Tuesday’s Lecture 1

Physics Simulations with Python: prerequisites, tools and basic concepts

Laboratorio di Fisica Computazionale
Computational Physics Laboratory

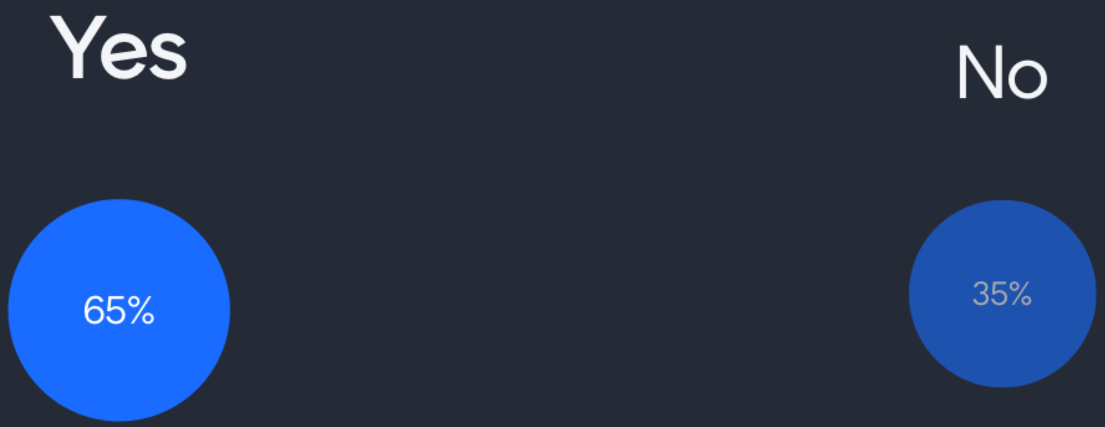
Antimo Marrazzo (Physics Department, UniTS)
AA 2022/23 II semester

Go to **www.menti.com** and use the code **1841 3112**

Lost contact with audience

Mentimeter

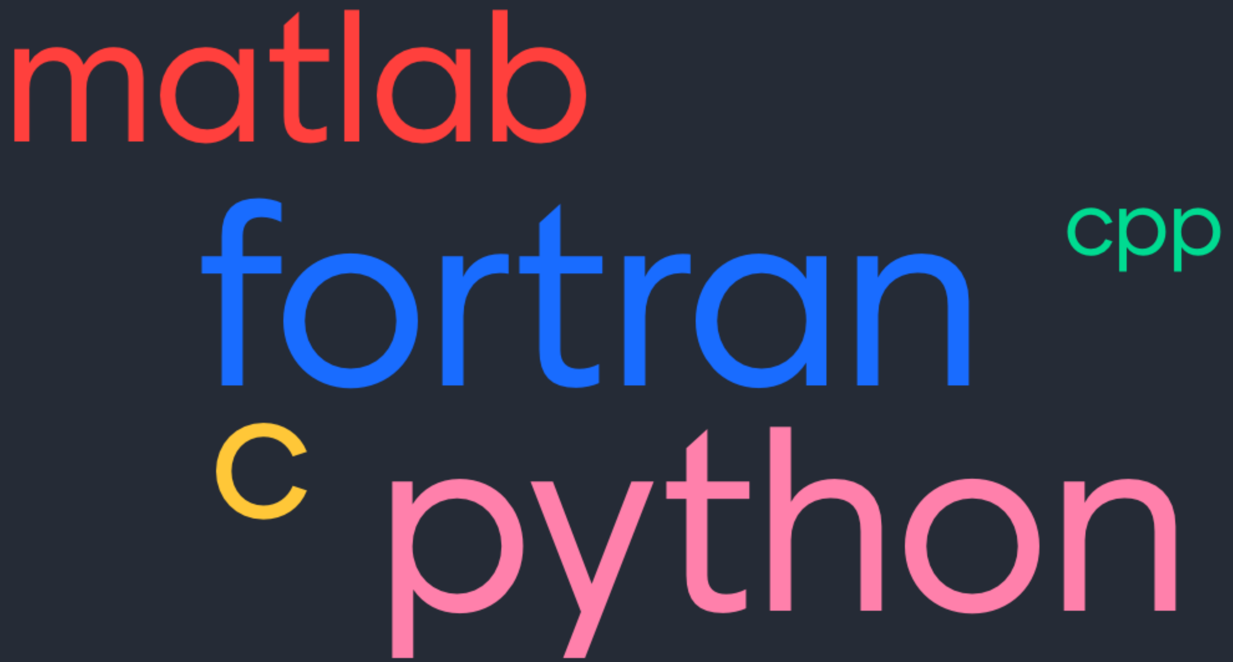
Have you ever coded in Python?



