In lecture on Thursday, we saw a divide-and-conquer algorithm, due to Karatsuba, that multiplies two n-digit integers using $O(n^{\lg 3})$ single-digit additions, subtractions, and multiplications. In this lab, we'll look at one application of Karatsuba's algorithm: converting a number from binary to decimal.

(The standard algorithm that computes one decimal digit of x at a time, by computing $x \bmod 10$ and then recursively converting $\lfloor x/10 \rfloor$, requires $\Theta(n^2)$ time.)

1. Consider the following recurrence, originally used by the Sanksrit prosodist Piṅgala in the second century BCE, to compute the number 2^n:

$$2^n = \begin{cases} 1 & \text{if } n = 0 \\ (2^{n/2})^2 & \text{if } n > 0 \text{ is even} \\ 2 \cdot (2^{\lfloor n/2 \rfloor})^2 & \text{if } n \text{ is odd} \end{cases}$$

   We can use this algorithm to compute the decimal representation of 2^n, by representing all numbers using arrays of decimal digits, and implementing squaring and doubling using decimal arithmetic. Suppose we use Karatsuba's algorithm for decimal multiplication. What is the running time of the resulting algorithm?

2. We can use a similar algorithm to compute the decimal representation of any integer. Suppose we are given an integer x as an array of n bits (binary digits). Write $x = a \cdot 2^{n/2} + b$, where a is represented by the top $n/2$ bits of x, and b is represented by the bottom $n/2$ bits of x. Then we can convert x into decimal as follows:

   (a) Recursively convert a into decimal.
   (b) Recursively convert $2^{n/2}$ into decimal.
   (c) Recursively convert b into decimal.
   (d) Compute $x = a \cdot 2^{n/2} + b$ using decimal multiplication and addition.

   Now suppose we use Karatsuba's algorithm for decimal multiplication. What is the running time of the resulting algorithm? (For simplicity, you can assume n is a power of 2.)

3. Now suppose instead of converting $2^{n/2}$ to decimal by recursively calling the algorithm from problem 2, we use the specialized algorithm for powers of 2 from problem 1. Now what is the running time of the resulting algorithm (assuming we use Karatsuba's multiplication algorithm as before)?

Harder problem to think about about later:

4. In fact, it is possible to multiply two n-digit decimal numbers in $O(n \log n)$ time. Describe an algorithm to compute the decimal representation of an arbitrary n-bit binary number in $O(n \log^2 n)$ time.