INTRO to PYTHON

What is Python?

- Python is an interpreted, high-level, general-purpose programming language.

- Great for:
  - Beginner level programmers
  - Data Scientists
  - You - for this lab!

What is Python?

- **Python** is general-purpose open-source programming language that is suitable for nearly every task imaginable, like statistical analysis, scientific and financial computing, machine learning, software development, creating websites, building web applications, data analysis, website parsing etc.
- It is now an extremely popular programming language in finance, and especially in quant finance. It is used for:

❖**Creating financial software** – backtesters, trading engines, research environments, tools for financial instrument pricing etc.

❖**Building smart web applications** – tools that employ collaborative filtering, sentiment analysis (mining the Social Web), deep learning etc.

❖**Carrying out statistical and econometric analysis** – modeling financial time series, doing simulations, running regressions etc.

- **Nearly all quant-oriented vacancies** in London/US require competence in Python from candidates or consider it as a tangible advantage.
- Google, Yahoo!, Facebook, Dropbox, IBM, CERN and NASA are known to use Python in their operation tasks.
- YouTube is written entirely in Python.

Why Python?

Feature	Python	R	SAS	MATLAB	Java, C#, C/C++
General-purpose programming language	✓	✓	✗	✗	✓
Convenient for data science	✓	✓	✓	✓	✗
Flexibility	Very good	Poor	Abysmal	Abysmal	Very good
Data processing capability	Very good	Good	Very good	N/A	Good
Scientific computing capability	Excellent	Excellent	Good	Excellent	Poor
Code readability	Excellent	Good	Poor	Excellent	Poor
Execution speed	Very good	Good	Very good	Very good	Excellent
System integration	Very good	Good	Poor	Poor	Very good
Package quality	Excellent	Variable	N/A	N/A	Excellent
Excel integration	Very good	Good	Good	None	None

Python environments and development tools

	Python console	Python IDLE	IPython	Jupyther Qt console	Jupyther notebook	Spyder	PyCharm
Interactivity	✓	✓	✓	✓	✓	✓	✓
Text highlight	✗	✓	✓	✓	✓	✓	✓
File/directory manipulation	✗	✗	✓	✓	✓	✓	✓
Inline graphics	✗	✗	✗	✓	✓	✓	✓
Execution of scripts	✗	✗	✓	✓	✓	✓	✓
Display of arrays and dataframes	✗	✗	✗	✗	✓	✓	✓
Display of scripts	✗	✗	✗	✗	✗	✓	✓
Convenient for version control	✗	✗	✗	✗	✗	✗	✓

Excel vs Python

The evidence suggests that both Excel and Python have their place with certain applications.

- Excel is a great entry-level tool and is a quick-and-easy way to analyse a dataset.
- But for the modern era, with large datasets and more complex analytics and automation, Python provides the tools, techniques and processing power that Excel, in many instances, lacks. After all, Python is more powerful, faster, capable of better data analysis and it benefits from a more inclusive, collaborative support system.

What is Jupyter Notebook?

- Jupyter notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text.
- It will allow us to::
  - Execute
  - Modify
  - Document

