22C:196 Homework 1

Due: Tuesday, 9/24

Notes: (a) Computer Science Ph.D. students are required to solve Problems 4-8. 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 6-8.

1. Bowl I contains 3 red chips and 7 blue chips. Bowl II contains 6 red chips and 4 blue chips. A bowl is selected (uniformly) at random and then one chip is drawn (again, uniformly at random) from the selected bowl. (a) Compute the probability that this chip is red. (b) Relative to the hypothesis that the chip is red, find the conditional probability that it is drawn from Bowl II.

2. Implement the FLT Primality testing algorithm so that it runs in $O(k \log^c n)$ rounds, where n is the input integer, k is the number of iterations of the loop in the algorithm, and c is some small constant. Use your implementation to test if the following numbers are prime:

   - 5991810554633396517767024967580894321153
   - 19822271254366240129112696248055903545291688310293
   - 27175146095341224357465037532218133092930145221379
   - 4702877858580764415667235078667510929270158248348819067635 07
   - 693711969678975263512873427191894879124339838606362751311911118403883

   As we did in class, use $k = 10$ when you run your implementation on the above inputs.

   Your answer should consist of a printout of your code along with a printout of the output produced. Given that the FLT Primality testing algorithm has a one-sided error, for each of your outputs, indicate whether there is a possiblity that your program's output may be wrong.
   Note: You can use any high-level programming language for this, but your entire code should be no longer than about 20 lines.

3. Describe an algorithm that takes as input a positive integer n and returns a random permutation of $I = \{1, 2, \ldots, n\}$, chosen uniformly at random from all $n!$ permutations of I. The algorithm should use, as its source of randomness, a procedure that returns a random bit in $O(1)$ time. Assume that the two bits, 0 and 1, are equally likely to be returned by a call to the procedure and that each call returns a bit that is independent of past history. Your algorithm is required to run in $O(n \log n)$ time.

   Your answer should consist of a brief description of the algorithm, followed by a brief analysis that shows that the algorithm runs in $O(n \log n)$ time and produces every permutation of I with equal probability.

4. Suppose that at every iteration of Karger's min-cut algorithm, instead of choosing a random edge for contraction, we choose two vertices u and v at random (i.e., uniformly at random from all unordered vertex pairs) and contract the u-v pair. Show that there are graph examples on which the probability that this modified algorithm finds a min-cut is exponentially small. More precisely, show that there is a family $\mathcal{F}$ of graphs such that for every n large enough, $\mathcal{F}$ contains a graph G_n with n vertices such that the modified Karger's algorithm finds a min-cut on G_n with probability at most $1/a^{n/b}$ for constants $a > 1$ and $b \geq 1$.

5. Suppose we have a sequence of items passing by one at a time. At all times, we want to maintain a sample of one item with the property that it is uniformly distributed over all items that have been seen. Moreover, we want to accomplish this without knowing the total number of items in advance or storing all of the items we see.

   Consider the following algorithm, which stores just one item in memory at all times. When the first item appears, it is stored in the memory. When the kth item appears, it replaces the item in memory with probability $1/k$. Explain why this algorithm solves the problem.
   Note: This is a simple example of a *data streaming algorithm* called *reservoir sampling*.

6. Suppose that we have $n \log n$ balls that we throw into n bins, choosing a bin uniformly at random for each ball. Show that with probability $1 - o(1)$ every bin contains $O(\log n)$ balls.

7. Problem 1.18. In addition to describing your algorithm, you should also prove that the algorithm computes $F(z)$ with probability at least $1/2$.
   Hint: To compute $F(z)$ correctly with sufficiently high probability, pick $z' \in \{0, 1, \ldots, n-1\}$ uniformly at random and use the given property of F with respect to z and z'.

8. Problem 2.16.
