Homework 4
Due: End of Day Dec 3 (Wednesday)

1. In this problem, we examine the searching of a particular element x in an unsorted array A consisting of n distinct elements. We will examine the following strategies:

Strategy 1: Pick a random index i, and compare $A[i]$ with x. If $A[i] = x$, terminate. Otherwise, we continue the search by picking a new random index other than i. What is the expected running time for this randomized algorithm?

Strategy 2: We search A in increasing order of the indices, i.e., $A[1], A[2], \ldots, A[n]$. Assuming that all possible permutations of the elements in A are equally likely. What is the expected running time of this strategy?

Strategy 3: Pick a random index i, and compare A[i] with x. If $A[i] = x$, terminate. Otherwise, we continue the search by picking another random index (that could be i again). What is the expected running time for this strategy?

2. Assume that you are given a list of real numbers. Your goal is to determine if there is a pair of numbers in this list whose product is exactly 1. What is the fastest deterministic algorithm you can devise to solve this problem? Can you do better in terms of expected running time if you are allowed to use a randomized algorithm such as a hash function?

3. One application of a Bloom filter is that it can be used to estimate the size of the intersection and union of two or more sets. More specifically, assume you are given two sets S and T with $|S| = m$ and $|T| = m$. Assume a Bloom filter B with n bits and a single hash function h is used. Describe how B can be used to calculate $S \cap T$ and $S \cup T$. What is the expected error of the intersection and union in terms of cardinality?

4. Suppose a good friend told you that a particular problem — call it Problem A — must be NP-hard because he was able to reduce Problem A to a well-known NP-hard problem the 3-SAT. Since you are interested in NP-hard problems, you decide to take a closer look.

   After working hard over the Thanksgiving weekend, you discovered an $O(n^{10})$ algorithm that correctly solves Problem A. To be sure, you showed your solution to your professor, who confirmed that your algorithm was indeed correct.

   Is your friend's claim that Problem A is NP-hard actually correct? Explain why or why not.

5. The k-leaf problem is as follows: Given an Graph $G(V, E)$, and an integer $k \geq 2$, determine if the graph has a spanning tree with exactly k leaves. Determine if the k-leaf spanning tree problem is NP-complete. If yes, prove it. If no, develop an efficient polynomial time algorithm for it.

   (Hint: you may consider using the known NP-hard Hamiltonian Path problem.)