CS 3510 - Spring 2009
Practice Problems 2

Here are some easier exercises to review (or in some cases introduce) modulus and big O notation to make sure you are comfortable with them.

1. Recall that $x \bmod n$ is a function that maps x to a number between 0 and $n-1$ and is the remainder when you divide x by n. We say that $x \equiv y \pmod{n}$ if $x \bmod n = y \bmod n$, which is equivalent to saying that n is a divisor of $x - y$.

   Show that $(x \bmod n)(y \bmod n)$ might not equal $xy \bmod n$. Now show that $(x \bmod n)(y \bmod n) \equiv xy \pmod{n}$.

2. Calculate (by hand):
   (a) $-354 \bmod 500$.
   (b) $453 * 243 \pmod{1000}$.
   (c) $25483 \mod 742$.
   (d) $848332 + 85392$ (as an integer).
   (e) $257 * 834 \pmod{53}$.

3. Given two functions f and g, we say that $f = O(g)$ if there exists integers $c > 0, n$ such that for all $n' \geq n$, $f(n') \leq cg(n')$. Show the following are true:
   (a) $3n = O(n)$.
   (b) $3n = O(n + 5)$.
   (c) $3n + 7 = O(n^2)$.
   (d) $n \log n = O(n^2)$.

4. We showed in class that the product of two n digit numbers can be calculated by the simple (fourth grade) algorithm in $O(n^2)$ operations. Show that we can calculate $k * x$ faster if x is an n digit number, but k is a 1 digit number. How many operations does your algorithm require? What if k had 7 digits?