**Grokking Algorithms**

* Binary Search Method, Takes O(log n) logarithmic time to compute as compared to the O(n) Linear Time fo simple search
* Big O notation tells you how fast an algorithm is.
* Big O doesn’t tell you the speed in seconds. Big O notation lets you compare the number of operations.
* It tells you how fast the algorithm grows.

**Some common Big O run times**

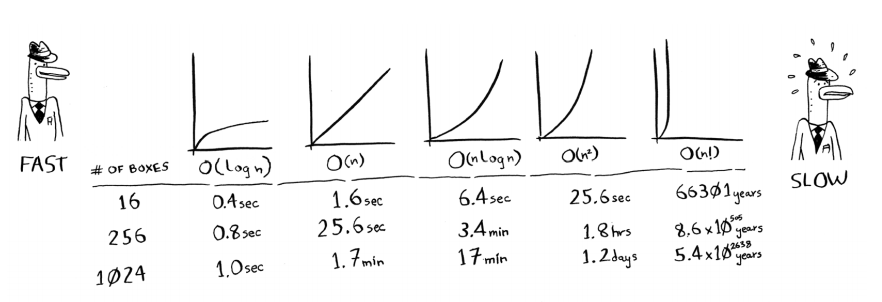
• O(log n), also known as log time. Example: Binary search.

• O(n), also known as linear time. Example: Simple search.

• O(n \* log n). Example: A fast sorting algorithm, like quicksort

• O(n2). Example: A slow sorting algorithm, like selection sort

• O(n!). Example: A really slow algorithm, like the traveling salesperson.



**Key Points**

* Algorithm speed isn’t measured in seconds, but in growth of the number of operations.

• Instead, we talk about how quickly the run time of an algorithm increases as the size of the input increases. • Run time of algorithms is expressed in Big O notation.

• O(log n) is faster than O(n), but it gets a lot faster as the list of items you’re searching grows.

* Constants like 1/2 are ignored in Big O notation

Selection sort is a neat algorithm, but it’s not very fast. Quicksort is a faster sorting algorithm that only takes O(n log n) time.