Fall 2018, CMPSC 465
Homework Assignment #3

This homework is due **9 pm** on **September 21st**.
Collaboration is permitted on this homework. If you choose to collaborate, you are allowed to discuss each problem with at most three other students currently enrolled in class. Before working with others on a problem, you should think about it yourself for at least 45 minutes. *You must write up each problem solution by yourself without assistance, even if you collaborate with others to solve the problem.* You must also identify your collaborators. If you do not work with anyone, you should write "Collaborators: none." It is a violation of this policy to submit a problem solution that you cannot orally explain to an instructor or a TA. *Finding answers to problems on the web or from other outside sources (includes anyone not enrolled in the class) is strictly forbidden.*
You should aim to be as clear and concise as possible in your writeup of solutions. A simple and direct analysis is worth more points than a convoluted one. Each problem is worth 10 points. Points may be deducted for illegible handwriting. Partial credit will be given only for answers that make significant progress toward correct solution.

1

Suppose we are given a sequence S of n elements, each of which is colored red or blue. Assuming S is represented as an array, give an in-place $O(n)$ method for ordering S so that all the red elements are listed before all the blue elements.

2

Consider the following variant of Mergesort, where instead of partitioning the array into two equal halves, you split the array into 5 nearly-equal parts. What is the running time recurrence relation for this algorithm? How does the running time compare asymptotically to the running time of normal Mergesort? Clearly state any assumptions you make when deriving the closed form expression for the running time.

3

Suppose you have k sorted arrays, each with n elements, and you want to combine them into a single sorted array of kn elements. Give an efficient solution to this problem using divide-and-conquer. What is the recurrence relation for the running time? Solve the recurrence to determine a closed-form expression for the running time.

4

Professor Caesar wishes to develop a matrix-multiplication algorithm that is asymptotically faster than Strassen's algorithm. His algorithm will use the divide-and-conquer method, dividing each matrix into pieces of size $n/4 \times n/4$, and the divide and combine steps together will take $\Theta(n^2)$ time. He needs to determine how many subproblems his algorithm has to create in order to beat Strassen's algorithm. If his algorithm creates a subproblems, then the recurrence for the running time $T(n)$ becomes $T(n) = aT(n/4) + \Theta(n^2)$. What is the largest integer value of a for which Professor Caesar's algorithm would be asymptotically faster than Strassen's algorithm?

5

A complex number $a + bi$, where $i = \sqrt{-1}$, can be represented by the pair (a, b). Describe a method performing only three real-number multiplications to compute the pair (e, f) representing the product of $a + bi$ and $c + di$.

6

Suppose you are consulting for a bank that is concerned about fraud detection, and they come to you with the following problem. They have a collection of n bank cards that they have confiscated, suspecting them of being used in fraud. Each bank card is a small plastic object, containing a magnetic stripe with some encrypted data, and it corresponds to a unique account in the bank. Each account can have many bank cards corresponding to it, and we will say that two bank cards are equivalent if they correspond to the same account.

It is very difficult to read the account number off a bank card directly, but the bank has a high-tech "equivalence tester" that takes two bank cards and, after performing some computations, determines whether they are equivalent.

Their question is the following: among the collection of n cards, is there a set of more than $n/2$ of them that are all equivalent to one another? Assume that the only feasible operations you can do with the cards are to pick two of them and plug them in to the equivalence tester. Show how to decide the answer to their question with only $O(n \log n)$ invocations of the equivalence tester.

7

Suppose you are given a set of small boxes, numbered 1 to n, identical in every respect except that each of the first i contain a pearl whereas the remaining $n - i$ are empty. You also have two magic wands that can each test whether a box is empty or not in a single touch, except that a wand disappears if you test it on an empty box. Show that, without knowing the value of i, you can use the two wands to determine all the boxes containing pearls using at most $o(n)$ wand touches. Express, as a function of n, the asymptotic number of wand touches needed.