CSCI E-124–Spring, 2004 Homework 1 Out: February 10, 2004
Due: Feb. 21, 2004

Please see the Web site for instructions on mailing in your homework assignment. Homework will be due by 5 p.m. on the date given above. Also, please look at the Web site regularly for updates (bugs in problems, extensions, etc.) In particular, SEND US SOME SORT OF E-MAIL (as announced on the Web page) so we know you're there and paying attention. Try to make your answers as clear and concise as possible; style will count in your overall mark. Be sure to read and know the collaboration policy in the course syllabus. Be sure to check the back of the page; problems occasionally show up there too!

1. Suppose you are given a six-sided die, that might be biased. Explain how to use die rolls to generate unbiased coin flips, and determine the expected number of die rolls until a coin flip is generated. Now suppose you want to generate unbiased die rolls (from a six-sided die) given your potentially biased die. Explain how to do this, and again determine the expected number of biased die rolls until a unbiased die rolls is generated. For both problems, you need not give the most efficient solution; however, your solution should be reasonable, and exceptional solutions will receive exceptional scores.

2. On a platform of your choice, implement the three different methods for computing the Fibonacci numbers (recursive, iterative, and matrix) discussed in lecture. Use integer variables. How fast does each method appear to be? Give precise timings if possible. (This is deliberately open-ended; give what you feel is a reasonable answer.) Can you determine the first Fibonacci number where you reach integer overflow?

   Since you should reach integer overflow with the faster methods quite quickly, modify your programs so that they return the Fibonacci numbers modulo $65536 = 2^{16}$. (In other words, make all of your arithmetic modulo 2^{16} – this will avoid overflow!) For each method, what is the largest Fibonacci number you can compute in one minute of machine time?

   Attach your source code to your assignment.

3. Indicate for each pair of expressions (A, B) in the table below the relationship between A and B. Your answer should be in the form of a table with a "yes" or "no" written in each box. For example, if A is $O(B)$, then you should put a "yes" in the first box.

A	B	O	o	Ω	ω	Θ
$\log n$	$\log(n^2)$					
$\log(n!)$	$\log(n^n)$					
$\sqrt[3]{n}$	$(\log n)^6$					
$n^2 2^n$	3^n					
$(n^2)!$	n^n					
$\frac{n^2}{\log n}$	$n\log(n^2)$					
$(\log n)^{\log n}$	$\frac{n}{\log(n)}$					
$100n + \log n$	$(\log n)^3 + n$					

4. For all of the problems below, when asked to give an example, you should give an example whose domain and range is the positive integers. (No cheating with 0's!)

   - Find (with proof) a function f_1 such that $f_1(2n)$ is $O(f_1(n))$.
   - Find (with proof) a function f_2 such that $f_2(2n)$ is not $O(f_2(n))$.

- Show that if $f(n)$ is $O(g(n))$, and $g(n)$ is $O(h(n))$, then $f(n)$ is $O(h(n))$.
- Give a proof or a counterexample: if f is not $O(g)$, then g is $O(f)$.
- Give a proof or a counterexample: if f is $o(g)$, then f is $O(g)$.

5. Buffy and Willow are facing an evil demon named Stooge, living inside Willow's computer. In an effort to slow the Scooby Gang's computing power to a crawl, the demon has replaced Willow's hand-designed super-fast sorting routine with the following recursive sorting algorithm, known as StoogeSort. For simplicity, we think of Stoogesort as running on a list of distinct numbers. StoogeSort runs in three phases. In the first phase, the first 2/3 of the list is (recursively) sorted. In the second phase, the final 2/3 of the list is (recusively) sorted. Finally, in the third phase, the first 2/3 of the list is (recursively) sorted again.

   Willow notices some sluggishness in her system, but doesn't notice any errors from the sorting routine. This is because StoogeSort correctly sorts. For the first part of your problem, prove rigorously that StoogeSort correctly sorts. (Note: in your proof you should explain what should happen if the number of items to be sorted is not divisible by 3. You may assume all numbers to be sorted are distinct.) But StoogeSort can be slow. Derive a recurrence describing its running time, and use the recurrence to bound the asymptotic running time of Stoogesort.

6. Solve the following recurrences exactly, and then prove your solutions are correct by induction. (Hint: Graph values and guess the form of a solution: then prove that your guess is correct.)

   - $T(1) = 1, T(n) = T(n-1) + 3n - 3$.
   - $T(1) = 1, T(n) = 2T(n-1) + 2n - 1$.

7. Give asymptotic bounds for $T(n)$ in each of the following recurrences.

   - $T(n) = 4T(n/2) + n^3$.
   - $T(n) = 17T(n/4) + n^2$.
   - $T(n) = 9T(n/3) + n^2$.
   - $T(n) = T(\sqrt{n}) + 1$. (Hint: try to make this recurrence look like the others using a substitution.)