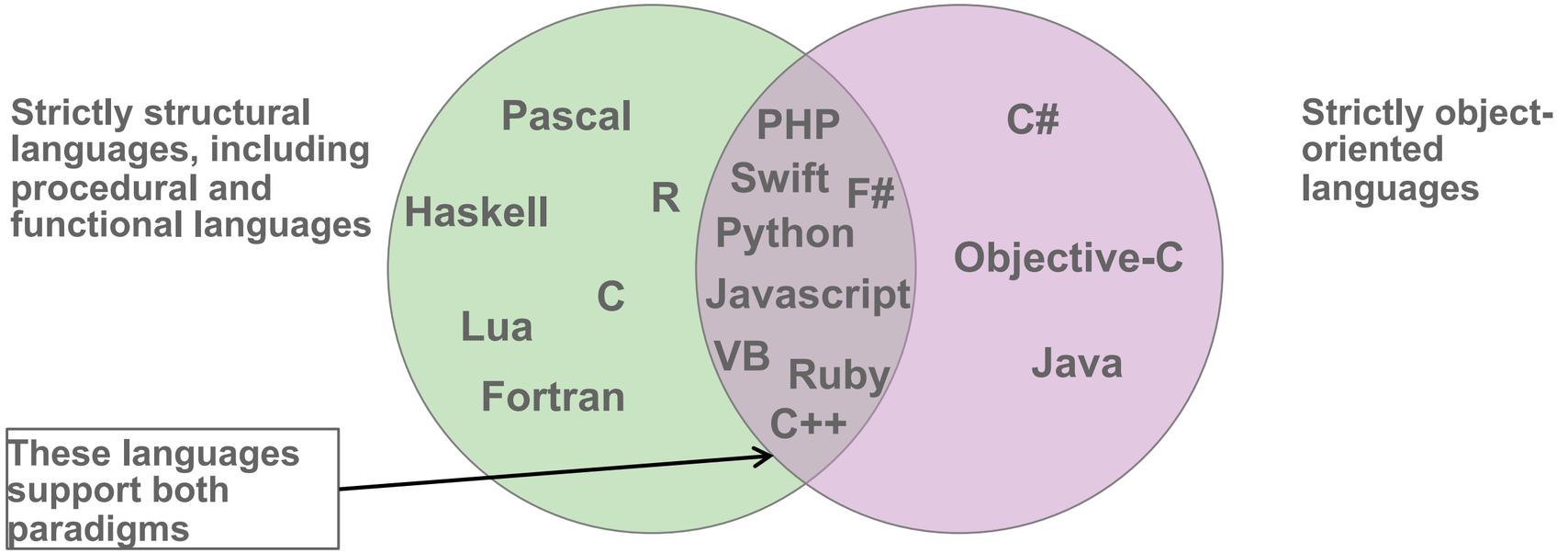
What is Google Colabatory?

- Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud of Google

- It will allow us to::
  - Execute
  - Save and Share
  - Access powerful resources

Structured or object-oriented?

- The language is described as belonging to the structured programming paradigm, if its programs are organized around its code (“code acting on data”).
- On the contrary, the language is described as having the object-oriented design, if its programs are organized around the data (“data controlling access to code”).
- The division is often approximate, since most languages are capable of both approaches to programming with a varying degree of completeness.



Interpreted or compiled?

Definitions

- The actual execution of your code is always performed by the Central Processing Unit (**CPU**) of your computer.
- CPU can understand and interpret native (machine) code only (i.e. the sequence of 0s and 1s turned into electronic impulses).
- Thus, there is an intermediary between you who writes code in a particular programming language and the CPU that executes it. This intermediary is called the **translator**.
- The **translator**, in its turn, may come in two forms*, depending on the particular language: the interpreter and the compiler.
- Depending on the form of translator, programming languages are referred to as interpreted or compiled, respectively.

Language type	Execution speed	Portability	Examples
Interpreted	Low/Medium	Good	R, Python, Ruby
Compiled	High/Extreme**	Variable	C/C++, C#, Java

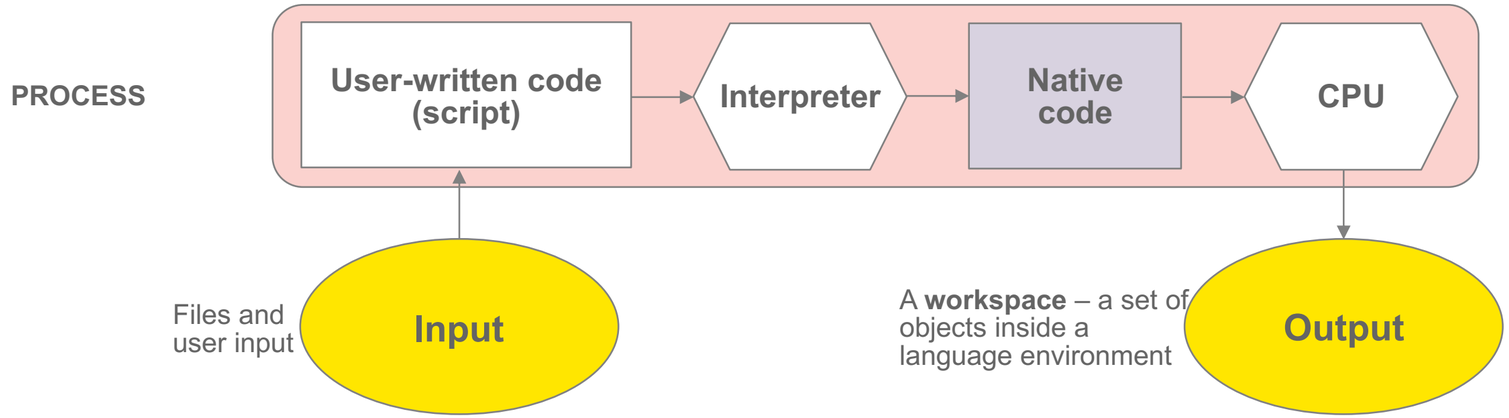
*The division is just a convenient approximation, however.

