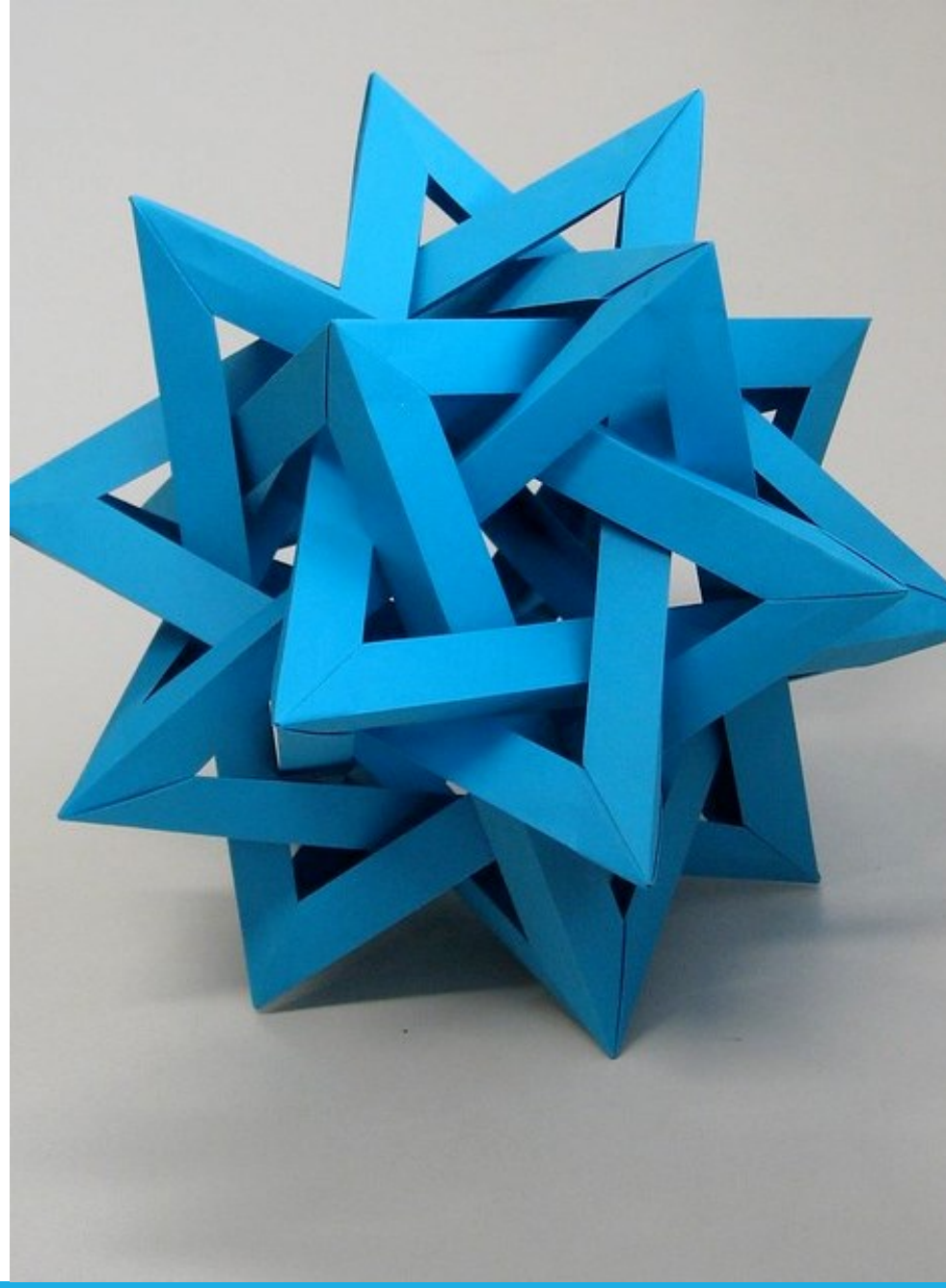
Politecnico di Torino

Dipartimento
di Automatica e Informatica

Ambienti di calcolo e simulazione per la ricerca sperimentale

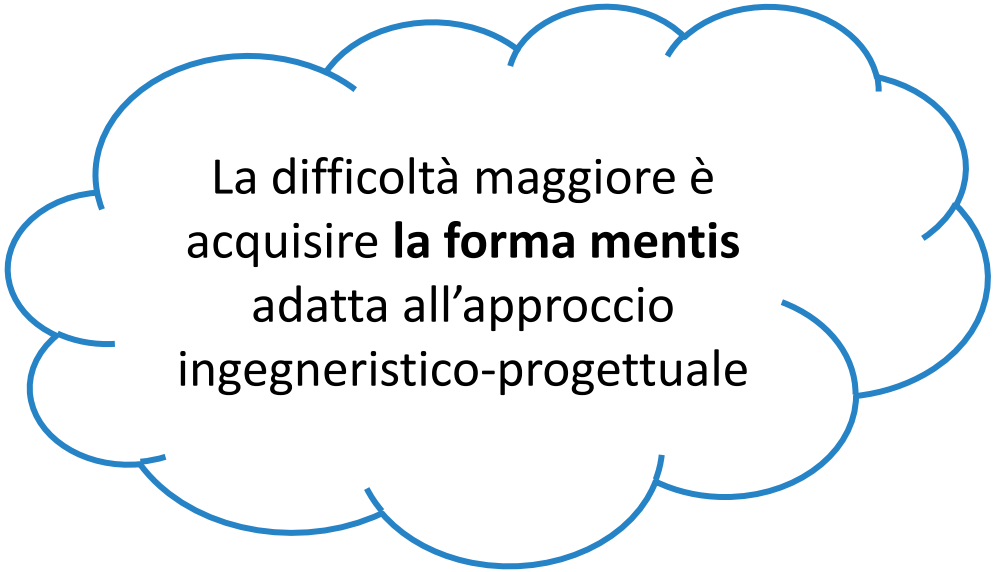
DOTTORATO DI RICERCA IN TECNOLOGIE E METODI PER LA FORMAZIONE UNIVERSITARIA

PROF. FULVIO CORNO



Un assaggio di… Ingegneria

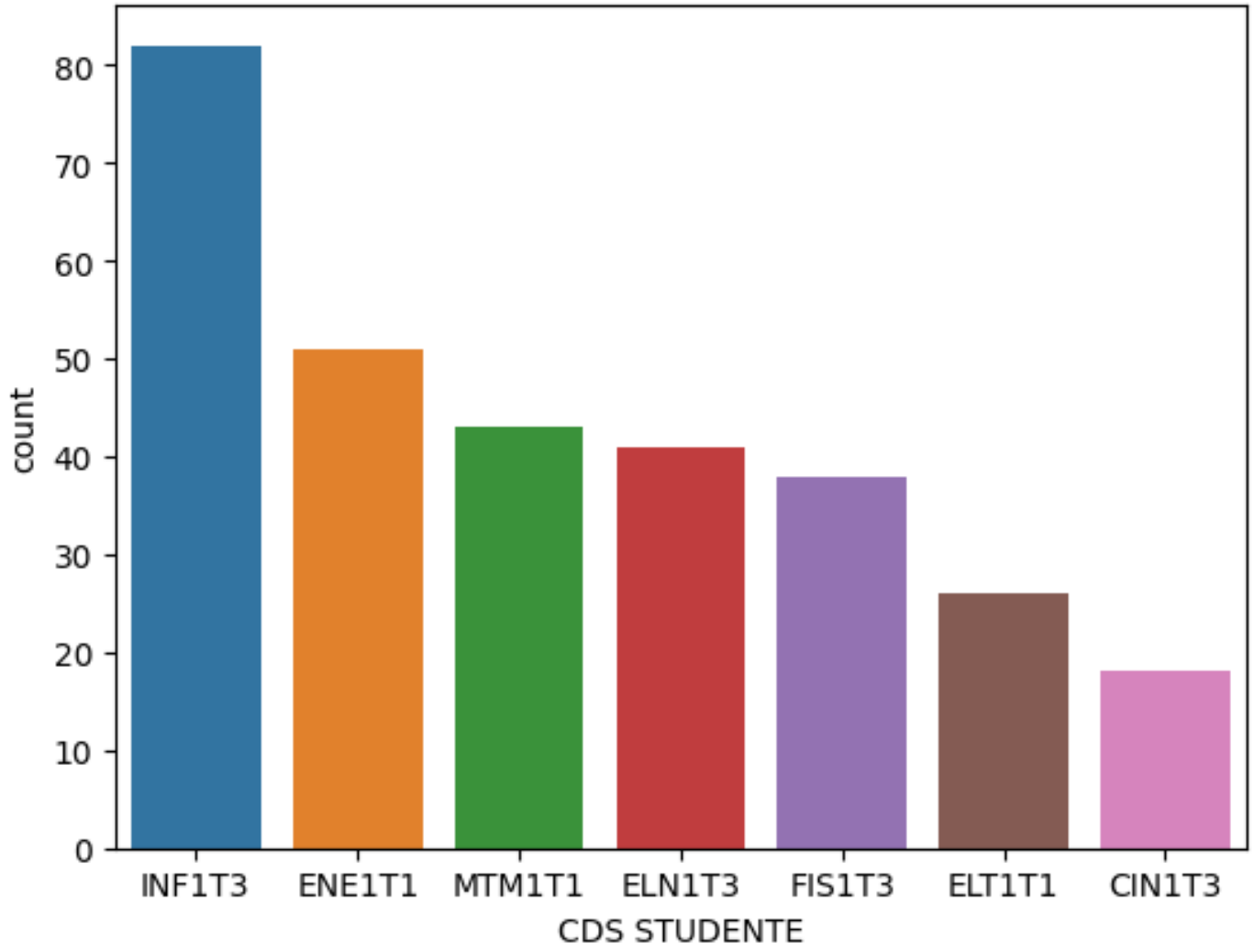
- Ingegneria =
  - Saper progettare
  - Risolvere problemi
  - Trovare soluzioni
  - Soddisfare le specifiche
  - Nel rispetto dei vincoli
  - Con gli strumenti disponibili



