

Comparación de diferentes lenguajes con respecto al tiempo de ejecución con ciertos algoritmos

Alumno: Felix David Prado Apaza



Introduccion

La necesidad de ser un genio de las matemáticas para aprender código es cosa del pasado. Más lenguajes de programación de alto nivel ofrecen una alternativa al código de máquina de bajo nivel, lo que hace que la codificación sea más accesible que nunca.

Pero con docenas de idiomas disponibles, ¿cuál es el mejor para desarrollar un proyecto? Independientemente de lo que se planea plasmar, lo mejor es aprender cuando y quienes usan un determinado lenguaje.

The slide features decorative geometric patterns in the corners, consisting of thin blue lines, dots, and circles. In the top-left, there are several parallel lines and a small cluster of dots. The top-right has a circle with a dot inside and a line with a dot. The bottom-left shows a circle with a dot, a line with a dot, and a small cluster of dots. The bottom-right features a circle with a dot, a line with a dot, and a small cluster of dots.

01

Algoritmos

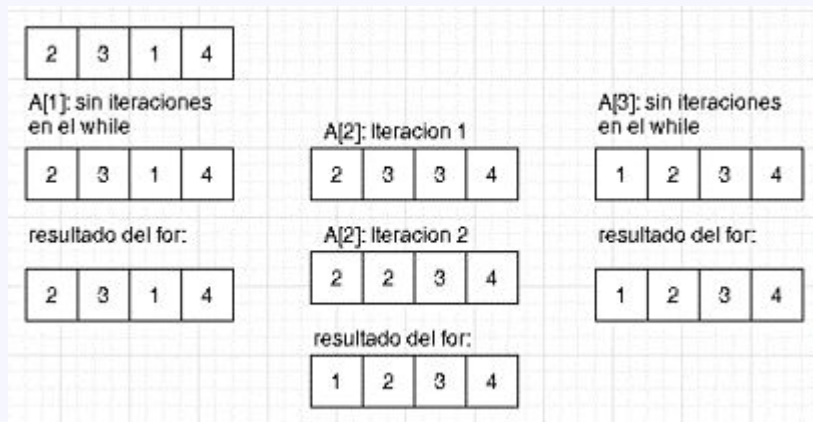
Quicksort

Insertion Sort

Insertion Sort

Este algoritmo es uno de los algoritmos más simples con una implementación simple.

Básicamente, la ordenación por inserción es eficiente para valores de datos pequeños. La ordenación por inserción es de naturaleza adaptativa, es decir, es adecuada para conjuntos de datos que ya están parcialmente ordenados.



Insertion Sort

Pseudocódigo

InsertionSort(A)

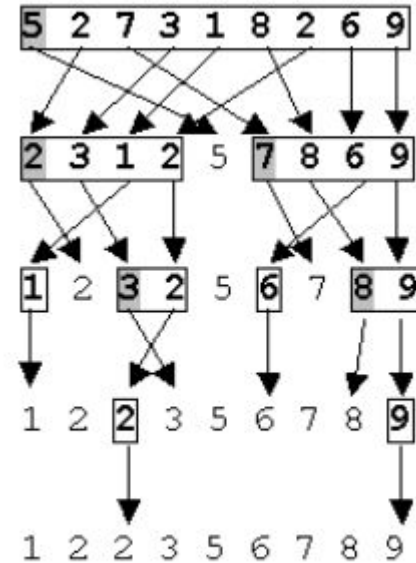
1. for $j \leftarrow 2$ to $length[A]$
2. $key \leftarrow A[j]$
3. $i \leftarrow j - 1$
4. while $i > 0$ and $A[i] > key$
5. $A[i + 1] \leftarrow A[i]$
6. $i \leftarrow i - 1$
7. $A[i + 1] \leftarrow key$

Quicksort

Quicksort es un algoritmo de clasificación basado en el enfoque divide y vencerás donde

Una matriz se divide en subarreglos seleccionando un elemento pivote (elemento seleccionado de la matriz).

Al dividir la matriz, el elemento pivote debe colocarse de tal manera que los elementos menores que el pivote se mantengan en el lado izquierdo y los elementos mayores que el pivote estén en el lado derecho del pivote.



