Princeton Univ Fall '25	COS 521: Advanced Algorithms
Homework 2	
Out: *Sep 23*	Due: *Oct 5*

Instructions:

- Upload your non-extra solutions to Gradescope in a single PDF file, and mark your solution to each problem. Please make sure you are uploading the correct PDF! Please anonymize your submission (i.e., do not list your name in the PDF), but if you forget, it's OK.
- If you choose to do extra credit, upload your solution to the extra credits as a single separate PDF file to Gradescope. Please again anonymize your submission.
- You may collaborate with any classmates, textbooks, the Internet, etc. Please upload a brief "collaboration statement" listing any collaborators as a separate PDF on Gradescope (if you forget, it's OK). But always **write up your solutions individually**.
- For each problem, you should have a solid writeup that clearly states key, concrete lemmas towards your full solution (and then you should prove those lemmas). A reader should be able to read any definitions, plus your lemma statements, and quickly conclude from these that your outline is correct. This is the most important part of your writeup, and the precise statements of your lemmas should tie together in a correct logical chain.
- A reader should also be able to verify the proof of each lemma statement in your outline, although it is OK to skip proofs that are clear without justification (and it is OK to skip tedious calculations). Expect to learn throughout the semester what typically counts as 'clear'.
- You can use the style of Lecture Notes and Staff Solutions as a guide. These tend to break down proofs into roughly the same style of concrete lemmas you are expected to do on homeworks. However, they also tend to prove each lemma in slightly more detail than is necessary on PSets (for example, they give proofs of some small claims/observations that would be OK to state without proof on a PSet).
- Each problem is worth twenty points (even those with multiple subparts), unless explicitly stated otherwise.

Problems:

§1 Recall that the collection of functions $h(x) = x_i$ for $i \in [d]$ is a good LSH family for the Hamming distance ($x \in \{0,1\}^d$), and when $r_1 = r, r_2 = cr$, we have $\gamma := \log(1/p_1)/\log(1/p_2) \approx 1/c$. This family can then be used to solve (r, cr)-PLEB, and by "guessing" the closest distance and constructing $O(\log(d_{\max}/d_{\min}))$ instances of (r, cr)-PLEB, this solves c-ANN.

Note that this is the same LSH family that is used in all (r, cr)-PLEB instances. Why didn't we directly use this LSH family to solve c-ANN at once? Specifically, what is different in these PLEB data structures and why do we need to guess the closest distance?

Hint: The answer is supposed to be simple, in one or two lines.

§2 Recall that the basic version of approximate counter covered in the lecture maintains X such that X is incremented with probability 2^{-X} each time we increment n, and at query time, the algorithm returns $2^X - 1$ as the estimator of n.

Suppose we make n increments for $n = 2^k$. Show that for any *constant* $c > 0$, the final X is at most $k - c$ with $\Omega(1)$ probability. Hence, the basic version of the algorithm may indeed incur a constant factor error with constant probability.

Remark: It is also true that with $\Omega(1)$ probability, X is at least $k + c$. You don't have to prove this.

§3 We say a random variable Z is *subgamma* with parameters (σ^2, B), if

$$\mathbb{E}\left[e^{\lambda(Z-\mathbb{E}[Z])}\right] \le e^{\lambda^2\sigma^2/2},$$

for all $|\lambda| \le B$.

(a) Let $Z_1, \ldots, Z_m$ be *independent* subgamma random variables with parameters (σ_i^2, B_i) respectively for $i \in [m]$. Prove that $\sum_{i\in[m]} Z_i$ is subgamma with parameters $(\sum_{i\in[m]} \sigma_i^2, \min_{i\in[m]} B_i)$.

(b) Show that if Z is subgamma with parameters (σ^2, B), then for any $t > 0$, both $\Pr[Z - \mathbb{E}[Z] > t]$ and $\Pr[Z - \mathbb{E}[Z] < -t]$ are at most $\max\left\{e^{-\frac{t^2}{2\sigma^2}}, e^{-\frac{tB}{2}}\right\}$.

(c) Let Z be a geometric random variable such that $\Pr[Z = k] = p \cdot (1-p)^{k-1}$ for all integers $k \ge 1$. Prove that Z is subgamma with parameters $(2/p^2, p/2)$.
Hint: The following two inequalities may be useful: $e^x \ge 1 + x$ for all $x \in \mathbb{R}$ and $1/(1-x) \le e^{x+x^2}$ for $|x| \le 1/2$.

§4 In this problem, we will analyze the *Morris counter* mentioned in the lecture, and prove that it solves approximate counting using $O(\log\log N + \log(1/\varepsilon) + \log\log(1/\delta))$ bits of space.

Recall that the approximate counting problem asks us to maintain a counter n (initialized to 0) up to N, supporting

- `inc()`: $n \leftarrow n + 1$;
- `query()`: output $\tilde{n}$ such that $\Pr[|\tilde{n} - n| > \varepsilon n] < \delta$.

A Morris counter has a parameter $\alpha > 0$. It maintains a variable X, initialized to 0. Each time `inc()` is called, X is incremented to $X + 1$ with probability $(1 + \alpha)^{-X}$. When `query()` is called, it returns $\tilde{n}(X) = ((1 + \alpha)^X - 1)/\alpha$.

(a) Let Y_k be the random variable denoting the number of `inc()` calls needed to increment X from $k-1$ to k. Derive the probability distribution of each Y_k, and prove that it is subgamma for some parameters (σ_k^2, B_k).

(b) When $\alpha n > C$ for a sufficiently large constant C and $\alpha < \varepsilon^2$, prove that after n `inc()` calls, $\tilde{n}(X) < (1 - \varepsilon)n$ with probability at most $e^{-\Omega(\varepsilon^2/\alpha)}$.

We can prove a similar upper bound on the probability that $\tilde{n}(X) > (1 + \varepsilon)n$, and you may use this bound without a proof in (c).

(c) Given N, ε, δ, choose the right parameters T and α, and prove that the Morris counter, together with an exact counter when $n \leq T$ using $O(\log T)$ bits, solves approximate counting with $O(\log\log N + \log(1/\varepsilon) + \log\log(1/\delta))$ bits.

§5 Fix a constant $c > 0$. Consider a stream of updates to a vector X of dimension n, where each update is of the form $x_i := x_i + \Delta$ for some $i \in [n]$ and (possibly negative) integer Δ. Each $|\Delta|$ is at most n^c, and the stream length is also at most n^c. Design an $O(\log n)$-bit space streaming algorithm that tests with high probability if all updates cancel out, i.e., suppose $X = \vec{0}$ initially, test if $X = \vec{0}$ after all updates. The algorithm must output the correct answer with probability at least $1 - 1/n^c$.

Hint: Let P be a non-zero degree-n polynomial, then P has at most n different roots. The same also holds modulo a prime q, i.e., at most n different integers $x \in [0, q-1]$ can have $P(x) = 0 \mod q$, if not every coefficient of P is a multiple of q. You may use this fact without a proof.