- Ingegneria informatica =
  - Problemi di ogni genere (calcolo, gestione dati, interazione, …)
  - Lo strumento è il calcolatore

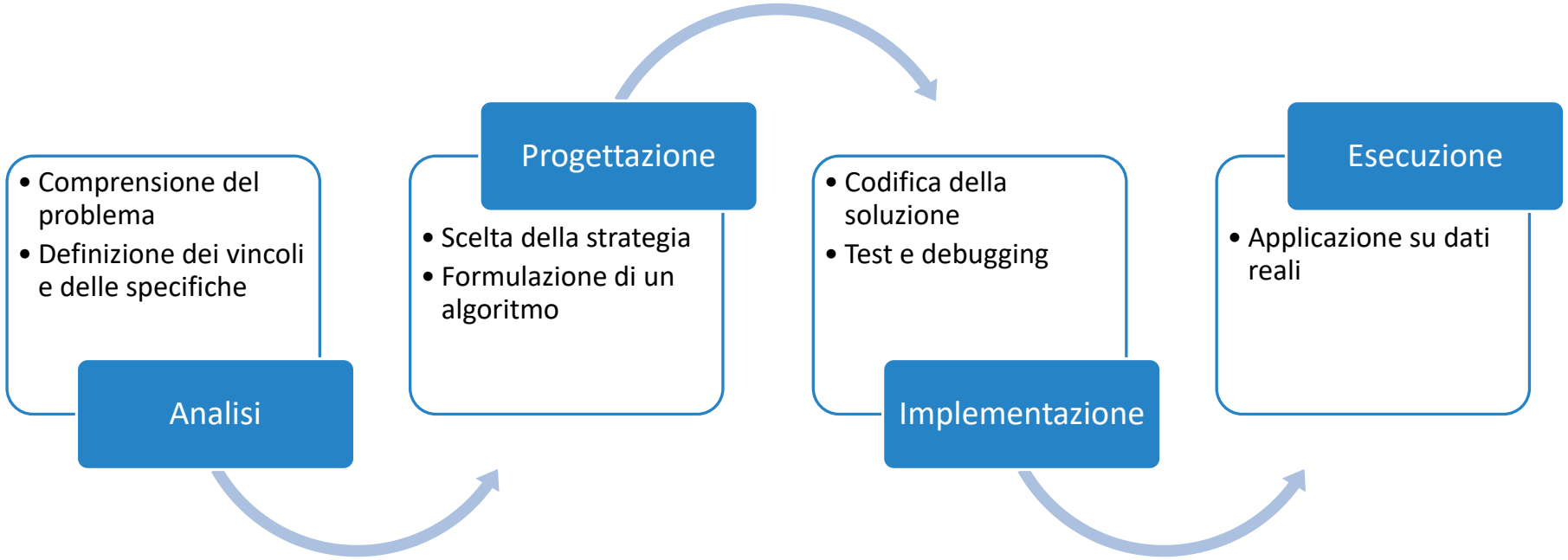
Statistiche studenti

```
import pandas as pd
import seaborn as sns
studenti = pd.read_csv('14BHDOA_2024.csv')
sns.countplot(data=studenti, x='CDS STUDENTE')
```



...e cioè cosa impariamo a fare?

- Quali sono i nomi più frequenti in quest'aula?

