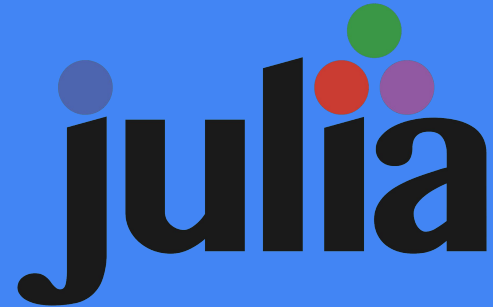


Teoría De Lenguajes


Teoría de la Programación

Aproximación de pi por Montecarlo



Historia

- Empezó a desarrollarse en 2009
- Creado en el 2012
- Actualización de 2018 a Julia 1.0



¿Qué lenguaje para programación general me recomendarías?


¿Y es muy rápido?

¿En esos se hacen los cálculos matemáticos más complejos?

Python, sobre todo si estás empezando a programar

Si buscas velocidad es mejor C o Fortran

Podrías, pero Matlab es mejor sobre todo en álgebra lineal



¿Alguno destaca en procesamiento cadenas?

¿Y para estadísticas?

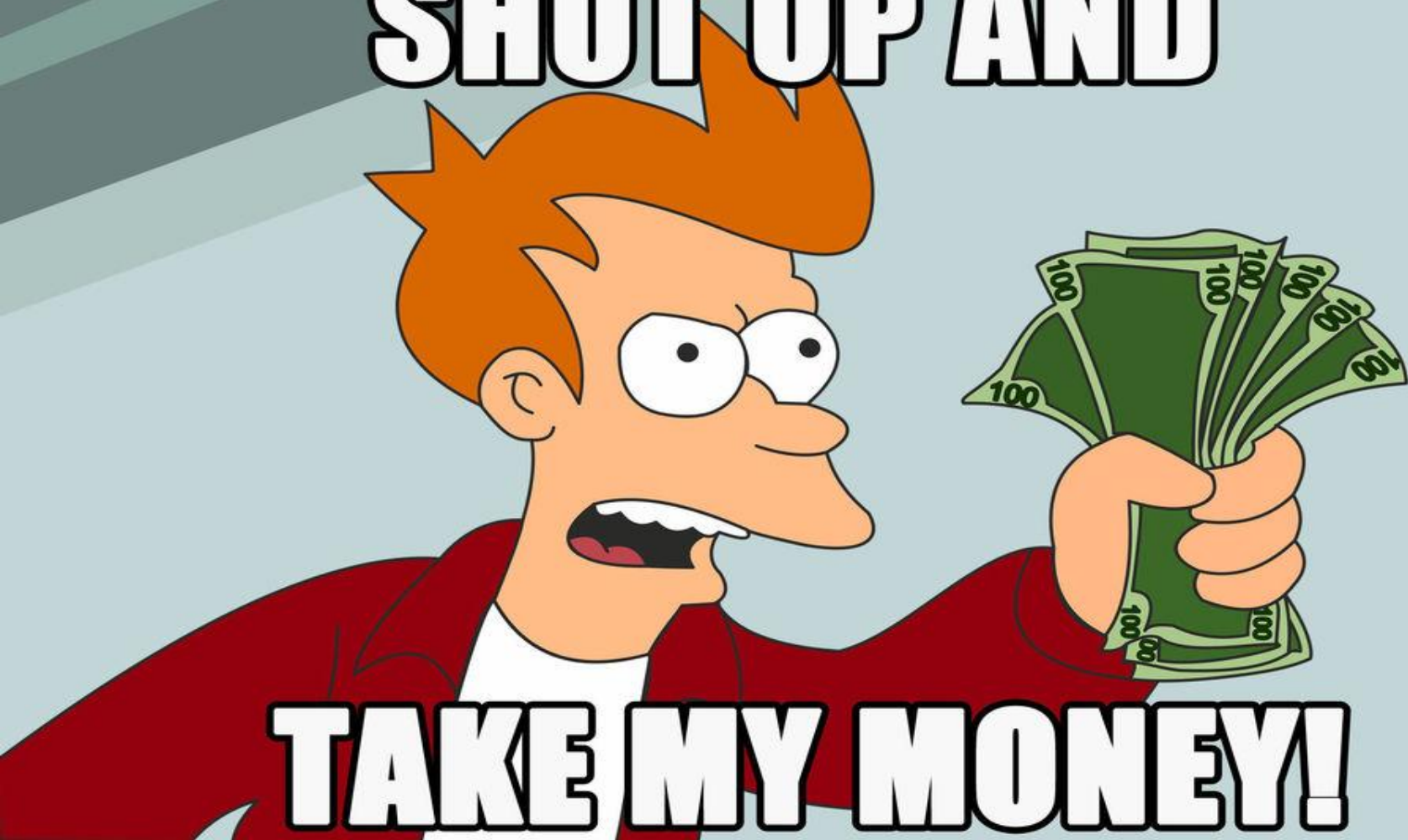
¿Ningún lenguaje es bueno en todo?

