Algorithms[1] Lab 2
Due: Tuesday, 1 October 2019 (afternoon), *at the beginning of class*

Notes on Grading (Reminder)

It is just as important to write in a style that is easy to understand and convincing as it is to get the correct answer. So, your assignment will be evaluated based not only on the correctness of your final answer, but also on clarity, neatness and attention to details.

Topics Covered For This Lab

- Summations
- Recurrences
- Mergesort

In Lab Exercises (COLLABORATION LEVEL 0)

The in-lab problems are meant to be be solved during the lab and to generate discussion and sharing of ideas. Your answers will not be graded, but you do need to work through these problems. You can turn in a single writeup, with all your answers stapled together. List the people in the group on the first page.

1. Find a tight bound for the solution of the following recurrences using iterative substitution.

   (a) $T(n) = T(n/2) + 1$
   (b) $T(n) = T(9n/10) + 1$
   (c) $T(n) = 3T(n/3) + n$
   (d) $T(n) = 3T(n/4) + n$
   (e) $T(n) = T(n/2) + \lg n$ (a little more difficult)

   "Extra Credit": Each of the first two pairs – (a) & (b) and (c) & (d) – makes an important point about the performance of algorithms with similar recurrences. What are these two points? (Actually, the two points are opposite sides of the same coin.)

Homework Problems (COLLABORATION LEVEL 1)

You are encouraged to collaborate, but only at Level 1, for these problems. Please refer to the class website for a description of this collaboration level. Write your name and list the people with whom you discussed the problems on a separate top sheet. It may be different from the people you worked with on the lab problems. Even if it's the same, list them again.

[1] CSCI 2200, Laura Toma and Stephen Majercik, Bowdoin College

Remember, turn in each answer separately. If you used more than one sheet *for one answer*, staple those together. But, do ***not*** staple the answers to all the problems together; ***clip*** them together. This will save me a huge amount of time removing staples and restapling.

1. Find a tight bound for the solution of the following recurrences using iterative substitution.
   (a) $T(n) = T(n/3) + 1$
   (b) $T(n) = T(n/3) + n$
   (c) $T(n) = T(\sqrt{n}) + 1$
   Hint: The base case does not have to be a problem of size 1; any small number is fine.
   (d) $T(n) = 7T(n/2) + n^3$
   (e) $T(n) = 7T(n/2) + n^2$

2. (a) Given an unsorted array and a number k, design an algorithm that finds two elements in the array whose sum is k or, if no such numbers exist, reports that fact. Your algorithm should be faster than the straightforward $O(n^2)$ algorithm. Analyze the running time of your algorithm.
   (b) Now generalize this to "3-sum," i.e. given an unsorted array and a number k, design an algorithm that finds three elements in the array whose sum is k or, if no such numbers exist, reports that fact. Analyze the running time of your algorithm.

3. Suppose you have a building with n stories and a bunch of eggs. An egg has a certain level l at which, if thrown from any level $\geq l$, it breaks. For example, an egg might have $l = 7$ meaning you can throw the egg down from levels 1 through 6 and it will not break but, if you through the egg from level 7 or higher, it breaks. All of the eggs are identical and so have the same level.

   Given a building with n stories and an unlimited number of eggs, your goal is to determine the level l of the eggs.

   (a) Describe an approach that only breaks one egg to determine l. How many throws does it take in the worst case?
   (b) Describe an approach that reduces the number of throws needed. The goal is to make the reduction as large as possible. What is the maximum number of eggs it will break?
   (c) Now assume that you have only two eggs. Describe an approach that minimizes the number of throws.

By the way, you may be thinking that this last problem has nothing to do with recursive sorting and solving recurrences. You would be right. But, it's good practice for designing algorithms and the problem has been used as an interview question.