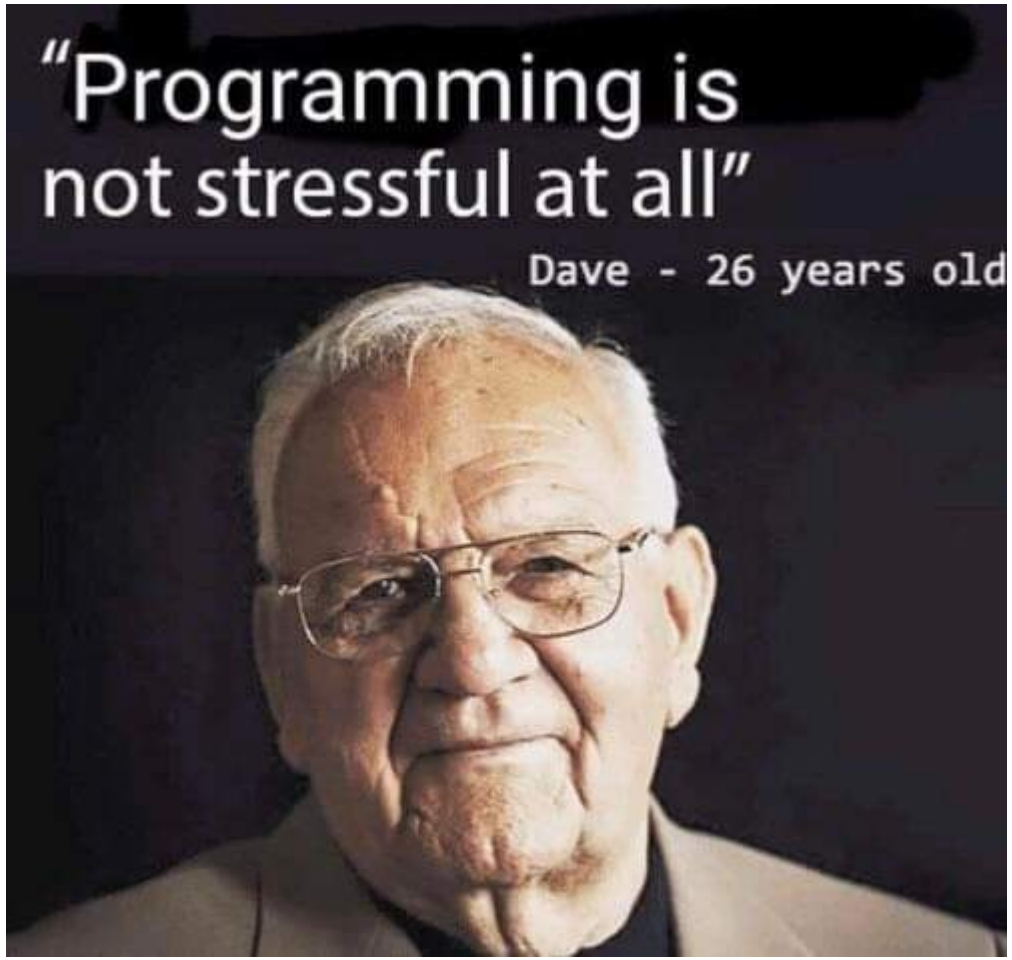


Una possibile soluzione… in Python

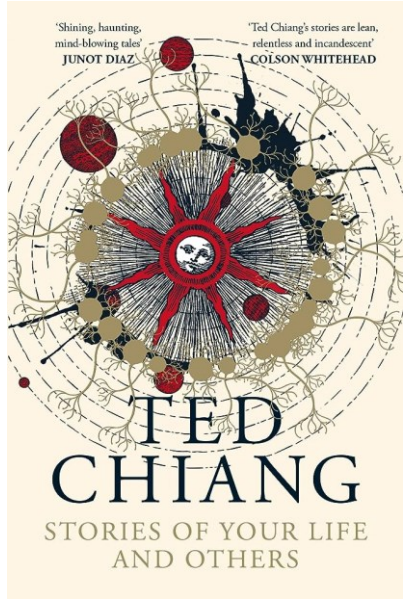
```python
import csv
from matplotlib import pyplot

# Leggi l'elenco degli studenti e salvalo in un'array
def leggi(nome_file):
    file = open(nome_file, 'r')
    reader = csv.reader(file)
    prima = True
    studenti = []
    for line in reader:
        if prima:  # skip first line (headers)
            prima = False
        else:
            studenti.append(line)
    file.close()
    return studenti

# estrai i nomi di battesimo da un elenco di studenti
def estrai_nomi(elenco):
    lista_nomi = []
    for riga in elenco:
        lista_nomi.append(riga[2])
    return lista_nomi

# Calcola le frequenze dei vari nomi presenti in un array
def frequenze(tokens):
    freq = {}
    for token in tokens:
        if token in freq:
            freq[token] = freq[token] + 1
        else:
            freq[token] = 1
    return freq
```

```python
# calcola il massimo valore presente nelle frequenze
def max_frequenza(freq):
    return max(freq.values())


def nomi_piu_frequenti(freq, max):
    return [nome for (nome, frequenza) in freq.items() if frequenza == max]


FILENAME = '01TXYOV_2020.csv'
def main():
    stud = leggi(FILENAME)
    nomi = estrai_nomi(stud)
    print(f"Nella classe ci sono {len(stud)} studenti")
    freq = frequenze(nomi)
    max_freq = max_frequenza(freq)
    print(f"Il nome più frequente compare {max_freq} volte")
    nomi_max = nomi_piu_frequenti(freq, max_freq)
    print(f"Si tratta di : {nomi_max}")
    # estrai solo i nomi che compaiono almeno 3 volte
    freq2 = {k: v for (k, v) in freq.items() if v >= 3}
    print(
        f"I nomi che compaiono più volte sono {', '.join(sorted(list(freq2.keys())))}."
    )

    pyplot.barh(list(freq2.keys()), freq2.values())
    pyplot.show()


main()
```

https://replit.com/@fulcorno/NomiFrequentiStudenti#main.py


“Programming is
not stressful at all”
Dave - 26 years old

A cosa serve imparare a programmare?