Quicksort

Pseudocodigo

```
quickSort(array, leftmostIndex, rightmostIndex)
  if (leftmostIndex < rightmostIndex)
    pivotIndex <- partition(array, leftmostIndex, rightmostIndex)
    quickSort(array, leftmostIndex, pivotIndex - 1)
    quickSort(array, pivotIndex, rightmostIndex)

partition(array, leftmostIndex, rightmostIndex)
  set rightmostIndex as pivotIndex
  storeIndex <- leftmostIndex - 1
  for i <- leftmostIndex + 1 to rightmostIndex
    if element[i] < pivotElement
      swap element[i] and element[storeIndex]
      storeIndex++
  swap pivotElement and element[storeIndex+1]
  return storeIndex + 1
```

The slide features a light blue background with abstract geometric patterns in the corners. These patterns consist of thin blue lines, dots, and circles, some of which are arranged in a way that suggests a circuit or a network. The patterns are more dense in the corners and become sparser towards the center.

02

Implementación

C++ - Python - Golang

Insertion Sort

Descripción de la implementación

Todos siguen la misma lógica del pseudocódigo, sin usar librerías para el ordenamiento o algún proceso del algoritmo; ya en el main de cada uno se realiza la lectura de los ficheros que contienen los elementos diferentes y repartidos aleatoriamente para generar el caso promedio, se lee cada uno de estos, se itera 5 veces, se calcula el tiempo promedio y desviación estándar y se imprime, siendo ese el output.



The background features decorative geometric patterns in the corners, consisting of thin blue lines, dots, and concentric circles. The top-left corner has a cluster of lines and dots. The top-right corner has a circle with a dot inside and a line with a dot. The bottom-left corner has a circle with a dot inside and a line with a dot. The bottom-right corner has a circle with a dot inside and a line with a dot.

03

Resultados de las pruebas

Pruebas en Insertion Sort

Complejidad $O(n^2)$


Nº de Pruebas 15

Nº de iteraciones por Prueba 5

C++ - Golang - Python

Tabla de Resultados en C++

n	Tiempo Promedio	Desviación Estándar
100	0.0002	0.000447214
1000	0.0016	0.000547723
2000	0.0102	0.00083666
3000	0.0242	0.00083666
4000	0.0454	0.00207364
5000	0.0698	0.00083666



6000	0.107	0.00678233
7000	0.142	0.00316228
8000	0.187	0.00367423
9000	0.2466	0.0186091
10000	0.295	0.00463681
20000	1.137	0.0317096
30000	2.679	0.0580991
40000	4.9324	0.0461335
50000	4.2508	0.0671245

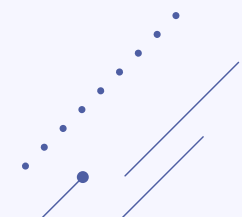



Tabla de Resultados en Golang

n	Tiempo Promedio	Desviación Estándar
100	0.005856	0.000000474552
1000	0.000404208	0.00000327922
2000	0.0125483826	0.00013310592
3000	0.0042027216	0.000693278
4000	0.0177015843	0.001901152
5000	0.0195619374	0.0286178306



6000	0.026065085	0.032125558
7000	0.0420151652	0.03160070
8000	0.059864852	0.02883441
9000	0.0678044218	0.03248487
10000	0.0969972826	0.03144509
20000	0.3137747325	0.11182786
30000	0.7502539888	0.10584663
40000	1.6356149908	0.118808819
50000	1.45911639	0.1198302

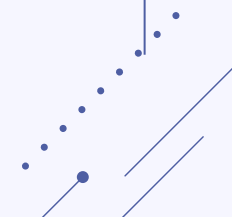

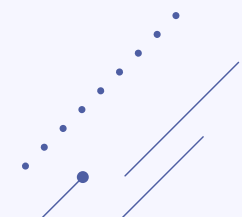


Tabla de Resultados en Python

n	Tiempo Promedio	Desviación Estándar
100	0.00100408	0.0000105034
1000	0.0766014	0.00461269
2000	0.408625	0.00815552
3000	1.06069	0.0371665
4000	1.83653	0.0235117
5000	2.92558	0.0462259



6000	4.23121	0.104465
7000	5.7844	0.160853
8000	7.59435	0.0836083
9000	9.81185	0.350118
10000	12.5024	0.618652
20000	46.8308	0.746774
30000	114.431	3.64621
40000	206.628	1.96436
50000	184.463	2.56335



Pruebas en Quicksort

Complejidad $O(n \log n)$


Nº de Pruebas 15

Nº de iteraciones por Prueba 5

C++ - Golang - Python

Tabla de Resultados en C++

n	Tiempo Promedio	Desviación Estándar
100	0.000000	0.000000
1000	0.000000	0.000000
2000	0.000400	0.000548
3000	0.000400	0.000548
4000	0.000800	0.000447
5000	0.001000	0.000000



