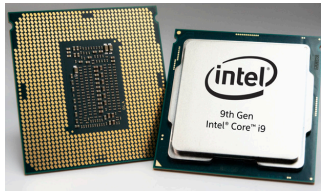
Python Background

Dr Ekaterina Abramova

Winter 2021

Current In-Demand Languages

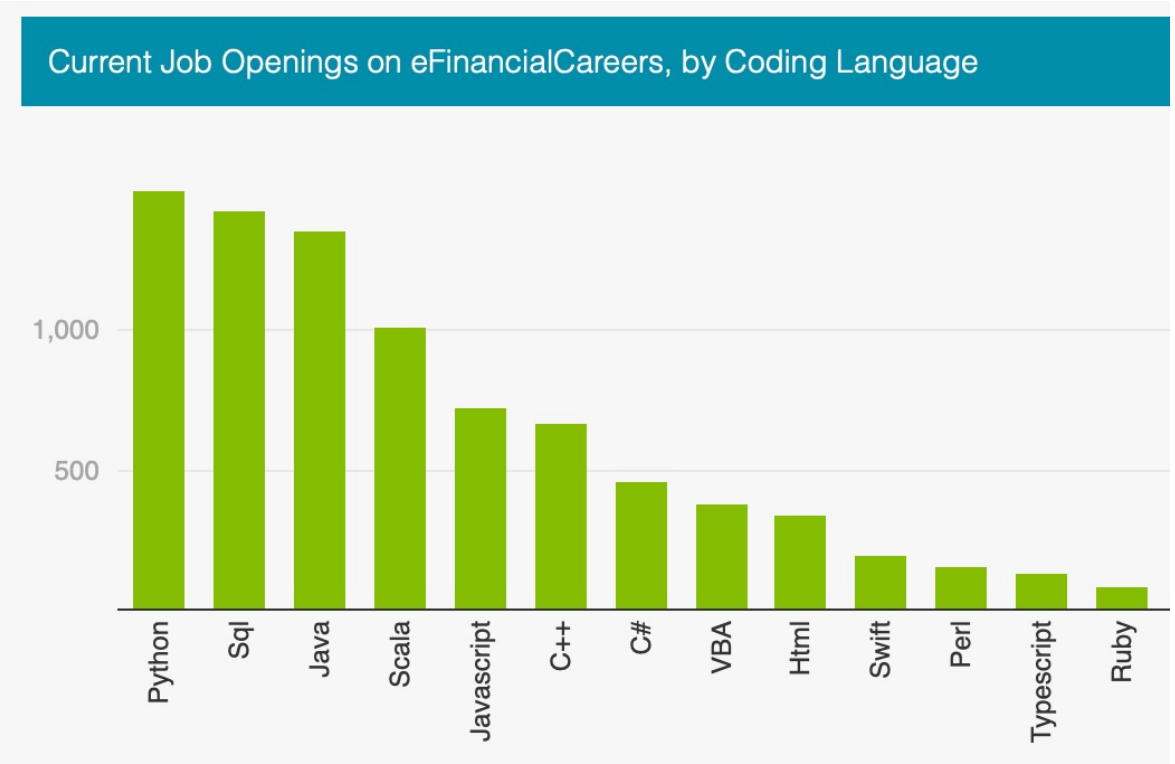
A programming language is a formal language designed to communicate instructions to a computer's Central Processing Unit (CPU) which is made of billions of transistors (that act as a switch between 0 and 1).



There are 600 programming languages out there, with 2 major distinguishing types:

- **Low-level languages**: are machine oriented with instructions using binary notation
- **High-level languages**: use English and symbols in its instructions

Which ones are the most popular on eFinancialCareers?



*Ranking: eFinancialCareers research 2020.

PYPL is the Popularity of Programming Language index created by analyzing how often language tutorials are searched on Google.