Perl, lo hace de forma muy natural

Hay varios pero R es el que usaría

Ojala algun dia llegue ese lenguaje

SHUT UP AND



TAKE MY MONEY!

Bibliografía

- Actualización a Julia 1.0: <https://julialang.org/blog/2018/08/one-point-zero-es/>
- Características de Julia: <https://learnxinyminutes.com/docs/es-es/julia-es/>

Introducción al método de Montecarlo



Un poco de historia

El método moderno lo ideó Stanislaw Ulam y a John von Neumann en la década del 40.

La creación se basó en la observación de que era más fácil analizar los resultados aleatorios del juego de cartas solitario que calcularlo analíticamente.

El método fue usado para resolver integrales y problemas físicos de dispersión de electrones.

Montecarlo la capital del juego de azar

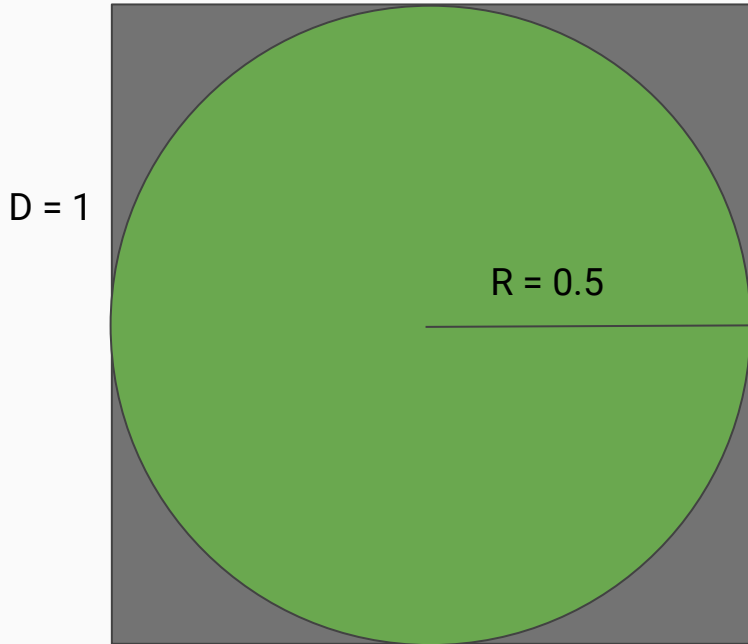


Puntos claves

- Es un método numérico para aproximar resultados
- Se utilizan números aleatorios como base
- Cada problema tiene su propia implementación del método más allá de la secuencia aleatoria de números
- Se basa en que es muy fácil generar muchos casos aleatorios computacionalmente para poder aproximar cálculos que analíticamente serían muy complicados

Calculo de pi

Búsqueda de la relación



D = Largo del lado del cuadrado
igual a 1

R = Radio del círculo ($D/2$)

Vamos a ir generando números aleatorios y mapeandolos en la superficie e ir contando los que nos caen en el círculo.

