Big O Notations

learn more

Understanding Big O

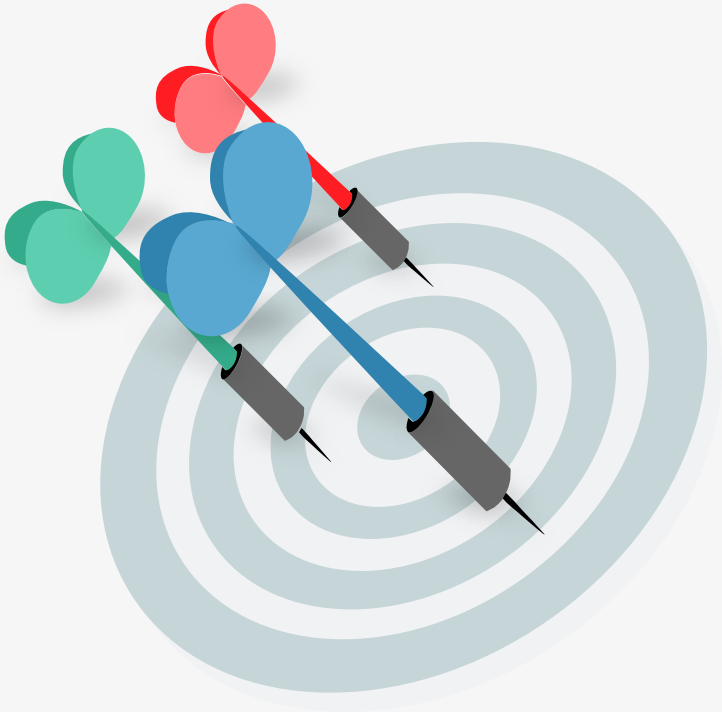
How efficient is an algorithm or piece of code?