20

Go to **www.menti.com** and use the code **1841 3112**

What is your favorite programming language?



19

What is Python?

- «Python is an easy to learn, powerful programming language» (source: official Python tutorial)
- Few catches:
  - Easy to *start* coding, difficult to loose accents from other languages
  - You only miss what you know about: several powerful featues potentially unexploited
  - Often harder (or not obvious) to produce efficient code for numerical simulations, especially at the HPC level. (Fortran is a Formula TRANslator, it was designed for number crunching; Python is more general purpose, from web design to data analysis)

What is Python? (really)

- Python is an **interpreted**, interactive, **object-oriented** programming language. It incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes.
- **It supports multiple programming paradigms beyond object-oriented programming, such as procedural and functional programming**.
- Python combines remarkable power with **very clear syntax**.
- It has **interfaces** to many system calls and **libraries**, as well as to various window systems, and is extensible in C or C++. It is also usable as an extension language for applications that need a programmable interface. Finally, Python is portable: it runs on many Unix variants including Linux and macOS, and on Windows.

[Source: docs.python.org]

Why Python?
(in a Computational Physics Laboratory)

- It dramatically reduces **the time to develop codes** (especially true if the programmer time is worth more than CPU time)
- **“Python as a glue”:** ease of integrating C, C++ and Fortran code
- Great for prototyping code
- Great for data analysis, machine learning & visualizing data
- It can be made efficient with extensions and libraries
- It has becoming extremely popular also in computational science (existing projects and available libraries)
- NB: *Pythonic* strategies, tools and style are radically different w.r.t compiled codes…*especially if you were originally trained with C or Fortran!*

A disclaimer

- This course is about **computational physics, *not* a coding class**

  ->For a dedicated introduction to Bash and the Python programming language **have a look at the I semester course 682SM *Abilità informatiche e telematiche*,** slides and other material available on the 682SM Teams channel (access code at **https://www.units.it/en/node/10905**)

- We will not teach you how to code in Python from scratch

  ->check out 682SM for that

  ->we will revise key concepts through short summaries and code examples

- We will not require you to know how to code in Python
- The proven capability to develop a code for numerical simulations in modern Fortran AND Python (i.e. using both!) will be evaluated very positively
- We will show you that implementing physics simulations sometimes requires different strategies in Python than in C or Fortran 90.