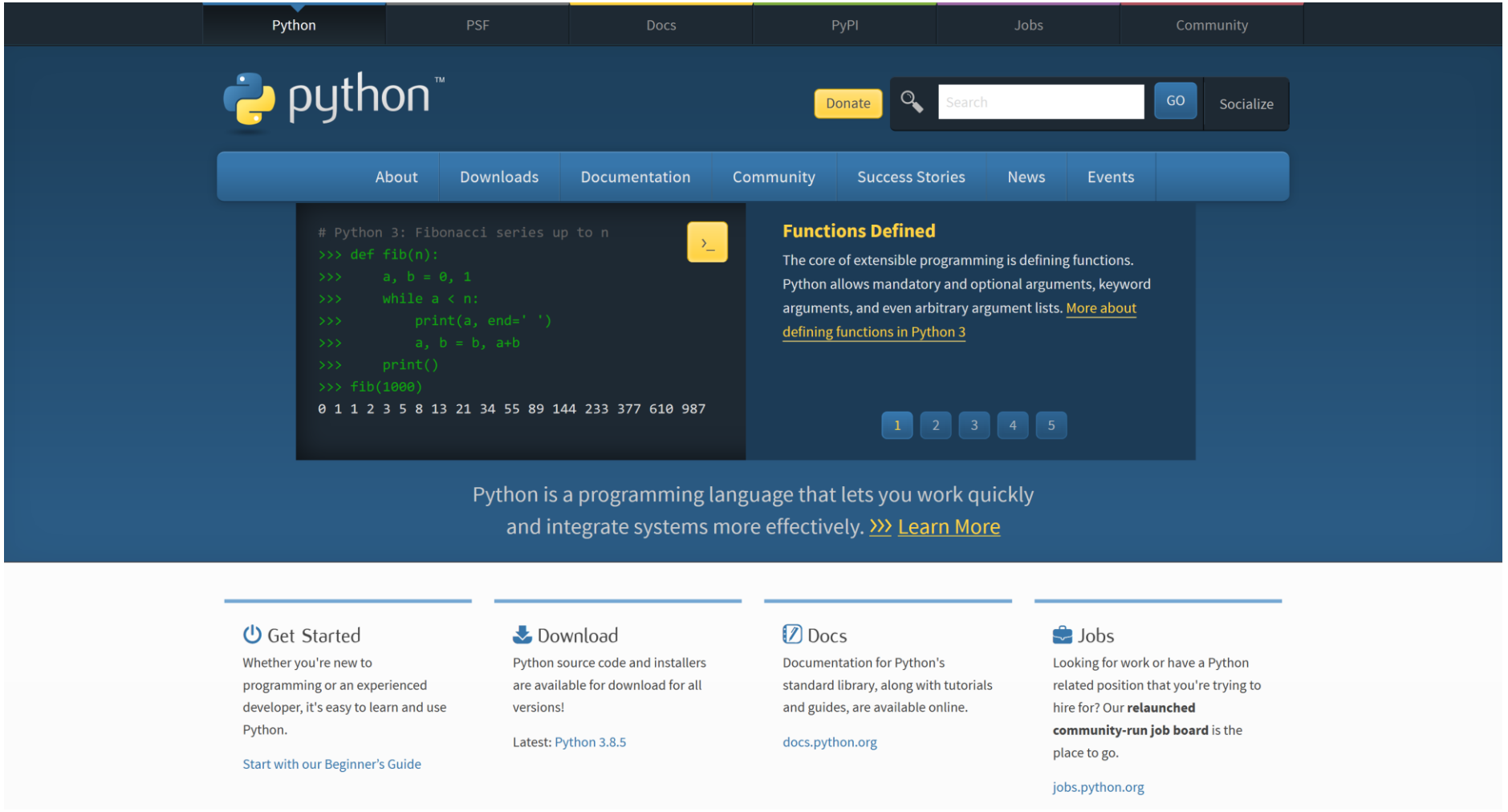
Uno sguardo a Python

VISIONE GENERALE DELL'ECOSISTEMA PYTHON

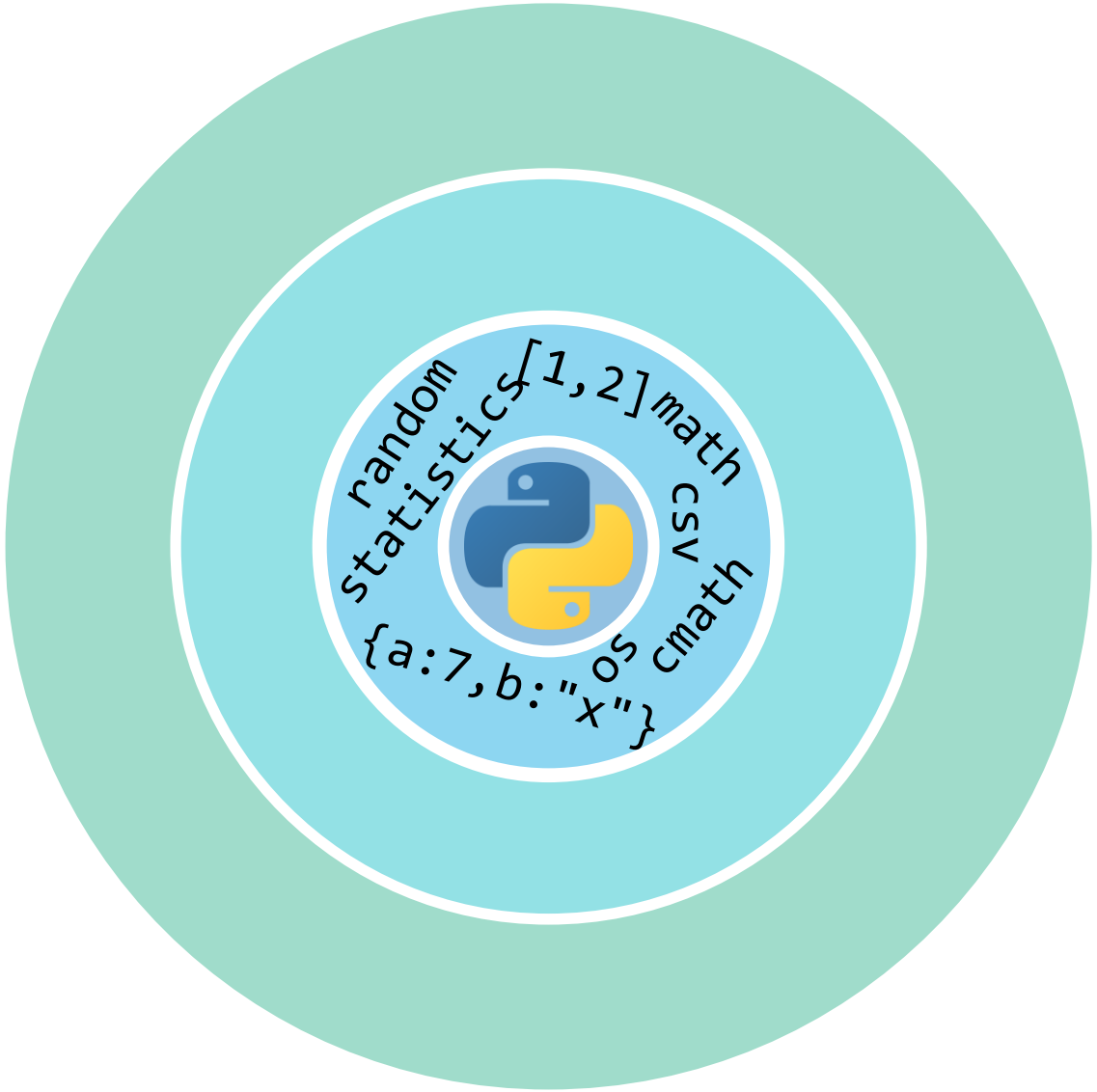
Il linguaggio Python

- Linguaggio gratuito ed open-source
- Disponibile per tutti i sistemi operativi
  - Windows, Mac OS X, Linux
  - Sistemi embedded, Raspberry PI, Android
- Progettato negli anni ’90 da Guido Van Rossum
  - Sintassi semplice, pulita, regolare
  - Approccio «batterie incluse»
    - Ampia libreria di funzioni standard
  - Basso gradino d’accesso
    - Linguaggio interpretato
- Sterminata documentazione on-line

https://www.python.org/



Batterie incluse

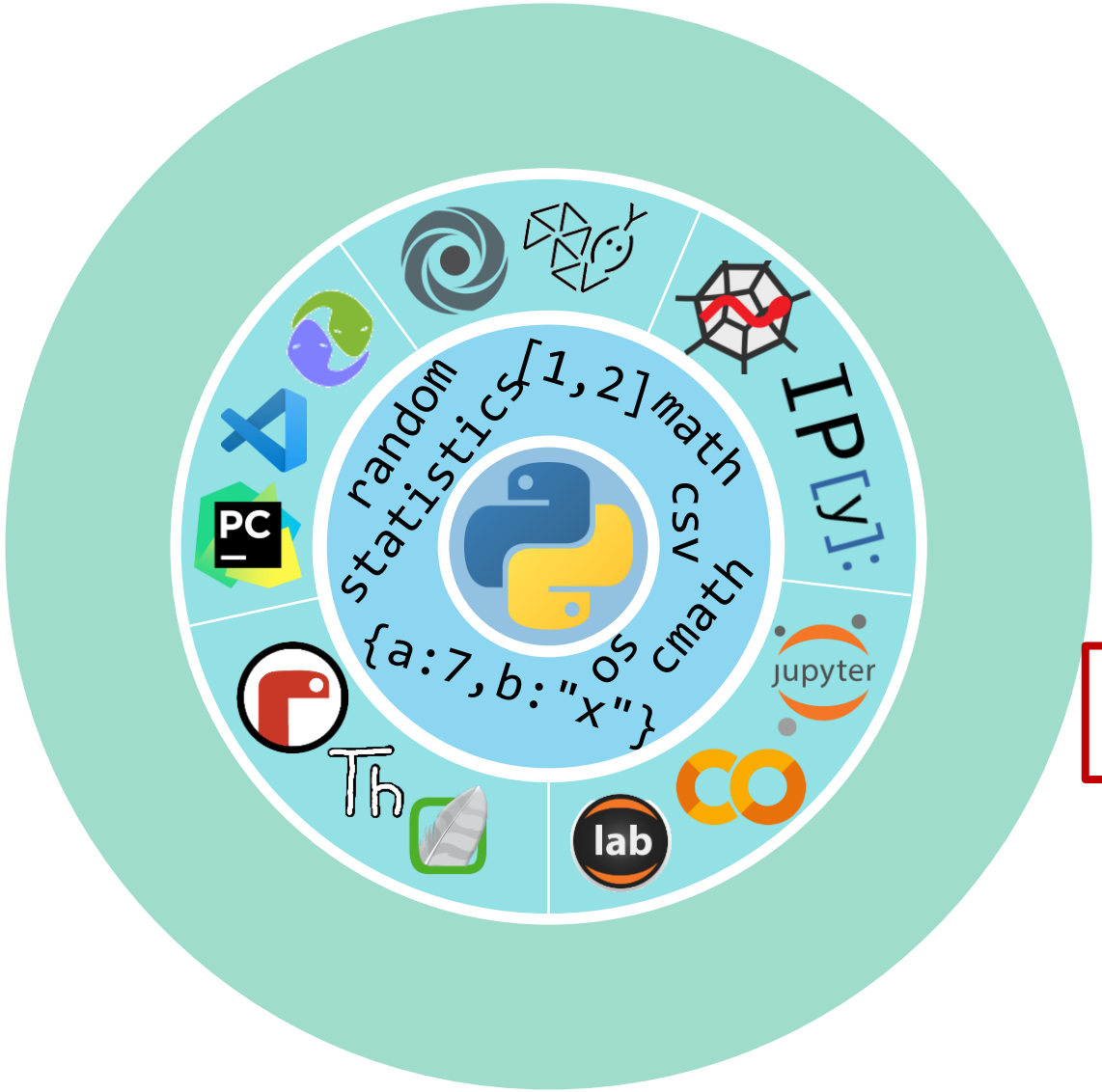


- Tipi di dato fondamentali
  - boolean, int, float, complex, string, regexp
- Strutture dati fondamentali
  - liste/array/matrici, tuple, insiemi, dizionari/mappe/hash, file, …
- Orientato agli oggetti
  - Utilizzo semplice e diretto di oggetti predefiniti
  - Possibilità di creare classi ed oggetti personalizzati (avanzato)
- 200+ Moduli nella libreria standard

