



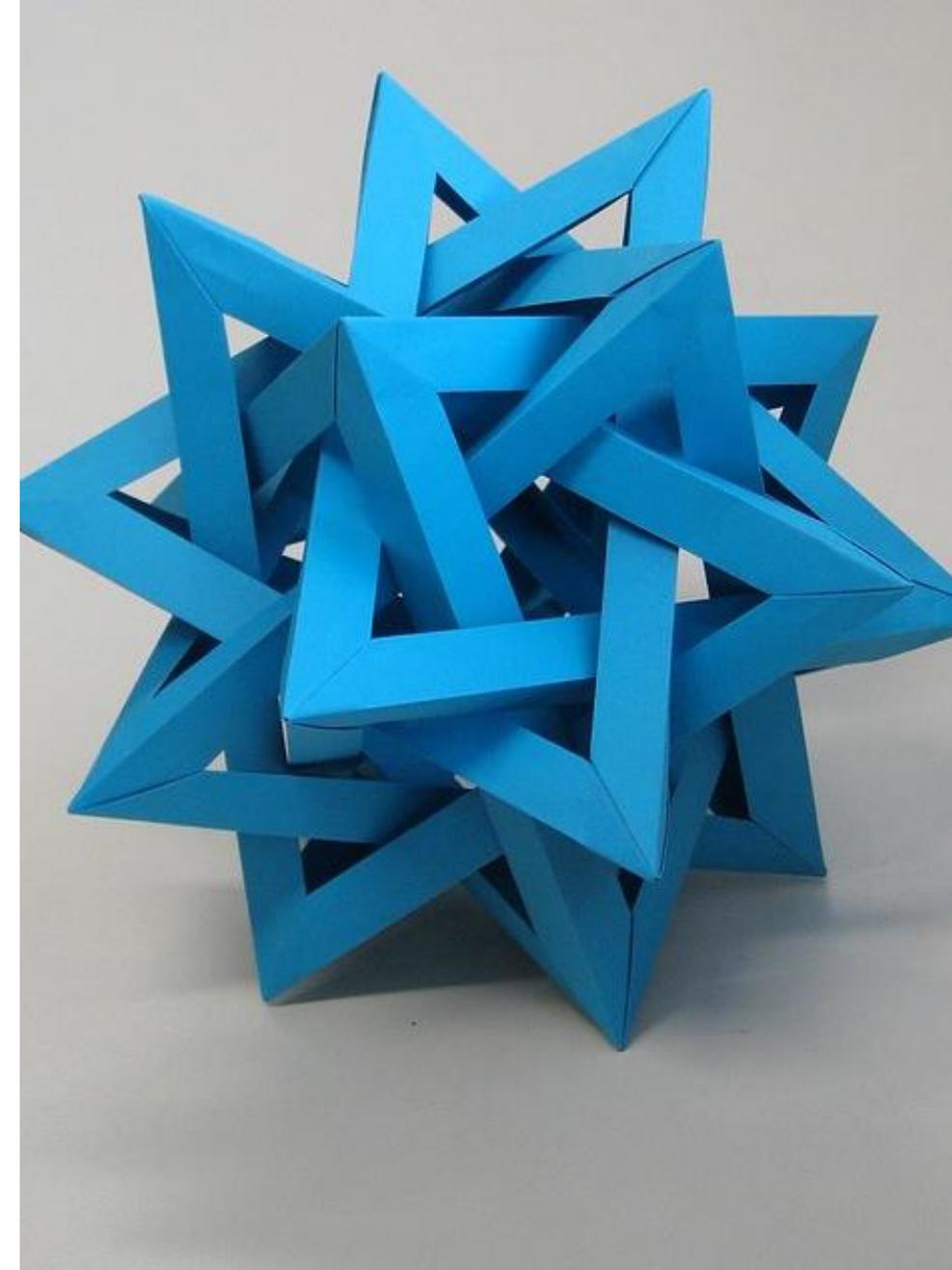
Politecnico
di Torino

Dipartimento
di Automatica e Informatica

14BHDxx Informatica

ING. INF/ELT/ENE/MTM/ELN/FIS/CIN – CORSO #3 (FAJ-KOR)

PROF. FULVIO CORNO



[This Photo](#) by Unknown Author is licensed under [CC BY-SA](#)

Welcome

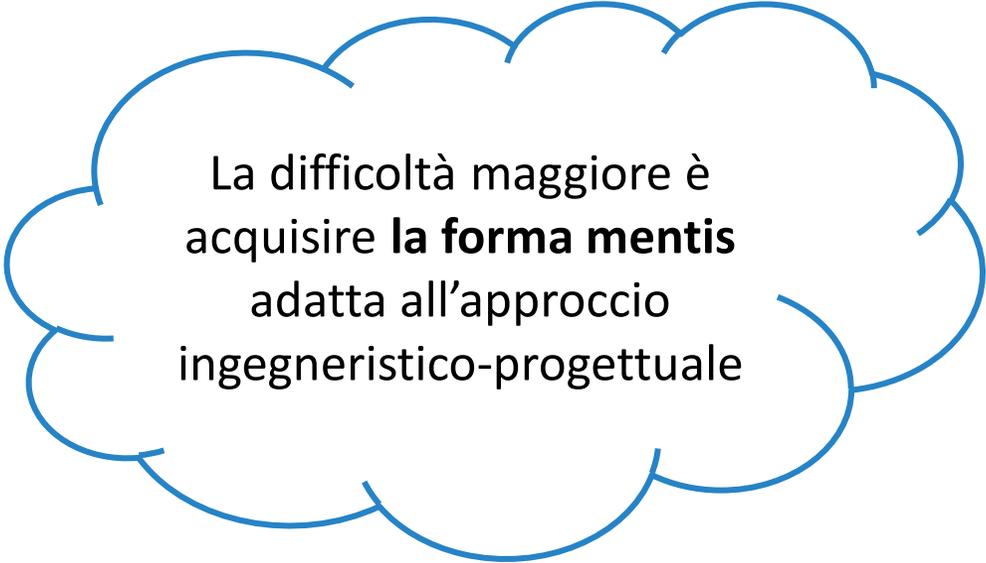
Corso di Informatica (14BHD)

- Insegnamento obbligatorio, 8 crediti, 1° Semestre
- Corso n. 3
 - Studenti con cognomi compresi tra FAJ e KOR
 - Iscritti ai corsi di ing. Informatica, Elettrica, Energetica, Matematica, Elettronica, Fisica, Cinema
- Docenti:
 - **Fulvio Corno**
(lezioni, esercitazioni in aula)
 - **Roberta Bardini**
(esercitazioni in laboratorio, esercitazioni in aula)
 - **Lorenzo Martini**
(esercitazioni in laboratorio)



Benvenuti ad Ingegneria

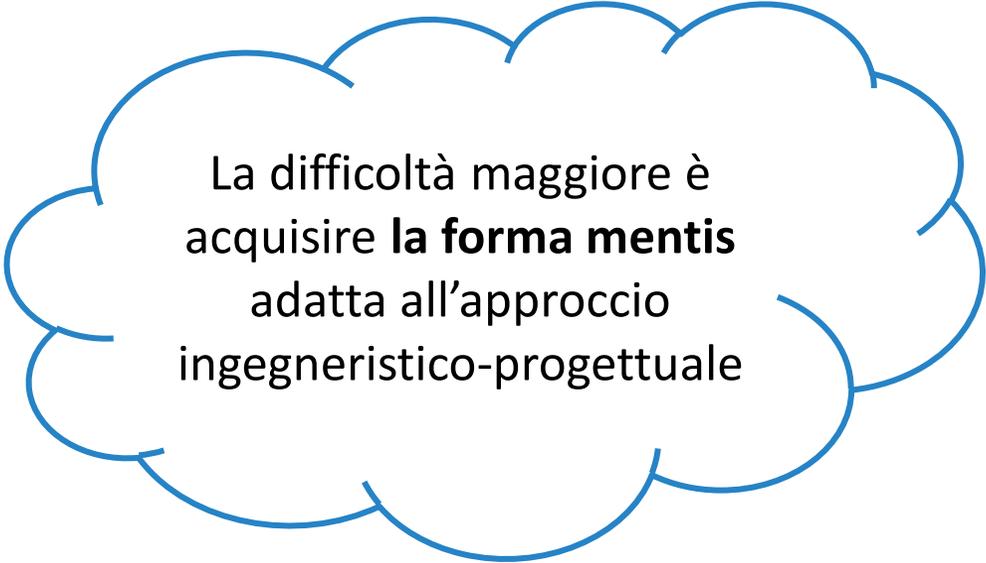
- Questo è il primo corso di Ingegneria che affronterete
- Ingegneria =
 - Saper progettare
 - Risolvere problemi
 - Trovare soluzioni
 - Soddisfare le specifiche
 - Nel rispetto dei vincoli
 - Con gli strumenti disponibili



La difficoltà maggiore è
acquisire **la forma mentis**
adatta all'approccio
ingegneristico-progettuale

Benvenuti ad Ingegneria

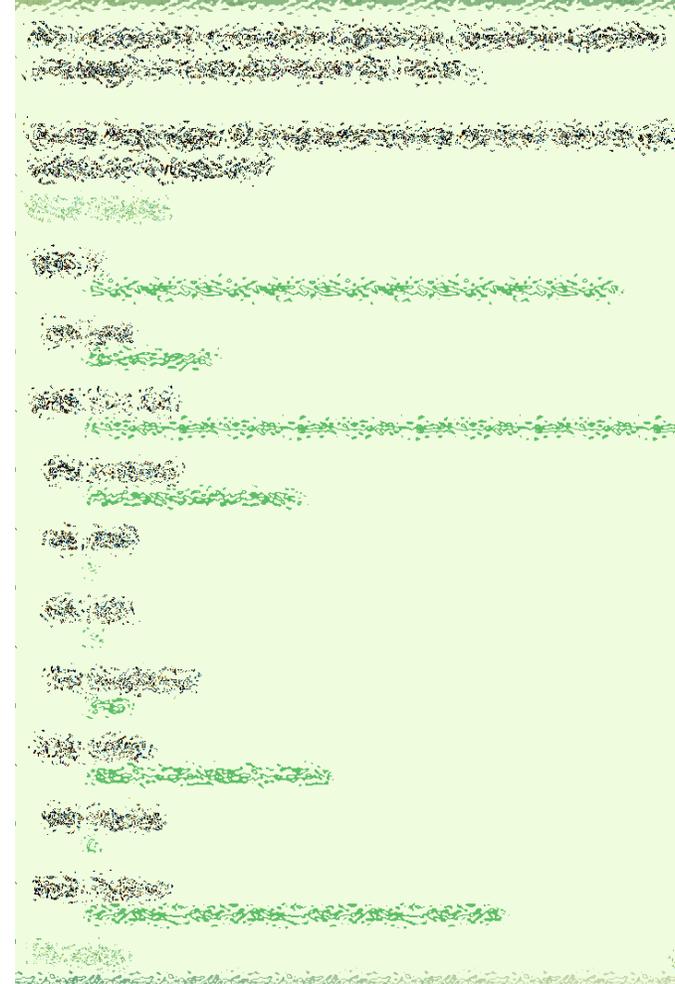
- Questo è il primo corso di Ingegneria che affronterete
- Ingegneria =
 - Saper progettare
 - Risolvere problemi
 - Trovare soluzioni
 - Soddisfare le specifiche
 - Nel rispetto dei vincoli
 - Con gli strumenti disponibili
- Ingegneria informatica =
 - Problemi di ogni genere (calcolo, gestione dati, interazione, ...)
 - Lo strumento è il calcolatore



