CS 3510 - Spring 2009
Pratice 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 $(x \bmod n)(y \bmod n) \equiv xy \pmod n$.

   For the first part we can just come up with a counterexample to show that these two terms are not always equal. Consider, for example, $x = 15$, $y = 15$ and $n = 10$. Then $(x \bmod n) = 5$, $(y \bmod n) = 5$ and their product (over the integers) is 25. Clearly this is not equal to $(xy \bmod 10) = 5$.

   However, we can indeed show that $(x \bmod n)(y \bmod n) \equiv xy \pmod n$, as follows.

   If $(x \bmod n) = q$, then there exists p such that

   $$x = pn + q.$$

   Similarly, if $(y \bmod n) = s$, then there exists r such that

   $$y = rn + s.$$

   Therefore

   $$\begin{aligned} xy &= (pn + q)(rn + s) \\ &= pr(n^2) + (ps + qr)n + qs, \end{aligned}$$

   so

   $$xy \equiv qs \pmod n.$$

2. Calculate (by hand):

(a) $-354 \bmod 500$.
Mod operations on negative numbers can be thought of as turning the second hand of a clock counter-clockwise. Thus $-354 \bmod 500$ is 354 units counter-clockwise on a clock with 500 seconds. Now we can say $-354 = 500 * -1 + 146$. Thus $-354 \bmod 500 = 146$.

(b) $453 * 243 \pmod{1000}$.
$= (400 + 50 + 3) * (200 + 40 + 3) \pmod{1000}$
Now we drop all subproducts whose expansion results in a multiple of 1000. For example, 50*40 = 2000 and is dropped as a result. Therefore,

$$\begin{aligned}
(400{+}50 + 3) * (200 + 40 + 3) &\pmod{1000} \\
&= 400 * 3 + 50 * 3 + 3 * 200 + 3 * 40 + 3 * 3 \pmod{1000} \\
&= 1200 + 150 + 243 * 3 \pmod{1000} \\
&= 1200 + 150 + 729 \pmod{1000} \\
&= 1350 + 729 \pmod{1000} \\
&= 2079 \pmod{1000} \\
&= 79.
\end{aligned}$$

(c) $25483 \bmod 742$.
$25483 \bmod 742 = 34$, so

$$\begin{aligned}
25483 &\equiv 25483 - 742 * 34 \\
&\equiv 25483 - 25338 \\
&\equiv 255 \pmod{742}.
\end{aligned}$$

(d) $848332 + 85392$ (as an integer).
$848332 + 85392 = 933724$. (Addition is $O(n)$.)

(e) $257 * 834 \pmod{53}$

$$\begin{aligned}
257 * 834 \pmod{53} &\equiv 257 \pmod{53} * 843 \pmod{53} \\
&\equiv 45 * 39 \pmod{53} \\
&\equiv 1755 \pmod{53} \\
&\equiv 6 \mod 53
\end{aligned}$$

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) Show $3n = O(n)$.
Let $c = 3$. For all $n \geq 0$, we have $3n \leq cn$. Thus, by definition $3n = O(n)$.

(b) Show $3n = O(n+5)$.
Let $c = 3$. For all $n \geq 0$, $3n \leq 3(n+5)$. Thus $3n = O(n+5)$.

(c) Show $3n + 7 = O(n^2)$.
Let $c = 10$. For $n \geq 0$, we have $3n + 7 \leq 3n^2 + 7n^2 = 10n^2$. Let's find where $3n + 7 = n^2$
Therefore $3n + 7 = O(n^2)$.

(d) Show $n \log n = O(n^2)$.
For all $x > 1$, $\log x < x$
Thus $n \log n \leq n^2$ for all $n \geq 1$. Therefore $n \log n = O(n^2)$.

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?
Let $x = a_1 a_2 a_3 \ldots a_n$ and $k = b_1$ be bit representations.
$\Rightarrow x * k = a_1 a_2 a_3 \ldots a_n * b_1$
One can see that it involves n n by 1 bit multiplications and $n - 1$ possible carry additions and so it is $O(n)$.Also consider the case where $k = b_1 b_2 b_3 \ldots b_7$.In this case, the same thing will occur seven timesas well as an additional shift and addition. Thus the $O(n)$ cost will be incurred seven times, which only differs from the original running time by a constant factor. So if the size of k remains constant the overall running time will be $O(n)$.