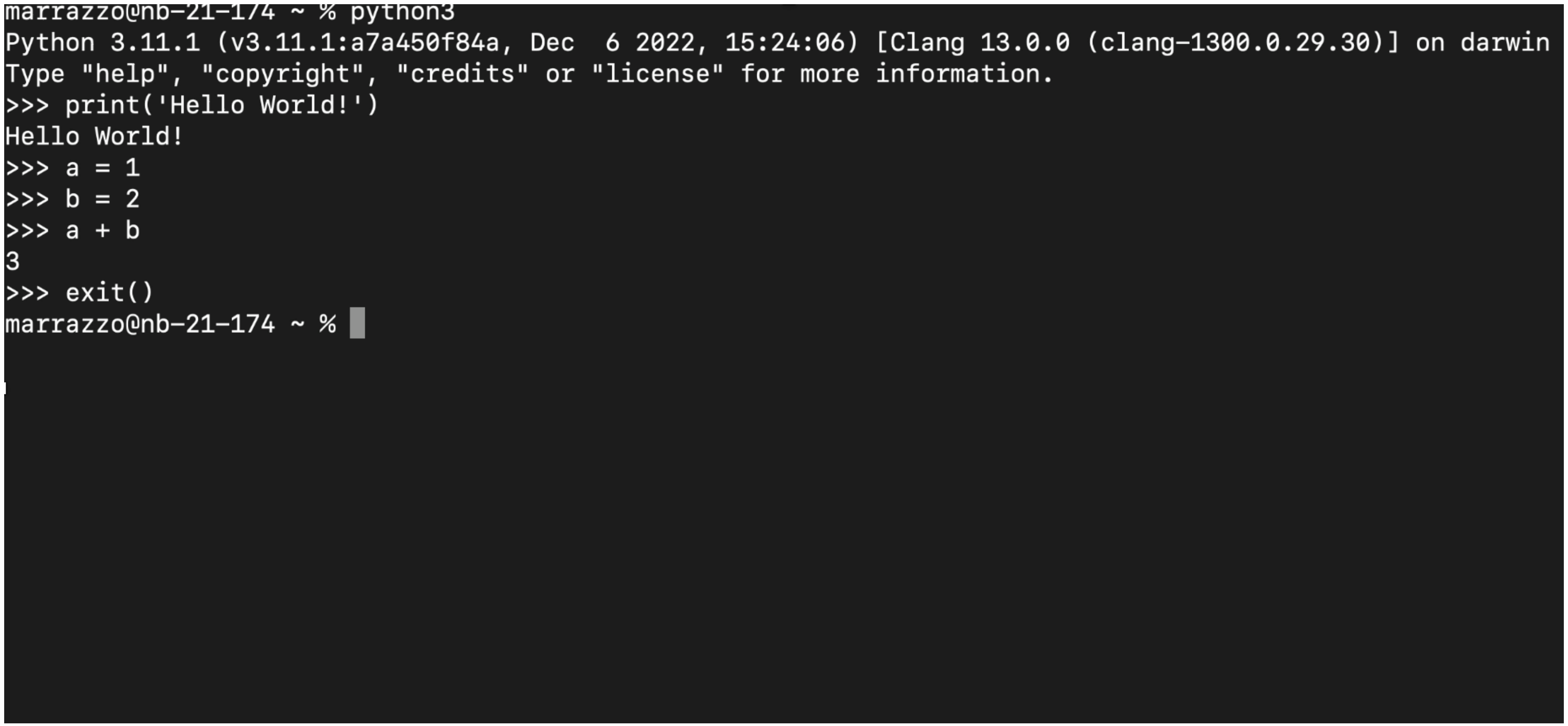
Homework #1

- Make sure you are familiar with these topics
  - Basic Python syntax.
  - Basic built-in datastructures (lists, tuples and dictionaries).
  - Control structures (if-else, while, for).
  - How to write and use functions and modules.
  - File I/O.

  at the level of the **course 682SM *Abilità informatiche e telematiche*.**
- If you come from modern Fortran, check out this Python-Fortran Rosetta Stone https://www.fortran90.org/src/rosetta.html (*Python with NumPy and Fortran are actually rather similar in terms of expressiveness and features)*