Cálculo de pi

$$\frac{\text{Áreal del círculo}}{\text{Áreal del cuadrado}} = \frac{\text{Puntos en el círculo}}{\text{Puntos en el cuadrado}} \quad (1)$$

$$\frac{\pi * R^2}{D^2} = \frac{\text{Puntos en el círculo}}{\text{Puntos en el cuadrado}} \quad (2)$$

Cálculo de pi

$$2 * R = D \quad (3)$$

$$\frac{\pi * R^2}{4R^2} = \frac{\text{Puntos en el circulo}}{\text{Puntos en el cuadrado}} \quad (4)$$

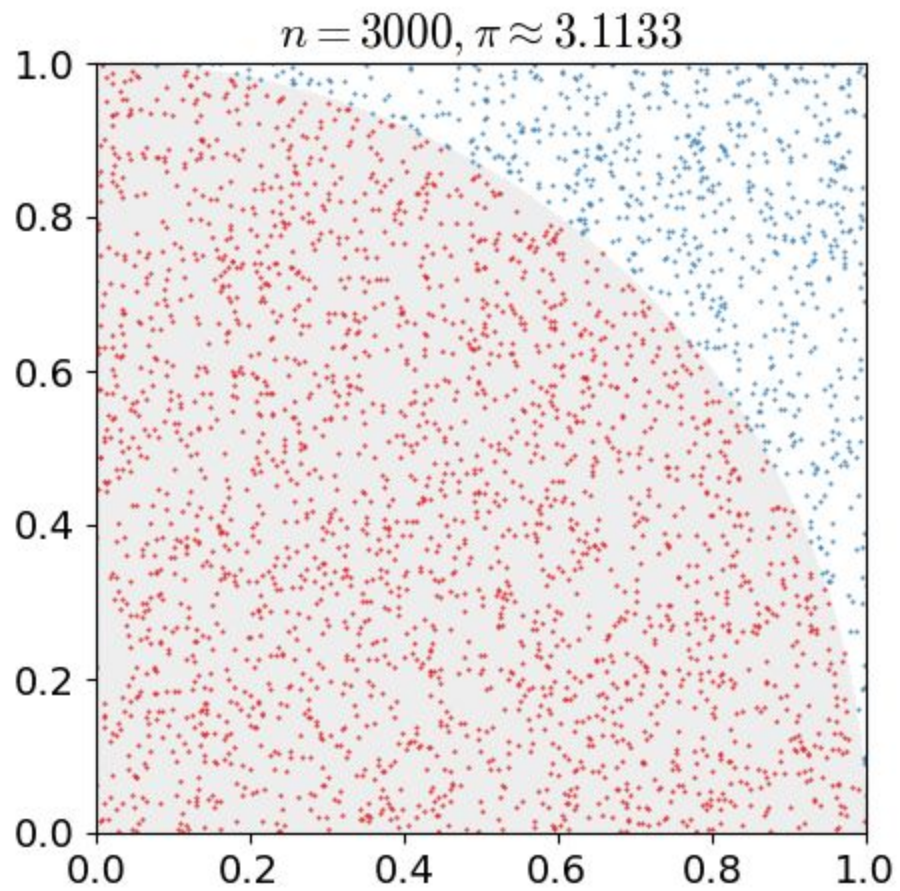
Cálculo de pi

$$\frac{\pi}{4} = \frac{\text{Puntos en el circulo}}{\text{Puntos en el cuadrado}} \quad (5)$$

$$\pi = 4 * \frac{\text{Puntos en el circulo}}{\text{Puntos en el cuadrado}} \quad (6)$$

Cálculo de pi

- Sabiendo los puntos a calcular
- Contamos los puntos que caen en el círculo
- Despejamos el valor de pi de la ecuación