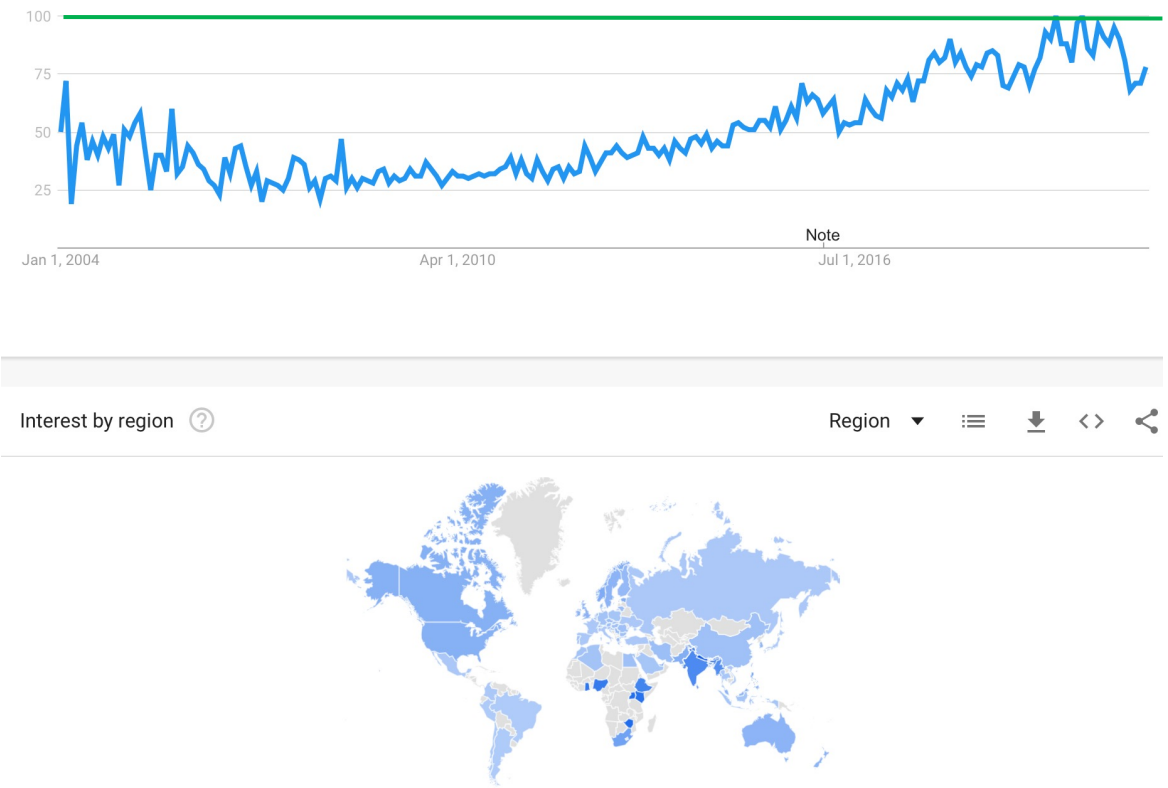
Worldwide, Feb 2021 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	30.06 %	+0.3 %
2		Java	16.88 %	-1.7 %
3		JavaScript	8.43 %	+0.4 %
4		C#	6.69 %	-0.6 %
5	↑	C/C++	6.5 %	+0.5 %
6	↓	PHP	6.19 %	-0.1 %
7		R	3.82 %	+0.0 %

*See PYPL webcite.