La difficoltà maggiore è
acquisire **la forma mentis**
adatta all'approccio
ingegneristico-progettuale

Linguaggi di programmazione

- In questo corso **si parte da zero**, non è richiesto conoscere già un linguaggio
- Le capacità di **problem solving** si trasferiscono facilmente da un linguaggio all'altro
- ⚠ Talvolta, dovrete dis-imparare alcune cattive abitudini apprese in precedenza



Programma del corso

Link utili

- Sito del corso (ufficiale):
 - <http://elite.polito.it/> → Teaching → Informatica (14BHD)
 - Link breve: <http://bit.ly/polito-informatica>
- Materiale, laboratori, esercizi
 - <https://github.com/polito-informatica>
- Gruppo Telegram
 - <https://t.me/politoinfo2023>



Programma dell'insegnamento

- Metodologie di *Problem Posing and Solving* (PPS)
 - Tecniche di analisi basate su flow-chart e pseudo-code
 - Implementazione attraverso programmi informatici
- Strutture dati e rappresentazione dell'informazione nel PPS
 - Numeri, Stringhe, Vettori, Sequenze, Liste, Insiemi, Dizionari, ...
- Linguaggio di programmazione Python
 - Maggior semplicità sintattica e maggior potenza espressiva
 - Possibilità di affrontare esercizi con uno scopo applicativo più diretto
 - Ambiente di lavoro adeguato ai sistemi operativi moderni
 - Disponibilità di numerose librerie adatte a diversi campi applicativi (che potranno essere introdotte negli insegnamenti successivi).

Contenuti

- Teoria (9h)
 - Cenni di Informatica generale ed impatti dell'informatica e del digitale
 - Struttura ed architettura del calcolatore, linguaggi, applicazioni
 - Rappresentazione dell'Informazione
- Problem Solving (12h)
 - Approccio alla logica dei problemi
 - I passaggi del processo di Problem Solving
 - Tipologie di problemi e di approcci risolutivi
- Programmazione Python (41h)
 - Numeri e Stringhe
 - Decisioni
 - Cicli
 - Liste (vettori)
 - Insiemi e dizionari (array associativi)
 - File
 - Funzioni
- Laboratori (12 x 1,5 = 18h)

...e cioè cosa impariamo a fare?

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



Una possibile soluzione... in 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
```

```
# 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 = '01TXYOYV_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())))}."
    )

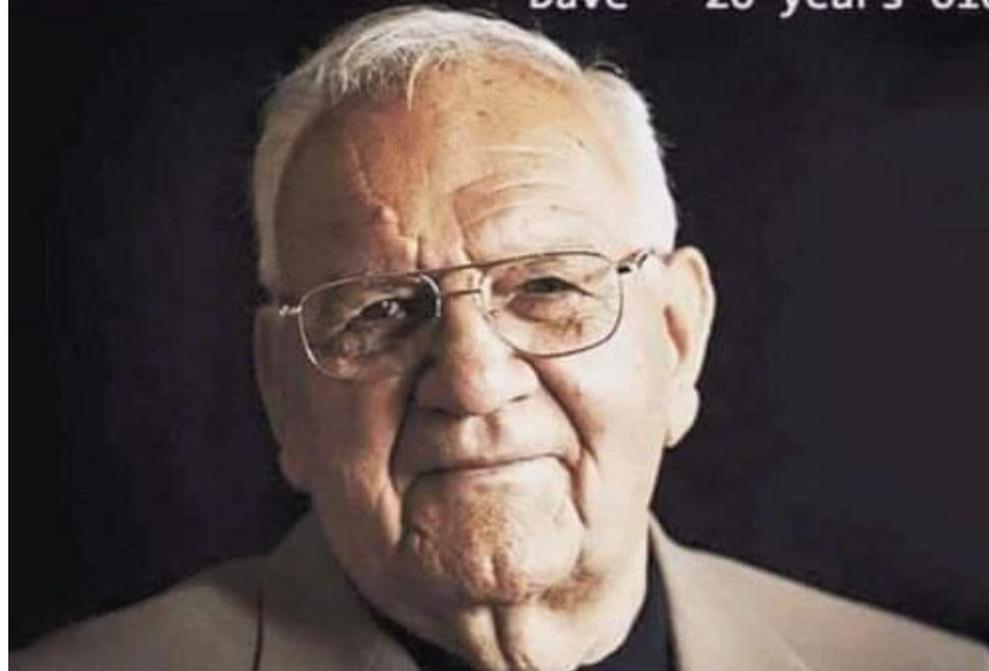
    pyplot.barh(list(freq2.keys()), freq2.values())
    pyplot.show()

