1. Consider the function \( \text{Factorial}(n) = n! \) for a nonnegative integer \( n \).

(a) Up to constant factors, many bits does it take to write \( n! \) down? You may use that \( (\frac{n}{2})^{n/2} \leq n! \leq n^n \).

(b) Consider the standard factorial implementation: \( f(0) = 1 \), and \( f(n) = n \cdot f(n - 1) \) for \( n \geq 1 \). How many multiplications does it perform?

(c) How much time does the standard factorial implementation take, using standard multiplication? Note that standard multiplication takes \( \Theta(kl) \) time to multiply a \( k \)-bit number by an \( l \)-bit number.

(d) Can you use Karatsuba multiplication to speed this up? If so, by how much?

(e) Now consider the following recursive implementation \( g(n, m) \) of \( \frac{n!}{(n-m)!} \), for \( 0 \leq m \leq n \):

\[
g(n, m) = \begin{cases} 
1 & \text{if } n = 0 \text{ or } m = 0 \\
n & \text{if } m = 1 \\
g(n, \lfloor m/2 \rfloor) \cdot g(n - \lfloor m/2 \rfloor, \lceil m/2 \rceil) & \text{otherwise}
\end{cases}
\]

Show that \( g(n, m) \) correctly computes \( \frac{n!}{(n-m)!} \), and so \( n! = g(n, n) \).

(f) Show that \( g(n, m) \) is \( \Theta(m \log n) \) bits long.

(g) Let \( M(k) \) denote the time to multiply two \( k \)-bit integers. Let \( T(m) \) be the maximum over all \( n' \leq n \) of the time to compute \( g(n', m) \). Ignoring the floors and ceilings in \( g \), show that

\[
T(m) \leq 2T(m/2) + M(m \log n).
\]

(h) What does this recurrence solve to, for standard and for Karatsuba multiplication? When \( m = n \), so \( g(n, n) = n! \), how does this compare to the standard factorial implementation?

2. There’s a Jupyter Notebook linked from the class webpage. Run through it, then answer the questions at the end. Don’t wait till the last day to do this: setting up the required libraries may take some time.