Randomized Quick Sort :-

Sorting is a fundamental problem in computer science. Given a list of n elements of a set with a defined order relation, the objective is to output the elements in sorted order. Quicksort [Hoa62] is a particularly efficient algorithm that solves the sorting problem. We demonstrate how Quicksort works using an example.

Suppose Quicksort was given the task to sort the following 8 numbers.

$$2, 7, 6, 3, 1, 8, 5, 4$$

Quicksort first selects a *pivot* element, say $\boxed{3}$. This pivot element is compared with all the elements on both sides, dividing the list of elements into two lists.

$$2, 7, 6, \boxed{3}, 1, 8, 5, 4$$

After comparisons with the pivot, the remaining elements are divided into two parts: one that consists of the elements smaller than the pivot, and one that consists of the elements larger than the pivot.

$$\underbrace{2, 1,}_{<3} \boxed{3}, \underbrace{6, 7, 8, 5, 4}_{>3}$$

We recurse on both parts.

$$\underbrace{2, 1,}_{\ldots} \boxed{3}, \quad \underbrace{6, 7, 8, 5, 4}$$

$$\underbrace{\boxed{2}, 1}_{\ldots} \qquad \underbrace{6, 7, \boxed{8}, 5, 4}$$

$$\underbrace{6, 7, 5, 4,} \quad \boxed{8}$$

$$\underbrace{6, 7, \boxed{5}, 4,}$$

$$\underbrace{4} \, \boxed{5} \, \underbrace{7, 6}_{\ldots}$$

Although the Quicksort algorithm is very efficient in practice, its worst-case running time is rather slow. When sorting n elements, the number of comparisons may be $O(n^2)$. The worst case happens if the sizes of the subproblems are not balanced. The selection of the pivot element determines the sizes of the subproblems. It is thus crucial to select a 'good' pivot.

In the following, we prove that if the pivot is selected *uniformly at random*, the expected number of comparisons of this randomized version of Quicksort is bounded by $O(n \log n)$.

Let $a = (a_1, a_2, \ldots a_i, \ldots a_j, \ldots a_n)$ denote the list of elements we want to sort, *in sorted order*. Note that, *for the analysis* we may assume that we know the order. The input is any permutation of a. For all $1 \leqslant i < j \leqslant n$, let $X_{i,j}$ denote the random variable indicating whether Quicksort compared a_i and a_j.

$$X_{i,j} = \begin{cases} 1 & \text{if Quicksort compares } a_i \text{ and } a_j \\ 0 & \text{otherwise} \end{cases}$$

Any two elements get compared at most once. The expected number of comparisons is thus

$$E\left[\sum_{i=1}^{n}\sum_{j=i+1}^{n} X_{i,j}\right] = E\left[\sum_{1\leqslant i<j\leqslant n} X_{i,j}\right] = \sum_{1\leqslant i<j\leqslant n} E[X_{i,j}]$$

Quicksort compares a_i and a_j if and only if either a_i or a_j is selected as pivot *before* any of the elements between a_i and a_j. Selecting an element between a_i and a_j as pivot will partition the list into two parts such that a_i and a_j lie in different parts and do not get compared. The number of elements between a_i and a_j is $j - i - 1$. Since in each step the pivot element is selected uniformly at random, the probability that either a_i or a_j is selected as pivot before any of the $j - i - 1$ elements between them is $2/(j - i + 1)$. If this happens, a_i and a_j are compared and $X_{i,j}$ is one. We thus have

$$E[X_{i,j}] = \frac{2}{j - i + 1}.$$

$$\begin{aligned}
E\left[\sum_{i=1}^{n}\sum_{j=i+1}^{n} X_{i,j}\right] &= \sum_{i=1}^{n}\sum_{j=i+1}^{n} E[X_{i,j}] \\
&= \sum_{i=1}^{n}\sum_{j=i+1}^{n} \frac{2}{j - i + 1} \\
&\leqslant 2\sum_{i=1}^{n}\sum_{k=1}^{n} \frac{1}{k} \\
&= 2n\sum_{k=1}^{n} \frac{1}{k} \\
&= O(n \ln n).
\end{aligned}$$