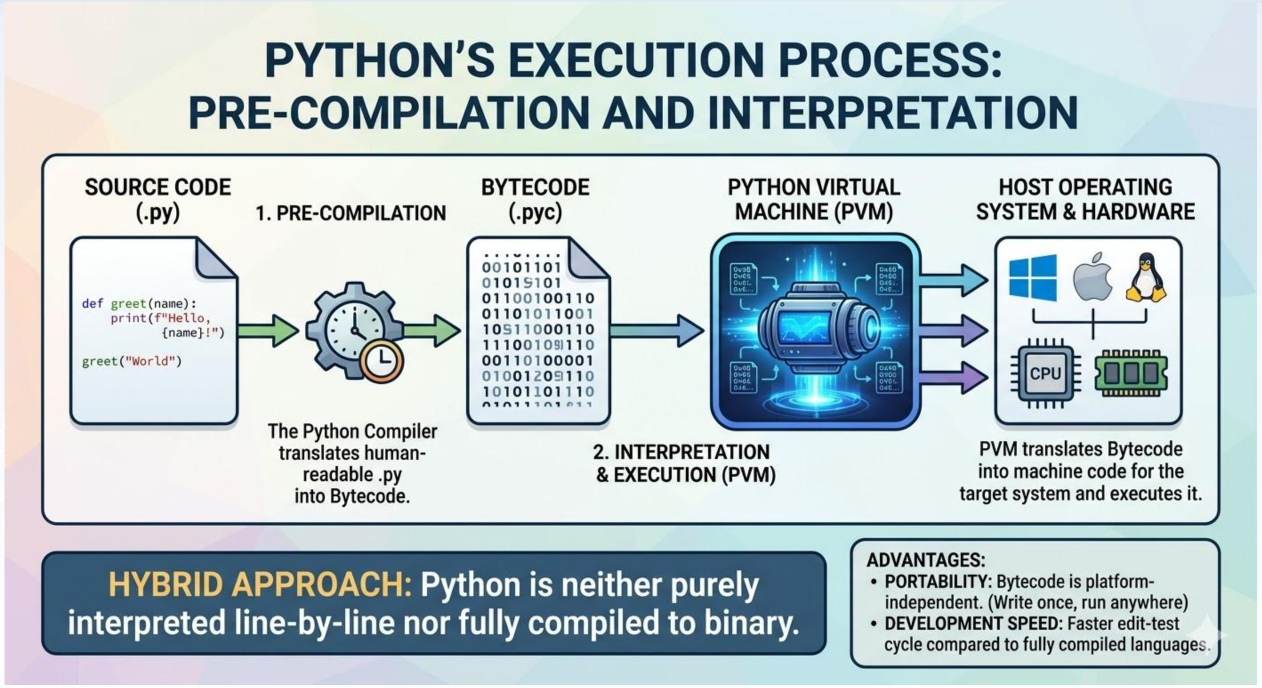
Corso Pratico Python

5. Ambiente di esecuzione
5.1. Intreprete, pre-compilatore,
5.2. Ambiente Variabili, convenzioni sulle chiamate
5.3. Environment, Pip, Conda
5.4. Spyder
5.5. Colab, Jupyter
5.6. Os, sys

Richiedente: Dott. Roberto Felici [CNR-ISM] • Insegnante: Dott. Scigè J. Liu [INAF-IAPS]

5. Ambiente di esecuzione
5.1. Intreprete, pre-compilatore

- **Linguaggio Ibrido:** Python non viene compilato direttamente in codice macchina (binario), né puramente interpretato riga per riga.
- **Fase 1: Pre-compilazione:** Il codice sorgente (.py) viene tradotto in **Bytecode** (.pyc). È un formato a basso livello indipendente dalla piattaforma.
- **Fase 2: Interprete (PVM):** La *Python Virtual Machine* legge il bytecode e lo esegue sul sistema operativo ospite.
- **Vantaggi:** Portabilità ("Write once, run anywhere") e velocità di sviluppo.


PYTHON'S EXECUTION PROCESS: PRE-COMPILATION AND INTERPRETATION
SOURCE CODE (.py)
1. PRE-COMPILATION
BYTECODE (.pyc)
PYTHON VIRTUAL MACHINE (PVM)
HOST OPERATING SYSTEM & HARDWARE
2. INTERPRETATION & EXECUTION (PVM)
CPU

```
def greet(name):
    print(f"Hello,
        {name}!")

greet("World")
```

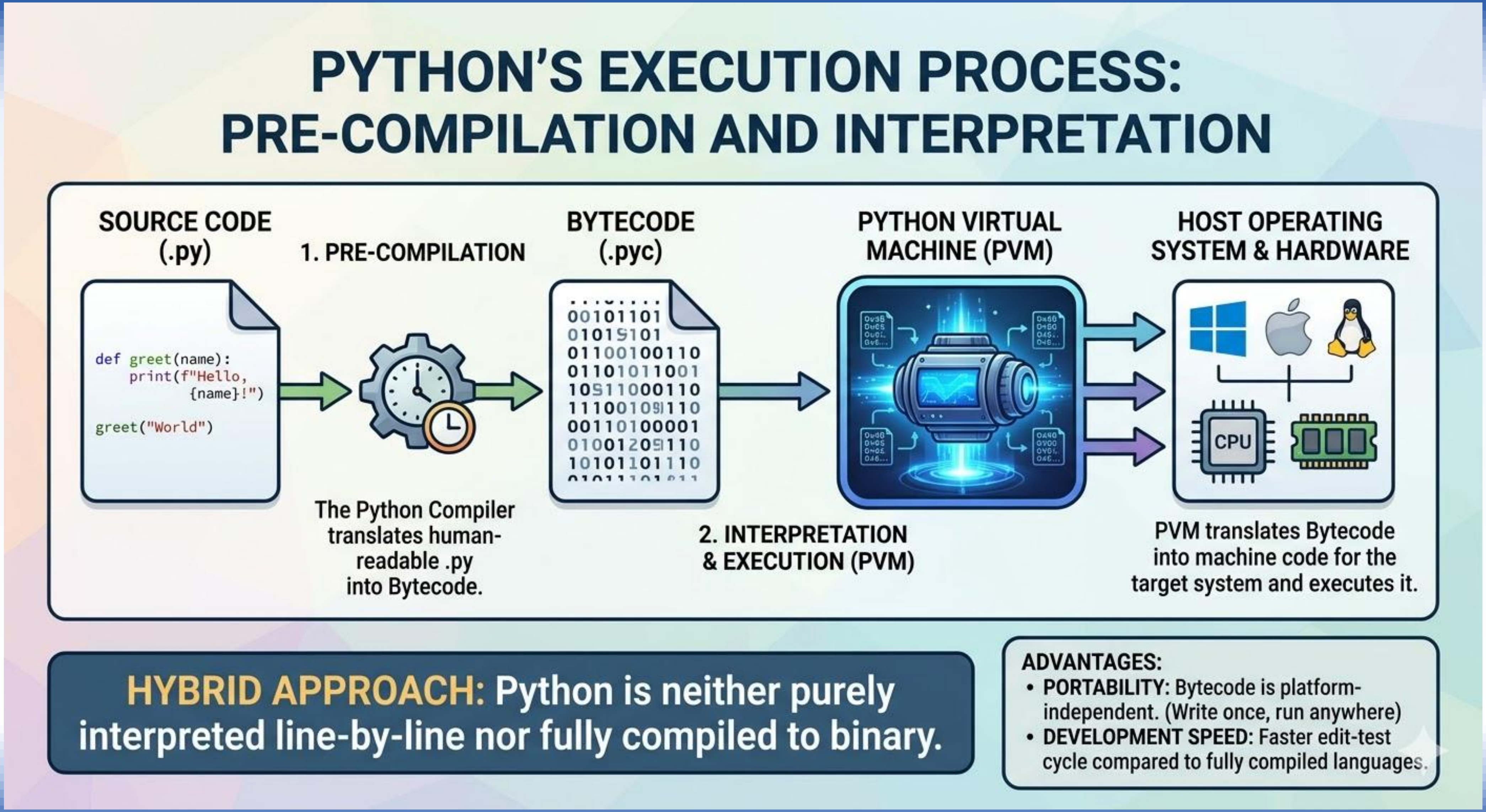
The Python Compiler translates human-readable .py into Bytecode.

PVM translates Bytecode into machine code for the target system and executes it.

HYBRID APPROACH: Python is neither purely interpreted line-by-line nor fully compiled to binary.

ADVANTAGES:
- PORTABILITY: Bytecode is platform-independent. (Write once, run anywhere)
- DEVELOPMENT SPEED: Faster edit-test cycle compared to fully compiled languages.

5. Ambiente di esecuzione



PYTHON'S EXECUTION PROCESS: PRE-COMPILATION AND INTERPRETATION
SOURCE CODE (.py)
1. PRE-COMPILATION
BYTECODE (.pyc)
PYTHON VIRTUAL MACHINE (PVM)
HOST OPERATING SYSTEM & HARDWARE
CPU

```
def greet(name):
    print(f"Hello,
          {name}!")

greet("World")
```

The Python Compiler translates human-readable .py into Bytecode.

2. INTERPRETATION & EXECUTION (PVM)

PVM translates Bytecode into machine code for the target system and executes it.

HYBRID APPROACH: Python is neither purely interpreted line-by-line nor fully compiled to binary.

ADVANTAGES:
- **PORTABILITY:** Bytecode is platform-independent. (Write once, run anywhere)
- **DEVELOPMENT SPEED:** Faster edit-test cycle compared to fully compiled languages.

5. Ambiente di esecuzione

5.2. Ambiente, Variabili, convenzioni sulle chiamate

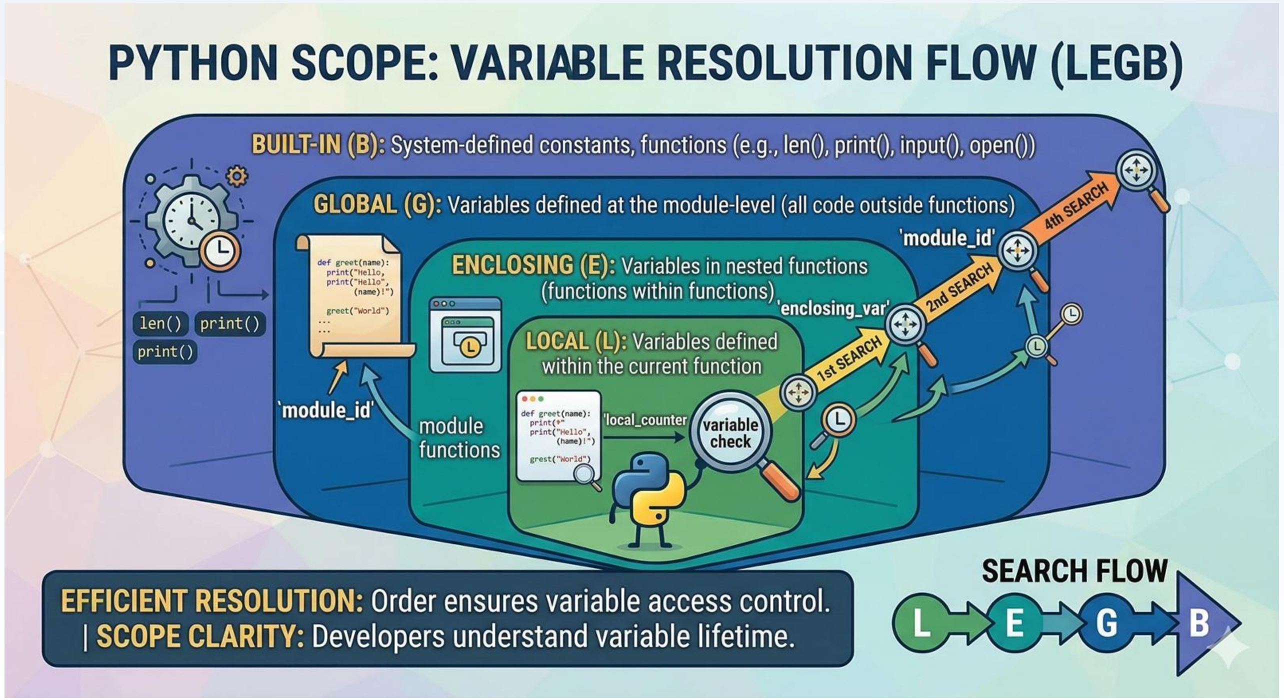
Ambiente Variabili e Visibilità

Namespace: Un sistema per garantire che i nomi siano univoci.

