22C:153 Homework 1

Due: Tuesday, 2/25

On basic probability theory:

(1) Problem C.2-9 (page 1106). Explain your answer by calculating the relevant probabilities.

(2) Problem C.3-6 (page 1111). Just show this assuming that X is a discrete random variable that takes on finitely many non-negative values.

On probabilistic analysis:

(3) Problem 5.2-5 (page 99).

(4) Problem 5.3-4 (page 105).

On randomized quicksort:

(5) Problem 7-2 (page 160).

(6) Problem 7-4 (page 162).

On randomized selection:

(7) In the analysis of randomized selection we derived the following sum:

$$\sum_{i=1}^{k-2}\sum_{j=i+1}^{k-1}\frac{2}{(k-i+1)}+\sum_{i=k+1}^{n-1}\sum_{j=i+1}^{n}\frac{2}{(j-k+1)}+\sum_{i=1}^{k}\sum_{j=k}^{n}\frac{2}{(j-i+1)}.$$

Show that for any integer k, $1 \le k \le n$, the above sum is $O(n)$.

On Karger's min-cut algorithm:

(8) Consider the modification of Karger's min-cut algorithm mentioned in class in which, instead of picking the endpoints of an edge to contract, we pick (uniformly at random) a pair of vertices and contract them. Show an example of a graph with n vertices such that the probability that th modified algorithm will find a min-cut in the graph is exponentially small (that is, the probability is bounded above by $\Theta(1/e(n))$ where $e(n)$ is an exponential function in n.

(9) We showed that the probability that Karger's min-cut algorithm does not find a min-cut is less than $1/e$. How many times does this algorithm have to be repeated if we want this probability to be less than 1/100? Justify your answer.

(10) Describe a data structure to store a multigraph (a graph with multiple edges between pairs of vertices) such that each `CONTRACT` can be performed in $O(n)$ time on an n-vertex graph. Furthermore, suppose that we start with the graph $G = (V, E)$ and have contracted the endpoints of edges $e_1, e_2, \ldots, e_k$. Let the resulting graph be called H. Each edge in H corresponds to an edge in the original graph G. Given an edge in H how quickly can you find the corresponding edge in G using your data structure?
(**Hint:** Consider a small modification of an adjacency matrix.)