main()
```

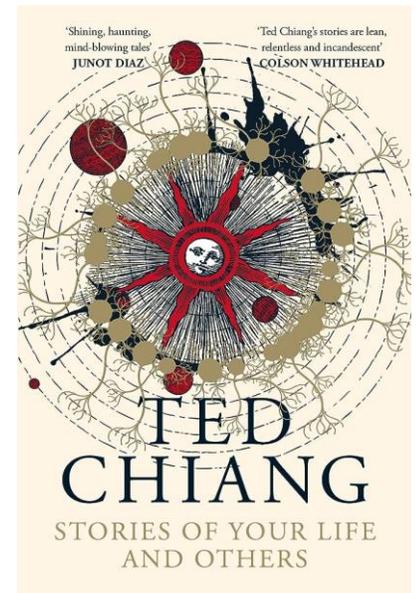
<https://replit.com/@fulcorno/NomiFrequentiStudenti#main.py>

“Programming is
not stressful at all”

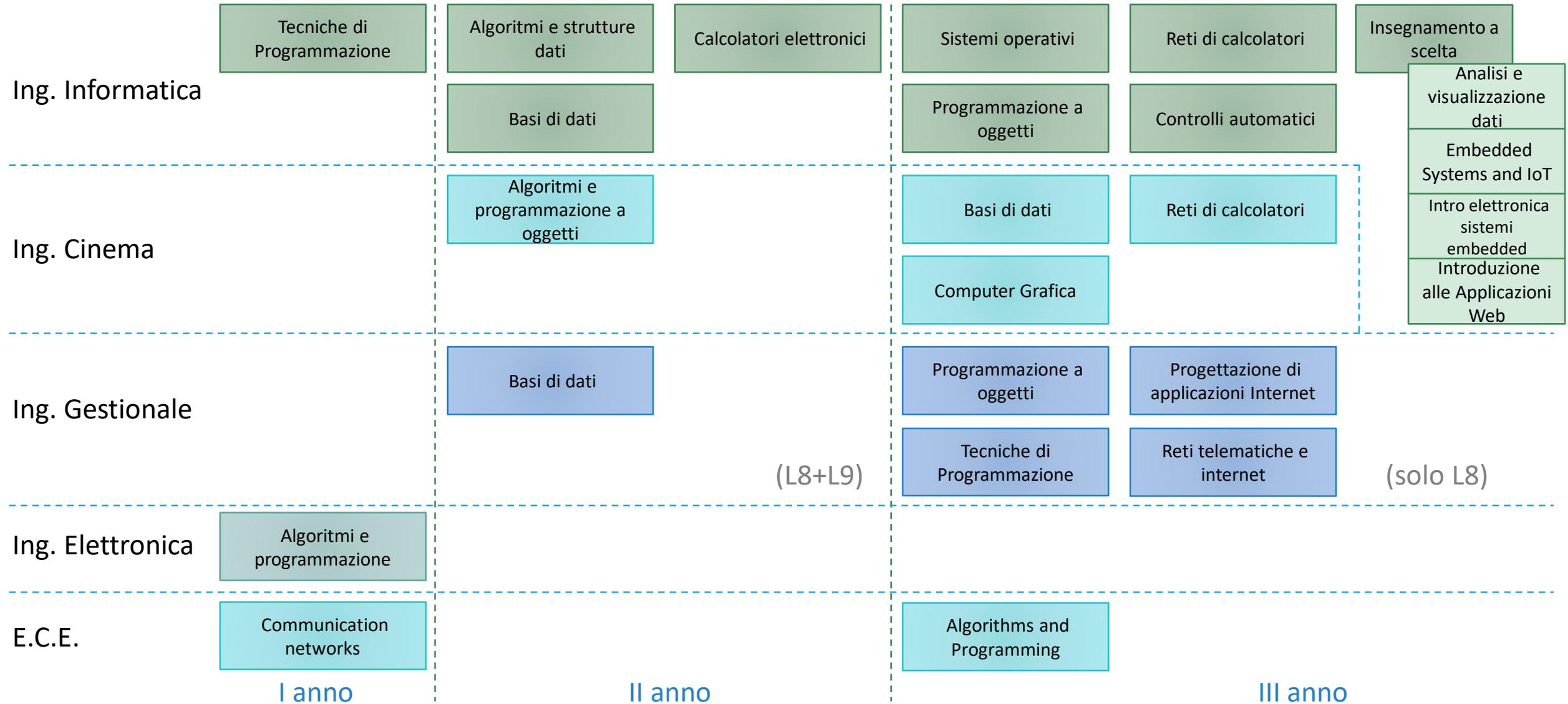
Dave - 26 years old



A cosa serve imparare a programmare?

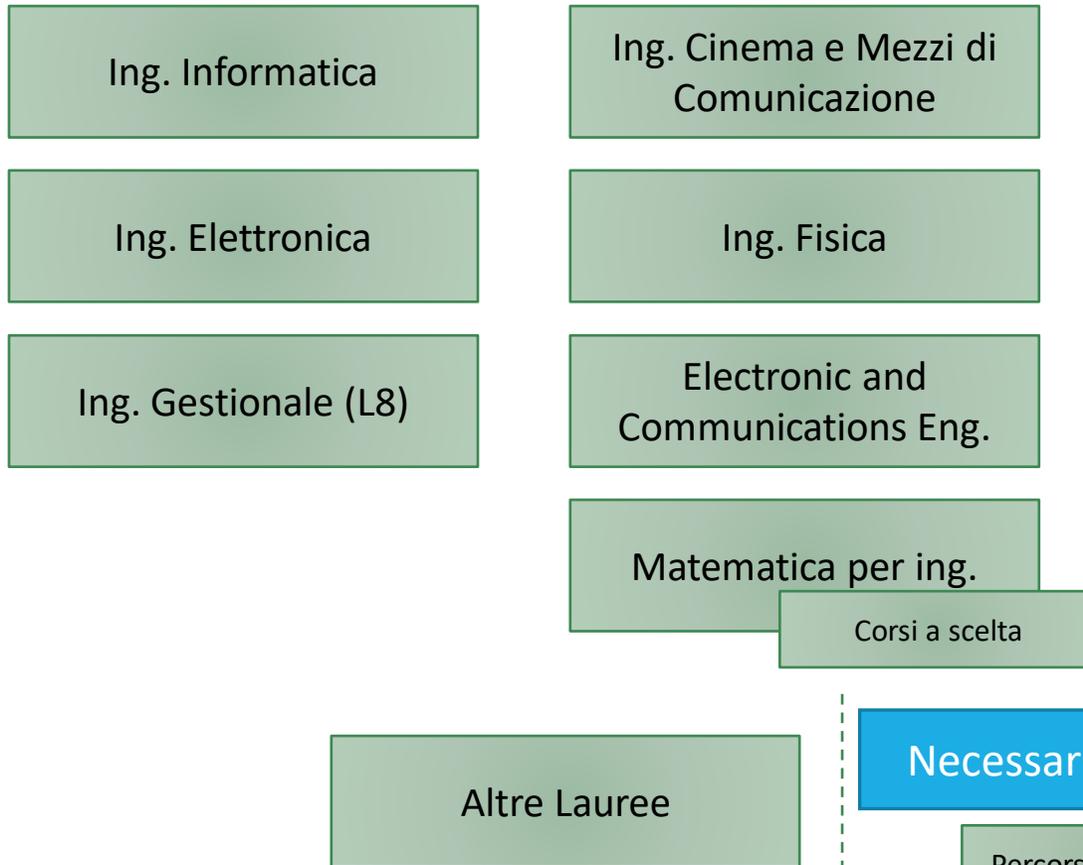


Dopo «Informatica»

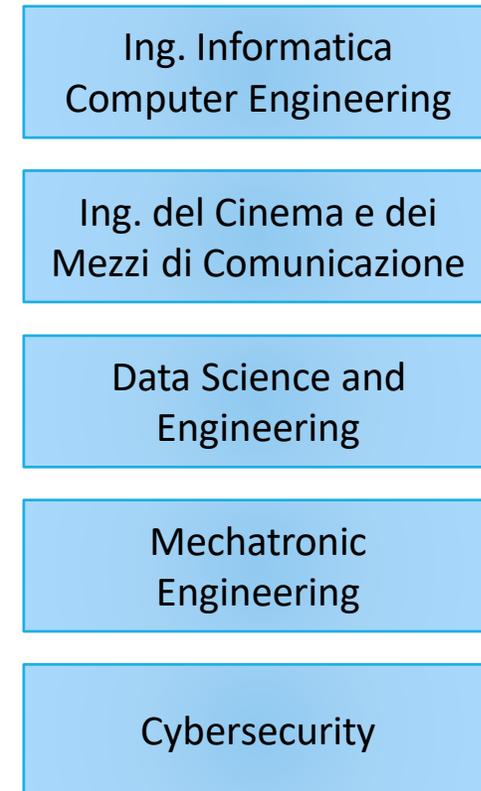


Dopo «Dopo «Informatica»»

Laurea



Laurea Magistrale



Uno sguardo a Python

VISIONE GENERALE DELL'ECOSISTEMA PYTHON

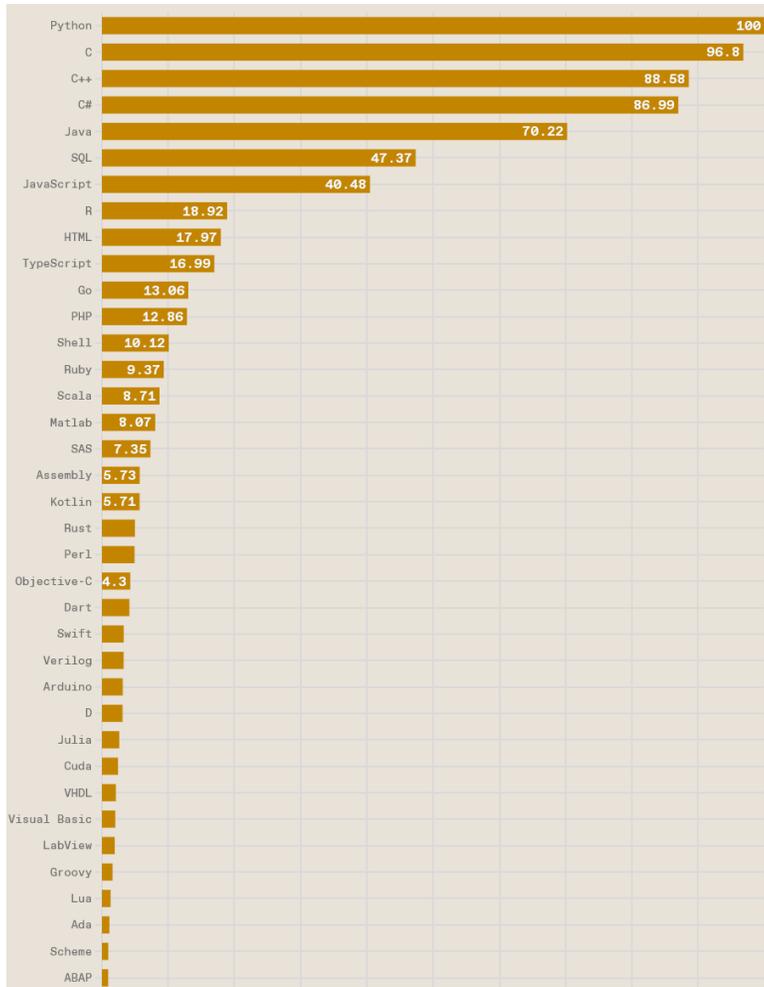
Il linguaggio Python



- 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



Diffusione del linguaggio Python

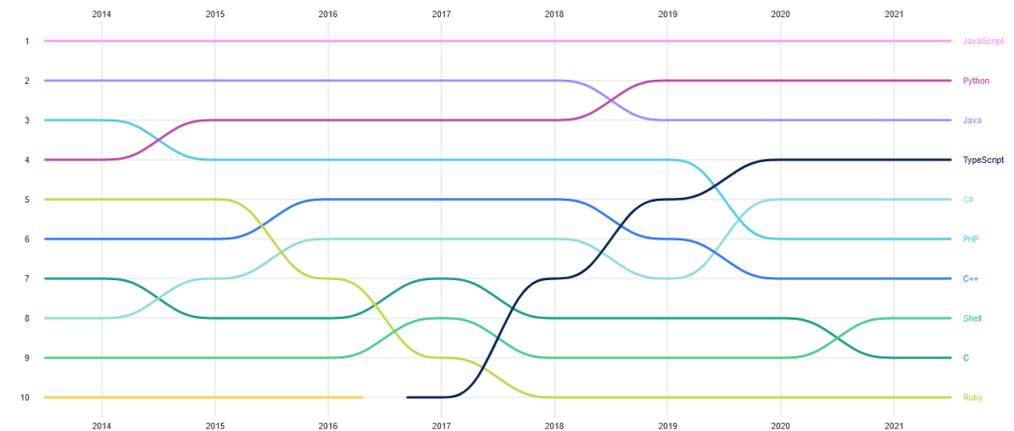


- IEEE Spectrum, 23 Aug 2022
<https://spectrum.ieee.org/top-programming-languages-2022>
 - Top Programming Languages 2022
- Altre statistiche, per i più curiosi:
 - <https://www.tiobe.com/tiobe-index/>
 - <http://pypl.github.io/PYPL.html>
 - <https://octoverse.github.com/>
 - ...

Diffusione del linguaggio Python

Aug 2022	Aug 2021	Change	Programming Language	Ratings	Change
1	2	▲	Python	15.42%	+3.56%
2	1	▼	C	14.59%	+2.03%
3	3		Java	12.40%	+1.96%
4	4		C++	10.17%	+2.81%
5	5		C#	5.59%	+0.45%
6	6		Visual Basic	4.99%	+0.33%
7	7		JavaScript	2.33%	-0.61%
8	9	▲	Assembly language	2.17%	+0.14%
9	10	▲	SQL	1.70%	+0.23%
10	8	▼	PHP	1.39%	-0.80%
11	16	▲▲	Swift	1.27%	+0.30%
12	12		Classic Visual Basic	1.27%	+0.04%
13	22	▲▲	Delphi/Object Pascal	1.22%	+0.60%
14	23	▲▲	Objective-C	1.22%	+0.61%
15	18	▲	Go	0.98%	+0.08%
16	14	▼	R	0.92%	-0.13%
17	17		MATLAB	0.90%	-0.08%
18	15	▼	Ruby	0.82%	-0.18%
19	13	▼	Fortran	0.81%	-0.32%
20	20		Perl	0.72%	-0.06%

Top languages over the years



<https://www.tiobe.com/tiobe-index/>

<https://octoverse.github.com/>

<https://www.python.org/>

The screenshot shows the Python.org homepage with a dark blue header and navigation menu. The main content area features a code editor with a Fibonacci function, a 'Functions Defined' section, and a central message about Python's capabilities. Below this are four columns of quick links: 'Get Started', 'Download', 'Docs', and 'Jobs'.

Python

PSF

Docs

PyPI

Jobs

Community

python™

Donate

Search

GO

Socialize

About

Downloads

Documentation

Community

Success Stories

News

Events