The Python interpreter

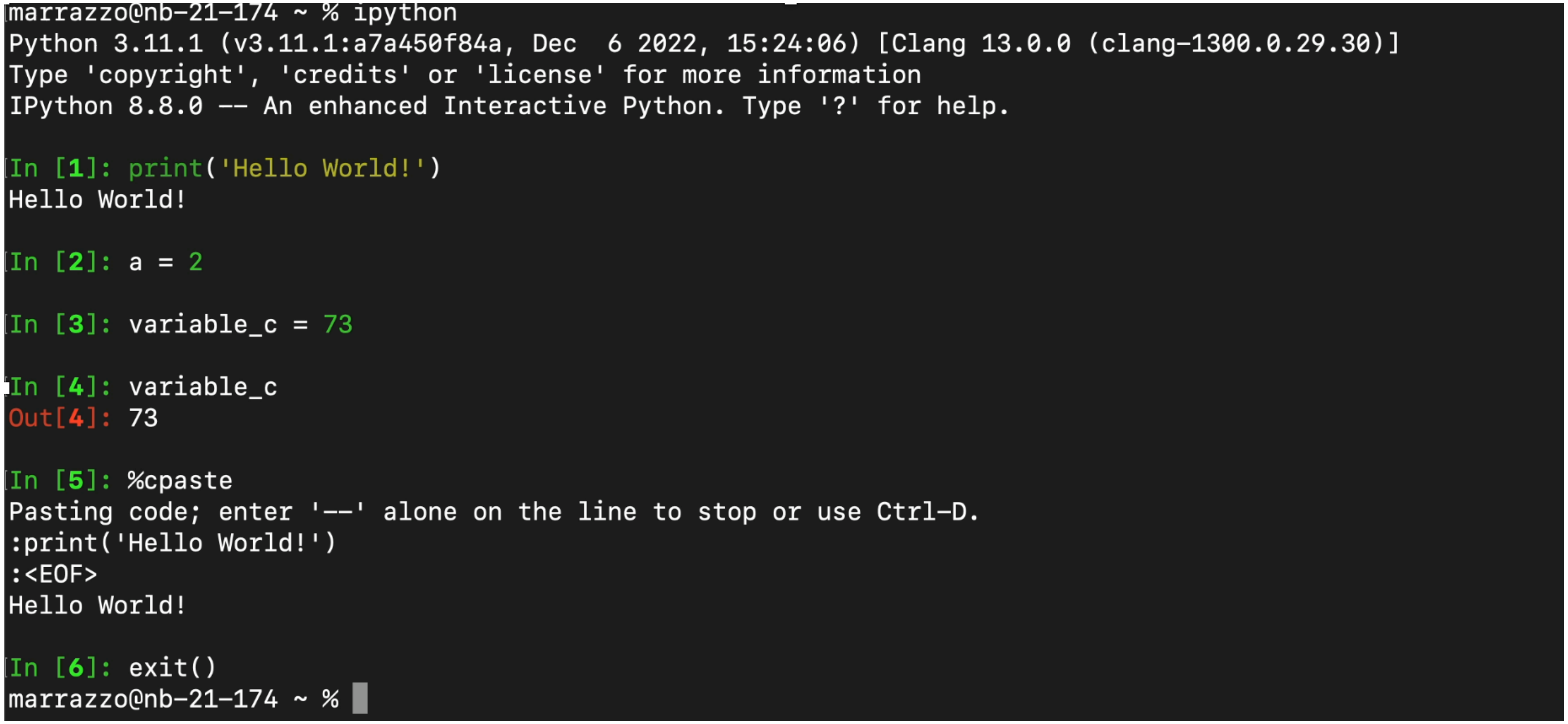
- Python is an interpreted language: the interpreter runs programs by executing one statement at a time.



```
marrazzo@nb-21-174 ~ % python3
Python 3.11.1 (v3.11.1:a7a450f84a, Dec  6 2022, 15:24:06) [Clang 13.0.0 (clang-1300.0.29.30)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Hello World!')
Hello World!
>>> a = 1
>>> b = 2
>>> a + b
3
>>> exit()
marrazzo@nb-21-174 ~ %
```

IPython

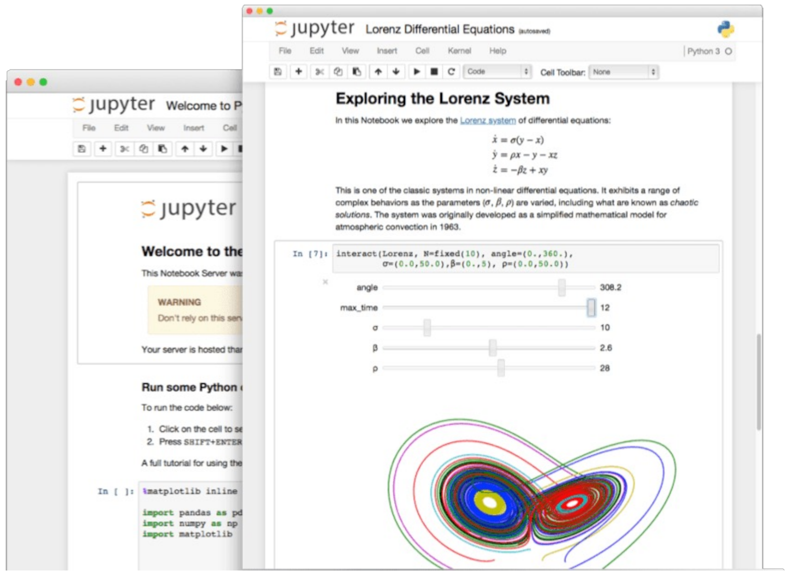
- **IPython is an enhanced Python interpreter** with tab completion, history and other advanced features, including the support for interactive data visualizations and tools for parallel computing.



```
marrazzo@nb-21-174 ~ % ipython
Python 3.11.1 (v3.11.1:a7a450f84a, Dec  6 2022, 15:24:06) [Clang 13.0.0 (clang-1300.0.29.30)]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.8.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: print('Hello World!')
Hello World!

In [2]: a = 2

In [3]: variable_c = 73

In [4]: variable_c
Out[4]: 73

In [5]: %cpaste
Pasting code; enter '--' alone on the line to stop or use Ctrl-D.
:print('Hello World!')
:<EOF>
Hello World!

In [6]: exit()
marrazzo@nb-21-174 ~ %
```