01 CPU (time) usage

02 Memory usage

03 Disk usage

04 Network usage



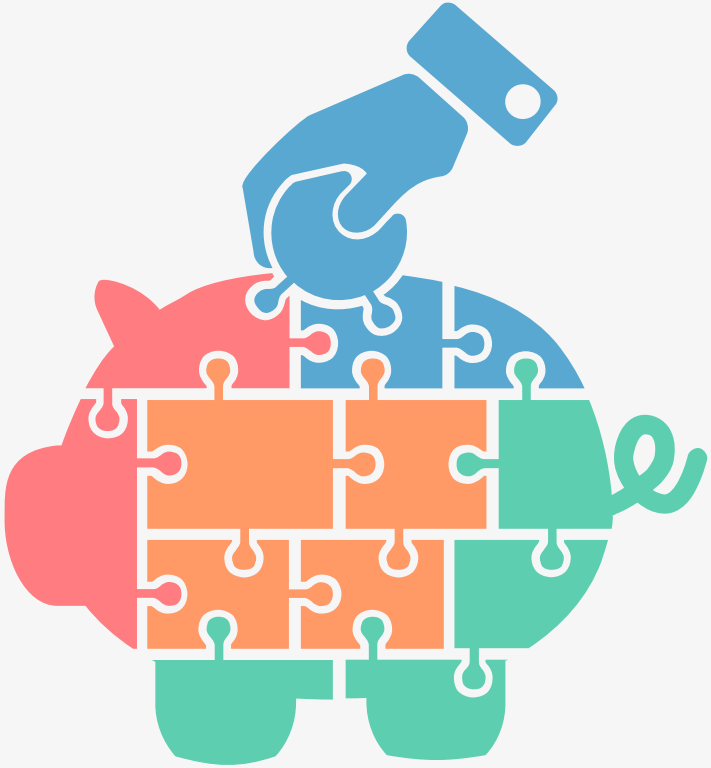
Common Confusion

PERFORMANCE

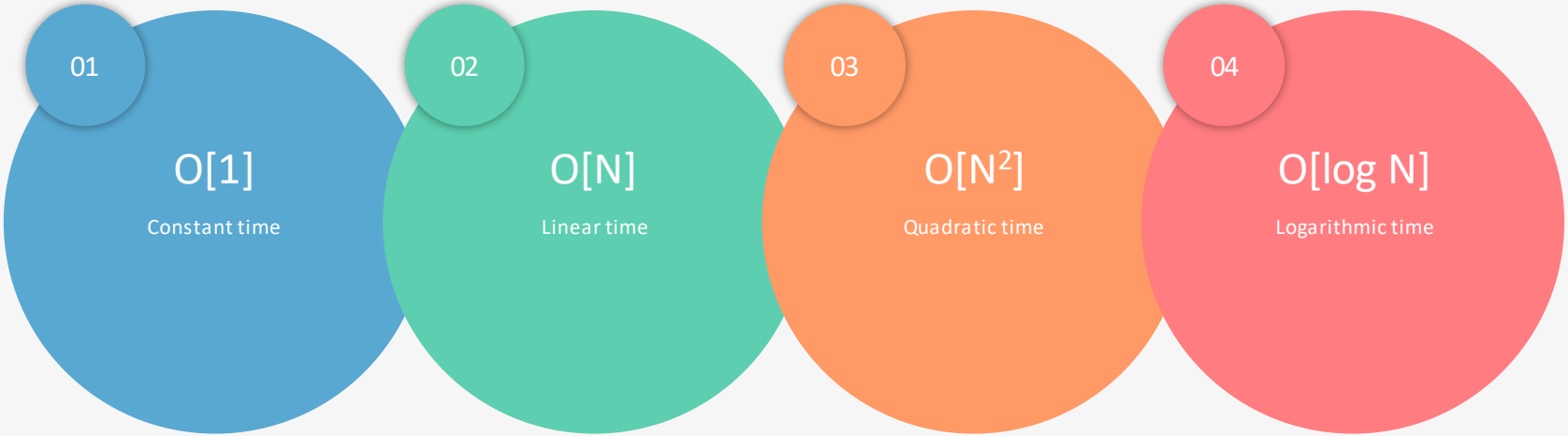
COMPLEXITY

Complexity Measurement

1. One Arithmetic Operation
2. One Assignment
3. One read
4. One write



Big-O Notation



Time Complexity

Constant Time

When we are adding two number programmatically
E.g.: 10 + 12 = 22

10 => **O[1]**

12 => **O[1]**

22 => **O[1]**

O[1] + O[1] + O[1] => **O[3]**

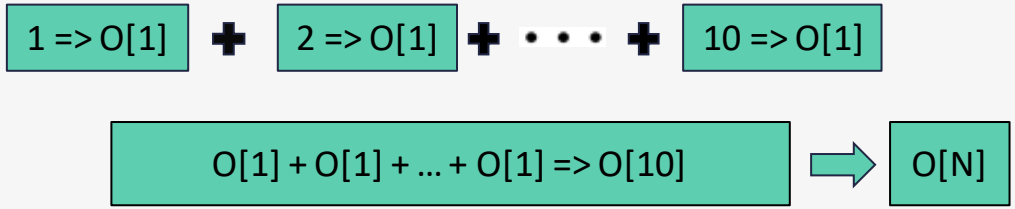
O[3] => **O[C]** ~ **O[1]**

Linear time

Linear to the number of inputs and time

Find the sum of all the elements in the Array

Input = {1,2,3,4,5,6,7,8,9,10}



Problem

Given an array of integers, sort the array without inbuilt function

15 Minutes

Quadratic time

Squared result in time for every increase in input

Input = {3,5,7,1,10,8}

Sorting 3 ⇨ O[N]

Sorting 5 ⇨ O[N]

⋮

Sorting 8 ⇨ O[N]