```
# Python 3: Fibonacci series up to n
>>> def fib(n):
>>>     a, b = 0, 1
>>>     while a < n:
>>>         print(a, end=' ')
>>>         a, b = b, a+b
>>>     print()
>>> fib(1000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
```

Functions Defined

The core of extensible programming is defining functions. Python allows mandatory and optional arguments, keyword arguments, and even arbitrary argument lists. [More about defining functions in Python 3](#)

1 2 3 4 5

Python is a programming language that lets you work quickly and integrate systems more effectively. >>> [Learn More](#)

Get Started

Whether you're new to programming or an experienced developer, it's easy to learn and use Python.

Start with our [Beginner's Guide](#)

Download

Python source code and installers are available for download for all versions!

Latest: [Python 3.8.5](#)

Docs

Documentation for Python's standard library, along with tutorials and guides, are available online.

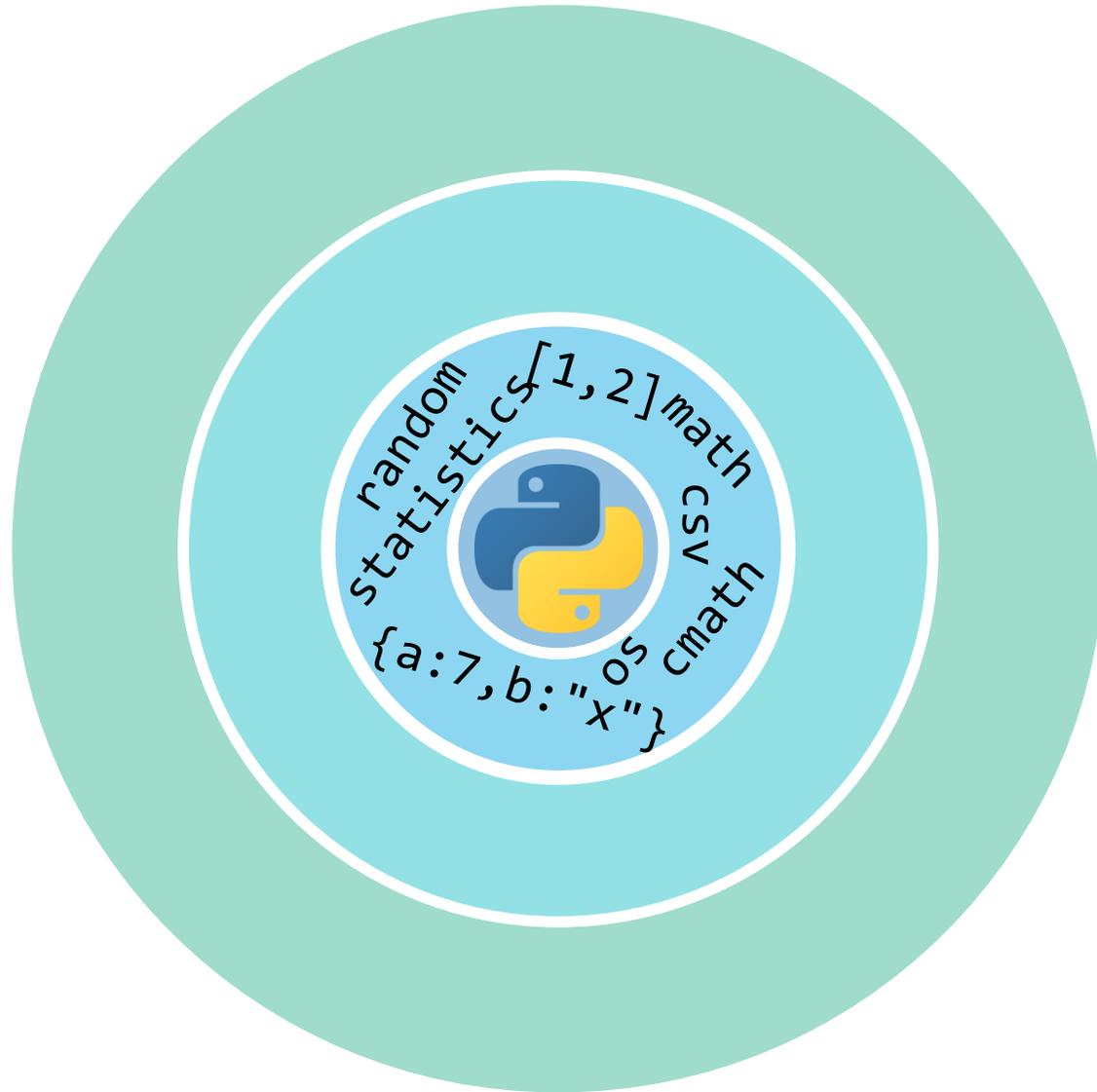
docs.python.org

Jobs

Looking for work or have a Python related position that you're trying to hire for? Our **relaunched community-run job board** is the place to go.

jobs.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	replib	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	imgchr	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



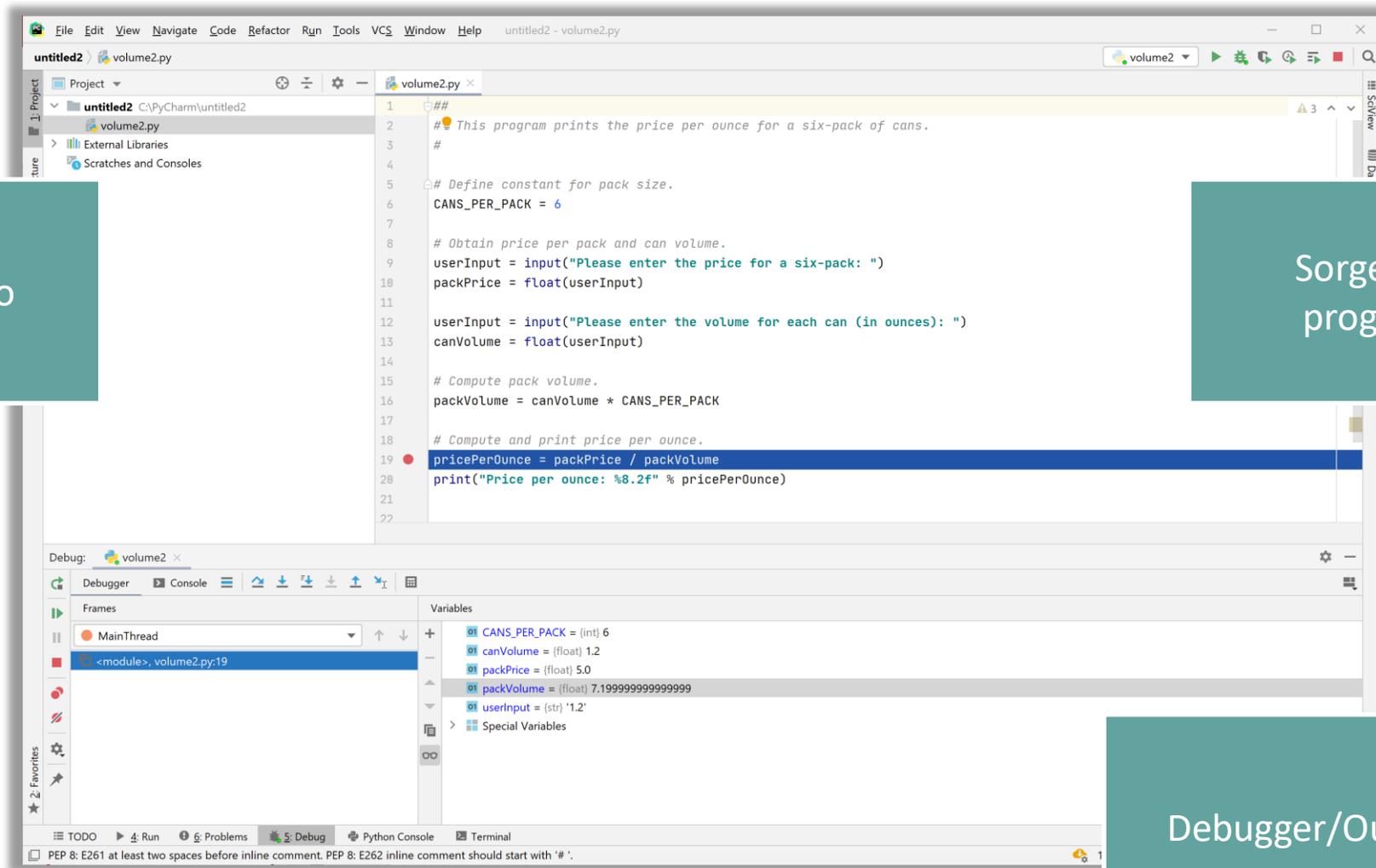
- Ambienti di sviluppo **tradizionali** (IDE)
 - IDLE, PyCharm, Visual Studio Code, Eclipse PyDev, ...
- Ambienti di sviluppo **on-line**
 - Repl.it, PythonAnywhere, Python Tutor
- Ambienti per il calcolo **interattivo**
 - Spyder, IPython
- **Notebook** Computazionali
 - Jupyter, JupyterLab, Google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

L'IDE di Visual Studio Code

The image displays the Visual Studio Code IDE interface with several key components highlighted by green boxes:

- File di progetto:** The Explorer sidebar on the left shows a project structure with folders for 'Settimana08' through 'Settimana14' and files like 'database.txt' and 'dati_aeroporto_torino.csv'.
- Sorgente del programma:** The main editor window shows the source code of a Python script named 'soluzione.py'. The code includes a list of coordinates and a function 'leggi_dati' that reads data from a file.
- Debugger:** The 'VARIABLES' pane on the left shows the current state of the program, including 'special variables', 'course: []', 'fermate: []', and 'nome_file: 'dati_aeroporto_torino.csv''.
- Errori:** The 'PROBLEMS' pane at the bottom shows an exception: 'FileNotFoundError: [Errno 2] No such file or directory: 'dati_aeroporto_torino.csv''. The error message is repeated in the 'TERMINAL' pane.
- Output:** The 'TERMINAL' pane at the bottom shows the command prompt output, including the PowerShell prompt and the execution of the Python script.

L'IDE di PyCharm



File di progetto

Sorgente del programma

Debugger/Output

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

The screenshot displays the Replit online IDE interface. At the top, the user profile is '@anonymous / GentleGiftedGeneric' with a Python logo, a dropdown menu, a refresh icon, and a green play button. On the right, there are 'talk' and 'Sign up' buttons. The main workspace is divided into three sections: a file explorer on the left, a code editor in the center, and a terminal on the right. The file explorer shows a file named 'main.py'. The code editor contains the following Python code:

```
1 # My first Python program
2 print("Hello, world")
```

The terminal on the right shows the output of the program: 'Hello, world'. Three blue callout boxes with arrows point to specific elements: 'File di progetto' points to the 'main.py' file in the file explorer; 'Sorgente del programma' points to the code in the editor; and 'Output del programma e console interattiva' points to the terminal output. A green rounded rectangle at the bottom contains the text: 'Opzione fantastica per esempi rapidi, per testare frammenti di programma, per evitare di creare un intero Progetto per una piccola parte di codice,, ...'.

Ambienti scientifici interattivi

SPYDER

4 From the SciPy Cookbook
5 """
6
7 from numpy import arange, cos, linspace, pi, sin, random
8 from scipy.interpolate import splprep, splev
9
10 # make ascending spiral in 3-space
11 t=linspace(0,1.75*2*pi,100)
12
13 x = sin(t)
14 y = cos(t)
15 z = t
16
17 #%% add noise
18 x+= random.normal(scale=0.1, size=x.shape)
19 y+= random.normal(scale=0.1, size=y.shape)
20 z+= random.normal(scale=0.1, size=z.shape)
21
22 #%% spline parameters
23 s=3.0 # smoothness parameter
24 k=2 # spline order
25 nest=-1 # estimate of number of knots needed (-1 = maximal,
26
27 #%% find the knot points
28 tckp,u = splprep([x,y,z],s=s,k=k,nest=-1)
29
30 #%% evaluate spline, including interpolated points
31 xnew,ynew,znew = splev(linspace(0,1,400),tckp)
32
33 import pylab

Object Inspector
Source Console Object numpy.mean
mean
Definition: mean(a, axis=None, dtype=None, out=None, keepdims=False)
Type: Function of numpy.core.fromnumeric module
Compute the arithmetic mean along the specified axis.
Returns the average of the array elements. The average is

IPython console
Python 3.4.0 on linux -- IPython 4.0.0
In [1]: runfile('/tmp/interpolation.py', wdir='/tmp')

JUPYTERLAB (ANCHE ON-LINE), GOOGLE COLAB

In this Notebook we explore the Lorenz system of differential equations:
$$\begin{aligned} \dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy \end{aligned}$$

Let's call the function once to view the solutions. For this set of parameters, we see the trajectories swirling around two points, called attractors.
In [4]: `from lorenz import solve_lorenz`
`t, x_t = solve_lorenz(N=10)`

Output View
sigma 10.00
beta 2.67
rho 28.00

```
9 def solve_lorenz(N=10, max_time=4.0, sigma=10.0, beta=8./3, rho=28.0):  
10     """Plot a solution to the Lorenz differential equations."""  
11     fig = plt.figure()  
12     ax = fig.add_axes([0, 0, 1, 1], projection='3d')  
13     ax.axis('off')  
14  
15     # prepare the axes limits  
16     ax.set_xlim((-25, 25))  
17     ax.set_ylim((-35, 35))  
18     ax.set_zlim((5, 55))  
19  
20     def lorenz_deriv(x,y,z, t0, sigma=sigma, beta=beta, rho=rho):  
21         """Compute the time-derivative of a Lorenz system."""  
22         x, y, z = x,y,z  
23         return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]  
24  
25     # Choose random starting points, uniformly distributed from -15 to 15  
26     np.random.seed(1)  
27     x0 = -15 + 30 * np.random.random((N, 3))  
28
```

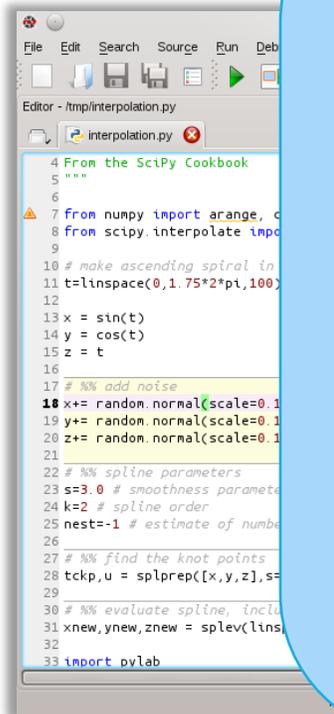
Ambienti scientifici interattivi

SPYDER

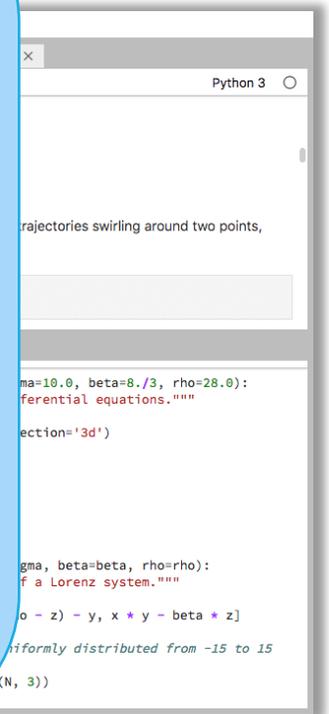
JUPYTERLAB (ANCHE ON-LINE), GOOGLE COLAB

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

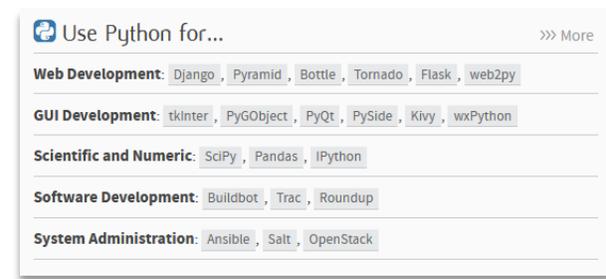


```
4 From the SciPy Cookbook
5 """
6
7 from numpy import arange, cos, sin, linspace
8 from scipy.interpolate import splprep, splev
9
10 # make ascending spiral in the xy plane
11 t=linspace(0,1.75*2*pi,100)
12
13 x = sin(t)
14 y = cos(t)
15 z = t
16
17 #%% add noise
18 x+= random.normal(scale=0.1, size=x.size)
19 y+= random.normal(scale=0.1, size=y.size)
20 z+= random.normal(scale=0.1, size=z.size)
21
22 #%% spline parameters
23 s=3.0 # smoothness parameter
24 k=2 # spline order
25 nest=-1 # estimate of number of knots
26
27 #%% find the knot points
28 tckp,u = splprep([x,y,z],s=s,k=k,nest=nest)
29
30 #%% evaluate spline, including noise
31 xnew,ynew,znew = splev(linspace(0,2*pi,100),tckp)
32
33 import pylab
```

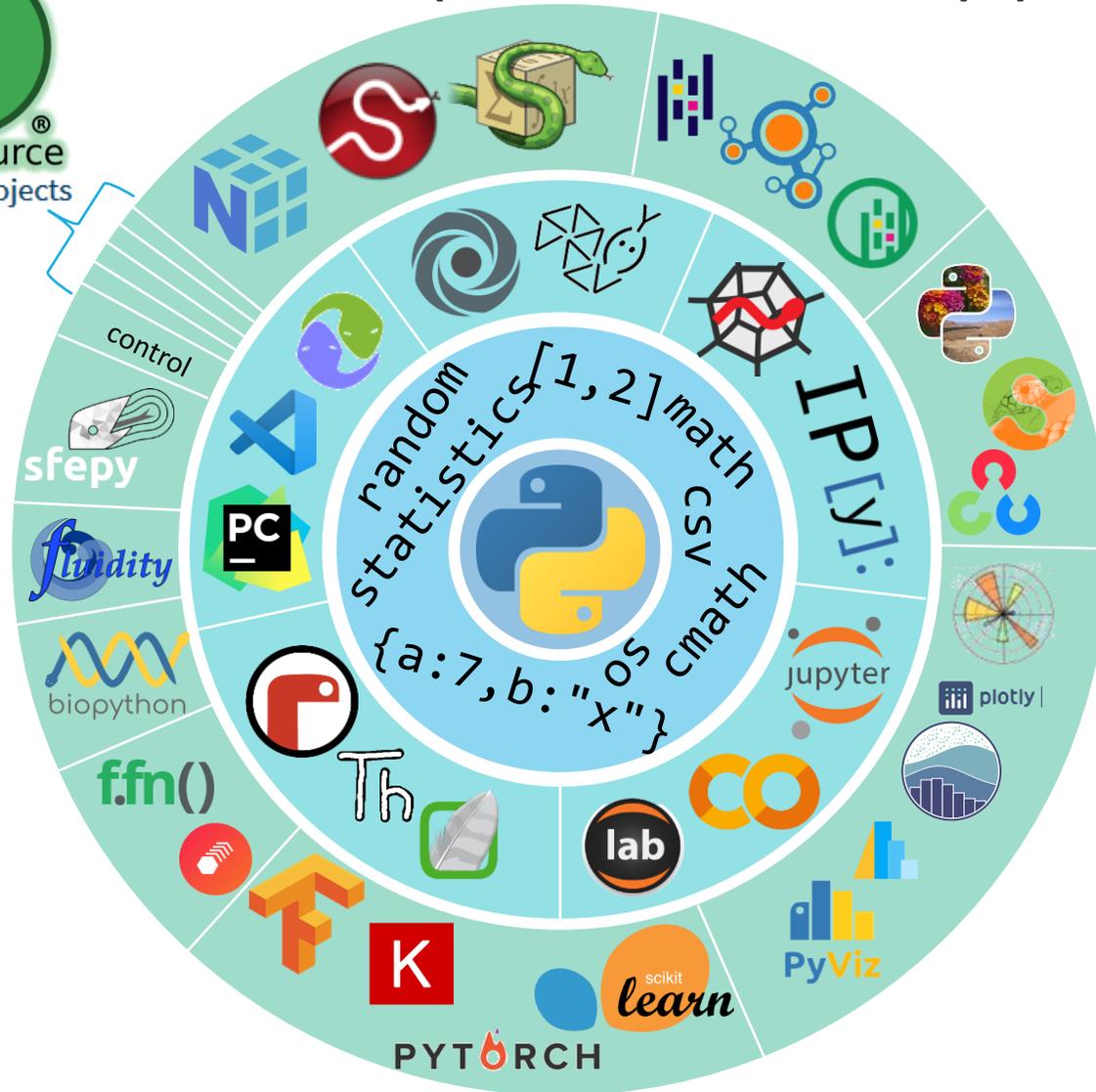