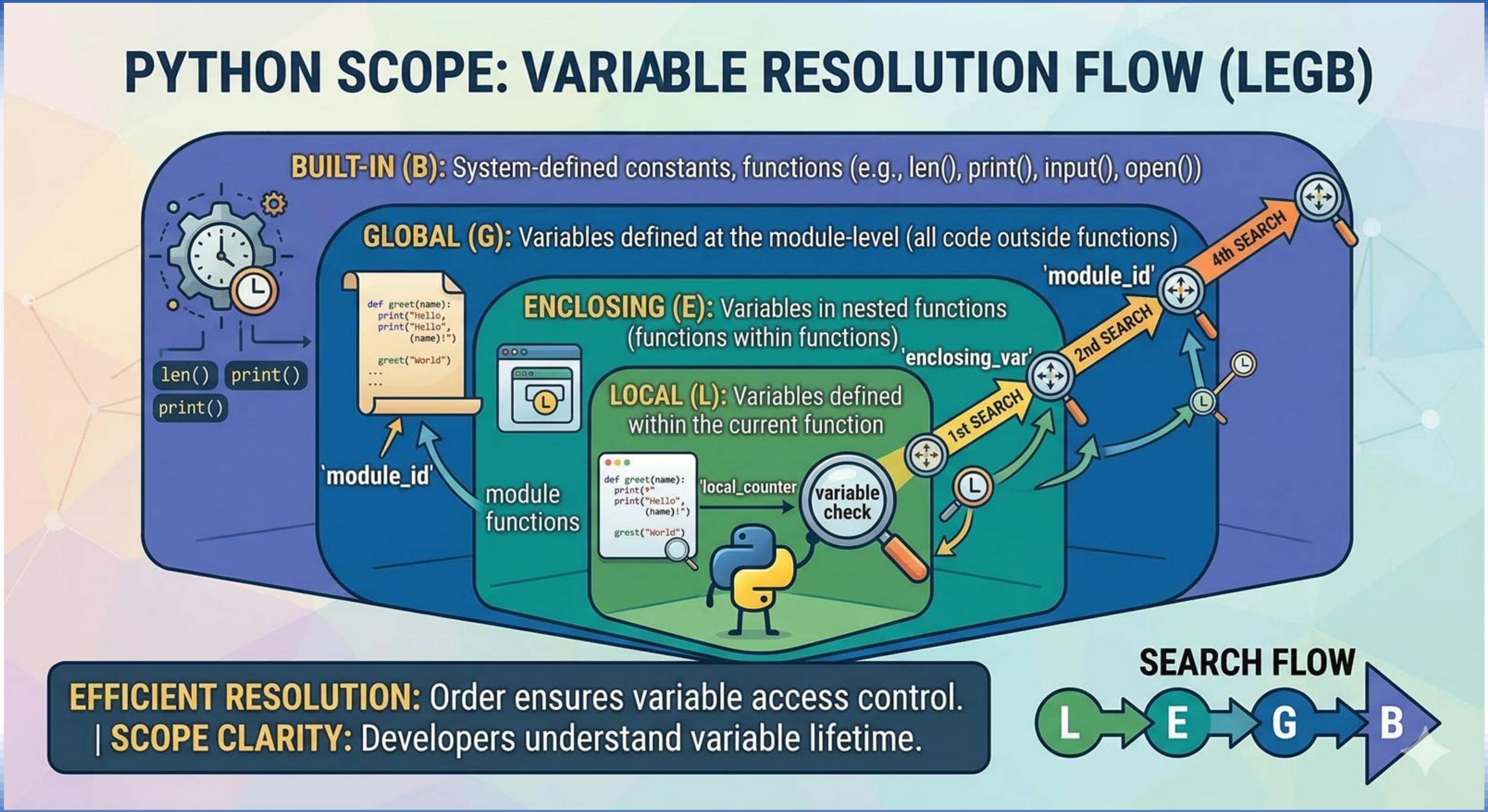
Python usa i **Dizionari** per gestire le variabili.

- **Regola LEGB:** L'ordine di ricerca delle variabili è:
  - **L**ocal: All'interno di una funzione.
  - **E**nclosing: In funzioni annidate.
  - **G**lobal: Variabili definite a livello di modulo.
  - **B**uilt-in: Funzioni standard di Python (es. print, len).

Lifetime: Le variabili locali "muoiono" al termine della funzione (Garbage Collection).


PYTHON SCOPE: VARIABLE RESOLUTION FLOW (LEGB)
BUILT-IN (B): System-defined constants, functions (e.g., len(), print(), input(), open())
GLOBAL (G): Variables defined at the module-level (all code outside functions)
'module_id'
ENCLOSING (E): Variables in nested functions (functions within functions)
'enclosing_var'
LOCAL (L): Variables defined within the current function
'local_counter'
variable check
module functions
`module_id'
len()
print()
print()
1st SEARCH
2nd SEARCH
4th SEARCH
EFFICIENT RESOLUTION: Order ensures variable access control.
| SCOPE CLARITY: Developers understand variable lifetime.
SEARCH FLOW
L
E
G
B

5. Ambiente di esecuzione



PYTHON SCOPE: VARIABLE RESOLUTION FLOW (LEGB)

BUILT-IN (B): System-defined constants, functions (e.g., len(), print(), input(), open())

GLOBAL (G): Variables defined at the module-level (all code outside functions)

ENCLOSING (E): Variables in nested functions (functions within functions)

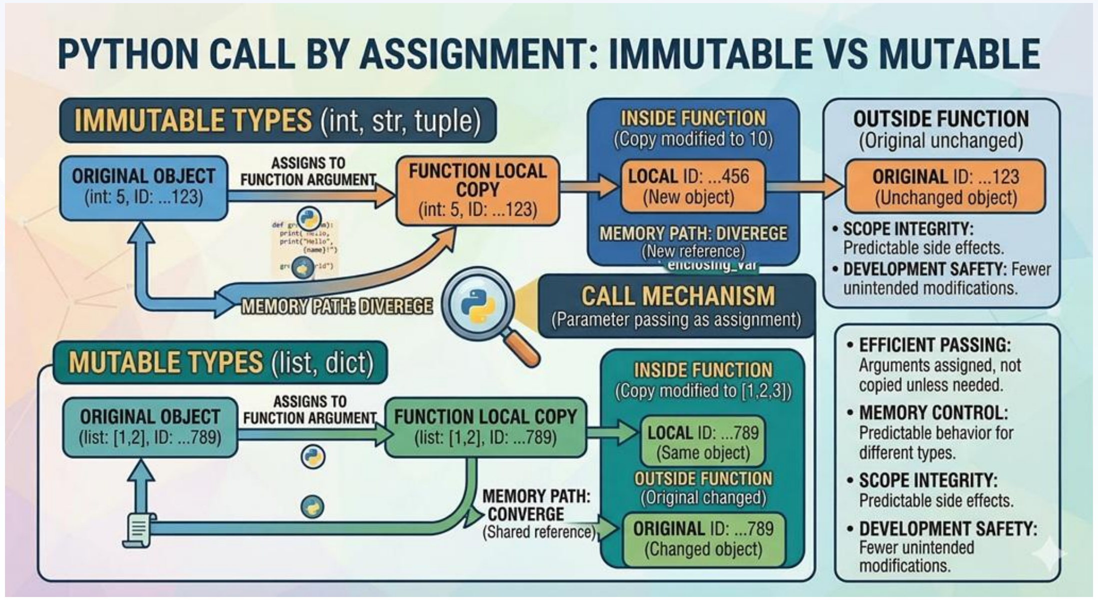
LOCAL (L): Variables defined within the current function

EFFICIENT RESOLUTION: Order ensures variable access control. | **SCOPE CLARITY:** Developers understand variable lifetime.

5. Ambiente di esecuzione
5.2. Ambiente, Variabili, convenzioni sulle chiamate

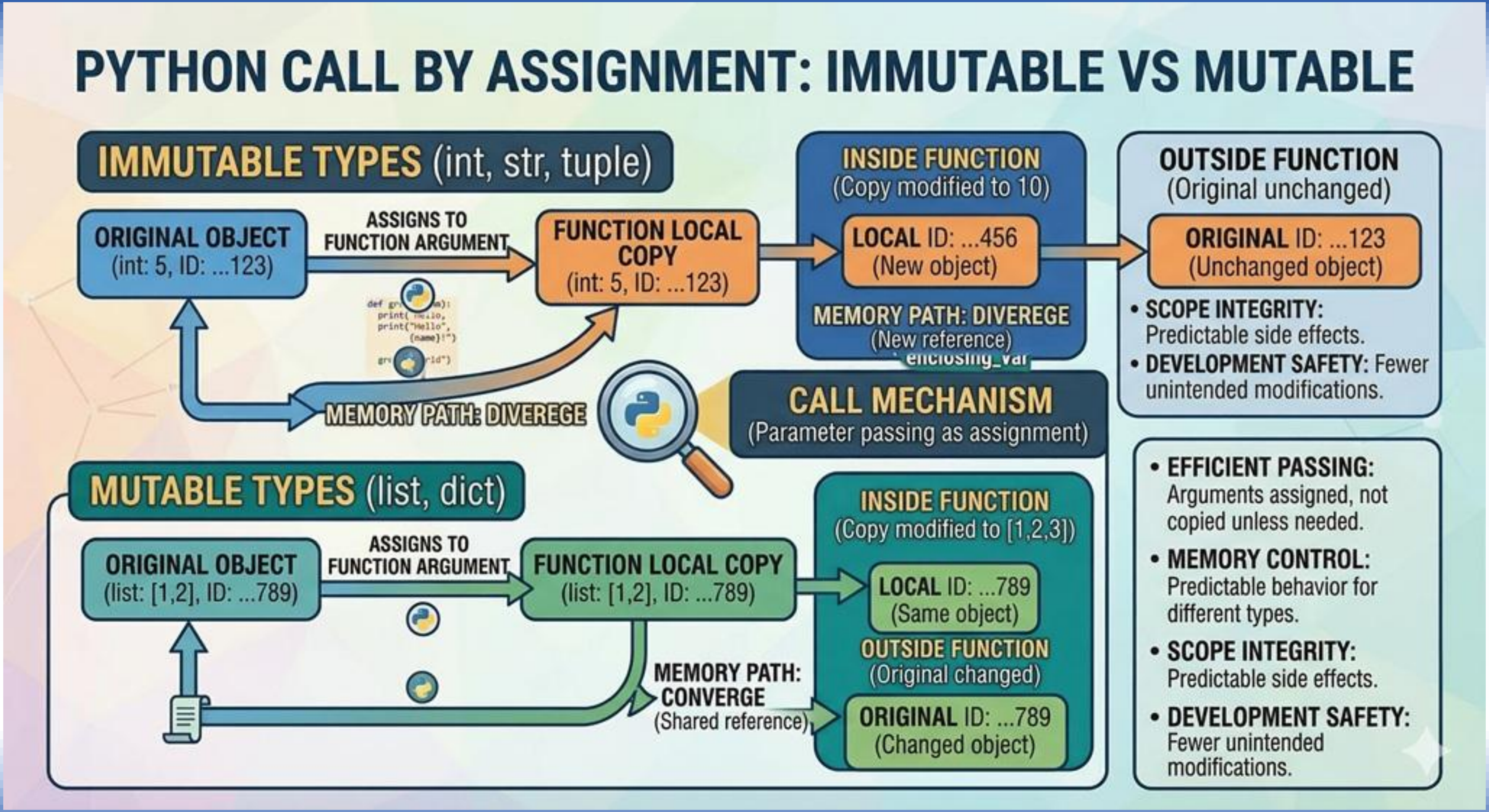
convenzioni sulle chiamate :
Passaggio dei Parametri

- **Call by Assignment:** In Python, tutto è un oggetto. Il passaggio non è nè "per valore" nè "per riferimento" in senso stretto.
- **Oggetti Immutabili (int, str, tuple):** Se modificati in una funzione, Python crea una nuova copia locale. L'originale non cambia.
- **Oggetti Mutabili (list, dict):** Se passati a una funzione, le modifiche interne si riflettono sull'oggetto originale.
- **Best Practice:** Attenzione a usare liste come valori di default negli argomenti!