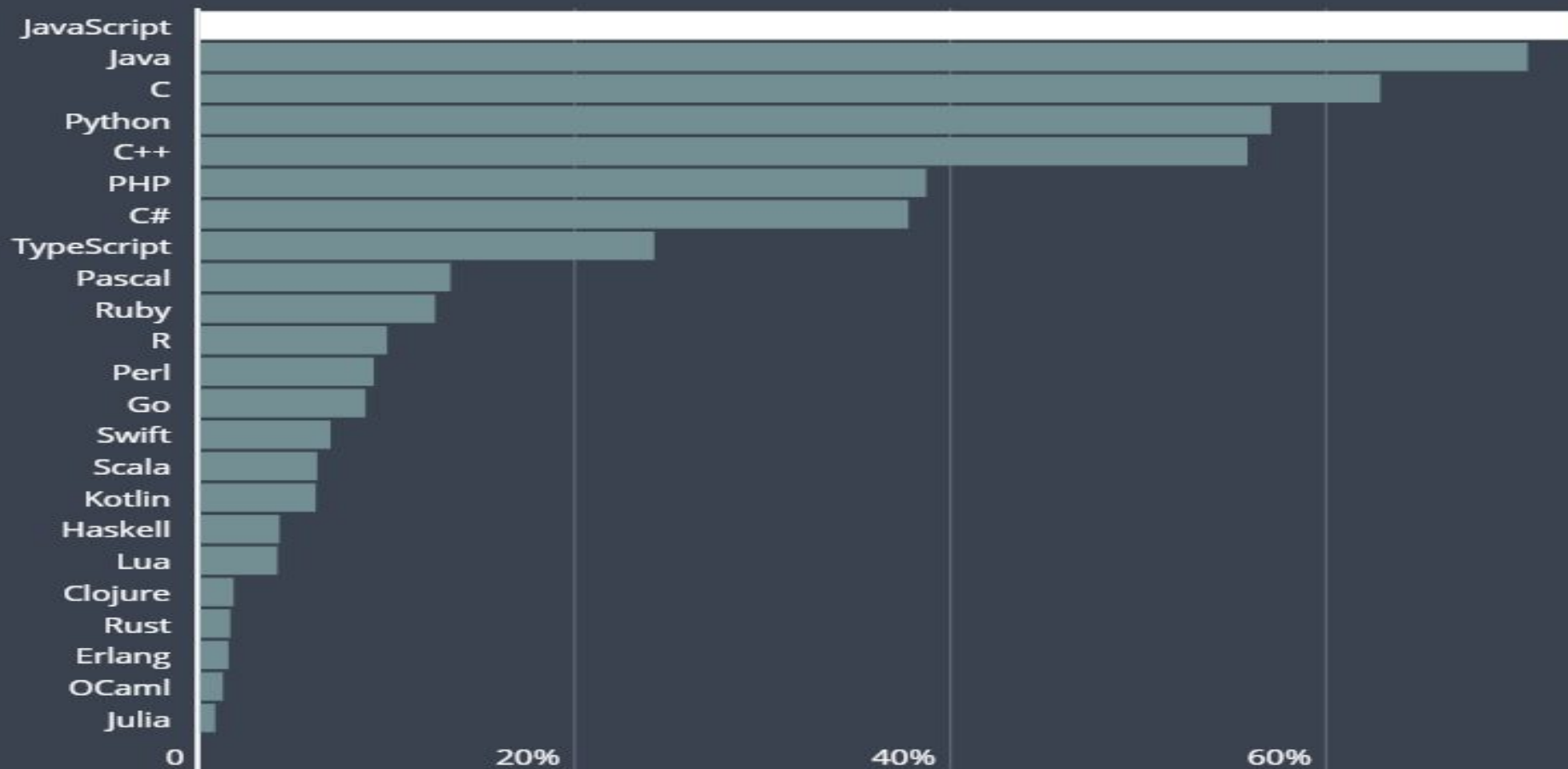


Bibliografía

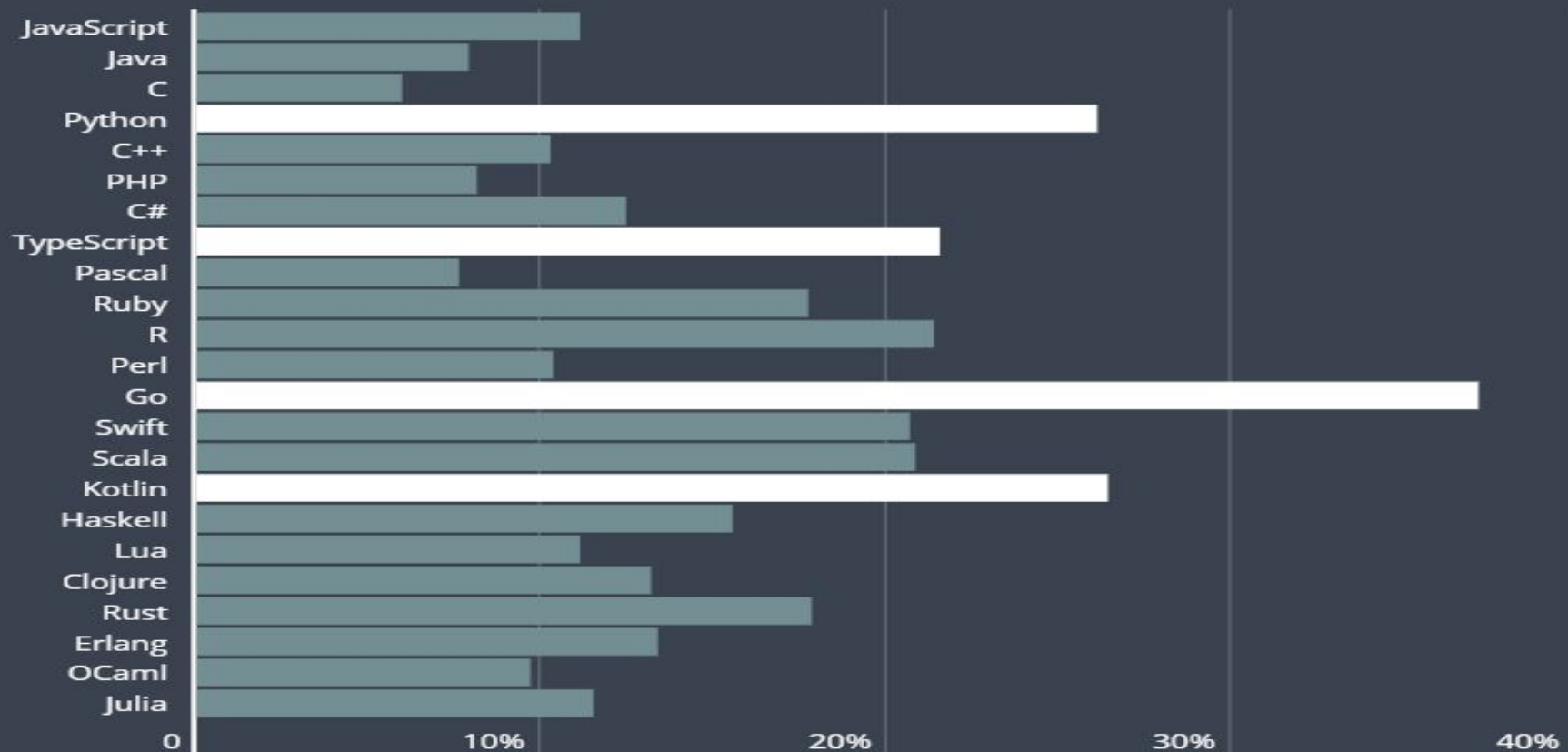
- Introducción al método de Montecarlo: https://en.wikipedia.org/wiki/Monte_Carlo_method
- Cálculo de pi con el método de Montecarlo:
https://github.com/BostonCollegeQAC/Intro_to_Monte_Carlo/blob/master/MC01_An_Invitation_to_Monte_Carlo.ipynb
- Documentación sobre Julia: <https://docs.julialang.org/en/v1/>