6000	0.001000	0.000000
7000	0.002000	0.000000
8000	0.002200	0.000447
9000	0.002600	0.000548
10000	0.002800	0.000447
20000	0.004600	0.000548
30000	0.007200	0.000447
40000	0.011000	0.000000
50000	0.010800	0.000447

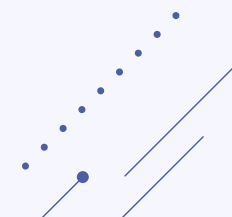



Tabla de Resultados en Golang

n	Tiempo Promedio	Desviación Estándar
100	0	0
1000	0.0001967	0.000439835
2000	0	0
3000	0.00019742	0.0004414445401
4000	0.00040008	0.000547832654
5000	0.0009999	0.00035270



6000	0.0004018	0.00055019
7000	0.00060324	0.00055076
8000	0.00040178	0.00055017
9000	0.00059998	0.000547747
10000	0.00060158	0.000549175
20000	0.0015107	0.000499919
30000	0.00180488	0.00044448697
40000	0.00240008	0.000539469
50000	0.00320328	0.000448416

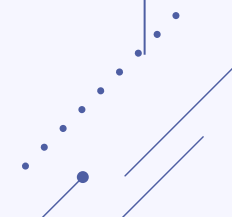

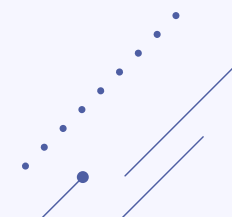


Tabla de Resultados en Python

n	Tiempo Promedio	Desviación Estándar
100	0.000200128555	0.000447501
1000	0.0032030582	0.000448823
2000	0.00720300674	0.0010911628
3000	0.01859903	0.01755869999
4000	0.02039723	0.004339802
5000	0.020998383	0.0000046586




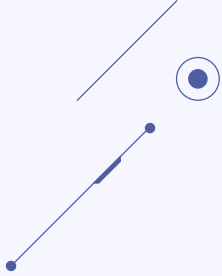
6000	0.03259859	0.00088951
7000	0.04120383	0.000838069
8000	0.049800539	0.00110208
9000	0.05339989662	0.00219090558
10000	0.06799669	0.0015753073
20000	0.1044034004	0.0033638437
30000	0.159396	0.0016763173
40000	0.25059995651	0.006692692
50000	0.2457973	0.0053565269



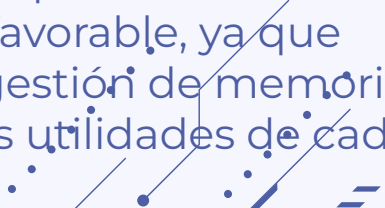
The background features abstract geometric patterns in the corners, consisting of thin blue lines, dots, and concentric circles. The top-left corner has a cluster of lines and dots. The top-right corner has a circle with a dot inside and a line with a dot. The bottom-left corner has a circle with a dot inside and a line with a dot. The bottom-right corner has a circle with a dot inside and a line with a dot.

04

Conclusiones



Bueno, con el resultado de las pruebas, se puede decir que Golang es el más rápido en comparación con los otros 2 (Python quedando como última opción), pero esto se debe a algunas características de Golang, ya que su lenguaje es de alto nivel (C++ es lenguaje de bajo nivel), apunta a un compromiso entre velocidad y simplicidad, logra concurrencia a través de goroutines (a comparación de C++ que tiene simultaneidad lograda a través de subprocesos del sistema operativo), en general, Golang supera a C++ en lo que respecta a la velocidad de codificación, pero esto no quiere decir que Golang sea el lenguaje predeterminado para cualquier proyecto o que Python no sea nada favorable, ya que Python es dinámico, tiene un intuitivo lenguaje, la gestión de memoria es automática, y otros, lo que importa está en ver las utilidades de cada lenguaje