```
gamma, beta=beta, rho=rho):
    f a Lorenz system. """
    rho - z) - y, x * y - beta * z]
    uniformly distributed from -15 to 15
    ((N, 3))
```

Librerie per ambiti applicativi



open source
304,797 projects



- 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



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

cluster

constants

fftpack

integrate

interpolate

io

linalg

ndimage

odr

optimize

signal

sparse

spatial

special

stats

Description

Clustering algorithms

Physical and mathematical constants

Fast Fourier Transform routines

Integration and ordinary differential equation solvers

Interpolation and smoothing splines

Input and Output

Linear algebra

N-dimensional image processing

Orthogonal distance regression

Optimization and root-finding routines

Signal processing

Sparse matrices and associated routines

Spatial data structures and algorithms

Special functions

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

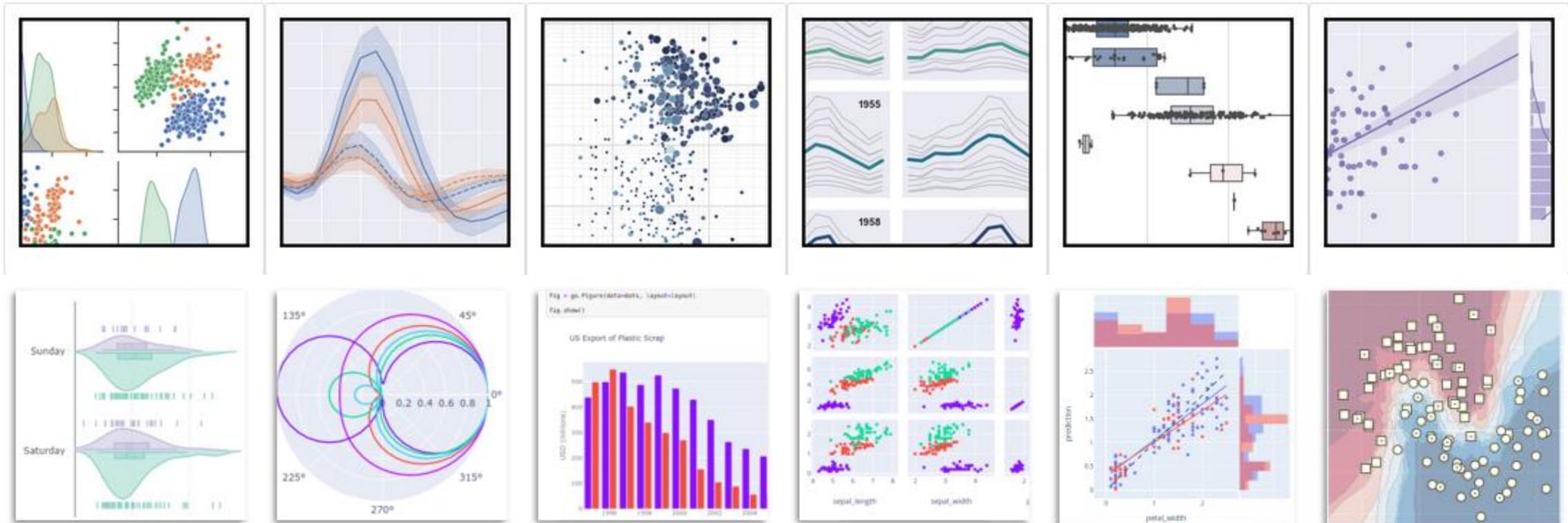
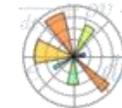
Features

- ✓ Core capabilities
- ✓ Polynomials
- ✓ Calculus
- ✓ Solving equations
- ✓ Combinatorics
- ✓ Discrete math
- ✓ Matrices
- ✓ Geometry
- ✓ Plotting
- ✓ Physics
- ✓ Statistics
- ✓ Cryptography
- ✓ Parsing
- ✓ Printing

Visualizzazione



plotly |



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)

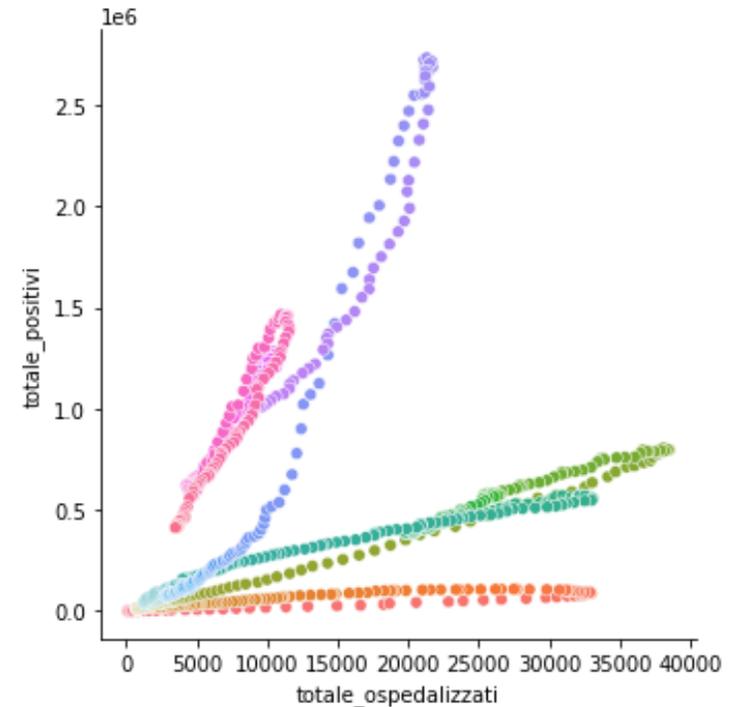
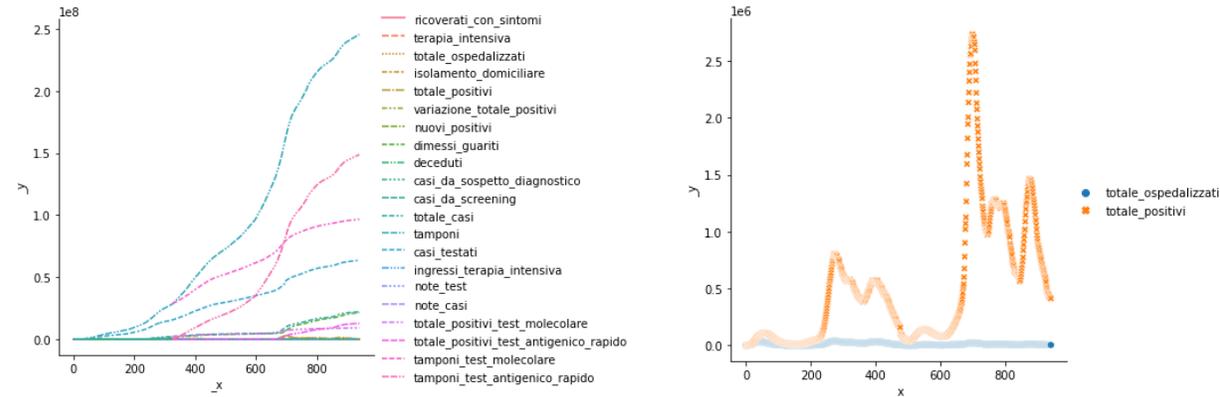
sns.relplot(data=covid, kind='line')

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



Organizzazione del corso

<http://bit.ly/polito-informatica>

Sito del corso

Tutto il materiale sarà disponibile su questo sito

- Slide
- Laboratori
- Esempi svolti
- Video Lezioni
- Temi d'esame
- Calendario lezioni
- ...

The screenshot shows the website for the Informatica (14BHD) course. The navigation menu includes e-Lite, News, People, Research, Teaching, and Thesis. The main content area features a sidebar with a list of course topics, including 'Informatica (14BHD)' which is currently selected. The main content area displays the course title 'Informatica (14BHD)' and a description: 'Pagina ufficiale del corso di Informatica (codice 14BHD, (DIQ - JZZ - Corso 3), anno accademico 2023/2024) presso il Politecnico di Torino.' Below this, there are six main content blocks: 'Introduzione' (Avvisi e informazioni di base), 'Materiale di studio' (Lucidi, videolezioni, esercizi, ...), 'Calendario delle lezioni' (Calendario delle lezioni e materiale utilizzato in ciascuna lezione), 'Laboratorio' (Testi e soluzioni delle esercitazioni di laboratorio), 'Esame' (Materiale per la preparazione all'esame), and a navigation bar with 'Previous < Exam' and 'Next Introduzione >' buttons.

Struttura del corso

- Programmazione e Python : 41 ore
- Teoria: 9 ore
- Problem solving : 12 ore
- Laboratorio : 18 ore (x 3 squadre)

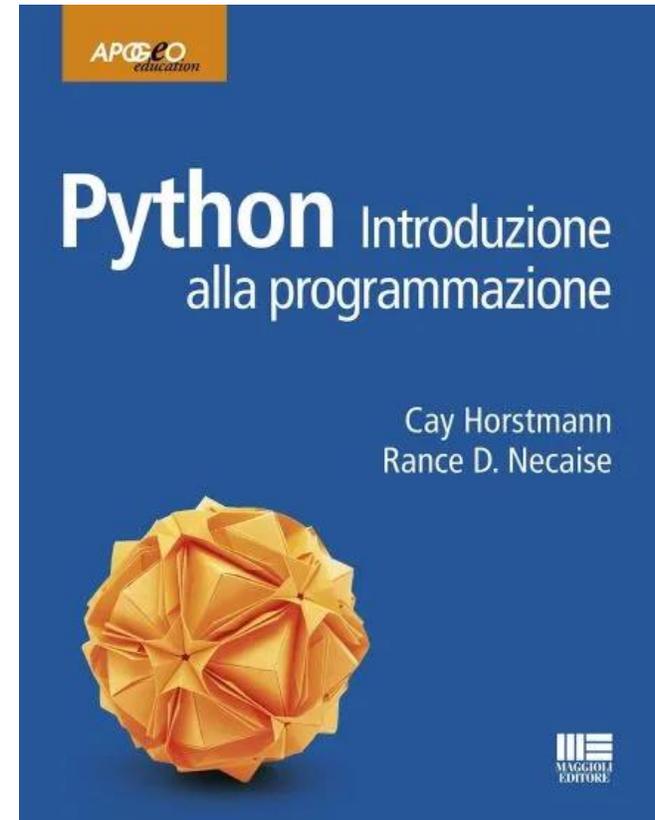
- Totale: 80 ore/studente

Libri di testo

TESTO COMPLETO IN ITALIANO



VERSIONE RIDOTTA (CAPITOLI 1-8)
CORRISPONDENTE AL PROGRAMMA SVOLTO



Libri di testo: Informazioni dettagliate

TESTO COMPLETO IN ITALIANO

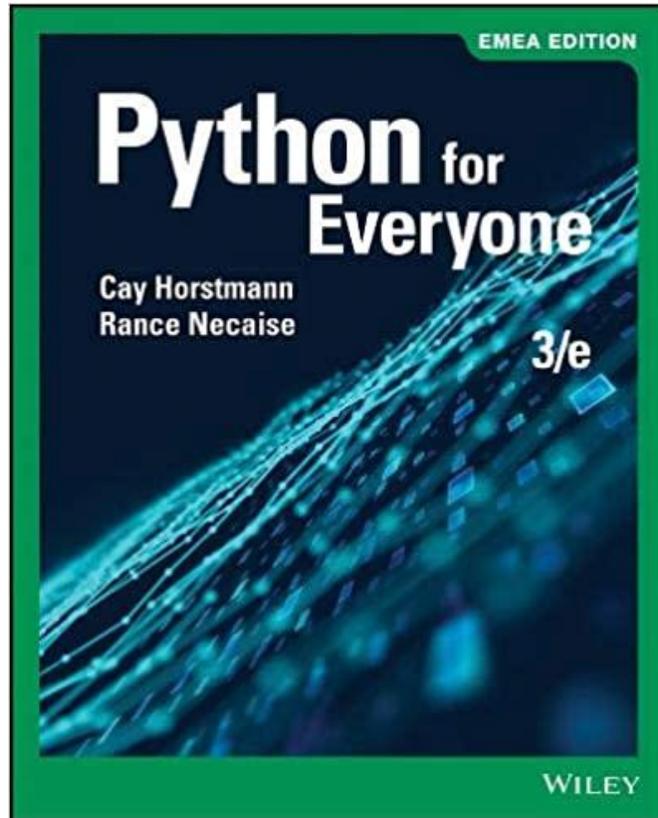
- Concetti di informatica e fondamenti di Python
- Seconda edizione - Giugno 2019 (II° Edizione)
- Cay Horstmann - Rance D. Necaie
- Maggioli Editore
- Giugno 2019
- ISBN 978-8891635433
- <https://www.maggiolieditore.it/concetti-di-informatica-e-fondamenti-di-python.html>

VERSIONE RIDOTTA (CAPITOLI 1-8) CORRISPONDENTE AL PROGRAMMA SVOLTO

- Python - Introduzione alla programmazione
- Versione ridotta della Seconda Edizione
- Cay S. Horstmann, Rance D. Necaie
- Maggioli Editore
- Luglio 2023
- ISBN: 978-8891663979
- <https://www.maggiolieditore.it/python-introduzione-alla-programmazione.html>

Libri di testo

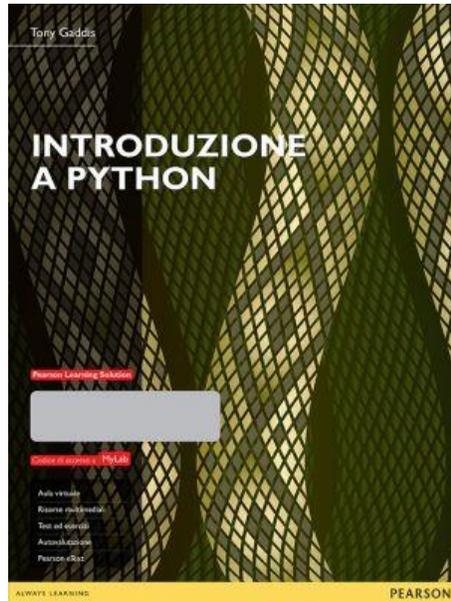
TESTO ORIGINALE IN LINGUA INGLESE



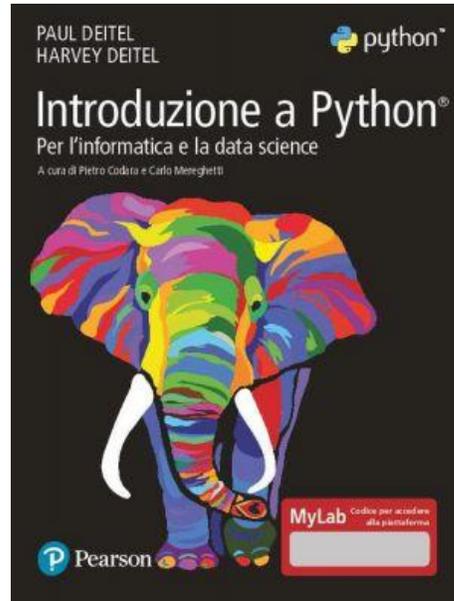
INFO DETTAGLIATE

- Python For Everyone
- 3rd Edition
- Cay S. Horstmann, Rance D. Neceise
- Wiley
- ISBN: 978-1-119-49853-7 December 2018
- <https://www.wiley.com/en-it/Python+For+Everyone,+3rd+Edition-p-9781119498537>

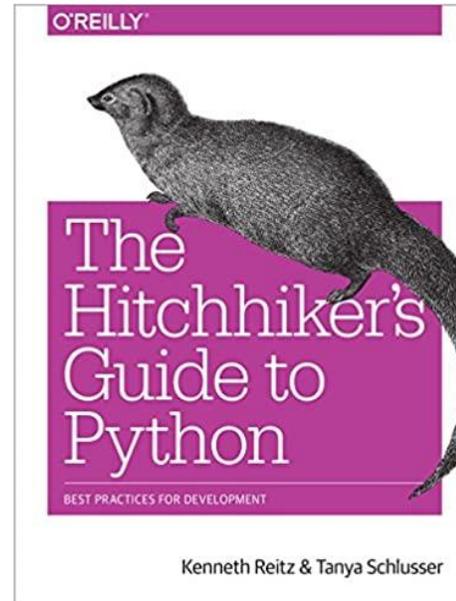
Altre risorse...



Introduzione a Python
Tony Gaddis
Pearson - ISBN: 9788891900999



Introduzione a Python per
l'informatica e la data science
Paul J. Deitel, Harvey M. Deitel,
Pietro Codara, Carlo Merghetti
Pearson - ISBN: 9788891915924



The Hitchhiker's Guide to Python:
Best Practices for Development
Kenneth Reitz, Tanya Schlusser
O'Reilly Media - ISBN-13: 978-
1491933176

Gratis su:

<https://docs.python-guide.org/>

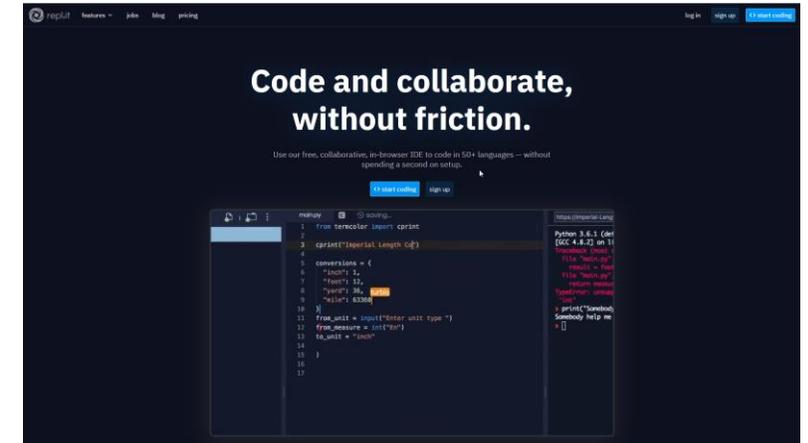
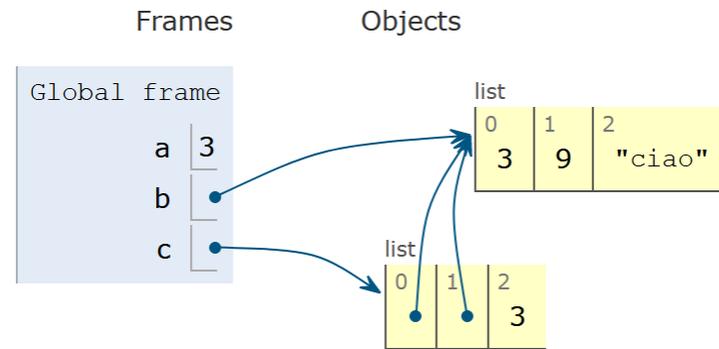


<https://www.python.org/>
<https://docs.python.org/3/>
<https://docs.python.org/3/tutorial/>



<https://realpython.com/>

Strumenti per programmare



<https://www.jetbrains.com/pycharm/>

- Professional (gratis per docenti e studenti)
- Community (gratis per tutti)

<http://pythontutor.com>

<https://replit.com>
(free online IDE)

Istruzioni di installazione

14BHDxx - Informatica

Installazione software (piattaforma Windows)

Sommario

FASE A: Installare l'interprete Python	2
Gli ambienti di sviluppo PyCharm	4
FASE B: Installazione di PyCharm Edu (opzione consigliata)	4
FASE C: Attivazione di PyCharm Edu	7
FASE D: Creazione di un nuovo progetto in PyCharm Edu	8
In alternativa (opzione avanzata): PyCharm Community o Professional	10
FASE B: Installazione di PyCharm Community o Professional	10
FASE C: Attivazione di PyCharm Community/Professional e della relativa licenza d'uso	12
FASE D: Creazione di un nuovo progetto in PyCharm Community/Professional	14

12BHDxx - Informatica

Installazione software (piattaforma macOS)

Sommario

FASE A: Installare l'interprete Python	2
FASE B: Installare l'ambiente di sviluppo PyCharm	4
FASE C: Attivazione di PyCharm e della relativa licenza d'uso	5
FASE D: Creazione di un nuovo progetto in PyCharm	8

<http://bit.ly/polito-informatica>

File Edit Selection View Go Run Terminal Help

nomi_frequenti_studenti.py x {} launch.json 1, U

Settimana01 > nomi_frequenti_studenti.py > main

