Big-O Notation

Column	
Created	@August 16, 2021 9:03 AM
Property	@August 16, 2021 9:26 AM
Property 1	Vatsal P

Time Complexity

A measure of how fast the algorithm runs, TC central concept. Expressed using O(n) notation

Space Complexity

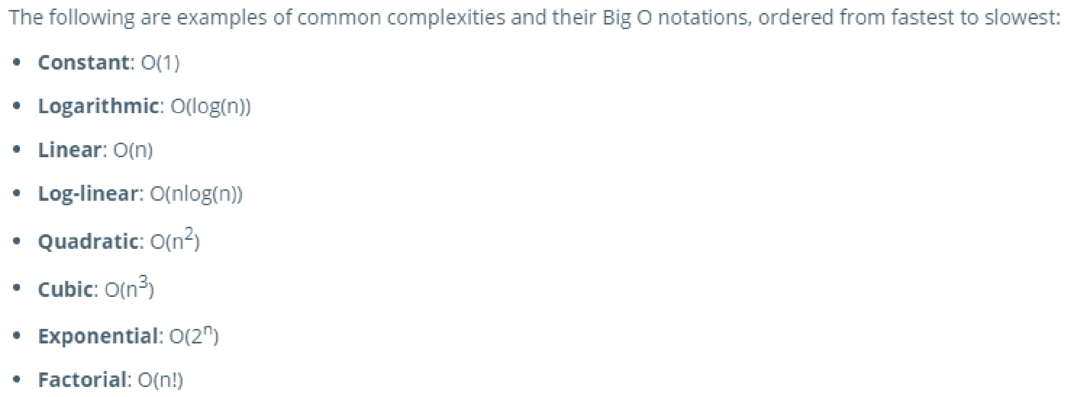
measure how much aux memory an algo takes up. SC central concept. Expressed using O(n) notation

Big O Notation

unit to measure the TC and SC of the algorithm variables in Big O notation denote the size of input to algos.

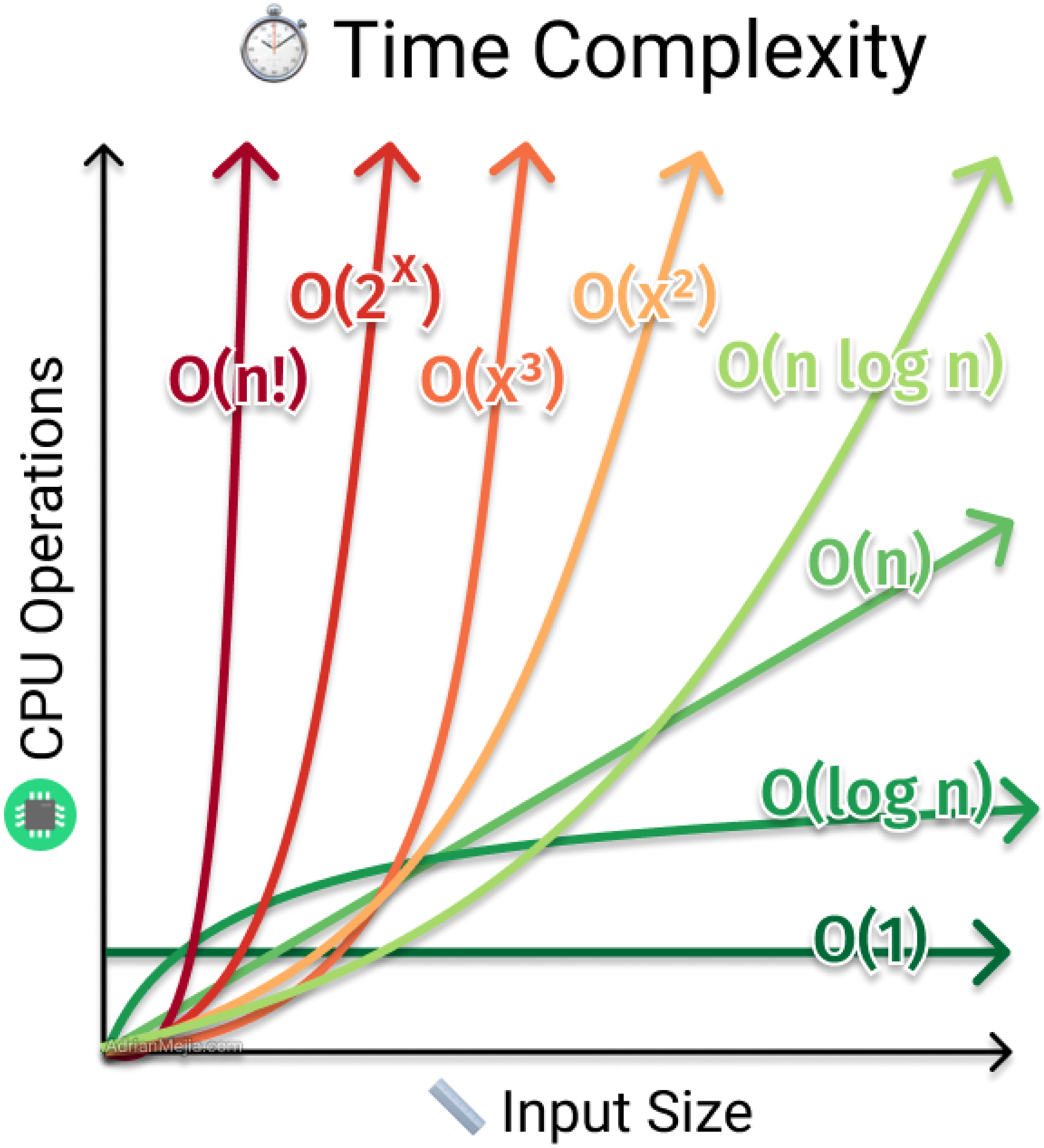
For ex O(n) ⇒ TC of algo, that traverses thru the array of length of n.

For ex O(n + m) ⇒ TC of algo, that traverses thru the array of length of n and thru the string of length of m.



The following are examples of common complexities and their Big O notations, ordered from fastest to slowest:

- **Constant**: O(1)
- **Logarithmic**: O(log(n))
- **Linear**: O(n)
- **Log-linear**: O(nlog(n))
- **Quadratic**: $O(n^2)$
- **Cubic**: $O(n^3)$
- **Exponential**: $O(2^n)$
- **Factorial**: O(n!)



Sequential Statements

statements with basic operations like comparisons, assignments, reading a variable take constant time each O(1).

Conditional Statements

```
if (isValid) {
  array.sort();
  return true;
} else {
  return false;
}
```

if block has a runtime of O(n log n) (that’s common runtime for efficient sorting algorithms). The else block has a runtime of O(1).

Loops

For-Loops / While-Loops

Linear Time Loops

find out the runtime of the block inside them and multiply it by the number of times the program will repeat the loop.

```
for (let i = 0; i < array.length; i++) {
  statement1;
  statement2;
}
```

All loops that grow proportionally to the input size have a linear time complexity O(N). If loop thru half array = O(N) → drop constants

Constant Time Loops

```
for (let i = 0; i < 4; i++) {
  statement1;
  statement2;
}
```

if a constant number bounds the loop, let’s say 4 (or even 400). Then, the runtime is constant O(4) -> O(1).