Comparación con otros lenguajes

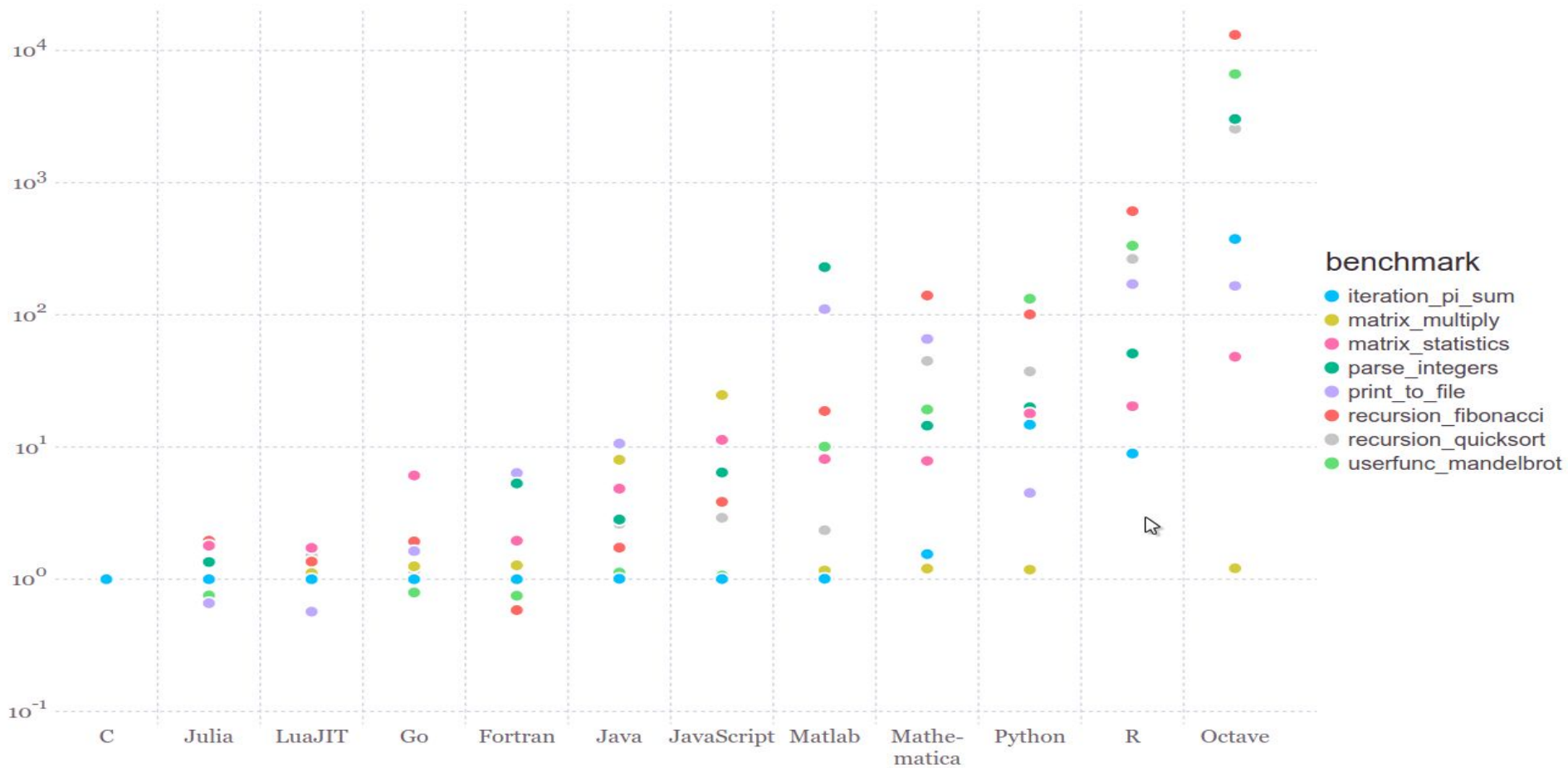
Lenguajes de programación más usados



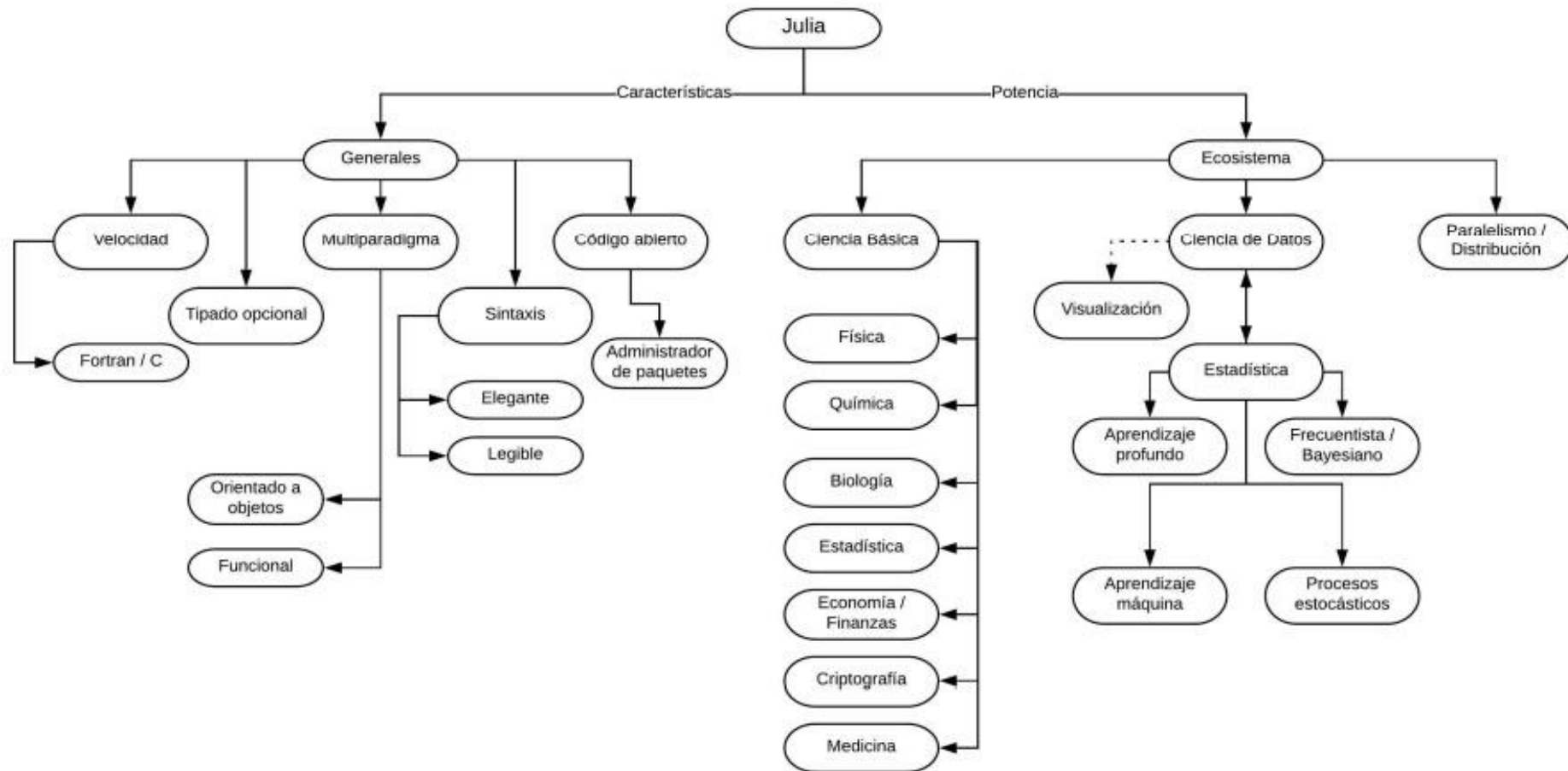
Lenguajes de programación que se desean aprender



Comparación de velocidad



Características y potencia de Julia



Bibliografía

- Graficos de uso:
<https://ingenieriadesoftware.es/lenguajes-programacion-populares-todos-rankings/>
- Comparación de rendimiento:
<https://www.analyticslane.com/2018/09/24/conoces-el-lenguaje-de-programacion-julia/>
- Un poco mas de Julia:
https://tecdigital.tec.ac.cr/revistamatematica/Secciones/Didactica_y_Software/RevistaDigital_Hcuevas_V20_n2_2020/RevistaDigital_HCuevas_V20_n2_2020.pdf
<https://riptutorial.com/Download/julia-language-es.pdf>
<https://www.youtube.com/watch?v=LbTbs-0pOuc>
<https://julialang.org/learning/>