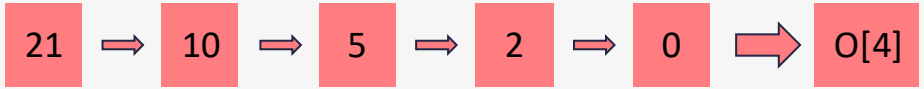
O[N] + O[N] + O[N] + . . . + O[N] ⇨ O[N*N]

~ **O[N^2]**

Logarithmic time

Running time grows in proportion to the logarithm of the input size

Find the atmost divisible by 2 for the any given number



~ **O[logn]**

Some rules to remember

Different Steps Added

O[a] + O[b] => O[a+b] => O[1]

2 conditions outside the loop*

Different inputs => Different variables

O[NXM] Not equal to $O[N^2]$

2 for loops inner with different variables

Drop Constants

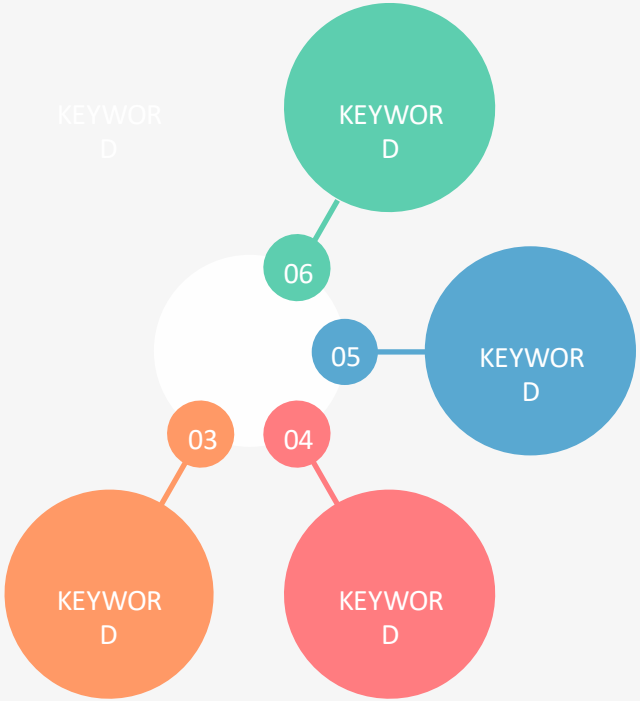
O[2N] => O[N]

2 individual for loops*

Drop Non dominate terms

$O[N^2] + O[N] + O[1] => O[N^2]$

3 loops with one inner and one external and with one condition outside of all loop



Space Complexity

Constant Time

When we are adding two number programmatically
E.g.: 10 + 12 = 22

10 => **4 Bytes**

12 => **4 Bytes**

22 => **4 Bytes**

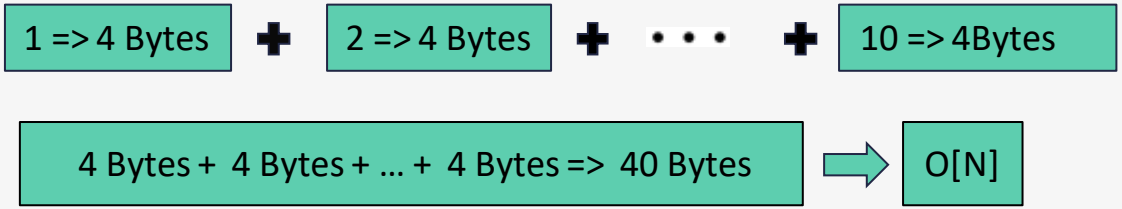
4 + 4 + 4 => **12**

12 => **O[C] ~ O[1]**

Linear time

Linear to the number of inputs and time

Input = {1,2,3,4,5,6,7,8,9,10}



Complexity

	Static Array	Dynamic Array
Access	O(1)	O(1)
Search	O(n)	O(n)
Insertion	N/A	O(n)
Appending	N/A	O(1)
Deletion	N/A	O(n)

First Problem

Given an array of integers, reverse the array without inbuilt function

5 Minutes