PYTHON CALL BY ASSIGNMENT: IMMUTABLE VS MUTABLE
IMMUTABLE TYPES (int, str, tuple)
ORIGINAL OBJECT (int: 5, ID: ...123)
ASSIGNS TO FUNCTION ARGUMENT
FUNCTION LOCAL COPY (int: 5, ID: ...123)
INSIDE FUNCTION (Copy modified to 10)
LOCAL ID: ...456 (New object)
MEMORY PATH: DIVEREGE (New reference)
OUTSIDE FUNCTION (Original unchanged)
ORIGINAL ID: ...123 (Unchanged object)
• SCOPE INTEGRITY: Predictable side effects.
• DEVELOPMENT SAFETY: Fewer unintended modifications.
MEMORY PATH: DIVEREGE
CALL MECHANISM (Parameter passing as assignment)
MUTABLE TYPES (list, dict)
ORIGINAL OBJECT (list: [1,2], ID: ...789)
ASSIGNS TO FUNCTION ARGUMENT
FUNCTION LOCAL COPY (list: [1,2], ID: ...789)
INSIDE FUNCTION (Copy modified to [1,2,3])
LOCAL ID: ...789 (Same object)
OUTSIDE FUNCTION (Original changed)
ORIGINAL ID: ...789 (Changed object)
MEMORY PATH: CONVERGE (Shared reference)
• EFFICIENT PASSING: Arguments assigned, not copied unless needed.
• MEMORY CONTROL: Predictable behavior for different types.
• SCOPE INTEGRITY: Predictable side effects.
• DEVELOPMENT SAFETY: Fewer unintended modifications.

5. Ambiente di esecuzione


PYTHON CALL BY ASSIGNMENT: IMMUTABLE VS MUTABLE
IMMUTABLE TYPES (int, str, tuple)
ORIGINAL OBJECT (int: 5, ID: ...123)
ASSIGNS TO FUNCTION ARGUMENT
FUNCTION LOCAL COPY (int: 5, ID: ...123)
MEMORY PATH: DIVEREGE
INSIDE FUNCTION (Copy modified to 10)
LOCAL ID: ...456 (New object)
MEMORY PATH: DIVEREGE (New reference)
enclosing_var
OUTSIDE FUNCTION (Original unchanged)
ORIGINAL ID: ...123 (Unchanged object)
CALL MECHANISM (Parameter passing as assignment)
MUTABLE TYPES (list, dict)
ORIGINAL OBJECT (list: [1,2], ID: ...789)
ASSIGNS TO FUNCTION ARGUMENT
FUNCTION LOCAL COPY (list: [1,2], ID: ...789)
MEMORY PATH: CONVERGE (Shared reference)
INSIDE FUNCTION (Copy modified to [1,2,3])
LOCAL ID: ...789 (Same object)
OUTSIDE FUNCTION (Original changed)
ORIGINAL ID: ...789 (Changed object)

- **SCOPE INTEGRITY:** Predictable side effects.
- **DEVELOPMENT SAFETY:** Fewer unintended modifications.

- **EFFICIENT PASSING:** Arguments assigned, not copied unless needed.
- **MEMORY CONTROL:** Predictable behavior for different types.
- **SCOPE INTEGRITY:** Predictable side effects.
- **DEVELOPMENT SAFETY:** Fewer unintended modifications.

5. Ambiente di esecuzione
Gestione Pacchetti e Ambienti (5.3)

Ambiente visto da Python

↓↓↓

Ambiente visto dall’Utente

Pip, Conda e Virtual Environments

Virtual Environments (venv): Isolano le dipendenze di un progetto.

Evitano il "**dependency hell**" (conflitti tra versioni diverse).

Pip: Il package manager standard. Installazione atomica da PyPI.

- `pip install pandas`

Conda: Gestore cross-platform (Anaconda/Miniconda).

Gestisce pacchetti Python e librerie di sistema (C++, CUDA).

Requirements.txt: File per replicare l'ambiente:

`pip freeze > requirements.txt.`

5. Ambiente di esecuzione

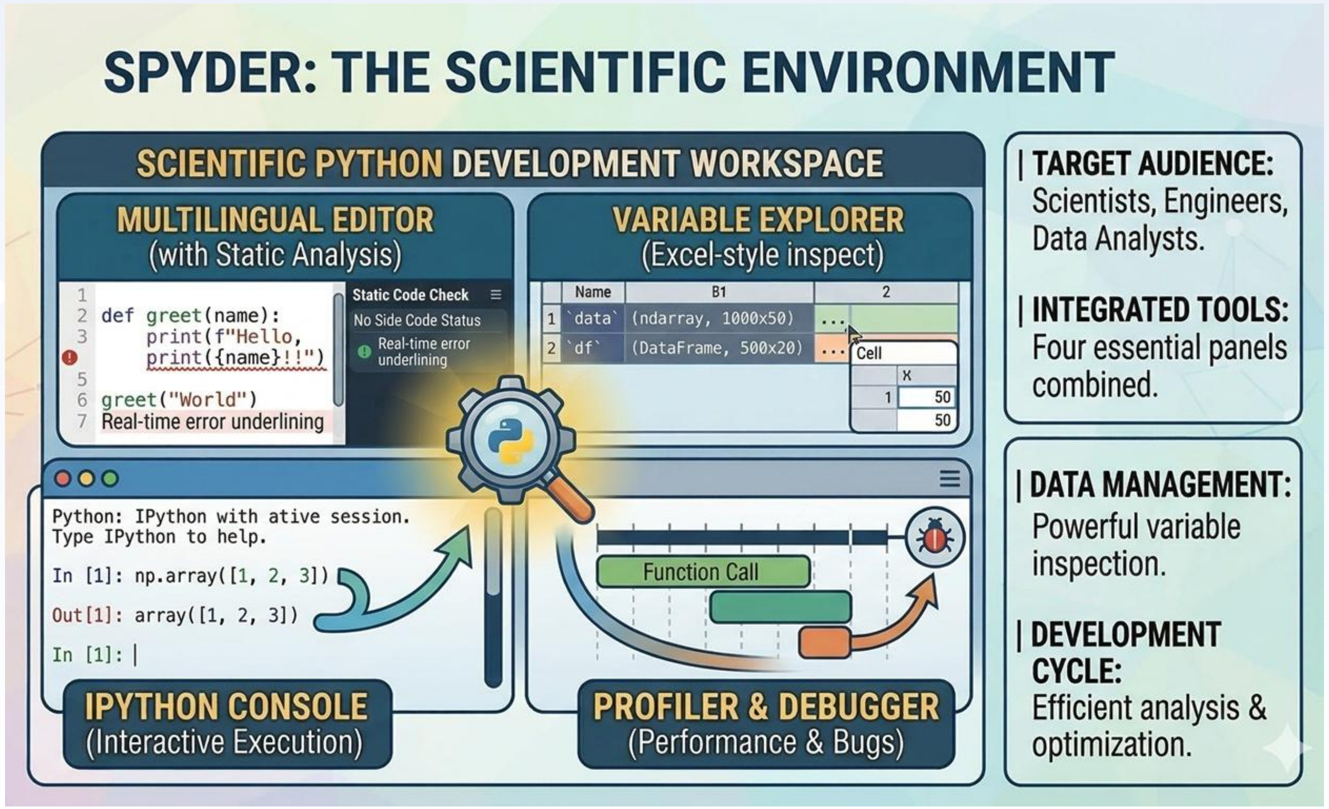
IDE Scientifico: Spyder (5.4)

Spyder – The Scientific Environment

Focus: Progettato per scienziati, ingegneri e analisti di dati.

Strumenti Integrati:

- **Editor Multilingua:** Con analisi statica del codice.
- **Variable Explorer:** Permette di ispezionare e modificare variabili (array NumPy, DataFrame Pandas) in una tabella stile Excel.
- **IPython Console:** Esecuzione interattiva immediata.
- **Profiler & Debugger:** Per ottimizzare le prestazioni del codice.



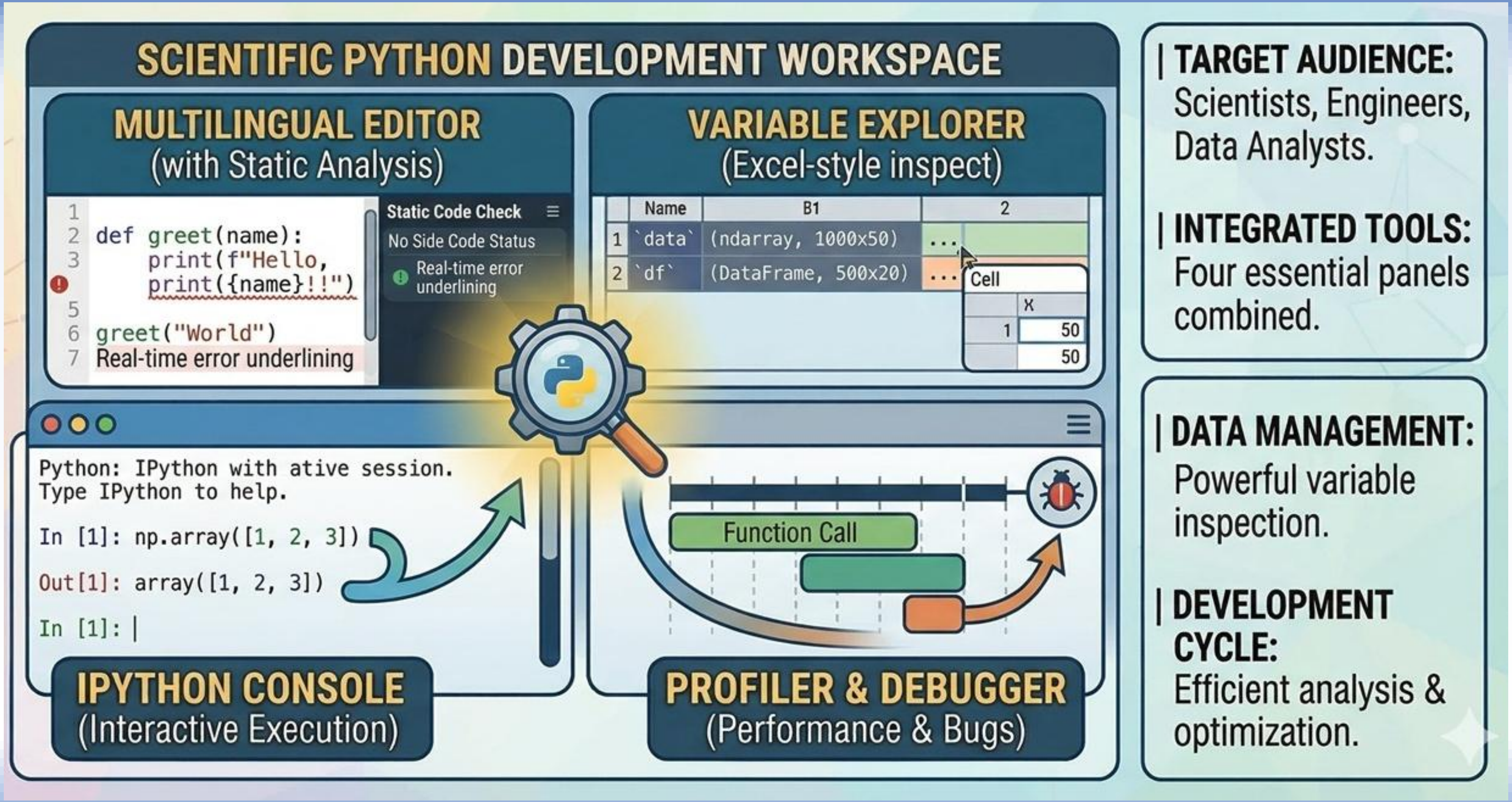
SPYDER: THE SCIENTIFIC ENVIRONMENT
SCIENTIFIC PYTHON DEVELOPMENT WORKSPACE
MULTILINGUAL EDITOR
(with Static Analysis)
Static Code Check
No Side Code Status
Real-time error underlining
Real-time error underlining
VARIABLE EXPLORER
(Excel-style inspect)
Name
B1
2
Cell
IPYTHON CONSOLE
(Interactive Execution)
Function Call
PROFILER & DEBUGGER
(Performance & Bugs)
TARGET AUDIENCE: Scientists, Engineers, Data Analysts.
INTEGRATED TOOLS: Four essential panels combined.
DATA MANAGEMENT: Powerful variable inspection.
DEVELOPMENT CYCLE: Efficient analysis & optimization.

```
def greet(name):
    print(f"Hello,
    print({name}!!")

greet("World")
```

```
1 `data` (ndarray, 1000x50)
2 `df`   (DataFrame, 500x20)
```

```
Python: IPython with ative session.
Type IPython to help.
In [1]: np.array([1, 2, 3])
Out[1]: array([1, 2, 3])
In [1]: |
```

5. Ambiente di esecuzione



```
def greet(name):
    print(f"Hello,
    print({name}!!")

greet("World")
Real-time error underlining
```

	Name	B1	2
1	`data`	(ndarray, 1000x50)	...
2	`df`	(DataFrame, 500x20)	...

