julia

Julia Programming Language

Julia Design

- General Purpose, but built specifically for scientific computing, machine learning, data mining, larger-scale linear algebra, distributed programming and parallel computing
- Multiparadigm
- Dynamic Programming Language
- Even though can be used as a general programming language since it’s Turing complete, it is mostly used for
- Array numbering begins at 1, not 0.
- Compiles to the LLVM Compiler

Julia Technical Perspective

“Looks like Python, feels like Lisp, runs like Fortran” - Julia motto

Looks

- Syntax most related to Python.
- Dynamic typing
- Has some similarities to the Lisp programming language. For example having “end” after the for loops.

Feels

- Some syntax is similar to Lisp:
  - Ex: Lisp `(+ 2 2)` , Julia: Popping from a list: push!(a, 3)
- Has some similarities in syntax to full lisp-style macros. Some properties similar to LISP

Runs

- Was made for fast performance … almost as fast as C and C++
- Does Just in Time Compiling.
- “Julia language includes built-in parallel computing support, multi-threading functionality and an implementation of multi-core distributed memory processing.”

Syntax comparison (Julia Vs python)

Julia

- **Print statements**

  “Hello World”

- **Creating variables**
- **For loop**

```
for iterator in range
    statements(s)
end
```

Python

- **Print statements**

  print(“Hello World”)

- **Creating variables**
- **For Loops**

```
for x in "banana":
  print(x)
```

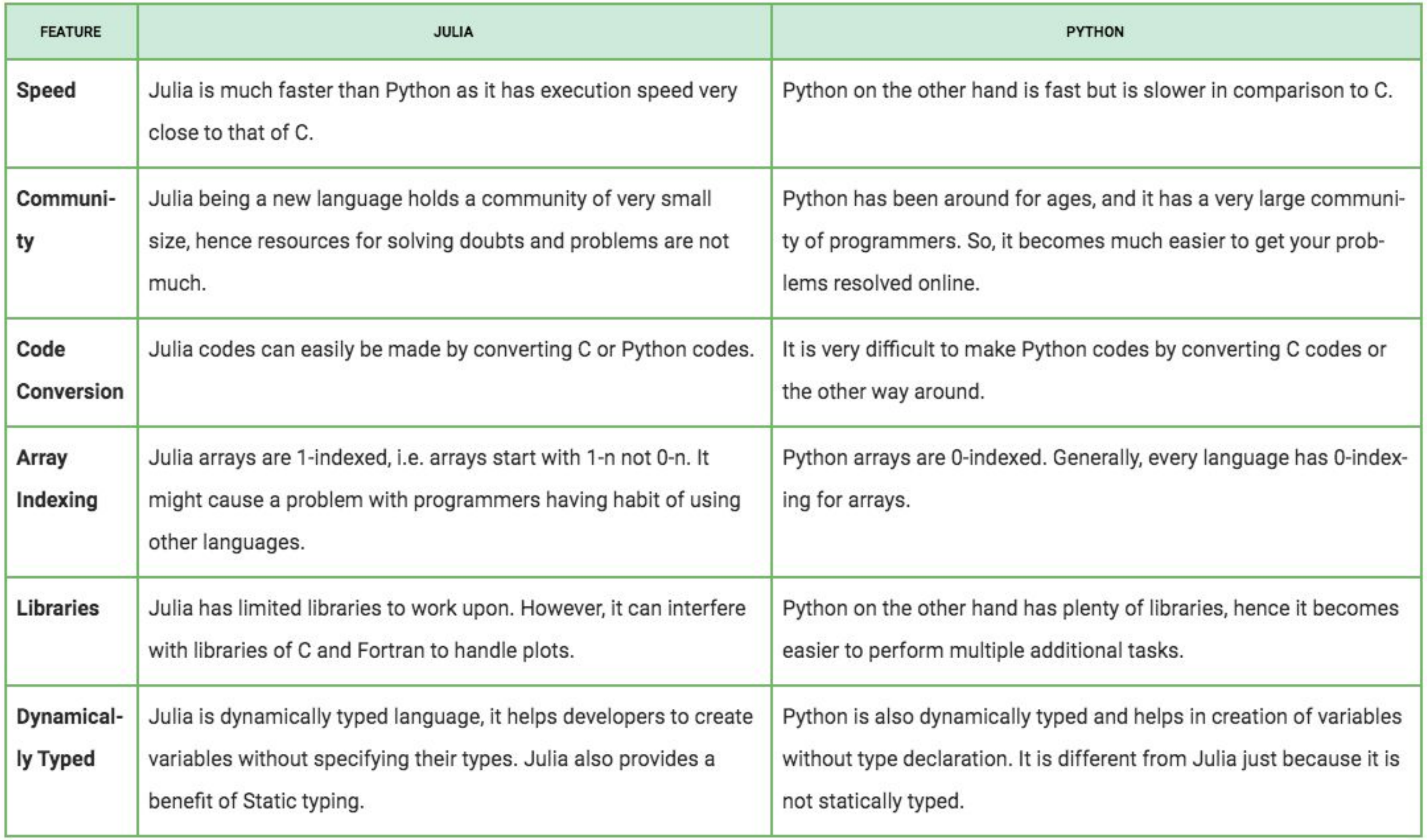
Syntax comparison (Julia Vs python)