Jupyter Notebooks and Jupyter Lab

- **Jupyter Notebooks**
  - Spin-off of IPython
  - Web-based application for creating & sharing computational documents
  - “Web-based” notebooks allow to mix code, text (e.g. Markdown, HTML) and interactive visualization
  - They can be used with any programming language (but particularly useful in Python)
- **Jupyter Lab**
  - Web-based interactive development environment for notebooks, code, and data


jupyter Lorenz Differential Equations
File Edit View Insert Cell Kernel Help
Python 3
Cell Toolbar: None
jupyter Welcome to P
angle
max_time
σ
β
ρ

Exploring the Lorenz System

In this Notebook we explore the Lorenz system of differential equations:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = \rho x - y - xz$$
$$\dot{z} = -\beta z + xy$$

This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters (σ, β, ρ) are varied, including what are known as chaotic solutions. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.

```
In [7]: interact(Lorenz, N=fixed(10), angle=(0.,360.),
                 σ=(0.0,50.0),β=(0.,5), ρ=(0.0,50.0))
```

```
angle     308.2
max_time  12
σ         10
β         2.6
ρ         28
```

Welcome to the

This Notebook Server wa

WARNING

Don't rely on this ser

Your server is hosted tha

Run some Python

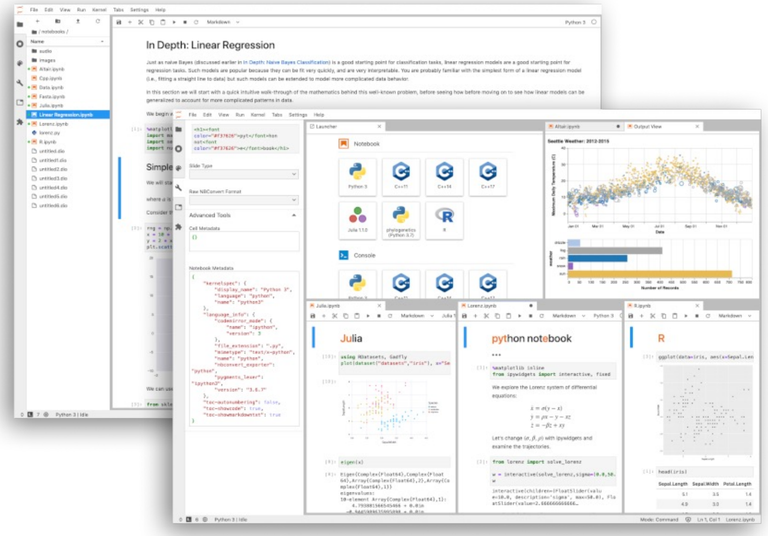
To run the code below:

1. Click on the cell to se
2. Press SHIFT+ENTER

A full tutorial for using the

```
In [ ]: %matplotlib inline

        import pandas as pd
        import numpy as np
        import matplotlib
```

Source: jupyter.org

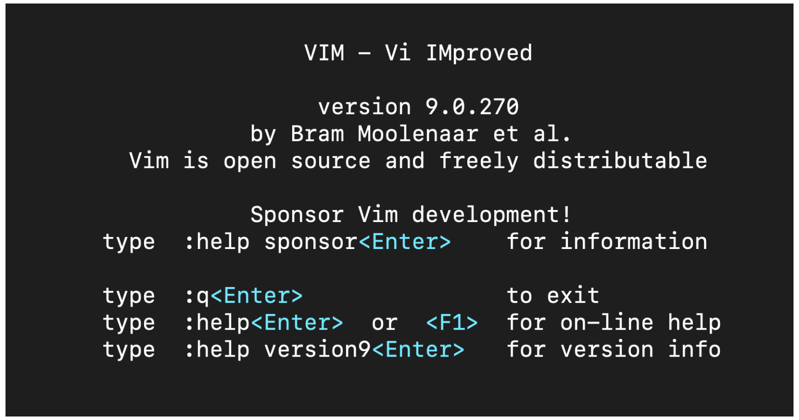


Integrated Development Environments (a.k.a. IDEs)