	X
1	50
	50

```
Python: IPython with ative session.
Type IPython to help.

In [1]: np.array([1, 2, 3])

Out[1]: array([1, 2, 3])

In [1]: |
```

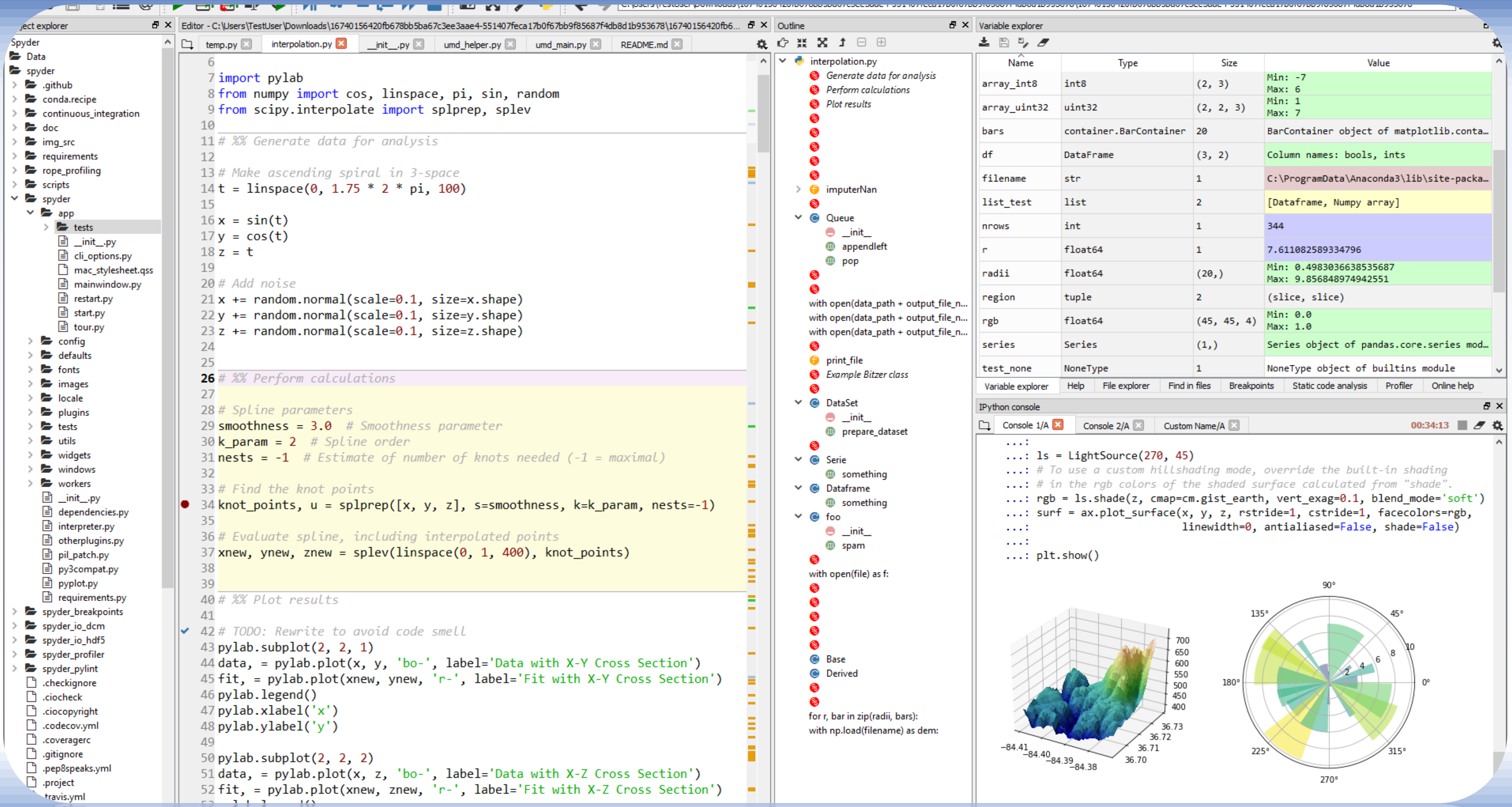
| TARGET AUDIENCE: Scientists, Engineers, Data Analysts.

| INTEGRATED TOOLS: Four essential panels combined.

| DATA MANAGEMENT: Powerful variable inspection.

| DEVELOPMENT CYCLE: Efficient analysis & optimization.

5. Ambiente di esecuzione



ject explorer

```
Spyder
  Data
  spyder
    > .github
    > conda.recipe
    > continuous_integration
    > doc
    > img_src
    > requirements
    > rope_profiling
    > scripts
    v spyder
      v app
        > tests
          __init__.py
          cli_options.py
          mac_stylesheet.qss
          mainwindow.py
          restart.py
          start.py
          tour.py
      > config
      > defaults
      > fonts
      > images
      > locale
      > plugins
      > tests
      > utils
      > widgets
      > windows
      > workers
        __init__.py
        dependencies.py
        interpreter.py
        otherplugins.py
        pil_patch.py
        py3compat.py
        pyplot.py
        requirements.py
    > spyder_breakpoints
    > spyder_io_dcm
    > spyder_io_hdf5
    > spyder_profiler
    > spyder_pylint
      .checkignore
      .ciocheck
      .ciocopyright
      .codecov.yml
      .coveragerc
      .gitignore
      .pep8speaks.yml
      .project
      .travis.yml
```

Editor - C:\Users\TestUser\Downloads\16740156420fb678bb5ba67c3ee3aae4-551407feca17b0f67bb9f85687f4db8d1b953678\16740156420fb6...

temp.py | interpolation.py | __init__.py | umd_helper.py | umd_main.py | README.md

```python

import pylab
from numpy import cos, linspace, pi, sin, random
from scipy.interpolate import splprep, splev

# %% Generate data for analysis

# Make ascending spiral in 3-space
t = linspace(0, 1.75 * 2 * pi, 100)

x = sin(t)
y = cos(t)
z = t

# Add noise
x += random.normal(scale=0.1, size=x.shape)
y += random.normal(scale=0.1, size=y.shape)
z += random.normal(scale=0.1, size=z.shape)


# %% Perform calculations

# Spline parameters
smoothness = 3.0  # Smoothness parameter
k_param = 2  # Spline order
nests = -1  # Estimate of number of knots needed (-1 = maximal)

# Find the knot points
knot_points, u = splprep([x, y, z], s=smoothness, k=k_param, nests=-1)

# Evaluate spline, including interpolated points
xnew, ynew, znew = splev(linspace(0, 1, 400), knot_points)


# %% Plot results

# TODO: Rewrite to avoid code smell
pylab.subplot(2, 2, 1)
data, = pylab.plot(x, y, 'bo-', label='Data with X-Y Cross Section')
fit, = pylab.plot(xnew, ynew, 'r-', label='Fit with X-Y Cross Section')
pylab.legend()
pylab.xlabel('x')
pylab.ylabel('y')

pylab.subplot(2, 2, 2)
data, = pylab.plot(x, z, 'bo-', label='Data with X-Z Cross Section')
fit, = pylab.plot(xnew, znew, 'r-', label='Fit with X-Z Cross Section')
```

Outline

```
interpolation.py
  Generate data for analysis
  Perform calculations
  Plot results
  > imputerNan
  v Queue
      __init__
      appendleft
      pop
  with open(data_path + output_file_n...
  with open(data_path + output_file_n...
  with open(data_path + output_file_n...
  print_file
  Example Bitzer class
  v DataSet
      __init__
      prepare_dataset
  v Serie
      something
  v Dataframe
      something
  v foo
      __init__
      spam
  with open(file) as f:
  Base
  Derived
  for r, bar in zip(radii, bars):
  with np.load(filename) as dem:
```

Variable explorer

Name	Type	Size	Value
array_int8	int8	(2, 3)	Min: -7 Max: 6
array_uint32	uint32	(2, 2, 3)	Min: 1 Max: 7
bars	container.BarContainer	20	BarContainer object of matplotlib.conta...
df	DataFrame	(3, 2)	Column names: bools, ints
filename	str	1	C:\ProgramData\Anaconda3\lib\site-packa...
list_test	list	2	[Dataframe, Numpy array]
nrows	int	1	344
r	float64	1	7.611082589334796
radii	float64	(20,)	Min: 0.4983036638535687 Max: 9.856848974942551
region	tuple	2	(slice, slice)
rgb	float64	(45, 45, 4)	Min: 0.0 Max: 1.0
series	Series	(1,)	Series object of pandas.core.series mod...
test_none	NoneType	1	NoneType object of builtins module

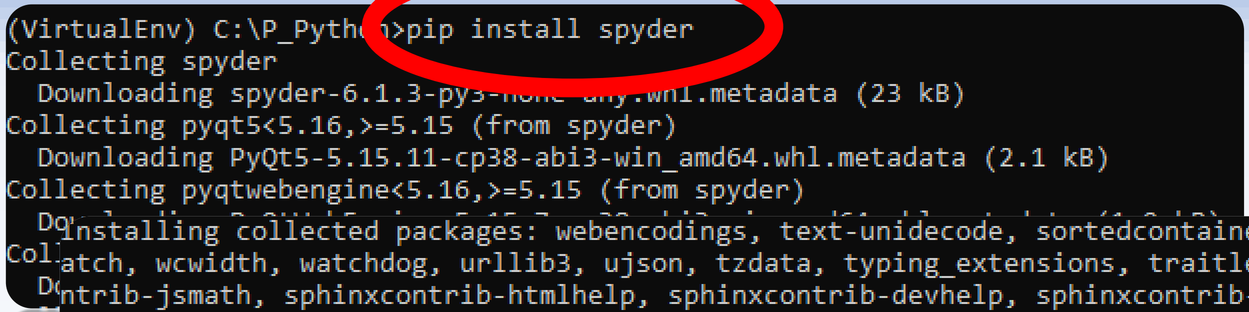
Variable explorer | Help | File explorer | Find in files | Breakpoints | Static code analysis | Profiler | Online help

IPython console

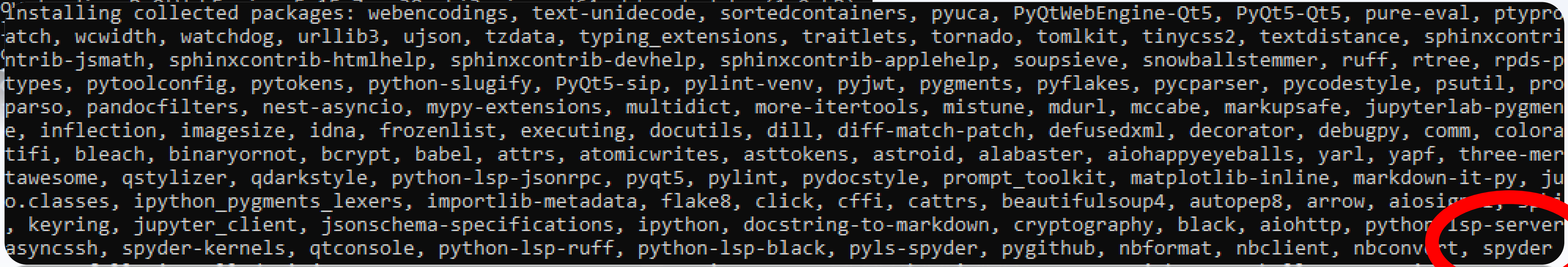
Console 1/A | Console 2/A | Custom Name/A   00:34:13

```
   ...:
   ...: ls = LightSource(270, 45)
   ...: # To use a custom hillshading mode, override the built-in shading
   ...: # in the rgb colors of the shaded surface calculated from "shade".
   ...: rgb = ls.shade(z, cmap=cm.gist_earth, vert_exag=0.1, blend_mode='soft')
   ...: surf = ax.plot_surface(x, y, z, rstride=1, cstride=1, facecolors=rgb,
   ...:                        linewidth=0, antialiased=False, shade=False)
   ...:
   ...: plt.show()
```

5. Ambiente di esecuzione

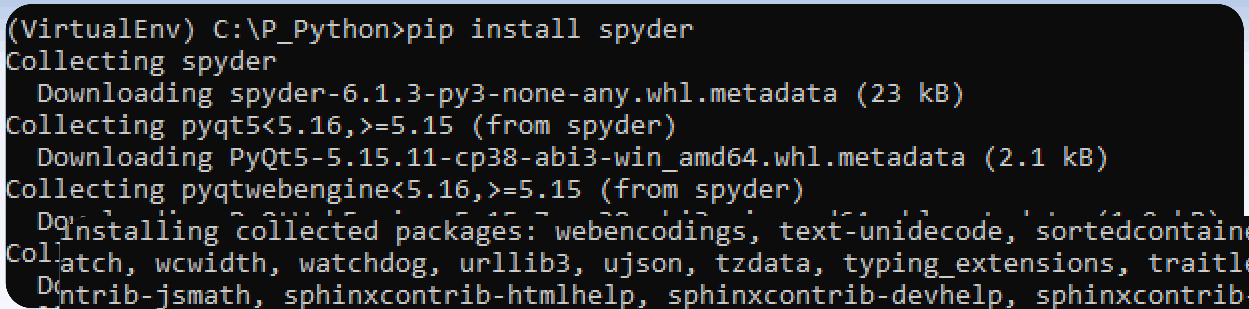


```
(VirtualEnv) C:\P_Python>pip install spyder
Collecting spyder
  Downloading spyder-6.1.3-py3-none-any.whl.metadata (23 kB)
Collecting pyqt5<5.16,>=5.15 (from spyder)
  Downloading PyQt5-5.15.11-cp38-abi3-win_amd64.whl.metadata (2.1 kB)
Collecting pyqtwebengine<5.16,>=5.15 (from spyder)
  Do
Col
  D
```