```

47 nomi = estrai_nomi(stud)
48 print(f"Nella classe ci sono {len(stud)} studenti")
49 freq = frequenze(nomi)
50 max_freq = max_frequenza(freq)
51 print(f"Il nome più frequente compare {max_freq} volte")
52 nomi_max = nomi_piu_frequenti(freq, max_freq)
53 print(f"Si tratta di : {' , '.join(nomi_max)}")
54 # estrai solo i nomi che compaiono almeno 3 volte
55 freq2 = {k: v for (k, v) in freq.items() if v >= 3}
56 print(
57     f"I nomi che compaiono almeno 3 volte sono {' , '.join(sorted(list(freq2.keys())))}."
58 )
59
60 hist = list(sorted(list(freq2.items()), key=operator.itemgetter(1), reverse=True))
61 pyplot.barh(y=[h[0] for h in hist], width=[h[1] for h in hist])
62 pyplot.show()
63
64
65 main()
66

```

VARIABLES

Locals

- freq: {'DANIELE': 4, 'MATILDE': 3, 'DOAA...'}
 - special variables
 - function variables
 - 'DANIELE': 4
 - 'MATILDE': 3
 - 'DOAA': 1
 - 'SARA': 3
 - 'LETIZIA': 2
 - 'NICOLO': 2
 - 'LEONARDO': 2
 - 'DILETTA': 1
 - 'SIMONE': 7
 - 'EDOARDO': 5
 - 'ANGELO': 1

WATCH

CALL STACK Paused on breakpoint

- main nomi_frequenti_studenti.py 60:1
- <module> nomi_frequenti_studenti.py 65:1

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL