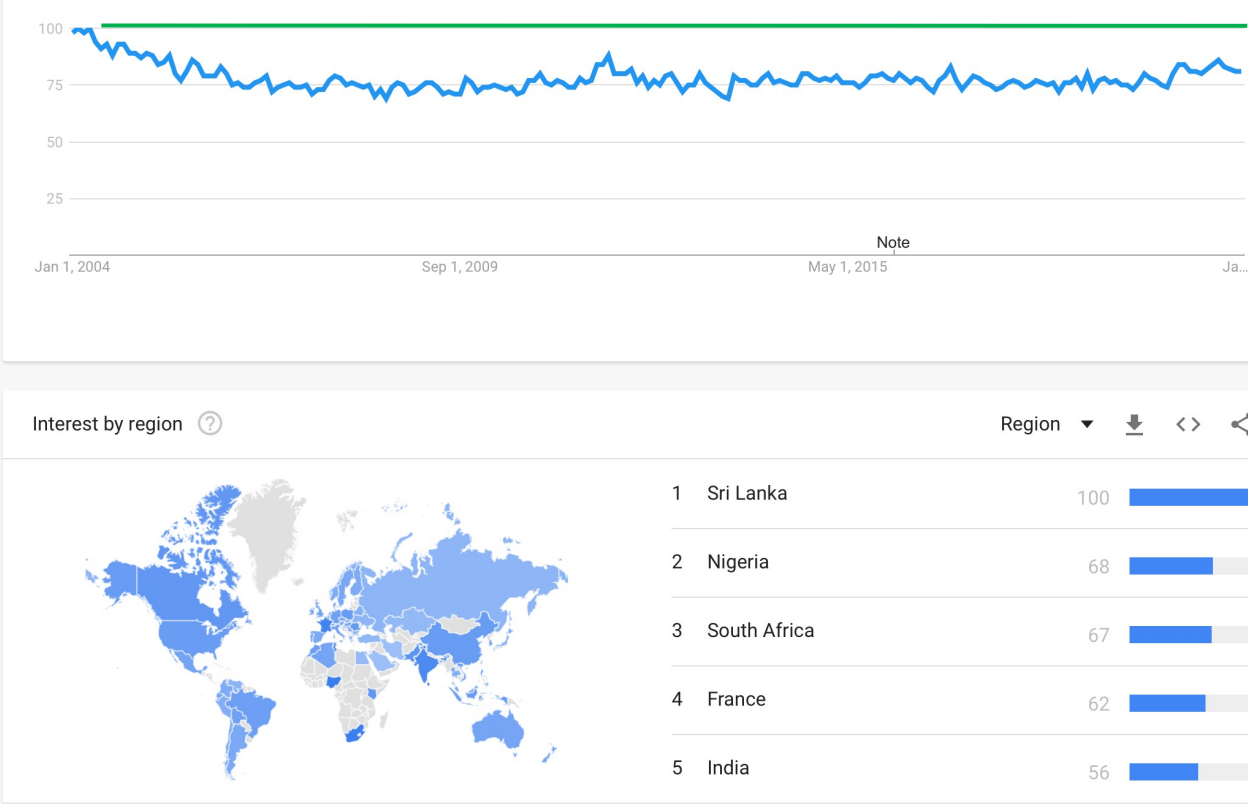
Google Trends – 2004 to 2021

“Python Programming” Searches



“C++ Programming” Searches

100% popularity



What about Python in Banking?

Language	Origin & Demand*	Current Use	Great For
C++	1985 High	Backbone of legacy banking systems	• Fast computations • High volumes of data (HFT)
Python	1989 High	Demand tripled over last 1.5 yrs Replacing Java Current go-to language	Pricing, trade and risk management • Analytic tools • Quant models
Java	1996 Medium	Demand down 25% over last 1.5 yrs Python/R: faster*, easier, more flexible	• Data security
R	1993 Medium	Data analytics: statistics, econometrics	• Statistical simulations • Predictive analysis
C#	2001 Medium	Still in use but only for particular tasks. Has similarities with Java.	• Quant tasks • Low-latency tasks
Matlab	1984 Low	Not used much in banking. Popular in engineering / scientific academic research (PhD, Post-Doc).	• Numerical analysis tasks • Matrix manipulations (inversion) • Financial engineering tasks

*Ranking varies according to industry insiders.

*Faster – in terms of development time NOT run time

Python – Financial Industry "Must"

Python Generally used for:

Quantitative problems / Algorithmic trading / Financial Engineering / Data Science

Industry	Implementations
FinTech Companies	Many supporting libraries for: Analytics & Data Science / Regulation & Compliance.
Investment Banking / Hedge Funds	*JP Morgan* **Athena** & *Bank of America Merrill Lynch* **Quartz** – cross-asset market risk and trading platforms (10 million lines of code). Pioneer of coding major investment banking platforms in Python is Kirat Singh.
Exchanges	Parts of exchange platforms are coded in Python.

Pros:

- Free software / numerous libraries / large developing & online support community
- Easy & elegant to code and deploy (e.g. easily integrated into front and back ends)
- Less lines of code (10 Python vs. 20 Java) / Quickest language for solving problems

Cons:

- Slower comparative execution time (vs. C++, Java)

Programming Support Website:
Stackoverflow

Example code Java vs Python

Fewer lines of code are required for the same job using Python!

