22C:196 Homework 2
Due: Tuesday, 10/17

Notes: (a) Computer Science Ph.D. students are required to solve Problems 3-7. The rest of the students are required to solve the first 5 problems. The problem numbers refer to problems in the textbook, by Mitzenmacher and Upfal. (b) It is possible that solutions to some of these problems are available to you via other textbooks, on-line lecture notes, etc. If you use any such sources, please acknowledge these in your homework *and* present your solutions in your own words. You will benefit most from the homework, if you sincerely attempt each problem on your own first, before seeking other sources. (c) As mentioned in the syllabus, it is *not* okay to discuss these problems with your classmates. But, you are welcome to come and chat with me about the problems. (d) Students who are not Computer Science Ph.D. students will receive extra credit for submitting correct solutions to any non-empty subset of Problems 6-7.

1. Suppose we roll a standard die 1000 times. Let X denote the sum of the numbers that appeared over the 1000 rolls. Use Chebyshev's inequality to upper bound $Pr(X \geq 5000)$.

2. In an election with two candidates using paper ballots, each vote is independently misrecorded with probability $p = 0.02$. Use a Chernoff bound to give an upper bound on the probability that more than 4% of the votes are misrecorded in an election of 1,000,000 ballots.

3. A *fixed point* of a permutation $\pi : [1, n] \rightarrow [1, n]$ is a value for which $\pi(x) = x$. Find the variance in the number of fixed points of a permutation chosen uniformly at random from all permutations.
   Hint: Let $X_i = 1$ if $\pi(i) = i$, so that $X = \sum_{i=1}^{n} X_i$ is the number of fixed points. Calculate $E[X_i]$ and $E[X_i \cdot X_j]$ in order to calculate $E[X]$ and $E[X^2]$ and use this to calculate $Var(X)$.

4. This problem is on the median finding algorithm that uses random sampling. This algorithm and its analysis appear in Section 3.4 in the textbook and all of this was discussed in class.

   (a) In the analysis of this algorithm, Chebyshev's inequality was used to derive the $\frac{1}{n^{1/4}}$ upper bound on the probability that the algorithm will fail to yield a median. Instead, could we use Chernoff bounds to derive a sharper upper bound on the failure probability? In particular, could we use Chernoff bounds to show that the algorithm fails with probability at most $1/n$? For your answer, you can either show how to obtain the $1/n$ upper bound on the failure probability via Chernoff bounds *or* explain why Chernoff bounds cannot be applied in this setting or do not yield the desired bound even when used.

   (b) In this algorithm we used a sample R of size $n^{3/4}$. Suppose we use a much smaller sample of size $O(\log n)$. With other appropriate changes to the algorithm (e.g., in the definitions of d and u) could we design an algorithm that runs in $O(n)$ time with failure probability bounded above by $1/n^c$ for a constant $c > 0$? Your answer should either be a restatement of the algorithm (with appropriate changes to parameter values) followed by a modified analysis *or* an explanation of why a sample size of $O(\log n)$ is too small.

5. There is a simple randomized median finding algorithm, that is very similar to the randomized quicksort algorithm discussed in class. Here is an informal description of this algorithm.

Suppose we want to find an element of rank k in the given set S (of n distinct elements). We pick a *pivot* $y \in S$ uniformly at random and construct subsets $S_1 = \{u \in S \mid u < y\}$ and $S_2 = \{v \in S \mid v > y\}$. If y has rank k, we are done. Otherwise, we recurse on one of S_1 *or* S_2 looking for an element of appropriate rank. State this algorithm precisely and show that the expected running time of this algorithm is $O(n)$.
Hint: Start by showing that the expected number of recursive calls that it takes for the problem size to shrink from n to $2n/3$ is $O(1)$. Then account for the total (expected) number of comparisons made by recursive calls whose input has more than $2n/3$ elements. Repeat this for size thresholds $(2/3)^2 n$, $(2/3)^3 n$, etc., to obtain a full accounting of all the work done.

6. Problem 3.22.

7. The following problem models a simple distributed system wherein "agents" contend for resources but "backoff" in the face of contention. This is a situation that arises in wireless networks when wireless nodes contend for access to the wireless medium to send messages.

   The system evolves in round. Every round, balls are thrown independently and uniformly at random into n bins. Any ball that lands in a bin by itself is *served* and removed from consideration. The remaining balls are all thrown again in the next round. We begin with n balls in the first round and we finish when every ball is served.

   (a) If there are b balls at the start of a round, what is the expected number of balls at the start of the next round?

   (b) Let x_j be the number of balls left after j rounds. Show that $x_{j+1} \leq x_j^2/n$. Use this to argue that if in every round the number of balls served was *exactly* the expected number of balls to be served, then all balls would be served in $O(\log \log n)$ rounds.