Problem Set 1 (due Monday, January 24)

1. (10 points) On the toughness of glass beakers

You are given the task of determining the toughness of a collection of identical glass beakers that are going to be used in space experiments by NASA. The question you need to answer is simple: what is the highest floor of the Empire State Building from which the glass beaker can be dropped safely without shattering it?

Of course, in order to answer this question satisfactorily, you should be allowed to break at least one beaker. One simple solution is to start from floor 1, and for each floor going up, repeatedly drop the beaker to the ground from the floor, until the beaker breaks. Let us call the drop from a floor as a *test*. If the Empire State Building has n floors, then this could require as many as n tests.

Suppose you are given a budget of k beakers. You would like to determine the highest floor from which it is safe to drop the beaker, using as few tests as possible, while breaking at most k beakers. Let $f_k(n)$ denote the maximum number of tests you need to do with a budget of k beakers. In our solution above with $k = 1$, we obtained $f_1(n) = n$.

(a) Describe your best possible strategy for $k = 2$ that aims to minimize the number of tests. What is $f_2(100)$? Derive an expression for $f_2(n)$.

(b) Can you generalize your approach to arbitrary k? Your strategy must have the property that $f_k(n)$ grows asymptotically slower than $f_{k-1}(n)$; i.e. $\lim_{n\to\infty} f_k(n)/f_{k-1}(n) = 0$, for all $k \geq 2$.

2. (10 points) Comparison of functions

Arrange the following functions in order from the slowest growing function to the fastest growing function. Briefly justify your answer. (*Hint:* It may help to express each function as a power of 2 and then compare. It may also help to plot the functions and obtain an estimate of their relative growth rates.)

$$(\lg n)^{\lg n} \quad (\frac{3}{2})^{2n} \quad n^{100} \quad 2^{\sqrt{\lg n}}$$

3. (10 points) An alternative gcd algorithm

Euclid's algorithm computes the gcd of two positive numbers a and b ($a \geq b$) by reducing the problem to that of computing the gcd of b and ($a \bmod b$), where $a \bmod b$ is often much smaller than a. Thus the algorithm rapidly converges to the final result. While the calculation of $a \bmod b$ is not hard, it requires division and may be more expensive to do on some very simple computing devices. In the world of binary arithmetic, division by 2 is much easier to compute. Consider the following alternative algorithm for gcd using subtraction and division by 2, developed below through a series of exercises.

(a) Show that if a and b are both even, then $\gcd(a, b) = 2 \cdot \gcd(a/2, b/2)$.

(b) Show that if a is even and b is odd, then $\gcd(a, b) = \gcd(a/2, b)$. (Similarly, if a is odd and b is even, then $\gcd(a, b) = \gcd(a, b/2)$.)

(c) Show that if a and b are both odd, then $\gcd(a, b) = \gcd((a - b)/2, b)$.

(*Hint:* Show that $\gcd(a, b) = \gcd(a - b, b)$; for inspiration, see the proof of correctness for the Euclidean Algorithm. If a and b are both odd, what is true about $a - b$? Now apply your result from part (b) above.)

(d) Apply the above three claims repeatedly to compute the following: $\gcd(323, 76)$, $\gcd(693, 378)$. Show your work.

4. (10 points) An addition circuit

Exercise 1.30 of text.