**Theory: Quick sort**

**Quicksort** is an efficient in-place sorting algorithm that is often faster in practice compared to other sorting algorithms. The algorithm is based on the divide-and-conquer paradigm.

Here are the steps of quicksort:

1. Pick some element from the array. We will call that element a **pivot**.
2. Reorder the array so that all values smaller than the pivot are positioned before it and all larger values come after; values equal to the pivot can go either way.
3. Recursively sort the subarrays of smaller and greater elements.

The base case of recursion is arrays the size of zero or one, which is in order by definition, so they never need to be sorted.

Quicksortcan be implemented as a recursive or iterative algorithm. Here we will consider only the recursive version.

The time complexity is O(n log n)*O*(*nlogn*) in the average case, and **O(n^2)*O*(*n*2)** in the worst case, but fortunately, it is usually average. We will consider some bad cases later.

Note, there are a lot of modifications that make the algorithm more efficient. The pivot selection and partitioning steps can be implemented in different ways. The choice of a specific implementation strategy greatly affects the algorithm's performance.

## Choosing a pivot

Here are some possible methods of choosing the pivot:

* Pick the leftmost or the rightmost element;
* Pick the middle element;
* Pick a random element;
* Take the first, middle and last value of the array and choose the median of these three numbers as the pivot.