Assignment 0

Due: Monday January 12th, in class.

Question 1:

(a): Consider the set \{1, 2, ..., n\} We generate a subset \(X\) of this set as follows: a fair coin is flipped independently for each element of this set: if the coin lands heads then the element is added to \(X\), and otherwise it is not. Argue that the resulting set \(X\) is equally likely to be any one of the \(2^n\) possible subsets.

(b): Suppose that two sets \(X\) and \(Y\) are chosen independently and uniformly at random from all the \(2^n\) subsets of \{1, 2, ..., n\}. Determine \(\Pr[X \subseteq Y]\) and \(\Pr[X \cup Y = \{1, ..., n\}]\).

Question 2: Suppose you are given a biased coin for which the probability of heads is \(p\). However, the value of \(p\) is unknown. How can you use this coin to generate unbiased coin-flips (where heads and tails are equally likely)? Give a scheme for which the expected number of flips of the biased coin for extracting one unbiased coin-flip is no more than \(1/(p(1-p))\).

Question 3: Consider the following balls-and-bin game. We start with one black ball and one white ball in a bin. We repeatedly do the following: choose one ball from the bin uniformly at random, and then put the ball back in the bin with another ball of the same color. We repeat until there are \(n\) balls in the bin. Show that the number of white balls is equally likely to be any number between 1 and \(n-1\).

Question 4: There are \(k+1\) coins in a box. The \(i^{th}\) coin will, when flipped, turn up heads with probability \(i/k, i = 0, 1, ..., k\). A coin is randomly selected from the box and is then repeatedly flipped. If the first \(n\) flips all result in heads, what is the conditional probability that the \((n+1)^{th}\) flip will do likewise?

Question 5: In the first lecture we discussed the testing equality problem and described a randomized algorithm in which the number of bits exchanged between you and the server is \(O(\log n)\) and the algorithm has constant probability of successfully testing equality of the vectors \(a\) and \(b\).

Suppose now that you and the server somehow share an identical random string of length \(\text{poly}(n)\). Give an algorithm that exchanges just \(k\) bits and succeeds in testing equality of \(a\) and \(b\) with probability at least \(1 - 2^{-k}\).