

O (NOTATION)

llll

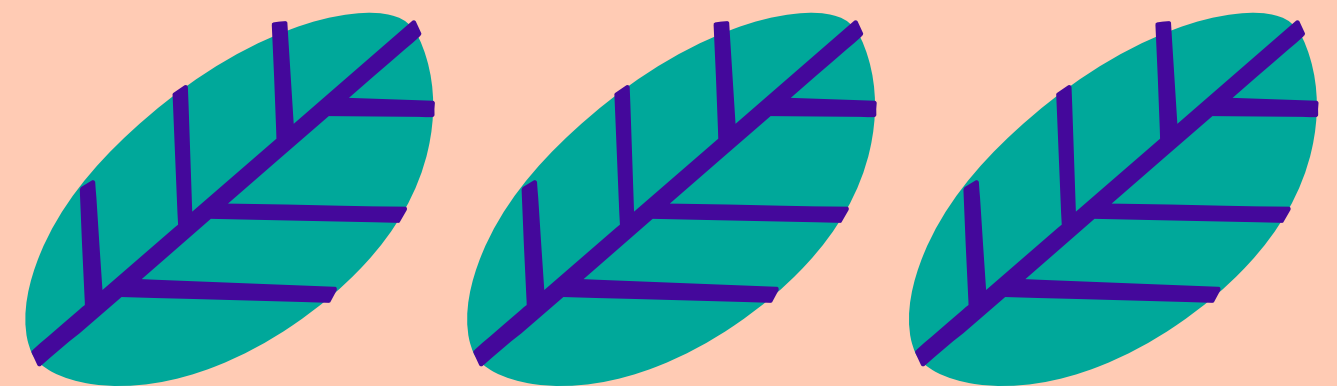
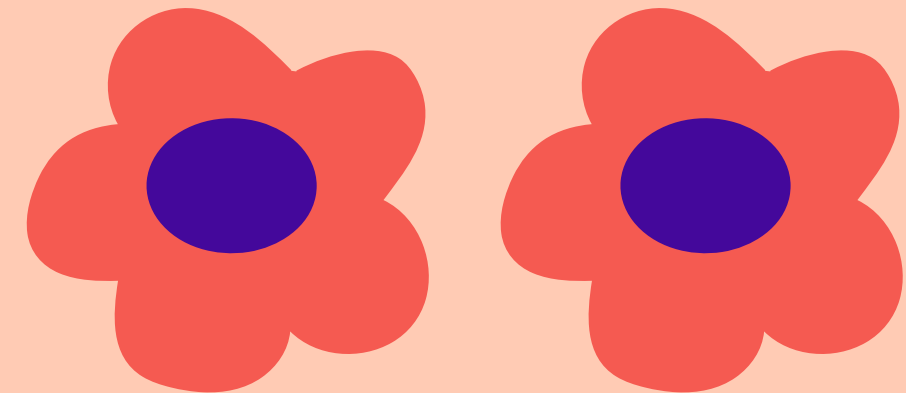
Nos permite dar una
nomenclatura a la
complejidad de un
algoritmo.