**C++, for example, is x1000 times faster than Python.

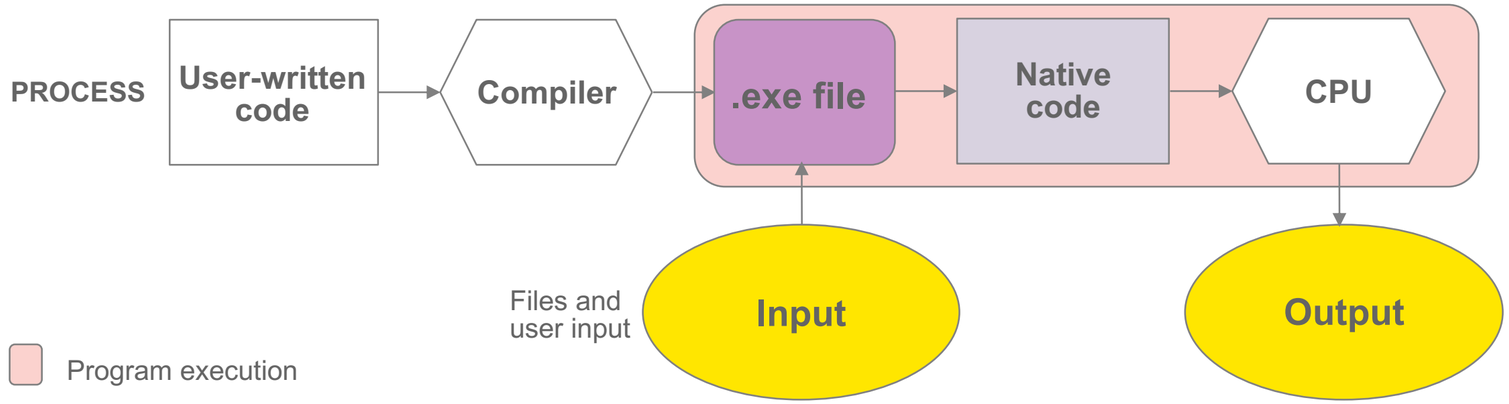
Interpreted or compiled?

How the translation works

- Code processing in interpreted languages:



- Code processing in interpreted languages:



Interpreted or compiled?

Comparison

Compiler	Interpreter
A compiler translates the entire source code in a single run.	An interpreter translates the entire source code line by line.
It consumes less time i.e., it is faster than an interpreter.	It consumes much more time than the compiler i.e., it is slower than the compiler.
It is more efficient.	It is less efficient.
CPU utilization is more.	CPU utilization is less as compared to the compiler.
Both syntactic and semantic errors can be checked simultaneously.	Only syntactic errors are checked.
The compiler is larger.	Interpreters are often smaller than compilers.
It is not flexible.	It is flexible.
The localization of errors is difficult.	The localization of error is easier than the compiler.
A presence of an error can cause the whole program to be re-organized.	A presence of an error causes only a part of the program to be re-organized.
The compiler is used by the language such as C, C++.	An interpreter is used by languages such as Java