BIG O

Really…how bad is it?

WHAT IS IT?

- “Order of Magnitude”
  - Gets at efficiency of an algorithm / performance at scale.
- Complexity of algorithm
  - Time
  - Space
- Big O refers to an upper bound
  - Worst case scenario

WHAT IT IS NOT?

- Best Case or Average Case scenario.
  - Best Case - Big Omega (Ω)
  - Average Case - Big Theta (Θ)
- Not useful when algorithm is small in size.
- Doesn’t provide real units of time/space just their magnitude.

HOW DO WE CALCULATE IT?

- **Count the steps**
  - **Anytime you hit a loop, multiply the number of times we iterate by the max complexity of each loop iteration.**
    - **Stuff inside loop** multiplies
    - **Nested Loops**: Multiply
    - **Sibling Loops:** Add
- **Drop the constants**
- **Drop less significant terms**

EXAMPLE 1

```
function doSomething(arr){
 for(let i=0; i < arr.length; i++){
  console.log(i);
 }
 for(let j=0; j < arr.length; j++){
  console.log(j);
 }
}
```

EXAMPLE 2

```
function doSomething(arr){
 for(let i=0; i < arr.length; i++){
  console.log(i);
  for(let j=0; j < arr.length; j++){
   console.log(j);
  }
 }
}
```

DON'T CONFUSE MULTIVARIATE TERMS

- Independent terms count as their OWN runtime.
  - Don't drop independent terms.

https://repl.it/repls/PlayfulSizzlingJaguar

COMMON RUNTIMES

- **O(1):** constant
- **O(log n):** logarthmic complexity
- **O(n):** linear
- **O(n*log n):** log-linear
- **O(n^2)**: quadratic
- **O(n^3), O(n^2), O(n^4):** polynomial
- **O(2^n)**: exponential complexity
- **O(n!)**: factorial

COMMON RUNTIMES

http://bigocheatsheet.com/

SPACE COMPLEXITY

- Really: memory
- Process is similar to deriving time complexity, except you count the space.