Aclaración

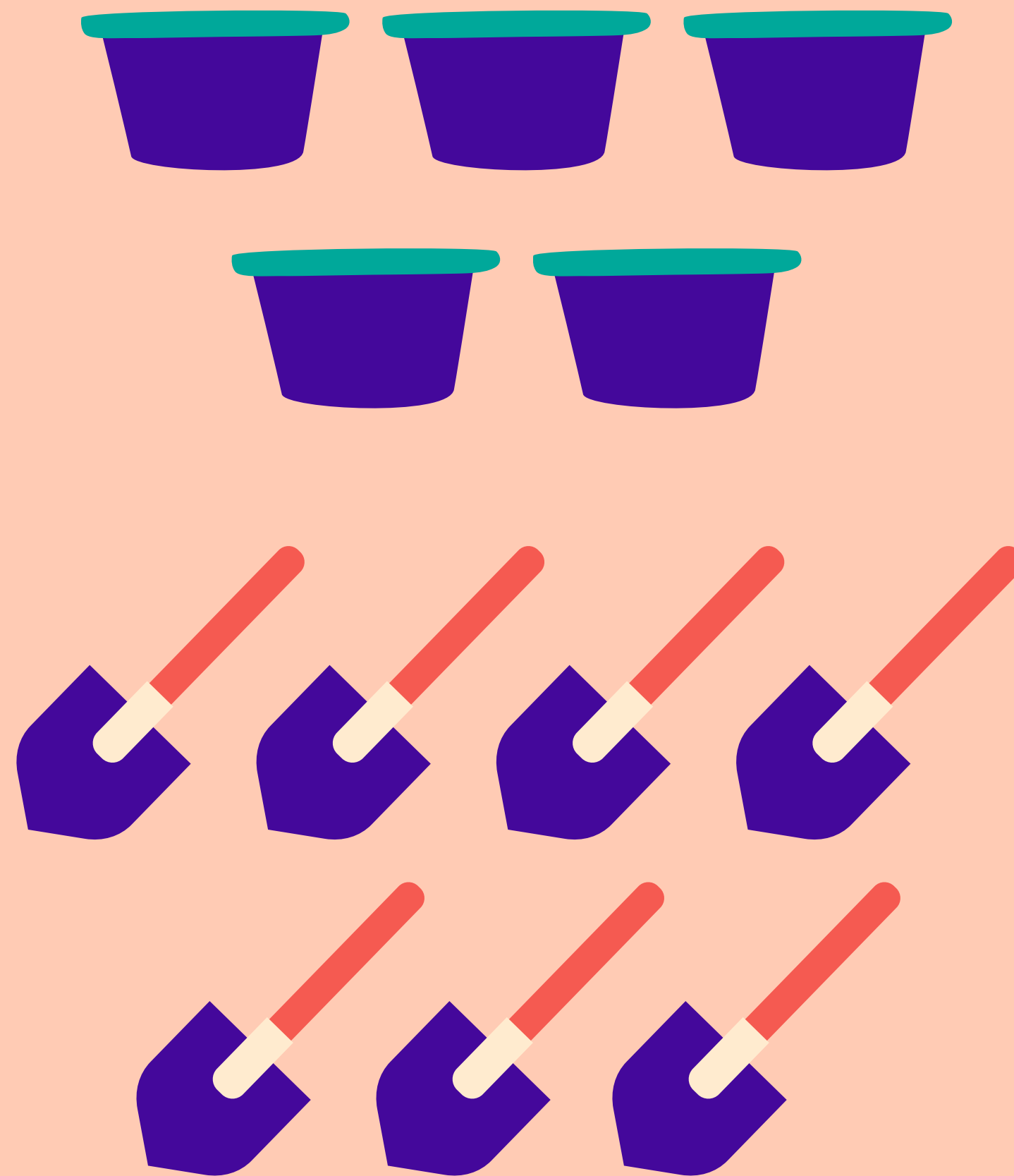


No nos permite
hablar de tiempo
ya que este
depende del
hardware en el
que se ejecute



Describe el
rendimiento y la
complejidad de un
algoritmo sin
importar el
tamaño de
muestras

(generalmente describe el
peor escenario)



$O(1)$

Constante

Esta expresión indica tiempo constante. El algoritmo se ejecuta con el mismo rendimiento sin importar el tamaño del input



FINDBYINDEX

$O(n)$

Notación lineal

Esta expresión hace referencia al crecimiento lineal. La complejidad del algoritmo aumenta de manera proporcional al tamaño del input

**CUALQUIER
ITERACIÓN**

$O(n^2)$

Notación cuadrática

Indica que el crecimiento en complejidad es proporcional al cuadrado del tamaño.

Son los menos eficientes.

