

- ▶ To install a library:
  - ▶ You may do it manually (usually not encouraged):
    - ▶ Download package, “compile” and install
  - ▶ You may use a package manager such as “pip”:
    - ▶ “pip install <package\_name>”
    - ▶ This will install all dependencies, in the version required by the package
    - ▶ Unless if you are a developer of an specific package, you will probably always use this method (or analog ones)

# Using Jupyter notebooks

- ▶ For simplicity, to follow the tutorials of this course we will be using Jupyter notebooks
  - ▶ Open-source web application that allows to edit and execute code
  - ▶ All code is executed locally within your python environment (even if it is shown within your browser)
- ▶ To begin the course, clone the gitHub repository of the course and execute the jupyter notebook “02\_01\_TheLanguage.ipynb”

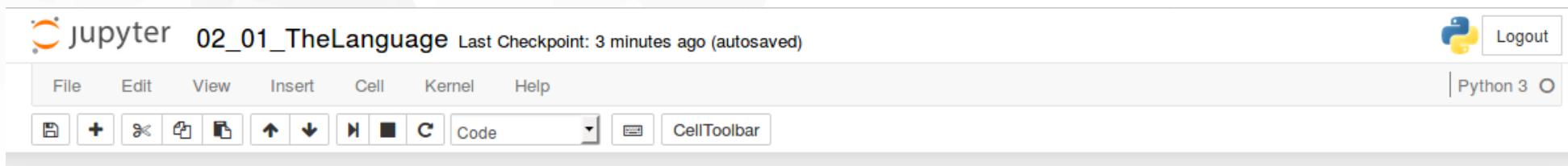


# Using Jupyter notebooks

```
thassan@tarek-pc: /tmp/PythonWorkshop-ICE/notebooks
thassan@tarek-pc:~$ cd /tmp/
thassan@tarek-pc:/tmp$ git clone https://github.com/Python4AstronomersAndParticlePhysicists/PythonWorkshop-ICE.git
Cloning into 'PythonWorkshop-ICE'...
remote: Counting objects: 218, done.
remote: Compressing objects: 100% (33/33), done.
remote: Total 218 (delta 22), reused 32 (delta 12), pack-reused 173
Receiving objects: 100% (218/218), 14.41 MiB | 1.94 MiB/s, done.
Resolving deltas: 100% (112/112), done.
Checking connectivity... done.
thassan@tarek-pc:/tmp$ cd PythonWorkshop-ICE/notebooks/
thassan@tarek-pc:/tmp/PythonWorkshop-ICE/notebooks$ ls
02_01_TheLanguage.ipynb          06_01_pandas.ipynb
03_01_GoodProgramming.ipynb      07_01_scipy.ipynb
03_02_TipsAndTricks.ipynb        08_01_MachineLearning.ipynb
04_01_numpy.ipynb                10_01_Astronomy_I.ipynb
05_01_matplotlib.ipynb          12_01_Databases.ipynb
05_02_matplotlib_animations.ipynb README.md
thassan@tarek-pc:/tmp/PythonWorkshop-ICE/notebooks$ jupyter notebook 02_01_TheLanguage.ipynb
```

# Using Jupyter notebooks

- From now on, the tutorial (and whole workshop) continues through notebooks (in your browser, similar to this):



## The language (1h)

- 1) Short historical overview (5m)
- 2) Briefly talk about python 2/3 (5m)
- 3) Getting started with python (15m)
  - Python terminal
  - iPython
  - Script with plain editors
  - Scripts with pycharm
- 4) How python works (20m)
  - binaries, libraries and environments
  - exercises/examples
    - example: import library included
    - example: import library not installed
    - exercise: install library manually
    - example: install library pip
- 5) Explain/show what jupyter notebooks are (15m)
  - Code is running locally
  - Typical shortcuts
  - Is possible to add comments, etc...