

Notación Big O.

Definición

La notación Big O se utiliza en ciencias de la computación para describir el rendimiento o la complejidad de un algoritmo. Específicamente, se enfoca en cómo el tiempo de ejecución o el uso de recursos (como la memoria) crece en función del tamaño de la entrada.

No es compleja

$$O(1)$$

Conforme mayor sea la
cantidad de elementos,
mayor será el beneficio

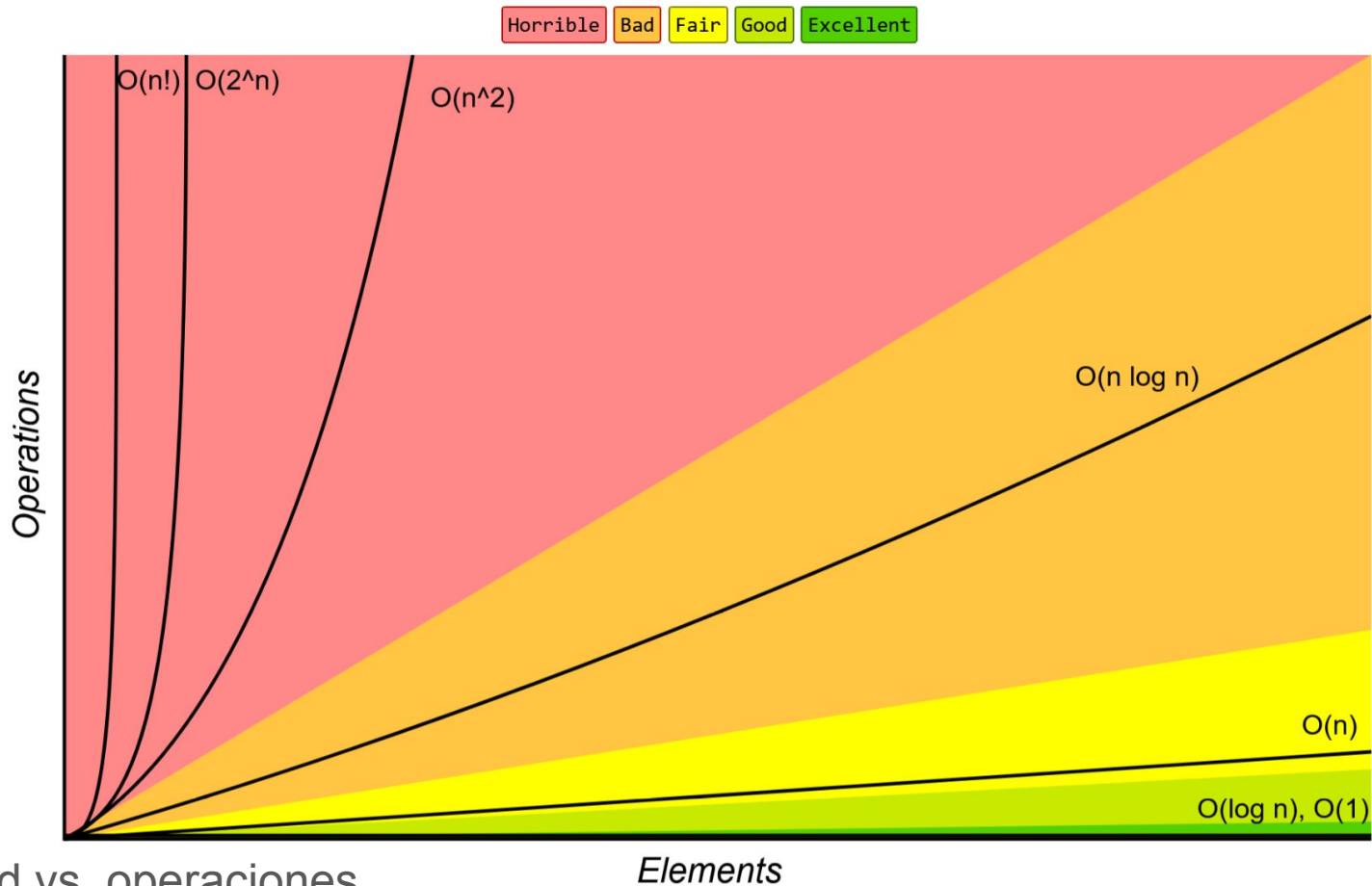
$$O(\log n)$$

Lineal al
número de
elementos

$$O(n)$$

Esta es la complejidad que representan.

Big-O Complexity Chart



DATA STRUCTURE OPERATIONS

Data Structure	Time Complexity								Space Complexity
	Average				Worst				Worst
	Access	Search	Insertion	Deletion	Access	Search	Insertion	Deletion	
Array	$O(1)$	$O(n)$	$O(n)$	$O(n)$	$O(1)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Stack	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$
Singly-Linked List	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$
Doubly-Linked List	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$
Skip List	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n \log(n))$
Hash Table	-	$O(1)$	$O(1)$	$O(1)$	-	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Binary Search Tree	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Cartesian Tree	-	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	-	$O(n)$	$O(n)$	$O(n)$	$O(n)$
B-Tree	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(n)$
Red-Black Tree	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(n)$
Splay Tree	-	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	-	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(n)$
AVL Tree	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(n)$

[Cheatsheet de estructuras de datos según su complejidad.](#)