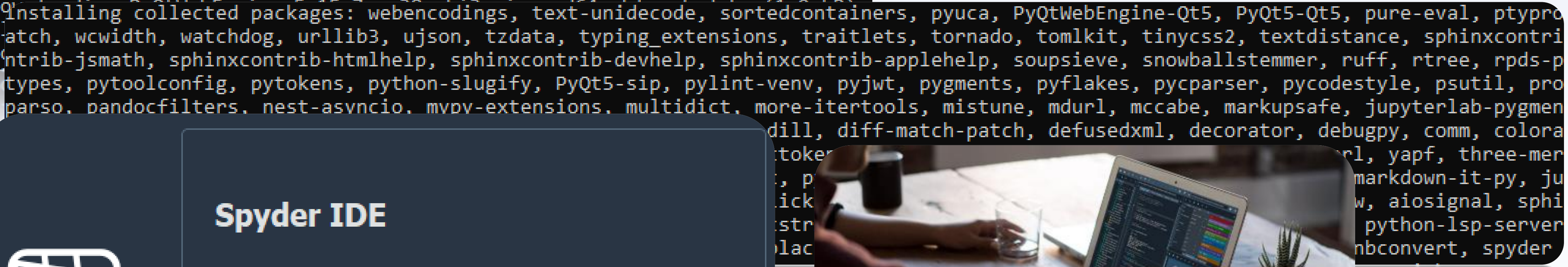


```
Installing collected packages: webencodings, text-unidecode, sortedcontainers, pyuca, PyQtWebEngine-Qt5, PyQt5-Qt5, pure-eval, ptypro
atch, wcwidth, watchdog, urllib3, ujson, tzdata, typing_extensions, traitlets, tornado, tomlkit, tinycss2, textdistance, sphinxcontri
ntrib-jsmath, sphinxcontrib-htmlhelp, sphinxcontrib-devhelp, sphinxcontrib-applehelp, soupsieve, snowballstemmer, ruff, rtree, rpds-p
types, pytoolconfig, pytokens, python-slugify, PyQt5-sip, pylint-venv, pyjwt, pygments, pyflakes, pycparser, pycodestyle, psutil, pro
parso, pandocfilters, nest-asyncio, mypy-extensions, multidict, more-itertools, mistune, mdurl, mccabe, markupsafe, jupyterlab-pygmen
e, inflection, imagesize, idna, frozenlist, executing, docutils, dill, diff-match-patch, defusedxml, decorator, debugpy, comm, colora
tifi, bleach, binaryornot, bcrypt, babel, attrs, atomicwrites, asttokens, astroid, alabaster, aiohappyeyeballs, yarl, yapf, three-mer
tawesome, qstylizer, qdarkstyle, python-lsp-jsonrpc, pyqt5, pylint, pydocstyle, prompt_toolkit, matplotlib-inline, markdown-it-py, ju
o.classes, ipython_pygments_lexers, importlib-metadata, flake8, click, cffi, cattrs, beautifulsoup4, autopep8, arrow, aiosig
, keyring, jupyter_client, jsonschema-specifications, ipython, docstring-to-markdown, cryptography, black, aiohttp, python-lsp-server
asyncssh, spyder-kernels, qtconsole, python-lsp-ruff, python-lsp-black, pyls-spyder, pygithub, nbformat, nbclient, nbconvert, spyder
```

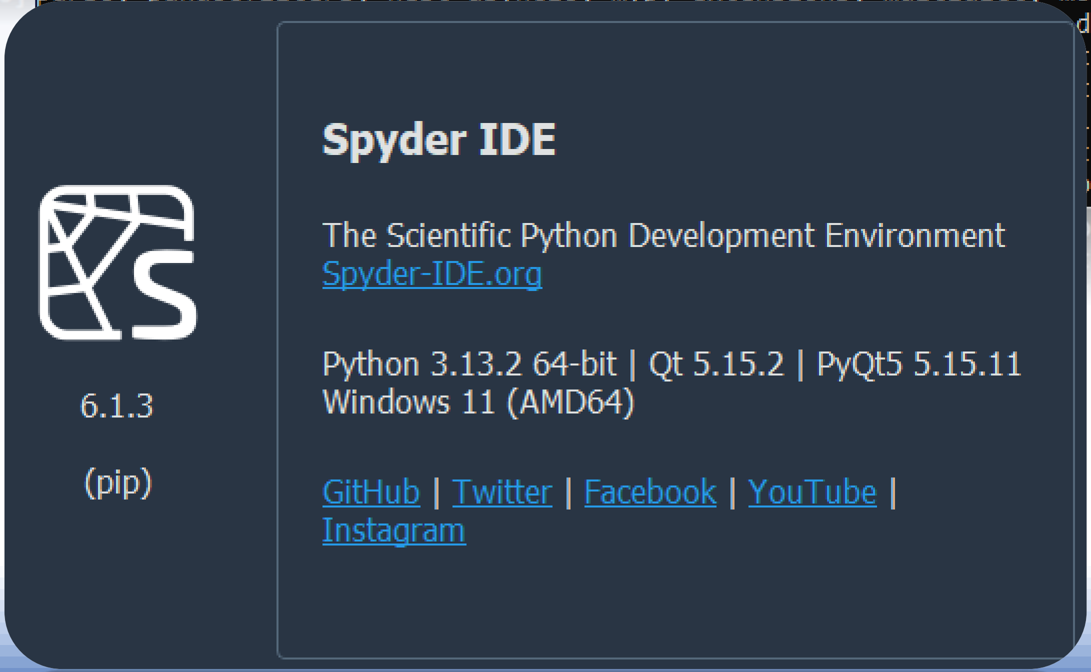
5. Ambiente di esecuzione



```
(VirtualEnv) C:\P_Python>pip install spyder
Collecting spyder
  Downloading spyder-6.1.3-py3-none-any.whl.metadata (23 kB)
Collecting pyqt5<5.16,>=5.15 (from spyder)
  Downloading PyQt5-5.15.11-cp38-abi3-win_amd64.whl.metadata (2.1 kB)
Collecting pyqtwebengine<5.16,>=5.15 (from spyder)
```



```
Installing collected packages: webencodings, text-unidecode, sortedcontainers, pyuca, PyQtWebEngine-Qt5, PyQt5-Qt5, pure-eval, ptypro
atch, wcwidth, watchdog, urllib3, ujson, tzdata, typing_extensions, traitlets, tornado, tomlkit, tinycss2, textdistance, sphinxcontri
ntrib-jsmath, sphinxcontrib-htmlhelp, sphinxcontrib-devhelp, sphinxcontrib-applehelp, soupsieve, snowballstemmer, ruff, rtree, rpds-p
types, pytoolconfig, pytokens, python-slugify, PyQt5-sip, pylint-venv, pyjwt, pygments, pyflakes, pycparser, pycodestyle, psutil, pro
parso, pandocfilters, nest-asyncio, mypy-extensions, multidict, more-itertools, mistune, mdurl, mccabe, markupsafe, jupyterlab-pygmen
                                                 dill, diff-match-patch, defusedxml, decorator, debugpy, comm, colora
                                                 toke                                              rl, yapf, three-mer
                                                 , p                                                markdown-it-py, ju
                                                 ick                                                w, aiosignal, sphi
                                                 str                                                 python-lsp-server
                                                 lac                                                nbconvert, spyder
```



Spyder IDE

The Scientific Python Development Environment
Spyder-IDE.org

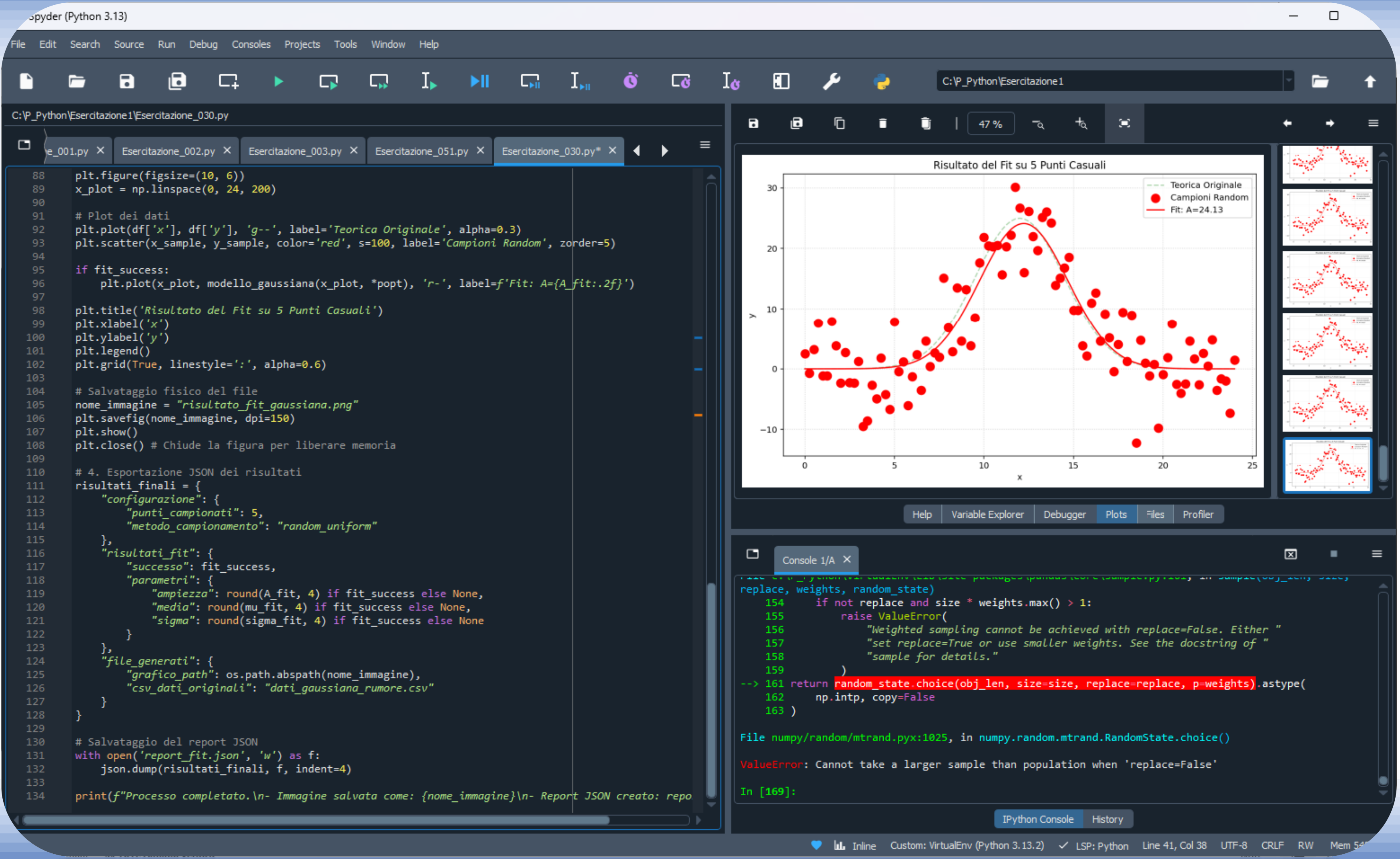
Python 3.13.2 64-bit | Qt 5.15.2 | PyQt5 5.15.11
Windows 11 (AMD64)

GitHub | Twitter | Facebook | YouTube |
Instagram

6.1.3

(pip)

