Quicksort

Quick Sort Performance

Quicksort's performance can be inefficient when the algorithm encounters imbalanced partitions. The worst case scenario is if the first or last element is always the partition point for an array or sub-array. In this case, one side of the partition will contain all the elements. This makes the recursive stack deeper, resulting in `O(N^2)` runtime.

Quick Sort General

Quicksort is a method for sorting an array by repeatedly partitioning it into sub-arrays by:

1. Selecting an element from the current array. This element is called the pivot element, and in our implementation we used the mid element.
2. Comparing every element in the array to the pivot element, swap the elements into sides greater than and less than. The partition point in the array is where we guarantee everything before is less and everything after is greater than.
3. Repeating this process on the sub-arrays separated by the partition point. Do this until a sub-array contains a single element. When the partitioning and swapping are done, the arrays are sorted from smallest to largest.

The worst case runtime for quicksort is `O(N^2)` and the average runtime for quicksort is `O(N logN)` . The worst case runtime is so unusual that the quicksort algorithm is typically referred to as `O(N logN)` "