Julia

- **Conditions**
- **Functions**

Python

- **Conditions**
- **Functions**



FEATURE	JULIA	PYTHON
Speed	Julia is much faster than Python as it has execution speed very close to that of C.	Python on the other hand is fast but is slower in comparison to C.
Community	Julia being a new language holds a community of very small size, hence resources for solving doubts and problems are not much.	Python has been around for ages, and it has a very large community of programmers. So, it becomes much easier to get your problems resolved online.
Code Conversion	Julia codes can easily be made by converting C or Python codes.	It is very difficult to make Python codes by converting C codes or the other way around.
Array Indexing	Julia arrays are 1-indexed, i.e. arrays start with 1-n not 0-n. It might cause a problem with programmers having habit of using other languages.	Python arrays are 0-indexed. Generally, every language has 0-indexing for arrays.
Libraries	Julia has limited libraries to work upon. However, it can interfere with libraries of C and Fortran to handle plots.	Python on the other hand has plenty of libraries, hence it becomes easier to perform multiple additional tasks.
Dynamically Typed	Julia is dynamically typed language, it helps developers to create variables without specifying their types. Julia also provides a benefit of Static typing.	Python is also dynamically typed and helps in creation of variables without type declaration. It is different from Julia just because it is not statically typed.

Julia Social Perspective

- Even though it can be used as a general language and has grown in many ways, including server side, Julia was originally designed for numerical/technical computing. Primarily intended for scientific computing and math.
- Creators wanted a language that would be as fast as C but as easy as Python.
- Developed in 2009 by Jeff Bezanson, Stefan Karpinski, Viral B. Shah, and Alan Edelman whom wanted to create a language that was easy to write and fast, with an intention of being used in Data Science

“We want a language that’s open source, with a liberal license. We want the speed of C with the dynamism of Ruby. We want a language that’s homoiconic, with true macros like Lisp, but with obvious, familiar mathematical notation like Matlab. We want something as usable for general programming as Python, as easy for statistics as R, as natural for string processing as Perl, as powerful for linear algebra as Matlab, as good at gluing programs together as the shell. Something that is dirt simple to learn, yet keeps the most serious hackers happy. We want it interactive and we want it compiled.” - Jeff Bezanson Stefan Karpinski Viral B. Shah Alan Edelman, some of the creators of Julia

Features

- Supports both static and dynamic typing
- Type inference
- Compiled (not interpreted)
- Supports C, Python, Fortran libraries
- Comes with its own extensive library for mathematical functions
- Open source and free
- Handles complex data analytics with ease
- Just-in-time compilation makes speed comparable to C

Pros

- Very fast language
- Concise
- Has garbage collection
- Math-friendly syntax … more math friendly than python
- Has parallelism … allowing more concurrency
- Straight-forward syntax
- Adds math symbols and other special characters to console for better understanding. Ex: showing √ instead of /sqrt
- Code in the backend of all packages is written in Julia, meaning it is all under one code base and easier to read

Cons

- New language, you must rely heavily on the site documentation
- Way more packages have been built for Python as well as a bigger community.
- Way less used in the industry, but popularity is growing
- Indexing starts at 1
- Few examples out there compared to popular languages
- Slower start up time than python
- “General purpose” but not like python. Syntax is geared towards math-heavy programming like machine learning and data-science

Applications and Use-Cases

- Scientific computations
- Working with big data
- Web Programming (packages available to make your code embeddable)
- Fast machine learning and artificial intelligence

Julia Extra Info

Julia Main Website

- https://julialang.org/learning/

Julia Documentation

- https://docs.julialang.org/en/v1/

Introduction to Julia courses provided by Julia

- https://juliaacademy.com/courses

Citations

https://www.geeksforgeeks.org/julia-language-introduction/