# Algorithms

Logic based step by step procedure of doing things

\*\*Big O focuses on the efficiency of the algorithm as input increases.

\*\* O(1) is not the fastest, but as input grows the efficiency should be constant.

Constant Complexity

Linear Complexity

-for loops

Quadratic Complexity

-nested loops

Logarithmic Complexity

-binary search: finding the **position** of the desired value in the array.

Factorial Complexity

-bogosort

Searching in an unsorted array

-Linear complexity, since you are going through each item one by one.

Searching through a sorted array

-Logarithmic complexity,

<https://www.bigocheatsheet.com/>

# Recursion

## When to use recursion

When we want to find all possible combinations of something.

Question:

We only know about time complexity,, what about space complexity.

<https://hackernoon.com/coding-interview-recursion-f0d60c9dbb60>

<https://www.byte-by-byte.com/recursion/>

## 6 types of recursions

* Iterations
* Subproblems
  + Towers of Hanoi Problem
  + Fibonacci
* Selection
  + Finding all possible combinations of something
* Ordering
* Divide & Conquer
* Depth First Search

# Sorting

Distribution sort

Comparison sort

Stability: preserve the initial order of the collection

### Bubble sort

1. For every element in the array, compare it with the element on the right, swap its position such that it is in ascending order.
2. The above is an iteration. After the end of the first iteration, the element with the largest value will be bubbled to the end of the array. 2 iteration – 2nd largest value, and so on.
3. There will be n iterations, and in each iteration there will be n steps in the worst case. Thus, the algorithm has a worst case complexity of O(n^2)

### Insertion sort

1. Imagine a temporary array with the first element in it.
2. The next element will be added into this array, where it will be swaps will happen until the temp array is sorted.
3. The temp array will then accept the next element, where it is again sorted internally in the temp array.
4. This happen until the size of the temp array is the size of the original array.

### Merge sort

1. There are two sub algorithms, divide and merge
2. After the array is broken down into 2 sub-arrays they are sorted into order with the merge algorithm.
3. Within each subarray we then apply the same algorithm. This happens recursively until the array is broken down into its individual elements and sorted from the “ground-up”.

## Distribution algorithm

* To determine the number of buckets, take the square-root of the number of elements.
* Elements should not be too dense (i.e. their values cannot be too close together) or the elements cannot be sorted into the different buckets. It doesn’t work if it is too wide, you cannot categorise them too neatly.

### Bucket Sort

### Radix Sort







