Example: Expected value of a roll of a fair dice is:

$$E[X] = \sum_{i=1}^{6} i \cdot \Pr(\text{dice rolls to side } i) = \sum_{i=1}^{6} i \cdot 1/6 = \frac{1+2+3+4+5+6}{6} = 3.5$$

Expected value of an indicator random variable Y for some condition C is:

$$E[Y] = 0.\Pr(\neg C) + 1.\Pr(C) = \Pr(C)$$

- **Linearity of expectation** is a rule that allows us to compute an expected value of sums of random variables:

$$E[X+Y] = E[X] + E[Y]$$

Example: Expected value of the sum of three dice rolls is $E[X] = E[X_1 + X_2 + X_3] = E[X_1] + E[X_2] + E[X_3] = 3 \cdot 3.5 = 10.5$

Running time of randomized algorithms:

- The **running time** $T_A(x, R)$ **of a randomized algorithm** A for a particular input x and a *sequence of random numbers* $r = (r_1, r_2, \ldots)$ is the time that the algorithm requires to solve the input x, if the random numbers generated during the program's run are $r_1, r_2, \ldots$ (in that order).
- The **expected running time** $T_A^{(exp)}(x)$ **of a randomized algorithm** A for a particular input x is the expected value of random variable $T_A(x, R)$, where R is a random variable corresponding to the sequence of random numbers generated during the run of the algorithm:

$$T_A^{(\exp)}(x) = E[T_A(x, R)] = \sum_{r=(r_1,r_2,\ldots)} T_A(x, r) \cdot \Pr(R = r)$$

- The **worst-case expected running time** $T_A^{(exp)}(n)$ **of a randomized algorithm** is a function of the size n of the input, where $T_A^{(exp)}(n)$ is the largest expected time required to solve an input of size n:

$$T_A^{(\exp)}(n) = \max\{T_A^{(exp)}(x) \mid |x| = n\}$$

3.1.2 Analyzing Randomized QuickSort

Assumption: In this analysis, we will assume that all input numbers are distinct. We can denote these numbers $z_1, z_2, \ldots, z_n$ so that $z_1 < z_2 < \ldots < z_n$. Note that at the beginning, these numbers are stored in array $A[1 \ldots n]$ in some order, while at the end of the sort we will have $A[i] = z_i$ for all i.

The running time of the randomized QuickSort is dominated by two factors:

- The number of recursive calls. In the worst case this is $O(n)$. (During each recursive call, one element needs to be chosen as a pivot, and each element is chosen as a pivot at most once.)
- The number of comparisons.

All other operations can be associated with one or the other of the two factors. Thus to compute the expected runtime of the randomized QuickSort, we need to answer the following question.

What is the expected number of comparisons? Let X be the random variable representing the number of comparisons during the execution of QuickSort, and let $X_{i,j}$ be an indicator random variable for condition "numbers z_i and z_j are compared at some point during the execution of QuickSort". Note, that the numbers only get compared in the partitioning phase, always with currently selected pivot element. Therefore, any two numbers are compared at most once during the QuickSort. Thus we can write:

$$X = \sum_{i,j} X_{i,j}$$

Consider sequence of pivots in the order as they are chosen during the execution of the QuickSort. To compute expected value of $X_{i,j}$, we need to recognize that from the point of view of the two numbers z_i and z_j (without loss of generality assume $z_i < z_j$) there are three phases during the execution of the QuickSort:

1. While the chosen pivot is z_k such that $k < i$ or $k > j$, it may happen that both z_i and z_j get compared to z_k. However, they do not get compared to each other, and they both stay in the same partition. These steps in the computation do not have any bearing on the value of random variable $X_{i,j}$.
2. First time the pivot z_k is chosen so that $i \leq k \leq j$, the elements z_i and z_j will be separated into two different partitions. At that point, both z_i and z_j are compared to z_k.
3. After that, the elements z_i and z_j are in two different partitions and they are processed by separate recursive calls. So they will never get compared again.

So the only way z_i and z_j may get compared to each other is in phase 2. That only happens, if either z_i or z_j are chosen as a pivot. Therefore the probability that z_i and z_j get compared is equal to the following probability:

$$\Pr(z_i \text{ or } z_j \text{ chosen as a pivot} \,|\, \text{pivot chosen from } z_i, z_{i+1}, \ldots, z_j) = \frac{2}{j-i+1},$$

and therefore $\boxed{E[X_{i,j}] = 2/(j-i+1).}$

Now, to compute the expected number for comparisons:

$$\begin{aligned}
E[X] &= E\left[\sum_{i,j} X_{i,j}\right] = \sum_{i,j} E[X_{i,j}] \\
&= \sum_{i=1}^{n} \sum_{j=i+1}^{n} 2/(j-i+1) \\
&= 2\sum_{i=1}^{n} \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n-i+1} \quad [k := n-i+1] \\
&= 2\sum_{k=1}^{n} \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{k} \\
&= 2\sum_{k=1}^{n} (H_k - 1) = 2\sum_{k=1}^{n} H_k - 2n \\
&= 2((n+1)H_n - n) - 2n = 2(n+1) \underbrace{H_n}_{\Theta(\log n)} - 4n = \Theta(n \log n)
\end{aligned}$$

(You can find definitions and formulas for algebraic manipulation of harmonic numbers H_k in TCSCS.)

Therefore the expected number of comparisons of QuickSort is $\Theta(n \log n)$ and we have proved the following claim:

Theorem 1. *Expected running time of randomized QuickSort is* $\Theta(n \log n)$.