Displaying ‘hello world’ string:

Hello World in Java:

```java
public class Main {
public static void main(String[] args)
{ System.out.println("hello world"); }
}
```

Hello World in Python:

```python
print "hello world";
```

Splitting a string into words:

String Operations in Java:

```java
public static void main(String[] args){
String test="compare Java with Python";
    for(String a : test.split(" "))
    System.out.print(a);
}
```

String Operations in Python:

```python
a="compare Python with Java";
print a.split();
```

Open in a file and print its contents:

File I/O in Java:

```java
// get current directory
File dir = new File(".");
File fin=new File(dir.getCanonicalPath()
        + File.separator + "Code.txt");

FileInputStream fis =
            new FileInputStream(fin);

//Construct the BufferedReader object
BufferedReader in = new BufferedReader
         (new InputStreamReader(fis));

String aLine = null;
while ((aLine = in.readLine()) != null)
{//Process each line, here we count
   empty lines
    if (aLine.trim().length() == 0) {
    }
}
// do not forget to close the buffer
reader
in.close();
```

File I/O in Python:

```python
myFile = open("/home/xiaoran/Desktop/
test.txt")

print myFile.read();
```

Image source: Quora.

Good Coding Practice

- File Names: use `.py`

```
# Good          # Bad
download.py      download.PY
```

- If Need Running Files in Sequence: use numbering

```
0-preprocess.py   analyse.py
1-analyse.py      preprocess.py
2-report.py       report.py
```

- Operators Spacing: use a space around operators (e.g. +, <, =) and after comma

```
# Good                    # Bad
result = 25 / (12 + 3)    result=25/(12+3)
```

```
# Good          # Bad
range(0, 100)   range(0,100)
```

- Function Call Spacing: no space before left parentheses; around code in parentheses or square brackets

```
# Good        # Bad
print(x, y)   print (x, y)
```

```
# Good        # Bad
print(x, y)   print( x, y )
```

```
# Good        # Bad
L[0]          L[ 0 ]
```

- Commenting guidelines: use dashes ---

```
# Question 1a -----------
# Question 1b -----------
```

- Variable Naming Convention:

```
# Good              # Bad
my_variable_name    myvariablename
```

NB: Official "Style Guide for Python Code": PEP8 (Author: Guido van Rossum).

Libraries

Library / Package – is pre-written collection of code containing variables, data sets and commands written in relation to performing a specific functionality.

Scientific Libraries – key computing tools utilized for scientific tasks.

NumPy

xarray

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

matplotlib

seaborn

scikit learn

scikit-image
image processing in python

SM StatsModels
Statistics in Python

Scrapy

PyTables

Cython

Link to Scientific Libraries

Library Name	Description	Import Commands and Conventions
numpy	Typically homogenous N-dim array object. Algebra. Easy integration with C/C++/Fortran and databases.	`import numpy as np`
xarray	Typically labelled N-dim array object	`import xarray as xr`
pandas	High performance, easy to use data structure for heterogenous data and time series. Regression.	`import pandas as pd`
scipy	Integration, optimization, interpolation, statistics tools	`from scipy import obj`
sympy	Calculus, polynomials, combinatorics, geometry, stats	`from sympy import obj`
matplotlib	Extensive plotting options using Matlab framework	`import matplotlib.pyplot as plt`
seaborn	Data visualization based on `matplotlib`	`import seaborn as sns`
scikit-learn	Data mining / data analysis / machine learning	`from sklearn.obj import obj`
scikit-image	Algorithms for machine learning image processing	`from skimage import obj`
statsmodels	Statistics, time series, econometrics	`import statsmodels as sm`
scrapy	Scraping data from websites	`import scrapy`
pytables	Management of hierarchical / very large datasets	`import tables`
cython	Allows object oritented C extensions for Python	`from Cython.Build import cythonize`

Links to Financial Libraries

Library Name	Description
statistics	Basic statistics in Python (pre-installed)
dynts / ARCH	Time series focused library built on `numpy` / ARCH and other tools for econometrics
pyfolio	Portfolio and risk analytics
pyrisk	Common financial risk and performance
qfrm	Measure, manage and visualize risk of financial instruments and portfolios
pyfin / ffn	Basic pricing / Financial function library, built on `numpy`, `pandas`, `scipy`.
vollib	Option prices, implied volatility, greeks using: Black, Black-Scholes, Black-Scholes-Merton
quantpy	Quantitative finance (CAPM, auto data downloader, event profiler)
pynance	Retrieving, analysing and visualizing data from stock and derivatives markets
quantlib	Structuring, valuation and risk (Python wrapped version of QuantLib C++)
pyalgotrade	Event driven (paper) algorithmic trading library, backtesting
zipline	Event driven (live) algorithmic trading library, backtesting
finmarketpy	Backtesting trading strategies and analysing financial markets
pybacktest	Vectorized backtesting built on Pandas.
TA-Lib	Technical Analysis of financial market data.