CS515: Algorithms and Data Structures, Fall 2015

Homework 1*

Due: Tue, 10/11/15

Homework Policy:

1. Students should work on homework assignments in groups of preferably three people. Each group submits to TEACH one set of typeset solutions, and hands in a printed hard copy in class or slides the hard copy under my door before the midnight of the due day. The hard copy will be graded.
2. The goal of the homework assignments is for you to learn solving algorithmic problems. So, I recommend spending sufficient time thinking about problems individually before discussing them with your friends.
3. You are allowed to discuss the problems with other groups, and you are allowed to use other resources, but you *must* cite them. Also, you must write everything in your own words, copying verbatim is plagiarism.
4. *I don't know policy:* you may write "I don't know" *and nothing else* to answer a question and receive 25 percent of the total points for that problem whereas a completely wrong answer will receive zero.
5. Algorithms should be explained in plain english. Of course, you can use pseudocodes if it helps your explanation, but the grader will not try to understand a complicated pseudocode.
6. More items might be added to this list. ☺

Problem 1. **[25 pts]** Describe a recursive algorithm to reconstruct an arbitrary binary tree, given its preorder and inorder node sequences as input. See Wikipedia (`https://en.wikipedia.org/wiki/Tree_traversal`) if you have forgotten preorder/inorder traversals.

Problem 2. **[25 pts]** Suppose you are given two sets of n points, one set $\{p_1, p_2, \ldots, p_n\}$ on the line $y = 0$ and the other set $\{q_1, q_2, \ldots, q_n\}$ on the line $y = 1$. Create a set of n line segments by connecting each point p_i to the corresponding point q_i. Suppose no three line segments intersect in a point.

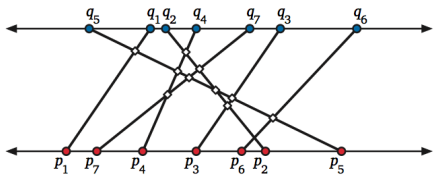

Figure 1: Eleven intersecting pairs of segments with endpoints on parallel lines.

*Some of the problems are from Jeff Erickson's lecture notes. Looking into similar problems from his lecture notes on recursion and dynamic programming is recommended.

Describe an algorithm to determine how many pairs of these line segments intersec. Prove that your algorithm is correct, and analyze its running time. For full credit your algorithm should work in $O(n \log n)$ time. A correct $O(n^2)$ time algorithm receives 10 points.

Problem 3. **[25 pts]** Design an algorithm that given an array of integers $A[1..n]$ (that contains at least one positive integer), computes

$$\max_{1 \le i \le j \le n} \sum_{k=i}^{j} A[i]$$

For example, if $A = [31, -41, 59, 26, -53, 58, 97, -93, -23, 84]$, the output must be 187. Prove that your algorithm is correct, and analyze its running time. For full credit your algorithm should work in linear time. For any credit (more than "I don't know") you algorithm should work in $O(n^2)$ time.

Problem 4. **[25 pts]** A palindrome is any string that is exactly the same as its reversal, like I, or DEED, or RACECAR, or AMANAPLANACATACANALPANAMA. Describe an algorithm to find the length of the longest subsequence of a given string that is also a palindrome. For example, the longest palindrome subsequence of MAHDYNAMICPROGRAMZLETMESHOWYOUTHEM is MHYMRORMYHM, so given that string as input, your algorithm should output the number 11. Prove that your algorithm is correct, and analyze its running time.

Here are a set of practice problems on asymptotic running time analysis, and recursion. Do *not* submit solutions for the following problems, they are just for practice.

Practice Problem A. Prove that $\log(n!) = \Theta(n \log n)$. (Logarithms are based 2)

Practice Problem B. For each of the following, indicate whether $f = O(g)$, $f = \Omega(g)$ or $f = \Theta(g)$.

(a) $f(n) = 2n - 5$, $g(n) = 1235813n + 2016$.

(b) $f(n) = n \log n$, $g(n) = 1235813n + 2016$.

(c) $f(n) = n^{2/3}$, $g(n) = 7n^{3/4}$.

(d) $f(n) = n^{1.00000001}$, $g(n) = n \log n$.

(e) $f(n) = n5^n$, $g(n) = 7^n$.

Practice Problem C. Suppose you are given a stack of n pancakes of different sizes. You want to sort the pancakes so that smaller pancakes are on top of larger pancakes. The only operation you can perform is a flip – insert a spatula under the top k pancakes, for some integer k between 1 and n, and flip them all over.

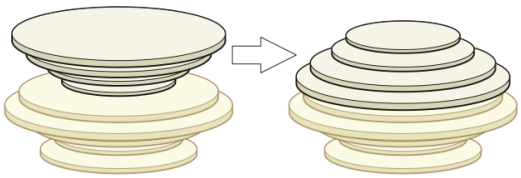

Figure 2: Flipping the top four pancakes.

(a) Describe an algorithm to sort an arbitrary stack of n pancakes using as few flips as possible. Exactly how many flips does your algorithm perform in the worst case?

(b) Now suppose one side of each pancake is burned. Describe an algorithm to sort an arbitrary stack of n pancakes, so that the burned side of every pancake is facing down, using as few flips as possible. Exactly how many flips does your algorithm perform in the worst case?