C++

Codificación a nivel
de hardware en
sistemas embebidos



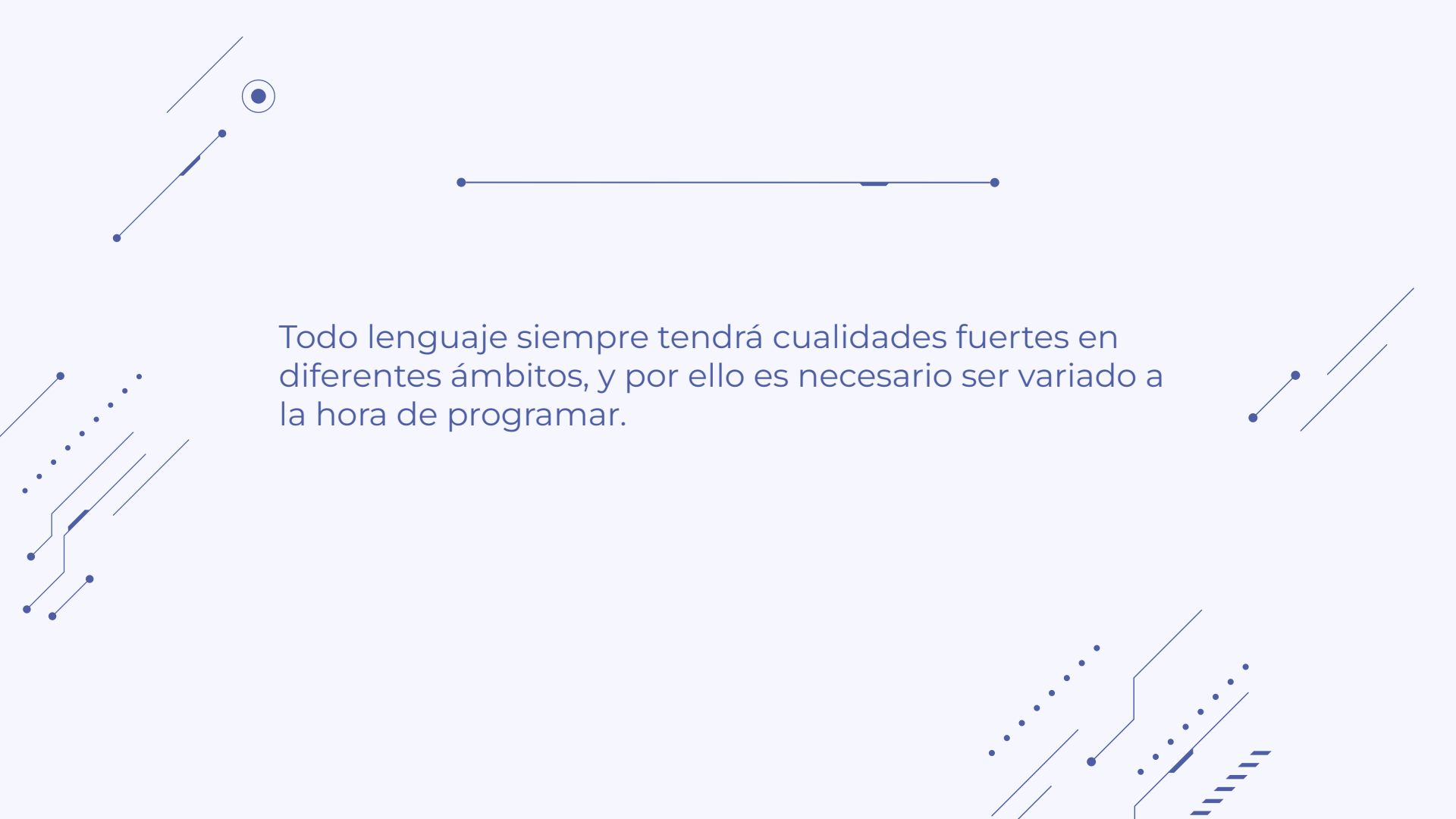
Python

Machine Learning
AI
Back-end



Golang

Backend
APIs RESTful
Scripts para
sistemas
Sockets

The background features a light blue gradient with abstract geometric elements in a darker blue. These include several thin lines of varying lengths and orientations, some with small circular dots at their ends. There are also some thicker, slightly blurred lines and a small cluster of dots in the bottom right corner. The overall aesthetic is clean and modern, suggesting a technical or digital theme.

Todo lenguaje siempre tendrá cualidades fuertes en diferentes ámbitos, y por ello es necesario ser variado a la hora de programar.

The top corners of the slide feature abstract geometric designs. These include thin blue lines, small solid blue dots, and concentric circles, all arranged in a way that suggests a technical or architectural theme.

¡Gracias

por la atención!

A large, empty rectangular box with a thin blue border is positioned on the right side of the slide, below the main text. It appears to be a placeholder for an image or additional content.The bottom corners of the slide continue the abstract geometric theme seen at the top, with thin blue lines, dots, and circles scattered across the space.