```

ine 294, in _get_code_from_file
code = compile(f.read(), fname, 'exec')
File "C:\Data\teaching\Info-2022\Settimane\.vscode\launch.json", line 9
"purpose": ["debug-in-terminal"]
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
SyntaxError: invalid syntax. Perhaps you forgot a comma?
(venv) PS C:\Data\teaching\Info-2022\Settimane\.vscode> C:; cd 'C:\Data\teaching\Info-2022\Settimane\Settimana01'; &
ttimane\venv\Scripts\python.exe 'c:\Users\Fulvio\.vscode\extensions\ms-python.python-2023.15.12301911\pythonFiles\lit
debugpy\launcher' '65381' '--' 'C:\Data\teaching\Info-2022\Settimane\Settimana01\nomi_frequenti_studenti.py'
Nella classe ci sono 280 studenti
Il nome più frequente compare 12 volte
Si tratta di : LORENZO
I nomi che compaiono almeno 3 volte sono ALBERTO, ALESSANDRO, ALESSIO, ANDREA, DANIELE, DAVIDE, EDOARDO, FEDERICO, FRA
IORGIA, GIOVANNI, GIULIA, JACOPO, LORENZO, LUCA, MARCO, MATILDE, MATTEO, MATTIA, PIETRO, SARA, SIMONE, SOFIA, STEFANO.
Backend TkAgg is interactive backend. Turning interactive mode on.

```

Visual Studio Code

Visual Studio Code

Version: 1.81.1 (user setup)
Commit: 6c3e3dba23e8fad360aed75ce363ba185c49794
Date: 2023-08-09T22:22:42.175Z (1 wk ago)
Electron: 22.3.18
ElectronBuildId: 22689846
Chromium: 108.0.5359.215
Node.js: 16.17.1
V8: 10.8.168.25-electron.0
OS: Windows_NT x64 10.0.22621

Copy OK

Files

- main.py
- 01TXYOV_2020.csv

```

main.py
1 import csv
2 # from matplotlib import pyplot
3
4 FILENAME = '01TXYOV_2020.csv'
5
6 # Leggi l'elenco degli studenti e salvalo in un array
7 def leggi(nomefile):
8     file = open(nomefile, 'r')
9     reader = csv.reader(file)
10    prima = True
11    studenti = []
12    for line in reader:
13        if prima: #skip first line (headers)
14            prima = False
15        else:
16            studenti.append(line)
17    file.close()
18    return studenti
19
20 # estrai i nomi di battesimo da un elenco di studenti
21 def nomi(elenco):
22     nomi = []
23     for riga in elenco:
24         nomi.append(riga[2])
25     return nomi
26
27 # Calcola le frequenze dei vari nomi presenti in un
    array
28 def frequenze(tokens):
29     freq = {}
30     for token in tokens:
31         if token in freq:
32             freq[token]= freq[token]+1
33         else:
34             freq[token] = 1

```

https://NomiFrequentiStudenti.fulcorno.repl.run

```

Nella classe ci sono 180 studenti
Il nome più frequente compare 9 volte
Si tratta di : ['ALESSANDRO']
I nomi che compaiono più di una volta sono ALESSANDRO, ANDREA
, CLAUDIO, DAVIDE, ENRICO, ETTORE, FEDERICA, FEDERICO, FRANCE
SCA, FRANCESCO, GABRIELE, GIANLUCA, GIOVANNI, GIUSEPPE, LOREN
ZO, LUCA, MARCO, MARTINA, MATTEO, MATTIA, MICHELE, PIETRO.

```

<https://replit.com>
(free online IDE)

Informazioni pratiche

	Lunedì	Martedì	Mercoledì	Giovedì	Venerdì
08:30-10:00					
10:00-11:30					
11:30-13:00			Lezione / Esercitazione Aula 4	Laboratorio Squadra 1 Aula 3D	
13:00-14:30			Lezione / Esercitazione Aula 4	Laboratorio Squadra 2 Aula 3D	
14:30-16:00					
16:00-17:30			Laboratorio Squadra 3 Aula 3D		
17:30-19:00	Lezione / Esercitazione Aula 4				

Orario Settimanale

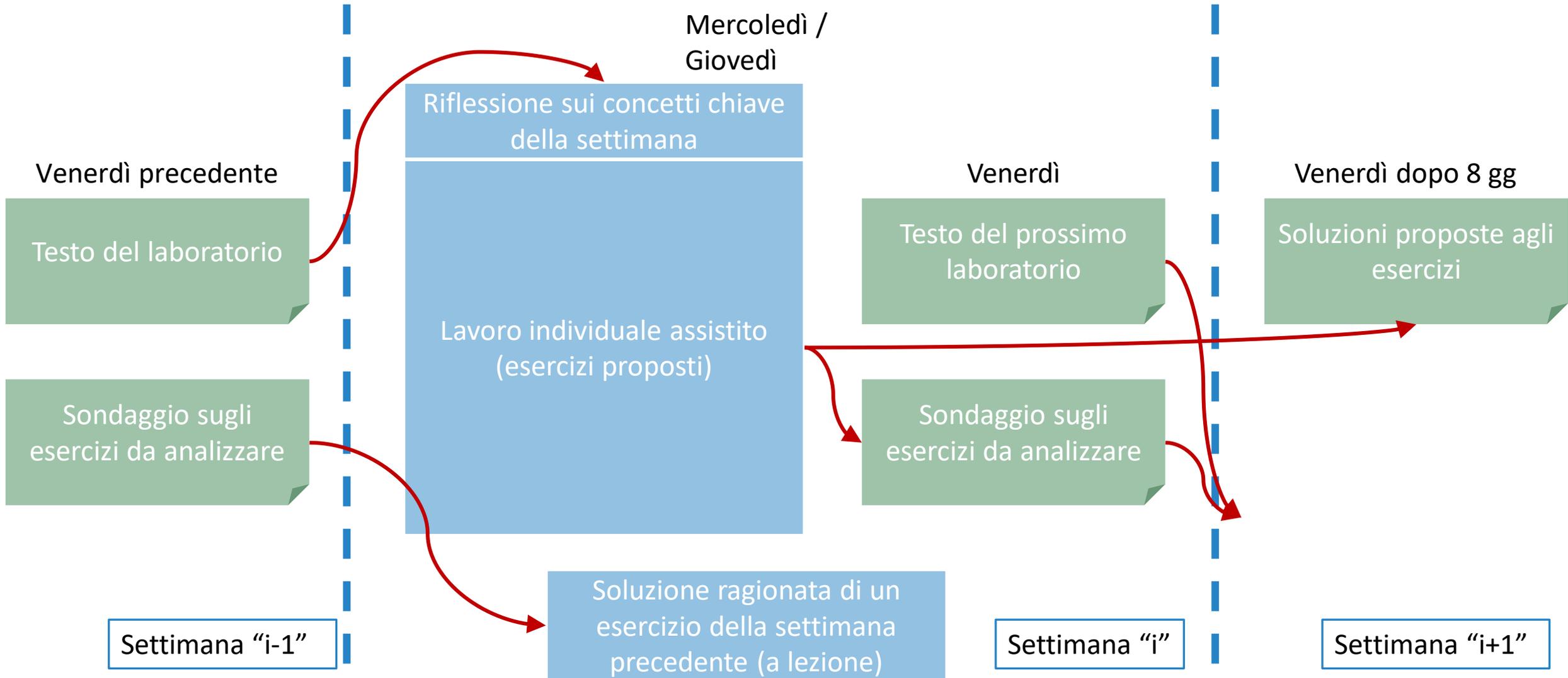
Laboratori

- La parte più importante del corso, in cui imparare a risolvere problemi e scrivere programmi
- Il corso è diviso in 3 squadre
 - Presso i Laboratori Informatici (LAIB)
- Inizio laboratori: **11-12/10/2023**
- Testo pubblicato sul sito del corso
- È necessario installare il software richiesto (*Python e Visual Studio Code*) - vedere le istruzioni di installazione sul portale

Suddivisione in squadre

- Squadra 1 (giovedì 11:30, Aula 3D): xxxxx
- Squadra 2 (giovedì 13:00, Aula 3D): xxxxx
- Squadra 3 (mercoledì 16:00, Aula 3D): xxxxx

Svolgimento dei laboratori



Comunicazioni

- **Tutti** i contatti con i docenti avverranno sulla piattaforma **Telegram**
 - Non inviare e-mail ai docenti, ma utilizzare il **gruppo Telegram**
 - Non inviare messaggi privati, se non richiesti dai docenti stessi
- **Iscriversi (obbligatorio!) all'indirizzo <https://t.me/politoinfo2023>**



Esame

Contenuti dell'esame

- 3 domande brevi sulla parte teorica del corso (6 punti)
- Un esercizio di programmazione (26 punti)
 - Con la possibilità di usare uno strumento di sviluppo per la scrittura del codice
 - Sarà consegnato il codice sorgente del programma sviluppato
 - Il codice verrà corretto manualmente (valuteremo la qualità della soluzione, e il rispetto delle richieste del problema, in maniera indipendente da eventuali errori sintattici e dall'effettivo funzionamento del programma)
- Nelle ultime 2 settimane del corso vi proporremo diversi esercizi di simulazione dell'esame, in modo da poter familiarizzare con le modalità di esame e con le conoscenze richieste per passarlo

Cosa serve per passare [bene] l'esame?

- Capacità logico-razionali di analisi e di sintesi
 - Comprendere i propri processi risolutivi e saperli formalizzare
- Svolgere tutti gli esercizi proposti
 - Davvero
 - Anche quelli [che sembrano] facili
 - Da soli
 - Su Personal Computer
 - Verificarli con dati diversi
 - Cercare di metterli in crisi
- Inventarsi nuovi problemi, o varianti di quelli proposti
 - E poi risolverli

Contatti

Docenti

- Fulvio Corno
 - Dipartimento di Automatica e Informatica (3° piano)
 - fulvio.corno@polito.it
- Roberta Bardini
 - Dipartimento di Automatica e Informatica (2° piano)
 - roberta.bardini@polito.it
- Lorenzo Martini
 - Dipartimento di Automatica e Informatica (2° piano)
 - lorenzo.martini@polito.it
- **Ci trovate su Telegram!**

Link utili

- Sito del corso (ufficiale):
 - <http://elite.polito.it/> → Teaching → Informatica (14BHD)
 - Link breve: <http://bit.ly/polito-informatica>
- Gruppo Telegram
 - <https://t.me/politoinfo2023>
- Materiale, laboratori, esercizi
 - <https://github.com/polito-informatica>



Licenza d'uso

- Queste diapositive sono distribuite con licenza Creative Commons “Attribuzione - Non commerciale - Condividi allo stesso modo 2.5 Italia (CC BY-NC-SA 2.5)”
- Sei libero:
 - di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire e recitare quest'opera
 - di modificare quest'opera
- Alle seguenti condizioni:
 - **Attribuzione** — Devi attribuire la paternità dell'opera agli autori originali e in modo tale da non suggerire che essi avallino te o il modo in cui tu usi l'opera.
 - **Non commerciale** — Non puoi usare quest'opera per fini commerciali.
 - **Condividi allo stesso modo** — Se alteri o trasformi quest'opera, o se la usi per crearne un'altra, puoi distribuire l'opera risultante solo con una licenza identica o equivalente a questa.
- <http://creativecommons.org/licenses/by-nc-sa/2.5/it/>