5. Ambiente di esecuzione

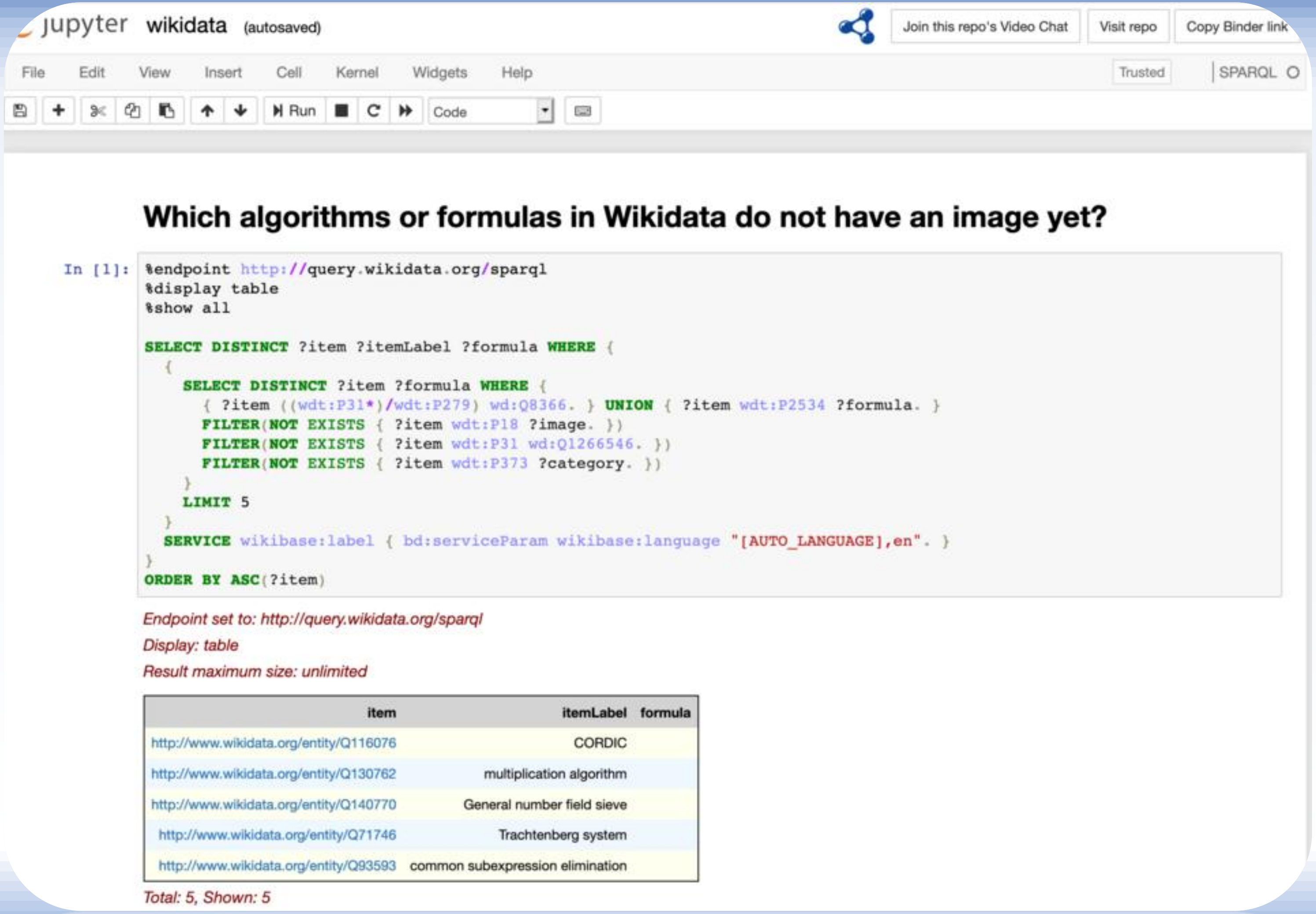


Spyder (Python 3.13)
File Edit Search Source Run Debug Consoles Projects Tools Window Help
C:\P_Python\Esercitazione1
C:\P_Python\Esercitazione1\Esercitazione_030.py
e_001.py
Esercitazione_002.py
Esercitazione_003.py
Esercitazione_051.py
Esercitazione_030.py*
47 %
Risultato del Fit su 5 Punti Casuali
Teorica Originale
Campioni Random
Fit: A=24.13
x
y
Help Variable Explorer Debugger Plots Files Profiler
Console 1/A
IPython Console History
Inline Custom: VirtualEnv (Python 3.13.2) LSP: Python Line 41, Col 38 UTF-8 CRLF RW Mem 54

```python
plt.figure(figsize=(10, 6))
x_plot = np.linspace(0, 24, 200)

# Plot dei dati
plt.plot(df['x'], df['y'], 'g--', label='Teorica Originale', alpha=0.3)
plt.scatter(x_sample, y_sample, color='red', s=100, label='Campioni Random', zorder=5)

if fit_success:
    plt.plot(x_plot, modello_gaussiana(x_plot, *popt), 'r-', label=f'Fit: A={A_fit:.2f}')

plt.title('Risultato del Fit su 5 Punti Casuali')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.grid(True, linestyle=':', alpha=0.6)

# Salvataggio fisico del file
nome_immagine = "risultato_fit_gaussiana.png"
plt.savefig(nome_immagine, dpi=150)
plt.show()
plt.close() # Chiude la figura per liberare memoria

# 4. Esportazione JSON dei risultati
risultati_finali = {
    "configurazione": {
        "punti_campionati": 5,
        "metodo_campionamento": "random_uniform"
    },
    "risultati_fit": {
        "successo": fit_success,
        "parametri": {
            "ampiezza": round(A_fit, 4) if fit_success else None,
            "media": round(mu_fit, 4) if fit_success else None,
            "sigma": round(sigma_fit, 4) if fit_success else None
        }
    },
    "file_generati": {
        "grafico_path": os.path.abspath(nome_immagine),
        "csv_dati_originali": "dati_gaussiana_rumore.csv"
    }
}

# Salvataggio del report JSON
with open('report_fit.json', 'w') as f:
    json.dump(risultati_finali, f, indent=4)

print(f"Processo completato.\n- Immagine salvata come: {nome_immagine}\n- Report JSON creato: repo
```

```
replace, weights, random_state)
    154     if not replace and size * weights.max() > 1:
    155         raise ValueError(
    156             "Weighted sampling cannot be achieved with replace=False. Either "
    157             "set replace=True or use smaller weights. See the docstring of "
    158             "sample for details."
    159         )
--> 161 return random_state.choice(obj_len, size=size, replace=replace, p=weights).astype(
    162     np.intp, copy=False
    163 )

File numpy/random/mtrand.pyx:1025, in numpy.random.mtrand.RandomState.choice()

ValueError: Cannot take a larger sample than population when 'replace=False'

In [169]:
```

5. Ambiente di esecuzione


jupyter wikidata (autosaved)
Join this repo's Video Chat
Visit repo
Copy Binder link
File Edit View Insert Cell Kernel Widgets Help
Trusted
SPARQL
Run
Code

Which algorithms or formulas in Wikidata do not have an image yet?

In [1]:

```
%endpoint http://query.wikidata.org/sparql
%display table
%show all

SELECT DISTINCT ?item ?itemLabel ?formula WHERE {
  {
    SELECT DISTINCT ?item ?formula WHERE {
      { ?item ((wdt:P31*)/wdt:P279) wd:Q8366. } UNION { ?item wdt:P2534 ?formula. }
      FILTER(NOT EXISTS { ?item wdt:P18 ?image. })
      FILTER(NOT EXISTS { ?item wdt:P31 wd:Q1266546. })
      FILTER(NOT EXISTS { ?item wdt:P373 ?category. })
    }
    LIMIT 5
  }
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en". }
}
ORDER BY ASC(?item)
```

Endpoint set to: http://query.wikidata.org/sparql

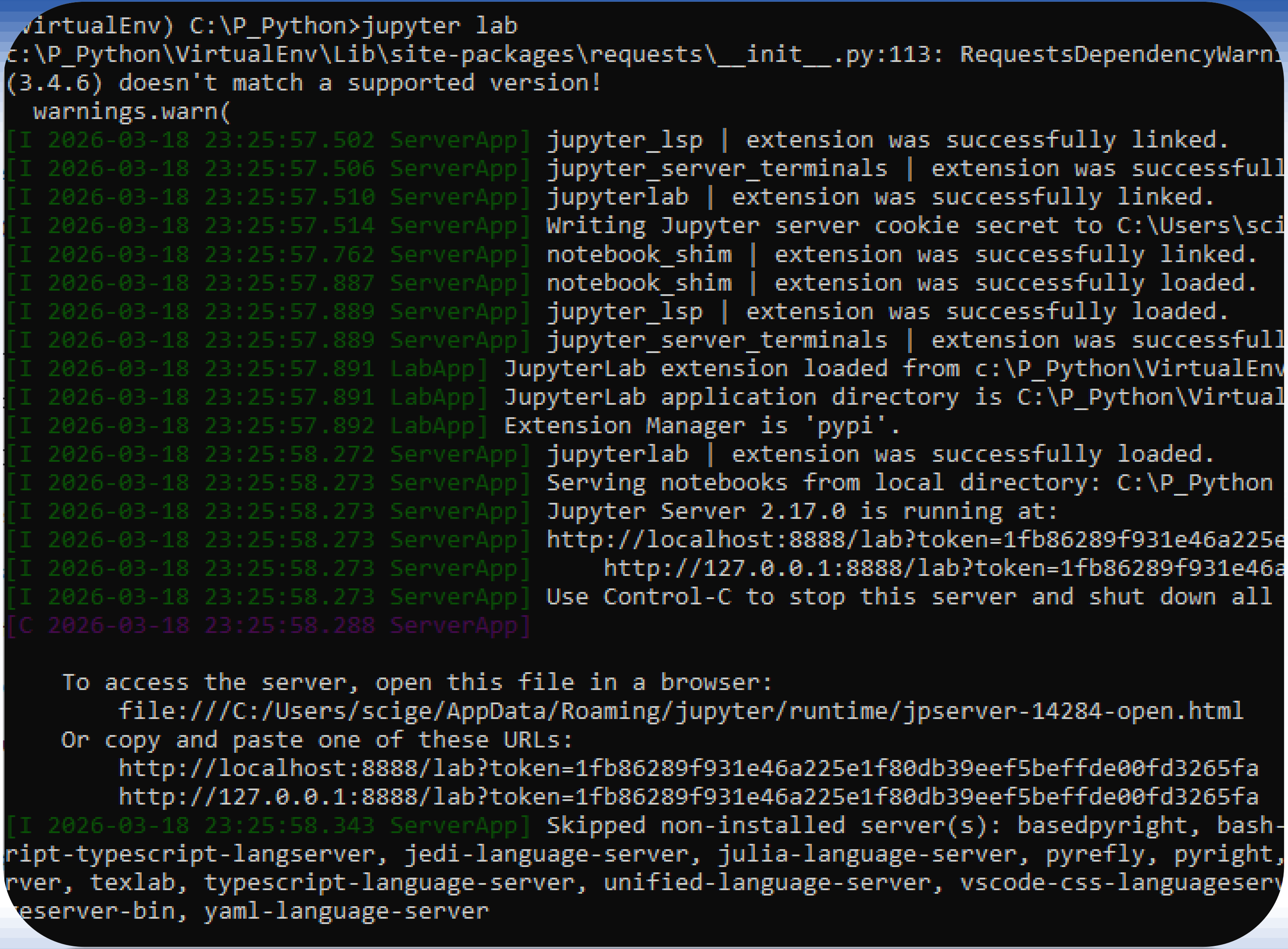
Display: table

Result maximum size: unlimited

item	itemLabel	formula
http://www.wikidata.org/entity/Q116076	CORDIC	
http://www.wikidata.org/entity/Q130762	multiplication algorithm	
http://www.wikidata.org/entity/Q140770	General number field sieve	
http://www.wikidata.org/entity/Q71746	Trachtenberg system	
http://www.wikidata.org/entity/Q93593	common subexpression elimination	