200 Moduli della libreria standard

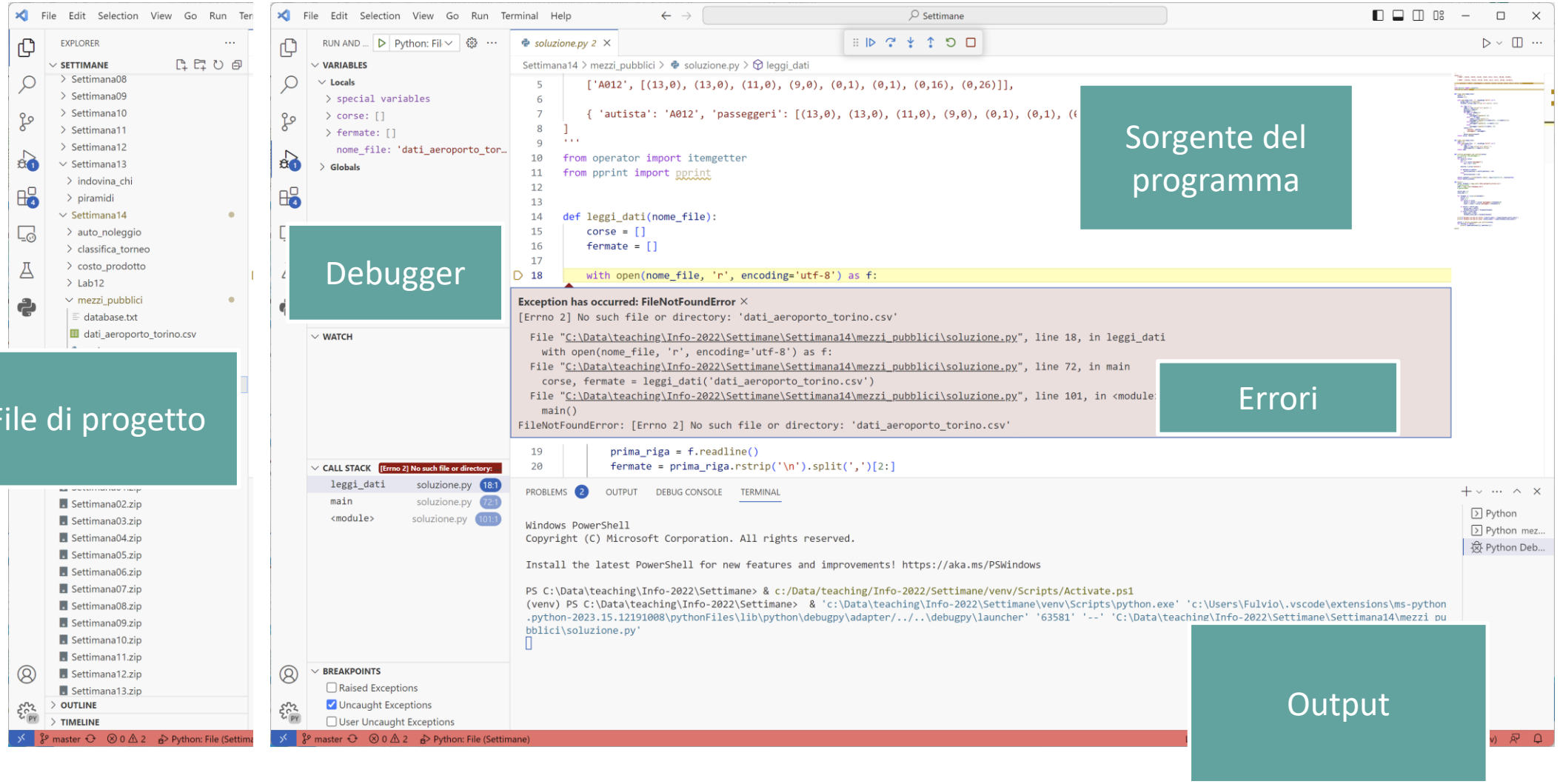
abc	chunk	**decimal**	getpass	keyword	optparse	**queue**	sndhdr	telnetlib	**unittest**
aifc	**cmath**	difflib	gettext	linecache	**os**	quopri	socket	**tempfile**	urllib
argparse	cmd	dis	glob	locale	ossaudiodev (Linux, FreeBSD)	**random**	socketserver	termios (Unix)	uu
array	**codecs**	distutils	**graphlib**	**logging**	parser	**re**	spwd (Unix)	**test**	**uuid**
ast	codeop	doctest	grp (Unix)	lzma	**pathlib**	readline (Unix)	**sqlite3**	**textwrap**	venv
asynchat	**collections**	**email**	gzip	mailbox	pdb	reprlib	ssl	threading	warnings
asyncio	colorsys	encodings	**hashlib**	mailcap	pickle	resource (Unix)	stat	**time**	wave
asyncore	compileall	ensurepip	**heapq**	marshal	pickletools	rlcompleter	**statistics**	**timeit**	weakref
atexit	configparser	enum	hmac	**math**	pipes (Unix)	runpy	**string**	**tkinter**	**webbrowser**
audioop	contextlib	errno	**html**	mimetypes	pkgutil	sched	stringprep	token	winreg (Win)
base64	contextvars	faulthandler	**http**	mmap	platform	secrets	struct	tokenize	winsound (Win)
bdb	**copy**	fcntl (Unix)	imaplib	modulefinder	plistlib	select	subprocess	trace	wsgiref
binascii	copyreg	**filecmp**	imghdr	msilib (Windows)	poplib	selectors	sunau	traceback	xdrlib
binhex	crypt (Unix)	fileinput	imp	msvcrt (Windows)	**pprint**	shelve	symbol	tracemalloc	**xml**
bisect	**csv**	fnmatch	importlib	multiprocessing	profile	shlex	symtable	tty (Unix)	xmlrpc
builtins	ctypes	**fractions**	inspect	netrc	pstats	**shutil**	**sys**	**turtle**	**zipapp**
bz2	curses (Unix)	ftplib	**io**	nis (Unix)	pty (Linux)	signal	sysconfig	turtledemo	zipfile
calendar	**dataclasses**	functools	ipaddress	nntplib	pwd (Unix)	site	syslog (Unix)	types	zipimport
cgi	**datetime**	gc	**itertools**	**numbers**	pyclbr	smtpd	tabnanny	typing	**zlib**
cgitb	dbm	getopt	json	**operator**	pydoc	smtplib	tarfile	unicodedata	zoneinfo

Gli ambienti di lavoro



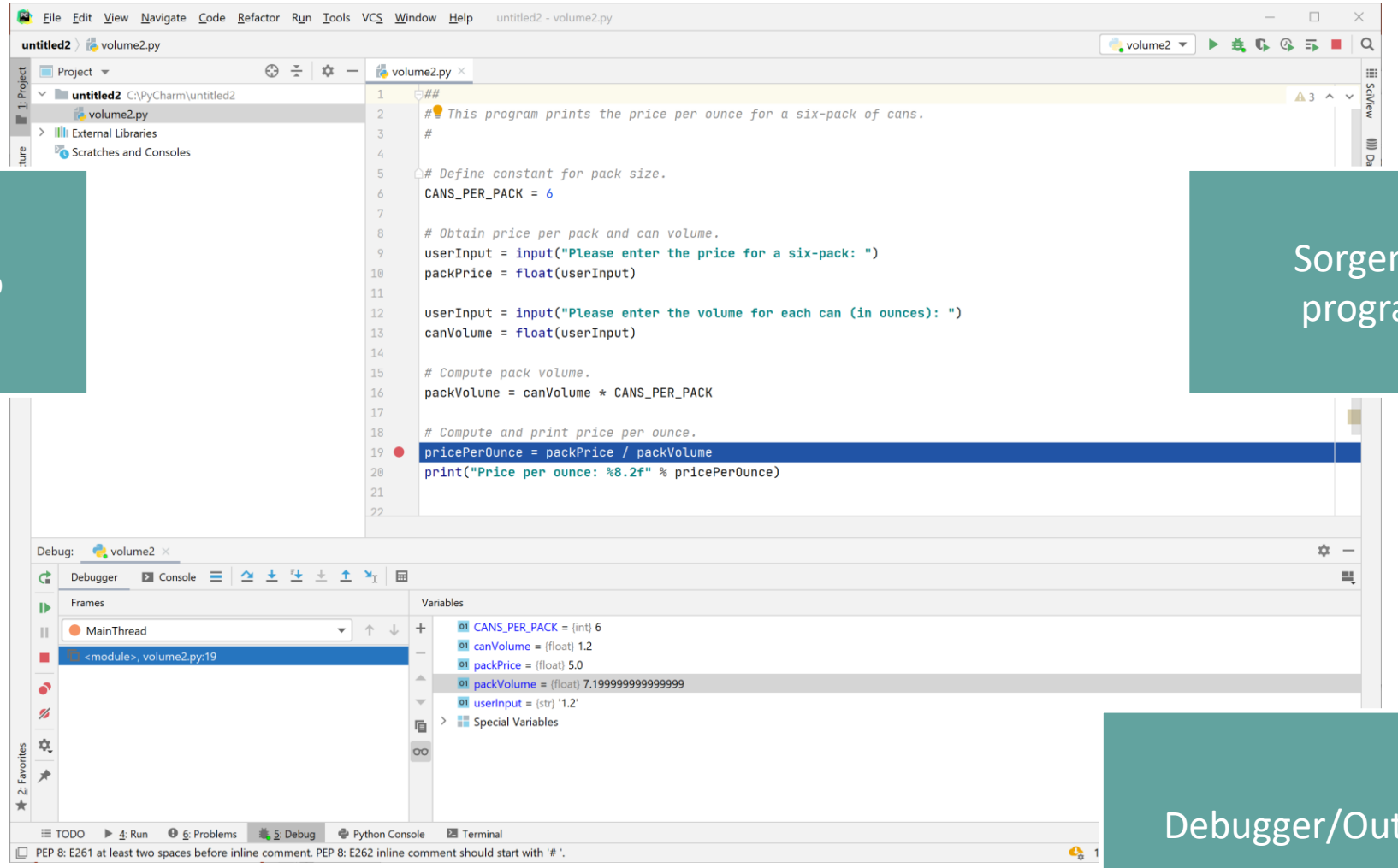
- Ambienti di sviluppo **tradizionali** (IDE)
  - IDLE, PyCharm, Visual Studio Code, Eclipse PyDev, …
- Ambienti di sviluppo **on-line**
  - Repl.it, PythonAnywhere, Python Tutor
- Ambienti per il calcolo **interattivo**
  - Spyder, IPython
- **Notebook** Computazionali
  - Jupyter, JupyterLab, Google Colab
- Ambienti per **l'apprendimento**
  - Mu, Thonny, Wing