No recomendable para muestras grandes

**ITERACIONES
ANIDADAS**

$O(\log n)$

Notación logarítmica

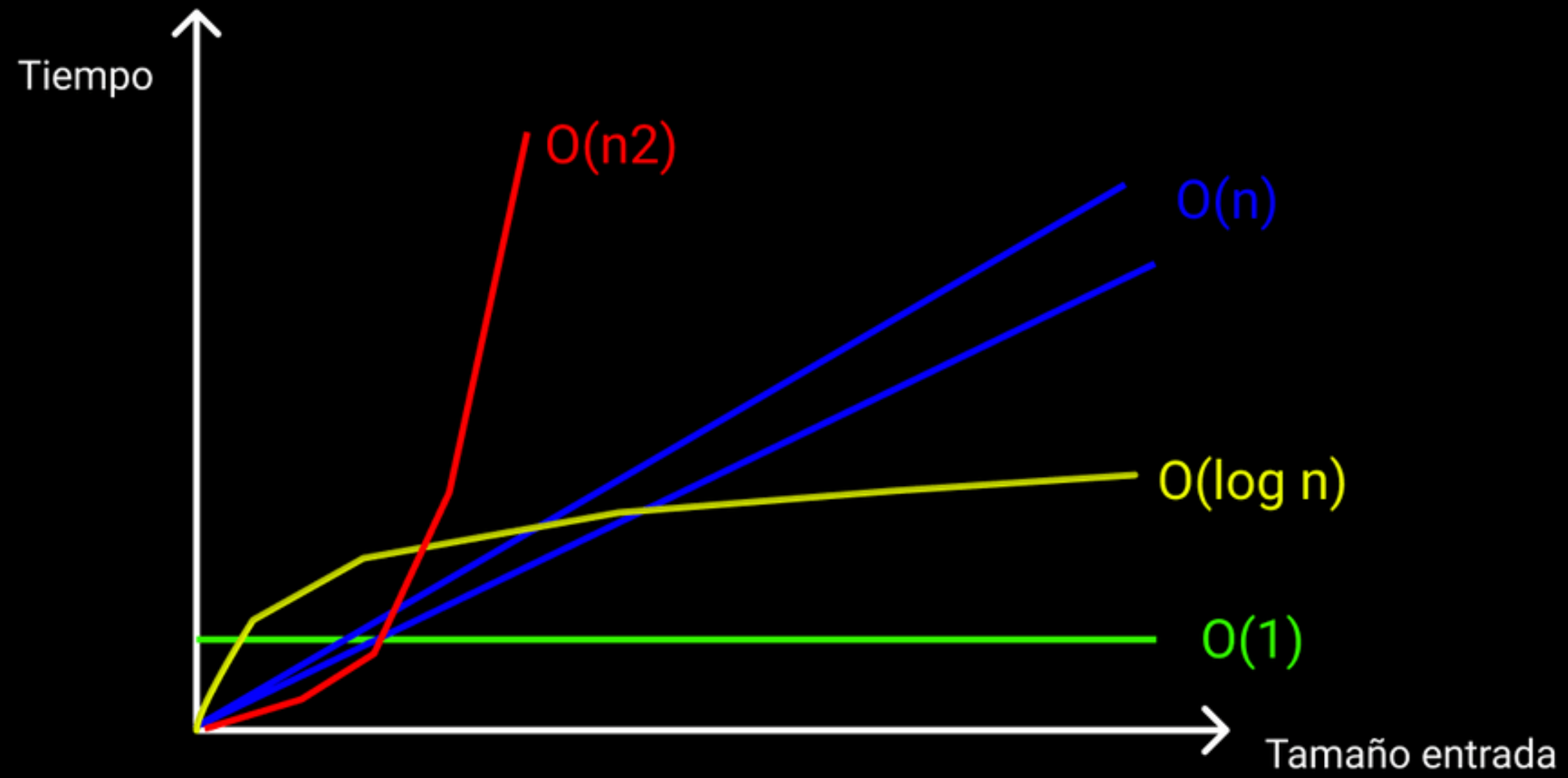
El tiempo aumenta de forma lineal
mientras las muestras son exponenciales.







1 segundo \rightarrow 10 elementos

2 segundos \rightarrow 100 elementos

BINARY SEARCH

Notación Big O



$O(1)$		Speed doesn't depend on the size of the dataset
$O(\log n)$		10x the data means 2x more time
$O(n)$		10x the data means 10x more time
$O(n \log n)$		10x the data means about 20x more time
$O(n^2)$		10x the data take 100x more time
$O(2^n)$		The dilithium crystals are breaking up!

Binary search

steps: 0



1	3	5	7	11	13	17	19	23	29	31	37	41	43	47	53	59
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
low								mid								high

Sequential search

steps: 0



1	3	5	7	11	13	17	19	23	29	31	37	41	43	47	53	59
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16