1. Write each of the functions below (n is the positive integer variable) in the Big-Oh notation.
   (a) $O(n^2)$
   (b) $O(log(n))$
   (c) $O(1/n)$
   (d) $O(1)$
   (e) $O(logn)$

2. (10 points) Let $A[1 \dots n]$ be an array of integers. Describe an algorithm that runs in time $O(n \log n)$ and returns an array B whose entries are all the *distinct* elements of A (i.e., with no duplicates).

3. (10 points) Suppose I tell you that there is an algorithm that can square any n digit number in time $O(n \log n)$, for all $n \geq 1$. Then, prove that there is an algorithm that can find the product of *any two* n digit numbers in time $O(n \log n)$. [*Hint:* think of using the squaring algorithm as a subroutine to find the product.]

4. Write the answers to the following as functions of k:
   (a) (4 points) Suppose we toss a fair coin k times. What is the probability that we see heads precisely once?
   (b) (6 points) Suppose we have k different boxes, and suppose that every box is colored uniformly at random with one of k colors (independently of the other boxes). What is the probability that all the boxes get distinct colors?

5. (10 points) Given an array $A[1 \dots n]$ of integers (not necessarily distinct), find if there exist indices i, j, k such that $A[i] = A[j] + A[k]$. Can you find an algorithm with running time $o(n^3)$? [NOTE: this is the little-oh notation, i.e., the algorithm should run in time $< cn^3$, for any constant c, as $n \to \infty$.] [*Hint:* aim for an algorithm with running time $O(n^2 \log n)$.]