L'IDE di Visual Studio Code



L’IDE di PyCharm

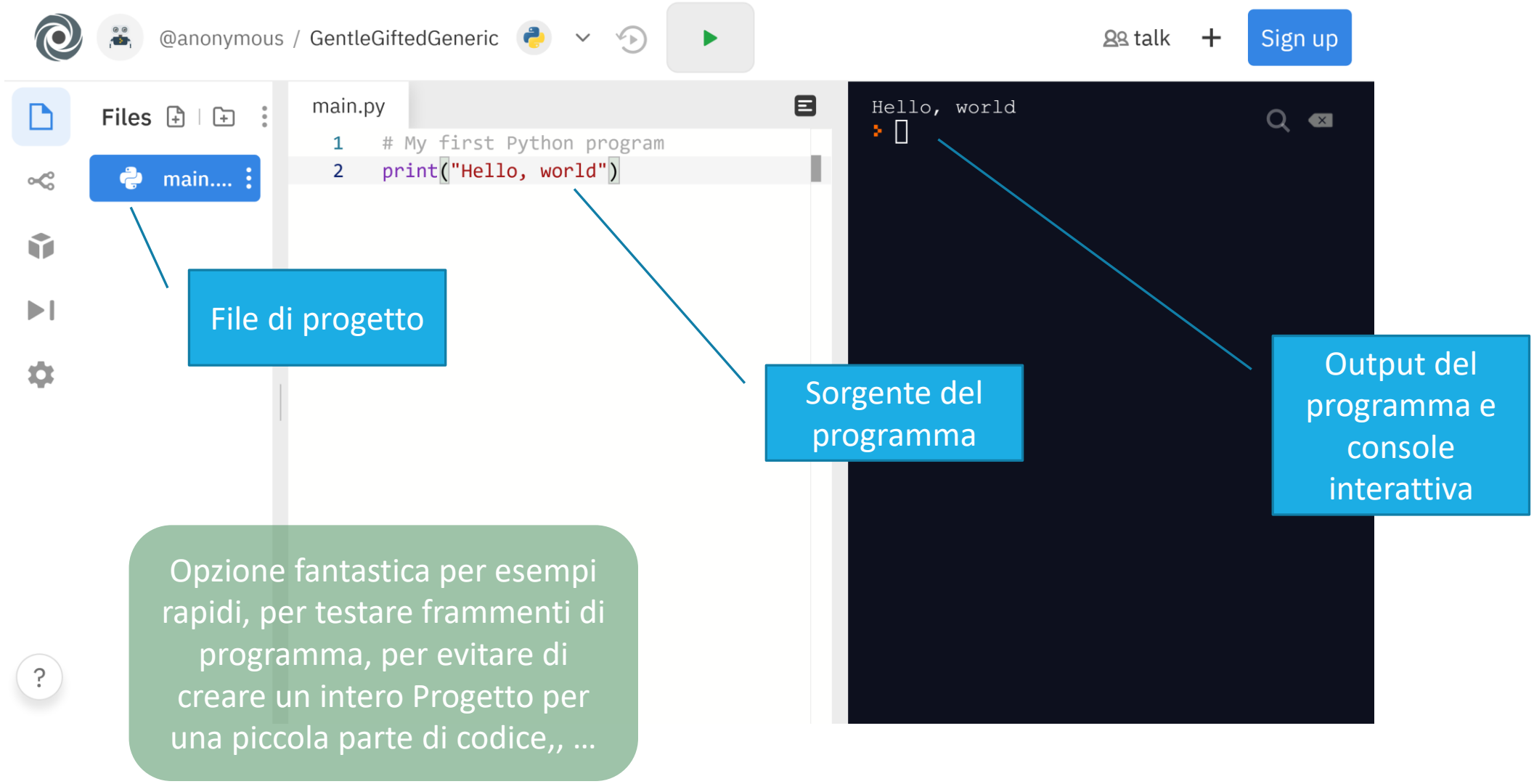
File di progetto



Sorgente del programma

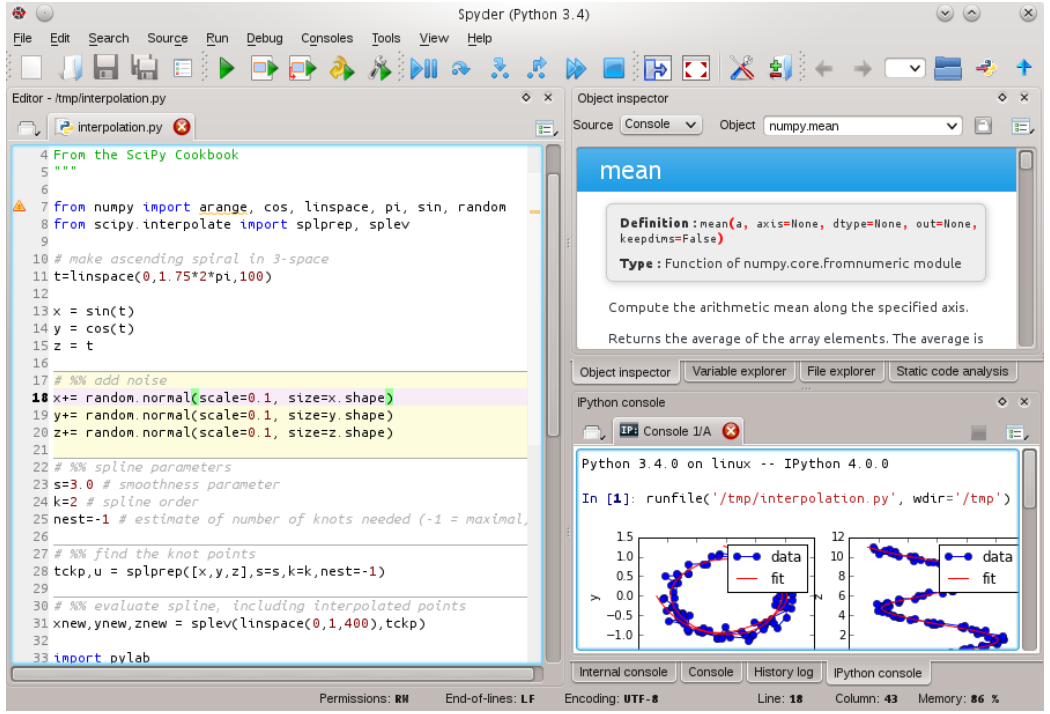
Debugger/Output

IDE On-line : https://replit.com/

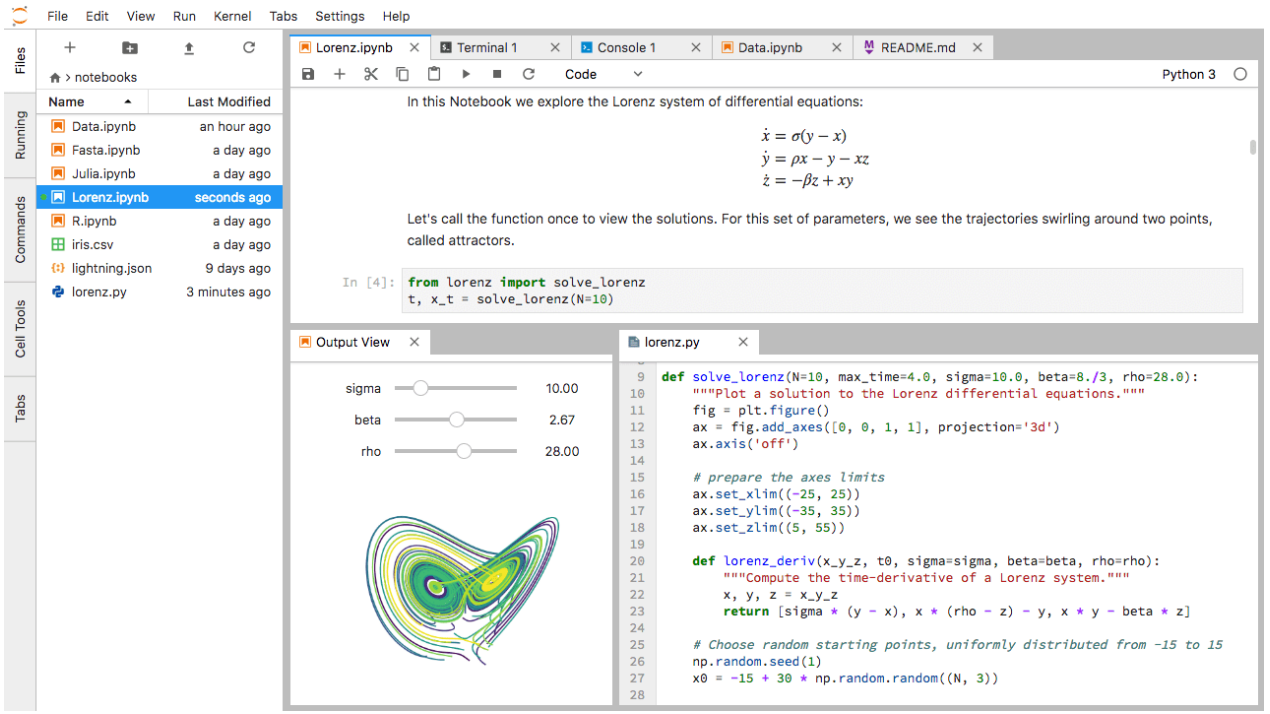


Ambienti scientifici interattivi

SPYDER



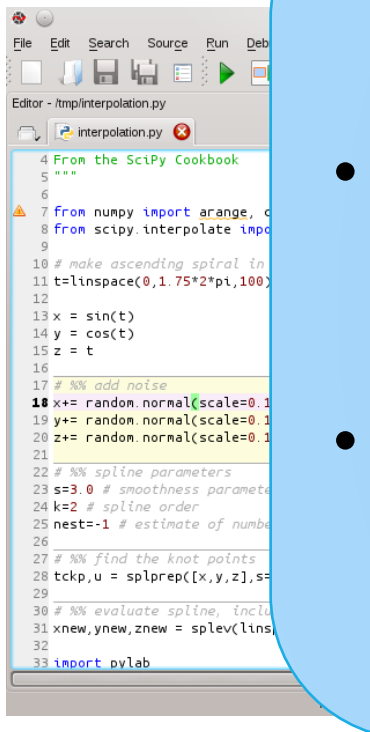
JUPYTERLAB (ANCHE ON-LINE), GOOGLE COLAB



Ambienti scientifici interattivi

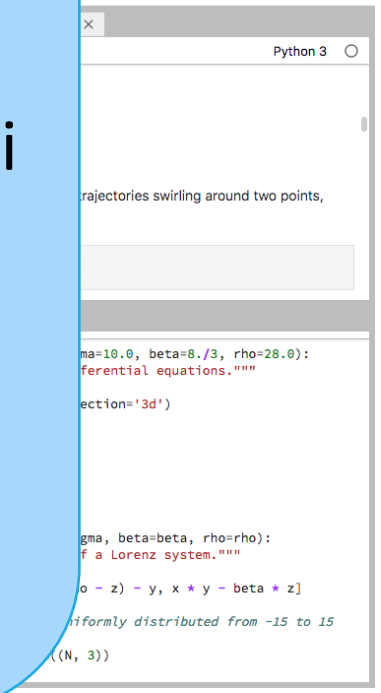
SPYDER

JUPYTERLAB (ANCHE ON-LINE), GOOGLE COLAB

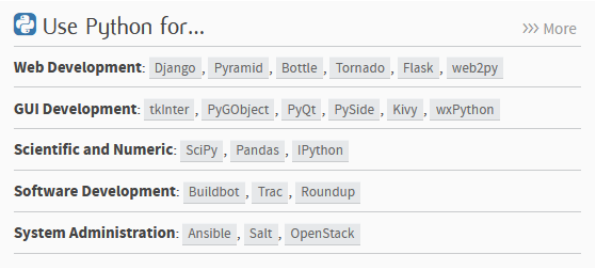


Nuove possibilità…

- Pubblicare on-line esercizi «interattivi» sotto forma di notebook
- Redigere le prime versioni di un articolo, inframmezzando il testo alle formule, con il [ri-]calcolo automatico di risultati e grafici



Librerie per ambiti applicativi



- Scientific computation
  - NumPy, SciPy, SymPy
- Data Analysis, Algoritmi, Grafi
  - Pandas, networkx, GeoPandas
- Image Processing
  - Pillow, scikit-image, OpenCV
- Visualization
  - Pyviz, matplotlib, plotly, seaborn, altair
- Machine Learning
  - Scikit-learn, tensorflow, pytorch, keras
- Fintech
  - f.fn, zipline, pyalgotrade
- Biology and Genome
  - Biopython
- Fluid Dynamics
  - Fluidity
- Finite Elements
  - Sfepy
- Control systems

python™ Package Index

Singoli moduli

ANACONDA®

Toolkit completo per data science

Calcolo scientifico

- NumPy
  - Array, vettori, algebra lineare

- SciPy
  - Package specializzati su diversi ambiti scientifici

- SymPy
  - Calcolo simbolico

- Pandas
  - Analisi e manipolazione dati

Subpackage	Description
cluster	Clustering algorithms
constants	Physical and mathematical constants
fftpack	Fast Fourier Transform routines
integrate	Integration and ordinary differential equation solvers
interpolate	Interpolation and smoothing splines
io	Input and Output
linalg	Linear algebra
ndimage	N-dimensional image processing
odr	Orthogonal distance regression
optimize	Optimization and root-finding routines
signal	Signal processing
sparse	Sparse matrices and associated routines
spatial	Spatial data structures and algorithms
special	Special functions
stats	Statistical distributions and functions

