Examples:

1. The expectation of the sum of two dice is 7, even if they are not independent.

2. Assume that we flip N coins, what is the expected number of heads?

Using linearity of expectation we get $N \cdot \frac{1}{2}$.

By direct summation we get $\sum_{i=0}^{N} i\binom{N}{i}2^{-N}$.

Thus we prove

$$\sum_{i=0}^{N} i\binom{N}{i}2^{-N} = \frac{N}{2}.$$

3. Assume that N people checked coats in a restaurants. The coats are mixed and each person gets a random coat.

How many people got their own coats?

It's hard to compute $E[X] = \sum_{k=0}^{N} kPr(X = k)$. Instead we define N 0-1 random variables X_i, where $X_i = 1$ iff i got his coat.

$$E[X_i] \;=\; 1 \cdot Pr(X_i = 1) \;+\; 0 \cdot Pr(X_i = 0) =$$

$$Pr(X_i = 1) = \frac{1}{N}.$$

$$E[X] \;=\; \sum_{i=1}^{N} E[X_i] \;=\; 1.$$

Randomized Quicksort

Procedure Q_S(S);

Input: A set S.

Output: The set S in sorted order.

1. If $|S| \leq 1$ then return S, else

2. (a) Choose a random element y uniformly from S.
   (b) Compare all elements of S to y. Let

$$S_1 = \{x \in S - \{y\} \mid x \leq y\}$$

$$S_2 = \{x \in S - \{y\} \mid x > y\}.$$

   (Elements in S_1 and S_2 are in the same order as in S.)
   (c) Return the list:

$$Q\_S(S_1), y, Q\_S(S_2).$$

Let T = number of comparisons in a run of QuickSort.

Theorem 1.

$$E[T] = O(n \log n).$$

Let $s_1, ..., s_n$ be the elements of S in sorted order.

For $i = 1, ..., n$, and $j > i$, define 0-1 random variable $X_{i,j}$, s.t.

$X_{i,j} = 1$ iff s_i is compared to s_j in the run of the algorithm.

The number of comparisons in running the algorithm is

$$T = \sum_{i=1}^{n} \sum_{j>i} X_{i,j}.$$

We are interested in $E[T]$.

What is the probability that $X_{i,j} = 1$?

s_i is compared to s_j iff either s_i or s_j is chosen as a "split item" before any of the $j - i - 1$ elements between s_i and s_j are chosen.

Elements are chosen uniformly at random $\rightarrow$ elements in the set $[s_i, s_{i+1}, ..., s_j]$ are chosen uniformly at random.

$$Pr(X_{i,j} = 1) = \frac{2}{j - i + 1}.$$

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

$$E[T] = E[\sum_{i=1}^{n} \sum_{j>i} X_{i,j}] =$$

$$\sum_{i=1}^{n} \sum_{j>i} E[X_{i,j}] = \sum_{i=1}^{n} \sum_{j>i} \frac{2}{j-i+1} \leq$$

$$\sum_{i=1}^{n} \sum_{k=1}^{n-i+1} \frac{2}{k} \leq 2 \sum_{i=1}^{n} \sum_{k=1}^{n} \frac{1}{k} = 2nH_n = n \log n + O(n)$$

Probabilistic Analysis of QuickSort

Theorem 2. *The expected run time of (deterministic) Quicksort on a random input, uniformly chosen from all possible permutation of S is $O(n \log n)$.*

Proof.

Set $X_{i,j}$ as before.

If all permutations have equal probability, all permutations of $S_i, ..., S_j$ have equal probability, thus

$$Pr(X_{i,j}) = \frac{2}{j-i+1}.$$

$$E[\sum_{i=1}^{n} \sum_{j>i} X_{i,j}] = O(n \log n).$$

□

Randomized Algorithms:

- Analysis is true for **any** input.
- The sample space is the space of random choices made by the algorithm.
- Repeated runs are independent.

Probabilistic Analysis:

- The sample space is the space of all possible inputs.
- If the algorithm is **deterministic** repeated runs give the same output.

Randomized Algorithm classification

A **Monte Carlo Algorithm** is a randomized algorithm that may produce an incorrect solution.

For decision problems: A **one-side error** Monte Carlo algorithm errs only on one possible output, otherwise it is a **two-side error** algorithm.

A **Las Vegas** algorithm is a randomized algorithm that **always** produces the correct output.

In both types of algorithms the run-time is a random variable.