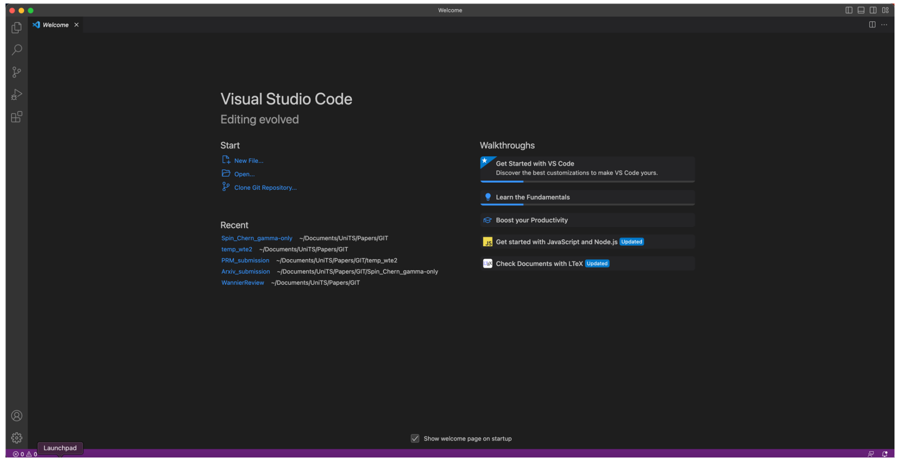
- IDEs are pieces of software which aid computer programmers to write codes
- IDEs are designed to maximise productivity (i.e. save time)
- There exists a large number of them, with a wide range of functionalities:
    - Basic features (code editor): vim, emacs, nano, …

      -> very useful to use on remote machines (e.g. HPC clusters)
    - More advanced (also GUI, compilers/interpreters, support for version control, …): Eclipse, Xcode, Visual Studio Code, …

      ->very powerful to develop code (and write in LaTeX as well!)



```
                  VIM - Vi IMproved

                   version 9.0.270
               by Bram Moolenaar et al.
      Vim is open source and freely distributable

               Sponsor Vim development!
type  :help sponsor<Enter>    for information

type  :q<Enter>               to exit
type  :help<Enter>  or  <F1>  for on-line help
type  :help version9<Enter>   for version info
```



Welcome

Visual Studio Code

Editing evolved

Start

New File...
Open...
Clone Git Repository...

Recent

```
Spin_Chern_gamma-only  ~/Documents/UniTS/Papers/GIT
temp_wte2  ~/Documents/UniTS/Papers/GIT
PRM_submission  ~/Documents/UniTS/Papers/GIT/temp_wte2
Arxiv_submission  ~/Documents/UniTS/Papers/GIT/Spin_Chern_gamma-only
WannierReview  ~/Documents/UniTS/Papers/GIT
```

Walkthroughs

Get Started with VS Code
Discover the best customizations to make VS Code yours.
Learn the Fundamentals
Boost your Productivity
Get started with JavaScript and Node.js Updated
Check Documents with LTeX Updated

Show welcome page on startup

Virtual environments

- It is good practices to develop projects in isolated **virtual environments** on top of an existing Python installation, essentially folders which contains all the necessary executables to use the packages that a Python project would need, including their own independent set of Python packages.
- Very easy through the package **virtualenv**
  - pip install virtualenv
  - **-> to create the environment:** virtualenv yourpythonenv
  - **-> to enter the environment**: source yourpythonenv/bin/activate (on Windows: yourpythonenv\Scripts\activate)
  - **-> to exit**: deactivate

…let’s use Jupyter Notebooks!
(now we switch to the .ipynb)

Some references

- An extensive list at https://wiki.python.org/moin/PythonBooks
- For beginners with a Physics background (very recommended!)

  Effective Computation in Physics: Field Guide to Research with Python
  (A. Scopatz & K. D. Huff, O'Reilly, 2015)
- For advanced Python users:

  Fluent Python: Clear, Concise, Effective Programming (L. Ramalho, O'Reilly, 2015)

But … nobody learns coding on books!

1) Learn by doing: practice, practice and practice
2) Python official documentation: https://docs.python.org/3/
3) Stackoverflow: https://stackoverflow.com

Installing a Python development environement (useful resources)

The Ubuntu VMs in *Aula Poropat* are already configured for the course, you can use those (even from remote!)

At home

- Option 0 (Linux): for Ubuntu
  - sudo apt-get install python3 ipython3 python3-pip python3-numpy python3-numexpr python3-matplotlib cython3 python3-cffi python3-scipy
  - pip3 install jupyter numba virtualenv
- Option 1 (Windows, Linux, MacOS): https://www.python.org/downloads/
- Option 2 (Windows, Linux, MacOS): Anaconda installer (most packages you need for this course are pre-installed)
- If you use Windows, consider also to install an Ubuntu VM with VirtualBox