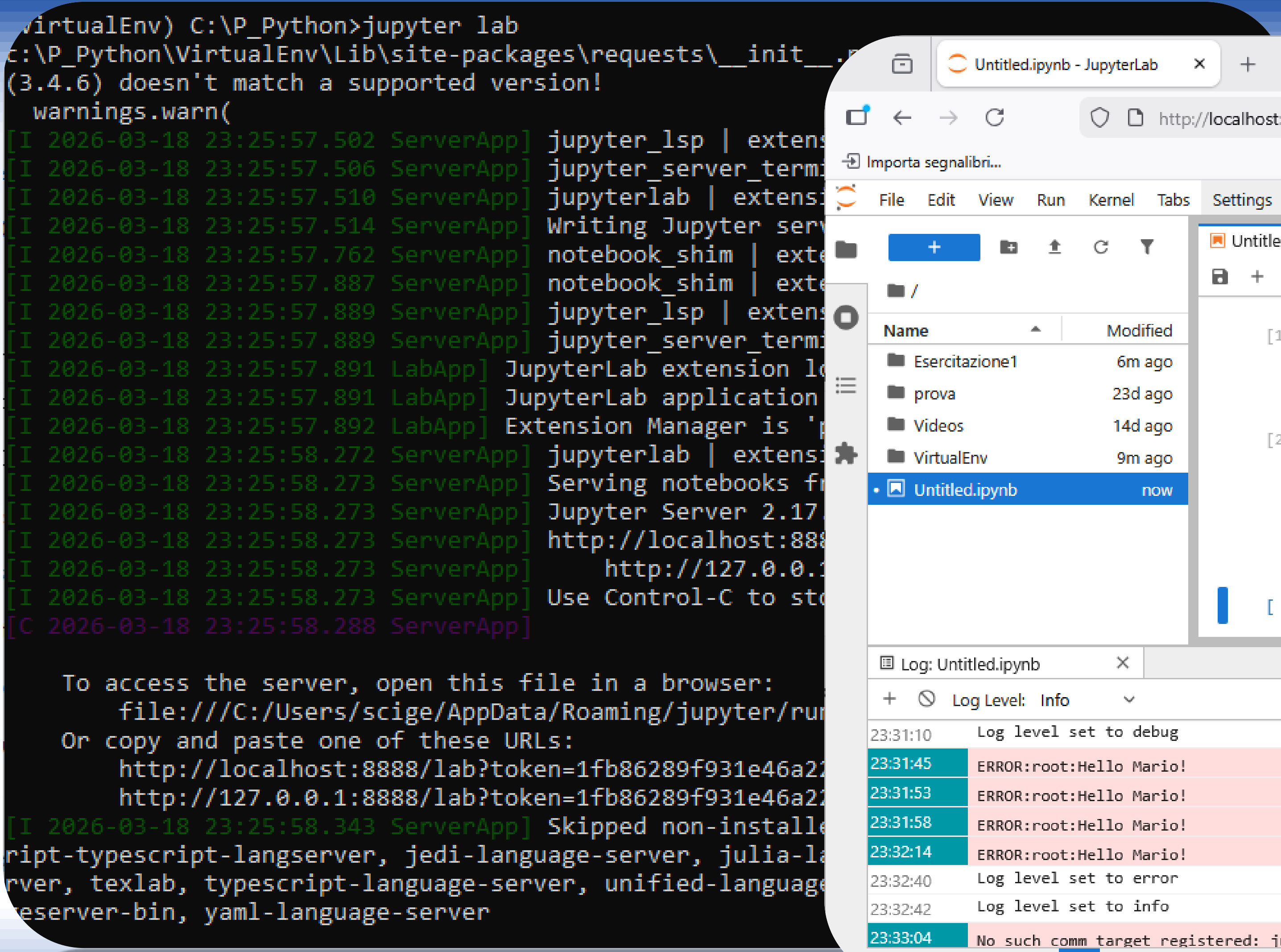
Total: 5, Shown: 5

5. Ambiente di esecuzione

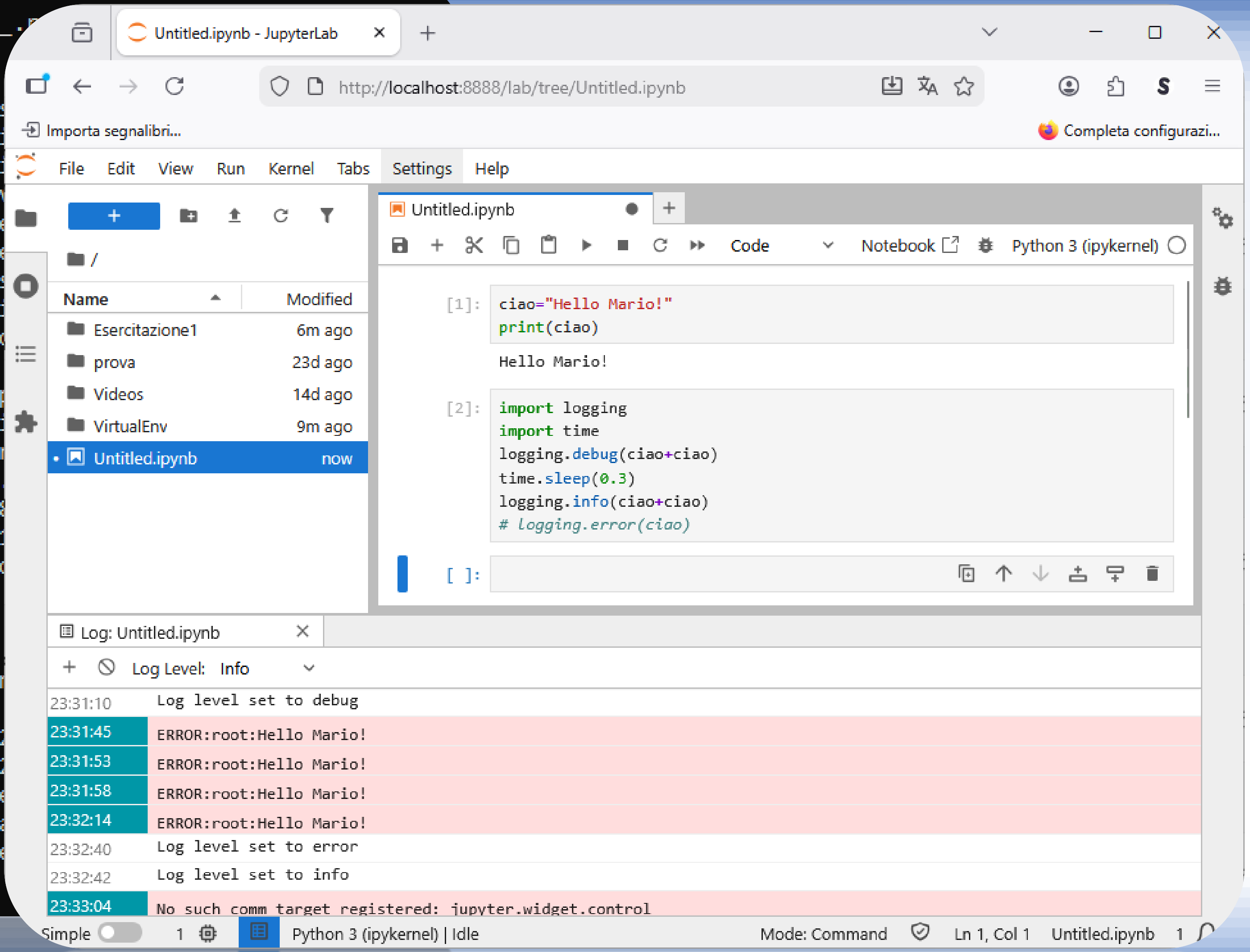


```
irtualEnv) C:\P_Python>jupyter lab
:\P_Python\VirtualEnv\Lib\site-packages\requests\__init__.py:113: RequestsDependencyWarn
(3.4.6) doesn't match a supported version!
  warnings.warn(
[I 2026-03-18 23:25:57.502 ServerApp] jupyter_lsp | extension was successfully linked.
[I 2026-03-18 23:25:57.506 ServerApp] jupyter_server_terminals | extension was successfull
[I 2026-03-18 23:25:57.510 ServerApp] jupyterlab | extension was successfully linked.
[I 2026-03-18 23:25:57.514 ServerApp] Writing Jupyter server cookie secret to C:\Users\sci
[I 2026-03-18 23:25:57.762 ServerApp] notebook_shim | extension was successfully linked.
[I 2026-03-18 23:25:57.887 ServerApp] notebook_shim | extension was successfully loaded.
[I 2026-03-18 23:25:57.889 ServerApp] jupyter_lsp | extension was successfully loaded.
[I 2026-03-18 23:25:57.889 ServerApp] jupyter_server_terminals | extension was successfull
[I 2026-03-18 23:25:57.891 LabApp] JupyterLab extension loaded from c:\P_Python\VirtualEnv
[I 2026-03-18 23:25:57.891 LabApp] JupyterLab application directory is C:\P_Python\Virtual
[I 2026-03-18 23:25:57.892 LabApp] Extension Manager is 'pypi'.
[I 2026-03-18 23:25:58.272 ServerApp] jupyterlab | extension was successfully loaded.
[I 2026-03-18 23:25:58.273 ServerApp] Serving notebooks from local directory: C:\P_Python
[I 2026-03-18 23:25:58.273 ServerApp] Jupyter Server 2.17.0 is running at:
[I 2026-03-18 23:25:58.273 ServerApp] http://localhost:8888/lab?token=1fb86289f931e46a225e
[I 2026-03-18 23:25:58.273 ServerApp]     http://127.0.0.1:8888/lab?token=1fb86289f931e46a
[I 2026-03-18 23:25:58.273 ServerApp] Use Control-C to stop this server and shut down all
[C 2026-03-18 23:25:58.288 ServerApp]

    To access the server, open this file in a browser:
        file:///C:/Users/scige/AppData/Roaming/jupyter/runtime/jpserver-14284-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/lab?token=1fb86289f931e46a225e1f80db39eef5beffde00fd3265fa
        http://127.0.0.1:8888/lab?token=1fb86289f931e46a225e1f80db39eef5beffde00fd3265fa
[I 2026-03-18 23:25:58.343 ServerApp] Skipped non-installed server(s): basedpyright, bash-
ript-typescript-langserver, jedi-language-server, julia-language-server, pyrefly, pyright,
rver, texlab, typescript-language-server, unified-language-server, vscode-css-languageserv
eserver-bin, yaml-language-server
```

5. Ambiente di esecuzione



```
VirtualEnv) C:\P_Python>jupyter lab
C:\P_Python\VirtualEnv\Lib\site-packages\requests\__init__.
(3.4.6) doesn't match a supported version!
  warnings.warn(
[I 2026-03-18 23:25:57.502 ServerApp] jupyter_lsp | extens
[I 2026-03-18 23:25:57.506 ServerApp] jupyter_server_termi
[I 2026-03-18 23:25:57.510 ServerApp] jupyterlab | extensi
[I 2026-03-18 23:25:57.514 ServerApp] Writing Jupyter serv
[I 2026-03-18 23:25:57.762 ServerApp] notebook_shim | exte
[I 2026-03-18 23:25:57.887 ServerApp] notebook_shim | exte
[I 2026-03-18 23:25:57.889 ServerApp] jupyter_lsp | extens
[I 2026-03-18 23:25:57.889 ServerApp] jupyter_server_termi
[I 2026-03-18 23:25:57.891 LabApp] JupyterLab extension lo
[I 2026-03-18 23:25:57.891 LabApp] JupyterLab application
[I 2026-03-18 23:25:57.892 LabApp] Extension Manager is 'p
[I 2026-03-18 23:25:58.272 ServerApp] jupyterlab | extensi
[I 2026-03-18 23:25:58.273 ServerApp] Serving notebooks fr
[I 2026-03-18 23:25:58.273 ServerApp] Jupyter Server 2.17.
[I 2026-03-18 23:25:58.273 ServerApp] http://localhost:888
[I 2026-03-18 23:25:58.273 ServerApp]     http://127.0.0.1
[I 2026-03-18 23:25:58.273 ServerApp] Use Control-C to sto
[C 2026-03-18 23:25:58.288 ServerApp]

    To access the server, open this file in a browser:
        file:///C:/Users/scige/AppData/Roaming/jupyter/run
    Or copy and paste one of these URLs:
        http://localhost:8888/lab?token=1fb86289f931e46a22
        http://127.0.0.1:8888/lab?token=1fb86289f931e46a22
[I 2026-03-18 23:25:58.343 ServerApp] Skipped non-installe
ript-typescript-langserver, jedi-language-server, julia-la
rver, texlab, typescript-language-server, unified-language
eserver-bin, yaml-language-server
```



Untitled.ipynb - JupyterLab
http://localhost:8888/lab/tree/Untitled.ipynb
Importa segnalibri...
Completa configurazi...
File Edit View Run Kernel Tabs Settings Help
Untitled.ipynb
Code
Notebook
Python 3 (ipykernel)

Name	Modified
Esercitazione1	6m ago
prova	23d ago
Videos	14d ago
VirtualEnv	9m ago
Untitled.ipynb	now

```
[1]: ciao="Hello Mario!"
     print(ciao)

     Hello Mario!

[2]: import logging
     import time
     logging.debug(ciao+ciao)
     time.sleep(0.3)
     logging.info(ciao+ciao)
     # logging.error(ciao)

[ ]:
```

Log: Untitled.ipynb

Log Level: Info

```
23:31:10  Log level set to debug
23:31:45  ERROR:root:Hello Mario!
23:31:53  ERROR:root:Hello Mario!
23:31:58  ERROR:root:Hello Mario!
23:32:14  ERROR:root:Hello Mario!
23:32:40  Log level set to error
23:32:42  Log level set to info
23:33:04  No such comm target registered: jupyter.widget.control
```