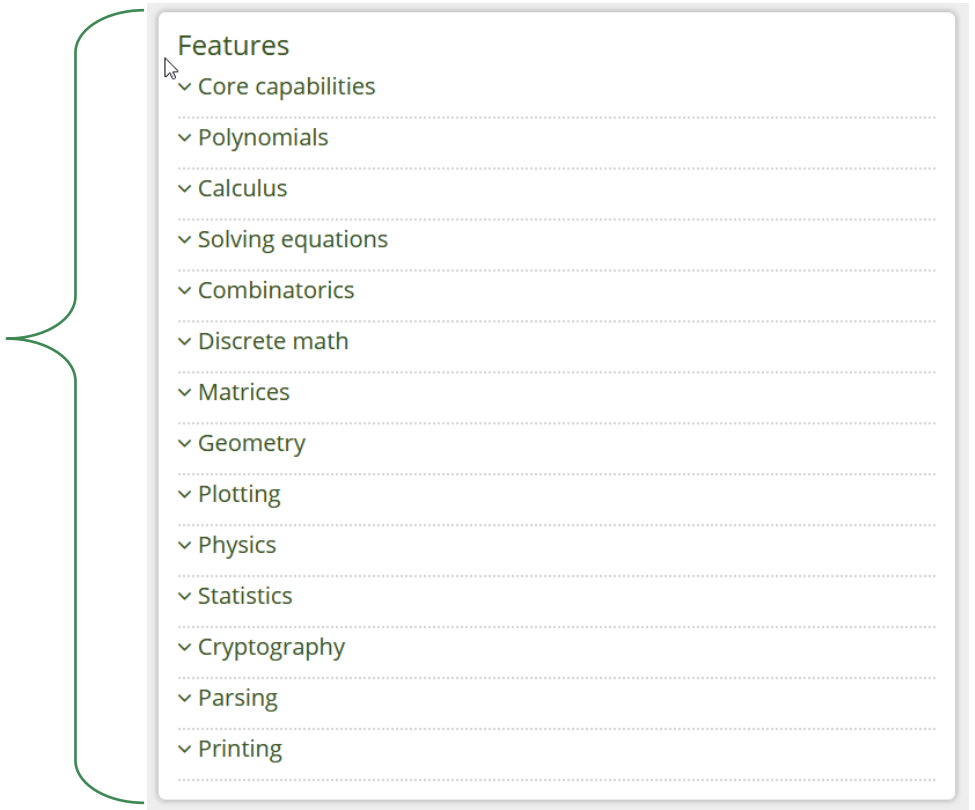
Calcolo scientifico

- NumPy
  - Array, vettori, algebra lineare

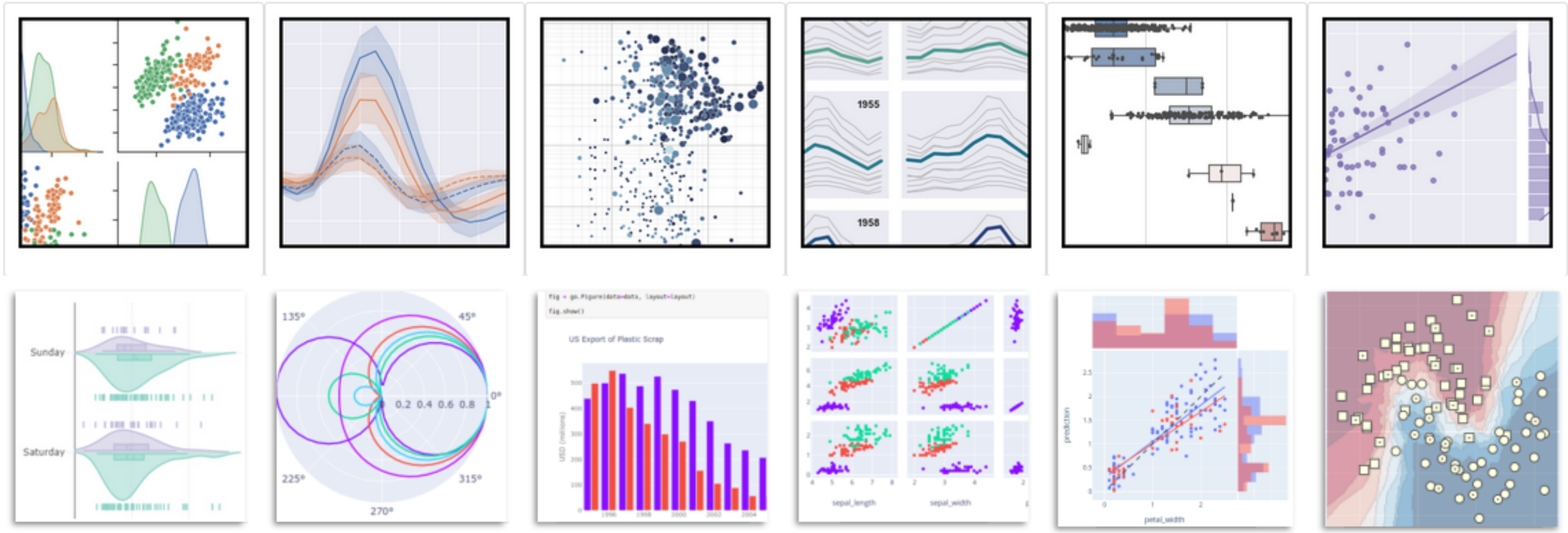
- SciPy
  - Package specializzati su diversi ambiti scientifici

- SymPy
  - Calcolo simbolico

- Pandas
  - Analisi e manipolazione dati



Visualizzazione

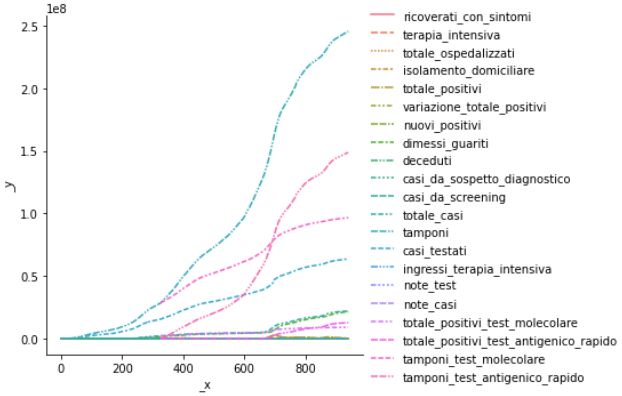


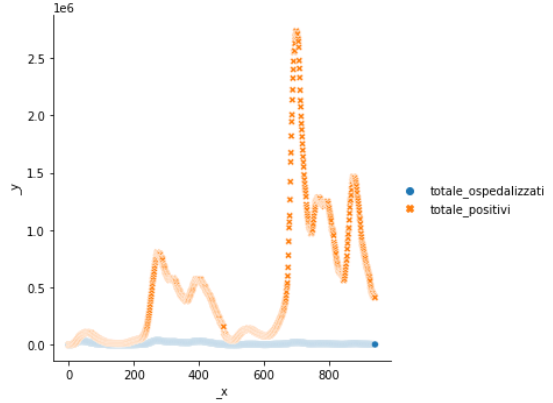
matplotlib, plotly, seaborn, …

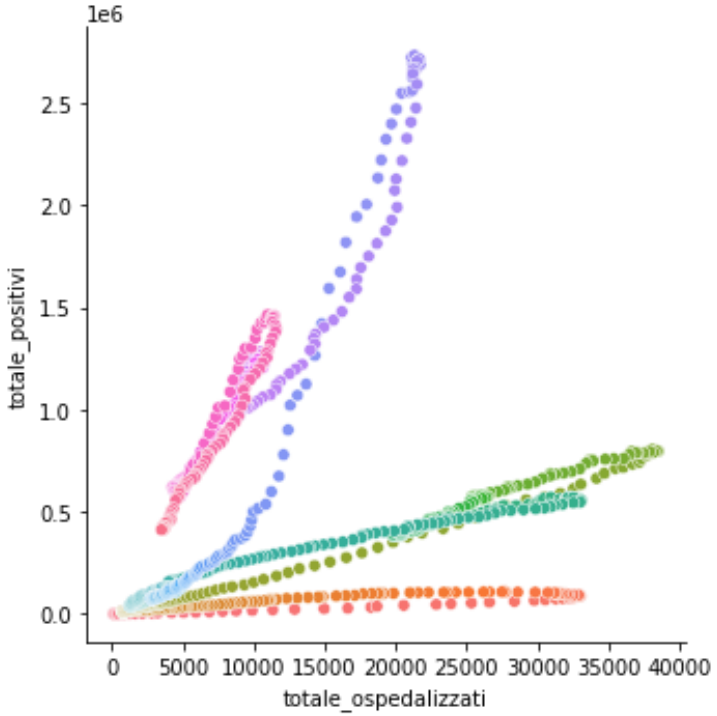
Esempio: dati ufficiali Covid-19 in real-time

```
import pandas as pd
import seaborn as sns
sns.set_style("whitegrid")

# Leggi dati aggiornati
covid = pd.read_json(
path_or_buf='https://raw.githubusercontent.com/pcm-dpc/COVID-
19/master/dati-json/dpc-covid19-ita-andamento-nazionale.json',
convert_dates=['data'])

covid.set_index('data', inplace=True)

sns.relplot(data=covid, kind='line')

dati_utili = covid[['totale_ospedalizzati', 'totale_positivi' ]]

sns.relplot(data=dati_utili, kind='line')

sns.relplot(data=dati_utili, kind='scatter',
x='totale_ospedalizzati', y='totale_positivi', hue='data',
legend=False)
```

Try me on Google Colab







Homework

- Choose one (quantitative) research question related to your PhD work
  - Concerning experimental data
  - Concerning user interviews
  - Concerning theoretical models
  - ….
- Write down the research question
- Analyze the question and try to re-write it in the most possible explicit and non-ambiguous way

- https://polito.padlet.org/fulcorno/ambienti-di-calcolo-e-simulazione-per-la-ricerca-sperimental-x2znowd1ibxngu0n