CSE 332 Data Abstractions, Spring 2013
Homework 1

Due: **Friday, April 12, 2013** at the BEGINNING of lecture. Your work should be readable as well as correct. You should refer to the written homework guidelines on the course website. This assignment has **5**, count them FIVE fabulous questions! Have fun! Please put your section (AA/AB) on your homework.

Problem 1. Some Important Sums

Much of algorithm analysis involves working with summations. Fortunately, finding the closed-form solutions to these sums often involve the simple algebraic manipulation. In the following two problems, you will compute two sums and prove a third to be true. They are the same in the 2nd and 3rd ed. of Weiss (p.27 in 3rd ed., p.26 in 2nd ed.).

(a) Weiss 1.8a

(b) Weiss 1.8b

(c) Weiss 1.12a

Problem 2. Recurrence Relations

Consider the following recurrence relation: $T(1) = 6$, and for $n > 1$, $T(n) = 1 + 2T(\lfloor n/2 \rfloor)$ Note. $\lfloor x \rfloor$ is the the *floor* function. It rounds x down to the nearest integer.

(a) Determine the value for $T(n)$ for integers n from 1 to 8.

(b) Expand the recurrence relation to get the closed form. Show your work; do not just show the final equation. For arithmetic simplicity, you may assume n is a sufficiently large power of 2 such that the floor function does not lead to rounding issues.

Problem 3. Budgeting Time

For each function $f(n)$ and time t in the following table, determine the largest size n of a problem that can be solved in time t, assuming that the algorithm to solve the problem takes $f(n)$ **micro**seconds (1 second equals 1 million microseconds). For large entries (say, those that warrant scientific notation), an estimate is sufficient. Note that for one of the rows, you will not be able to solve it analytically, and will need a calculator, spreadsheet, or small program.

f(n)	**1 second**	**1 minute**	**1 hour**	**1 day**	**1 month**	**1 year**
$500 \log_2 n$						
$1000n$						
$100n \log_2 n$						
$10n^2$						
$2n^3$						
$\frac{1}{20}2^n$						

This problem gives an orthogonal view of comparative running times from that given in lecture.

Be sure to look at the patterns in your table when you have completed it.

Problem 4. Big-O, Big-Θ

For each of the following statements, use our formal denitions of Big-O, Big-Θ, and Big-Ω either to prove the statement is **true** or to explain why it is **false**. You can assume that the functions will only have positive values (similar to what actual runtimes would be).

(a) If we have an algorithm that runs in $O(n)$ time and make some changes that cause it to run 10 times slower for all inputs, it will still run in $O(n)$ time.

(b) If $f(n) = O(g(n))$ and $h(n) = O(k(n))$, then $f(n)h(n) = O(g(n)k(n))$.

(c) If $f(n) = O(g(n))$ and $h(n) = O(k(n))$, then $f(n) + h(n) = O(g(n) + k(n))$.

(d) $2^{n+3} = \Theta(2^n)$

(e) $(2^n)^{1/3} = \Theta(2^n)$

Problem 5. Algorithm Analysis

These problems are the same in both the 2nd and 3rd edition of Weiss.

(a) Weiss 2.7a (Give the best big-O bound you can for the six program fragments. You do not have to explain why.)

(b) Weiss 2.11