Simple 1 Python 3 (ipykernel) | Idle Mode: Command Ln 1, Col 1 Untitled.ipynb 1

5. Ambiente di esecuzione
Notebook e Cloud (5.5)

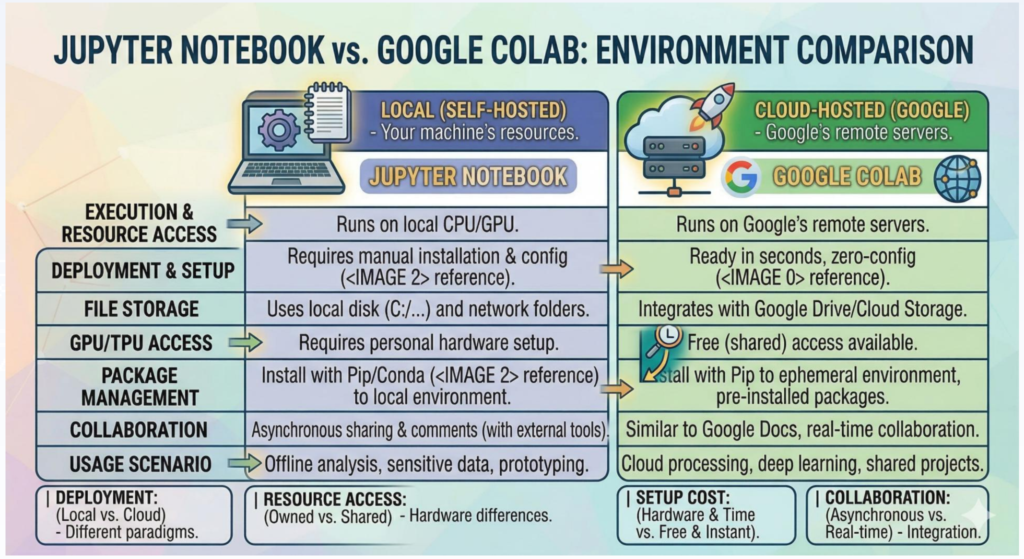
Jupyter e Google Colab

Jupyter Notebook: Interfaccia web che combina codice eseguibile, testo Markdown, equazioni LaTeX e grafici.

- *Uso:* Prototipazione rapida e reportistica.

Google Colab: Versione cloud basata su Jupyter.

- **Zero Configurazione:** Gira nel browser.
- **GPU/TPU Gratuite:** Fondamentale per il Deep Learning.
- **Collaborazione:** Condivisione simile a Google Docs.



JUPYTER NOTEBOOK vs. GOOGLE COLAB: ENVIRONMENT COMPARISON

	LOCAL (SELF-HOSTED) - Your machine's resources. JUPYTER NOTEBOOK	CLOUD-HOSTED (GOOGLE) - Google's remote servers. GOOGLE COLAB
EXECUTION & RESOURCE ACCESS	Runs on local CPU/GPU.	Runs on Google's remote servers.
DEPLOYMENT & SETUP	Requires manual installation & config (<IMAGE 2> reference).	Ready in seconds, zero-config (<IMAGE 0> reference).
FILE STORAGE	Uses local disk (C:/...) and network folders.	Integrates with Google Drive/Cloud Storage.
GPU/TPU ACCESS	Requires personal hardware setup.	Free (shared) access available.
PACKAGE MANAGEMENT	Install with Pip/Conda (<IMAGE 2> reference) to local environment.	stall with Pip to ephemeral environment, pre-installed packages.
COLLABORATION	Asynchronous sharing & comments (with external tools).	Similar to Google Docs, real-time collaboration.
USAGE SCENARIO	Offline analysis, sensitive data, prototyping.	Cloud processing, deep learning, shared projects.

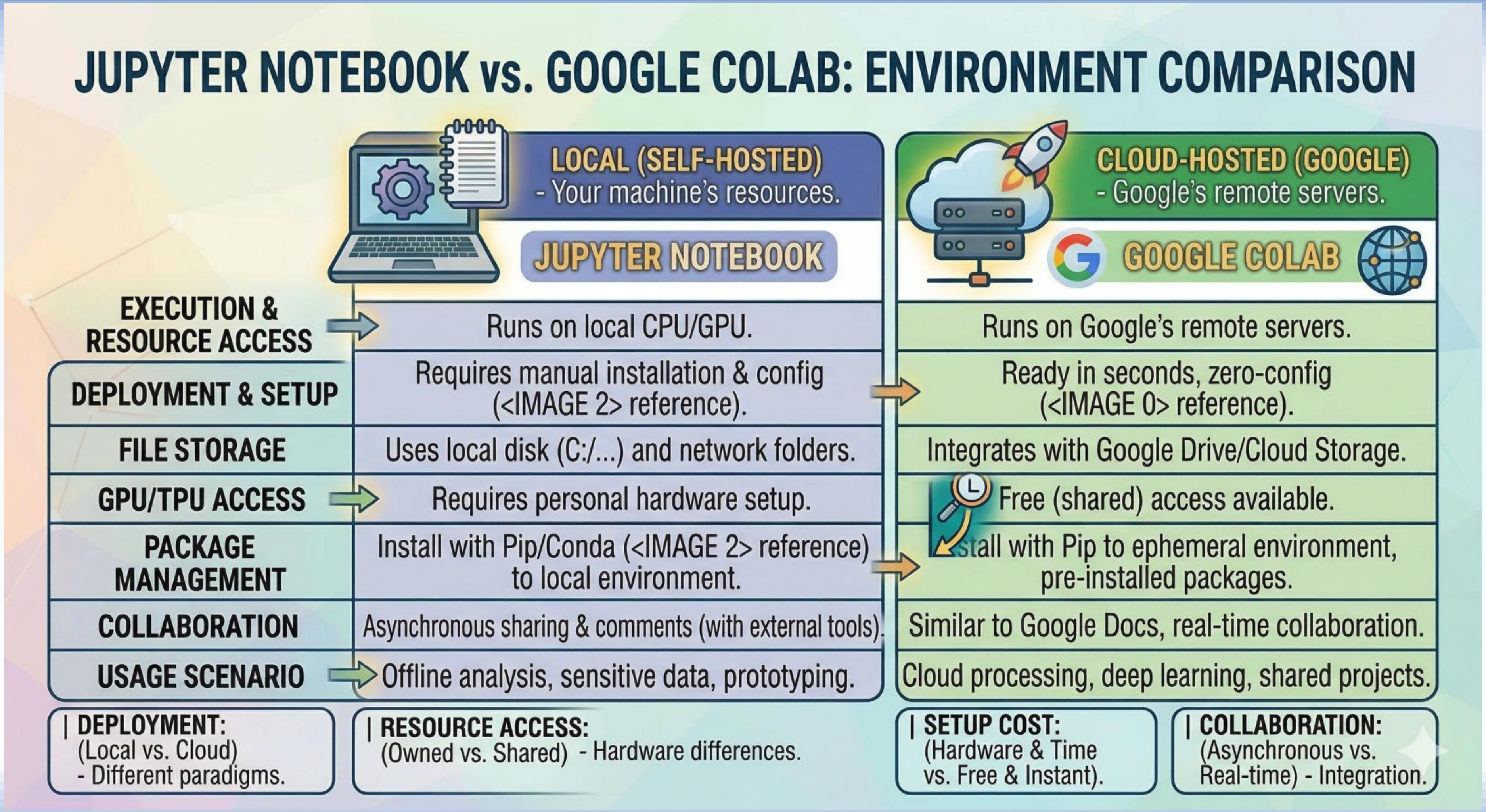
| DEPLOYMENT: (Local vs. Cloud) - Different paradigms.

| RESOURCE ACCESS: (Owned vs. Shared) - Hardware differences.

| SETUP COST: (Hardware & Time vs. Free & Instant).

| COLLABORATION: (Asynchronous vs. Real-time) - Integration.

5. Ambiente di esecuzione



JUPYTER NOTEBOOK vs. GOOGLE COLAB: ENVIRONMENT COMPARISON

	LOCAL (SELF-HOSTED) - Your machine's resources. JUPYTER NOTEBOOK	CLOUD-HOSTED (GOOGLE) - Google's remote servers. GOOGLE COLAB
EXECUTION & RESOURCE ACCESS	Runs on local CPU/GPU.	Runs on Google's remote servers.
DEPLOYMENT & SETUP	Requires manual installation & config (<IMAGE 2> reference).	Ready in seconds, zero-config (<IMAGE 0> reference).
FILE STORAGE	Uses local disk (C:/...) and network folders.	Integrates with Google Drive/Cloud Storage.
GPU/TPU ACCESS	Requires personal hardware setup.	Free (shared) access available.
PACKAGE MANAGEMENT	Install with Pip/Conda (<IMAGE 2> reference) to local environment.	stall with Pip to ephemeral environment, pre-installed packages.
COLLABORATION	Asynchronous sharing & comments (with external tools).	Similar to Google Docs, real-time collaboration.
USAGE SCENARIO	Offline analysis, sensitive data, prototyping.	Cloud processing, deep learning, shared projects.

| DEPLOYMENT: (Local vs. Cloud) - Different paradigms.

| RESOURCE ACCESS: (Owned vs. Shared) - Hardware differences.

| SETUP COST: (Hardware & Time vs. Free & Instant).

| COLLABORATION: (Asynchronous vs. Real-time) - Integration.

5. Ambiente di esecuzione
Interazione con il Sistema (5.6)

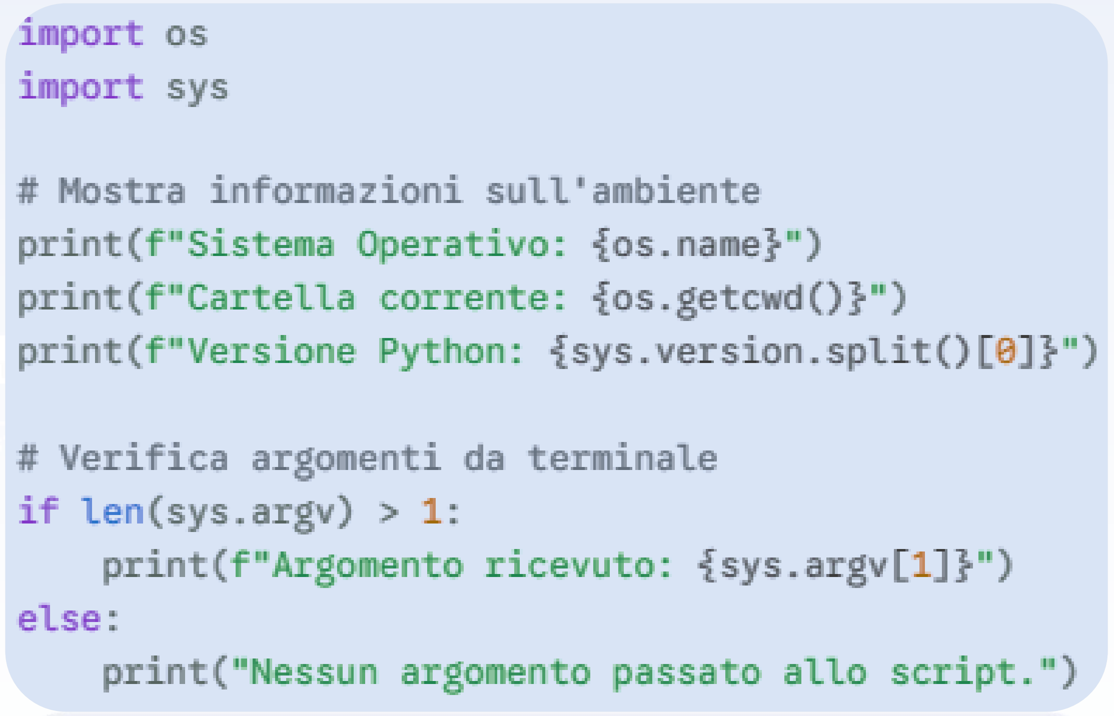
Moduli Os e Sys

Modulo os: Interfaccia con il Sistema Operativo.

- Navigazione directory (os.getcwd(), os.chdir()).
- Gestione file (os.remove(), os.path.exists()).
- Variabili d'ambiente (os.environ).

Modulo sys: Parametri e funzioni specifici dell'interprete.

- sys.argv: Lista argomenti riga di comando.
- sys.path: Lista di directory dove cerca i moduli.
- sys.exit(): Uscita pulita dal programma.



```python
import os
import sys

# Mostra informazioni sull'ambiente
print(f"Sistema Operativo: {os.name}")
print(f"Cartella corrente: {os.getcwd()}")
print(f"Versione Python: {sys.version.split()[0]}")

# Verifica argomenti da terminale
if len(sys.argv) > 1:
    print(f"Argomento ricevuto: {sys.argv[1]}")
else:
    print("Nessun argomento passato allo script.")
```