1. Call a 0 - 1 matrix “nine-free” if there is no $3 \times 3$ submatrix with all entries one. (Rows and columns need not be consecutive.) Let $f(n)$ denote the maximal number of ones in an $n \times n$ nine-free matrix. Find a lower bound for $f(n)$ — i.e., show, for $\alpha$ as large as possible, that there exists a nine-free $n \times n$ matrix $A$ with at least $\alpha$ ones. Hint: Use the deletion method, first letting $P(A_{ij} = 1) = p$ and then changing a one to a zero in every $3 \times 3$ submatrix with all entries one.

2. (a) Let $A$ be a random $n \times n$ matrix with entries chosen independently and uniformly in $\{0,1\}$. Find the expectations of the determinant and the permanent of $A$.

Note: You are certainly familiar with the first of these concepts, possibly not with the second; they are defined by the formulas

$$
\det(A) = \sum_{\sigma \in S_n} \text{sign}(\sigma) \prod_{1}^{n} A_{i,\sigma(i)}
$$

$$
\text{perm}(A) = \sum_{\sigma \in S_n} \prod_{1}^{n} A_{i,\sigma(i)}
$$

(b) Let $A$ be a random $n \times n$ matrix with entries chosen independently and uniformly in $\{\pm 1\}$. Find $E((\det(A))^2)$.

3. A monotone function $f : \{0, 1\}^n \to \{0, 1\}$ is one with the property that if $f(\ldots, x_{i-1}, 0, x_{i+1}, \ldots) = 1$ then $f(\ldots, x_{i-1}, 1, x_{i+1}, \ldots) = 1$. Here are some examples: the “dictator function” Dict$_{x,j}(x_1, \ldots, x_n) = x_j$; the AND$_n$ function which is 1 only for $x_1 = \ldots = x_n = 1$; the function MAJ$_n$ for odd $n$ which is 1 if more than half the inputs are 1’s; the function CLIQUE$_{n,k}(x_1,1, x_1, 2, \ldots, x_{n-1, n})$ which is 1 if the graph having an edge for each “1”, contains a clique of size $k$.

When we design boolean circuits for functions, we use a fixed (and constant-size) basis of gates. For instance the basis $\{\text{AND}_2, \text{NOT}\}$, or even just the basis with the single gate $\{\text{NAND}_2\}$. If we are only interested in computing monotone functions, however, then we can consider using a basis consisting only of monotone gates. This is not necessarily the simplest or most efficient way of constructing a circuit. For instance the simplest way to compute MAJ$_n$ is to use a general (non-monotone) basis to perform arithmetic, and add up the input bits and check whether the sum is $> n/2$. In order to understand the power of nonmonotonicity, even for computing monotone functions, we need to ask how efficiently we can compute functions like MAJ$_n$ using only a monotone basis. That is what we will do in this exercise.

The basis we consider is simple: it includes only the 3-input gate MAJ$_3$. Your task is to show something not at all obvious: there are log-depth circuits for MAJ$_n$ consisting solely of MAJ$_3$ gates.

Hints:

(a) There exists a circuit of the following simple form: the MAJ$_3$ gates form a complete 3-ary tree from the output gate all the way down to input wires at depth $O(\log n)$. Then each of these wires is randomly, independently, hooked up to one of the $n$ inputs. (Note, each input will be used many times.)

(b) A good approach is to show that for any particular $x = (x_1, \ldots, x_n)$, with very low probability the circuit you constructed at random gives the wrong answer.

(c) For any particular $x$, let $p_t$ be the probability that a wire at level $t$ of the circuit carries a value that disagrees with MAJ$_n(x)$. Show that $p_1 \leq (n - 1)/(2n)$ and $p_{t+1} = 3p_t^2 - 2p_t^3$. 


4. The following is an example of a heavy-tailed distribution. \( \mu \) is supported on the nonzero integers,

\[ \mu(m) = K/m^4 \]

for the appropriate normalizing constant \( K \) which is \( 45/\pi^4 \).

The first and second moments of \( \mu \) are well-defined; if you calculate you’ll see \( E(X) = 0, \ Var(X) = 15/\pi^2 \).

The purpose of this exercise is to demonstrate that for a heavy-tailed distribution like this, taking the average of a large number of independent samples does not create a light-tailed distribution.

Specifically, take \( n \) iid rvs \( X_1, \ldots, X_n \) with the distribution \( \mu \), and set \( \bar{X} = (1/n) \sum X_i \). The second-moment inequality tells us:

\[ \Pr(|X| \geq \lambda \sqrt{\Var(X)}) \leq \frac{1}{\lambda^2} \]

(Specifically \( \Pr(|X| \geq r) \leq \frac{15}{\pi^2 r^2} \).

Show that there is a polynomial \( p(\lambda, n) \) such that

\[ \Pr \left( \bar{X} > \lambda \sqrt{\Var(X)} \right) \geq 1/p(\lambda, n). \]

What does this tell you about the moment generating function of \( \mu \)?

5. You are trying to count sheep. There are a lot of sheep and you are a shepherd of very little brain: you don’t even have a memory of size \( \lg n \), which is what you would need to count \( n \) sheep.\(^{1}\)

(Not to be handed in: argue that any deterministic counting algorithm requires this much space.) Instead, you come up with the following mechanism whose goal is to estimate the number of sheep within a constant factor, using memory only \( O(\lg \lg n) \).

Initialize \( C := 0. \)

After a sheep walks by, flip a biased coin \( X \), \( \Pr(X = 1) = 2^{-C} \) (otherwise \( X = 0 \)).

Set \( C := C + X. \)

Denote by \( C(n) \) the random variable after \( n \) sheep have walked by. Show for any value of \( n \), that \( 2^{C(n)} \) probably approximates \( n \) within a constant factor. More specifically, show that \( \forall a > 0 \exists b > 0 \text{ s.t. with probability } \geq 1 - a, bn \leq 2^{C(n)} \leq n/b. \)

Also, suppose you do not have access to coins of arbitrary bias but only to a fair coin. Can you still solve the problem within the required memory limitation?

Note: It is possible, although slightly subtle, to show that \( E(2^{C(n)}) = n + 2, \ Var(2^{C(n)}) = n(n+1)/2. \) If you want to pursue this approach for credit, please don’t look it up, although it can be found in Approximate Counting: a Detailed Analysis, P. Flajolet, BIT 25 (1985), 113-134.

I recommend instead pursuing a simpler (if less precise) approach. Let’s turn things around and imagine there is an infinite list of sheep, and let \( N(c) \) be the index of the first sheep to bring the register to \( c. \)

E.g., for sure \( N(1) = 1. \) Show that \( \forall a > 0 \exists b > 0 \text{ s.t. with probability } \geq 1 - a, b2^a \leq N(c) \leq 2^c/b. \)

Finally, get the quantification right: argue what we asked for any fixed number of sheep \( n. \) (You might have to pay a little in \( a. \))

\(^{1}\)The convention is that \( \lg = \log_2. \)