SICP: Exercise 1.25

August 26, 2024

Alyssa P. Hacker complains that we went to a lot of extra work in writing expmod. After all, she says, since we already know how to compute exponentials, we could have simply written

```
(define (expmod base exp m)
  (remainder (fast-expt base exp) m))
```

Is she correct? Would this procedure serve as well for our fast prime tester? Explain.

There is a critical difference between the two implementations. Specifically, in Alyssa's implementation, the full exponential will be calculated first, which could result in us having to handle very large numbers. In comparison, the original `expmod` implementation only has to deal with numbers that never get bigger than `m` (since the outermost call at the end is always wrapped in a `remainder`), thus saving memory. We probably even save computational steps since we don't have to deal with numbers bigger than our registers.

For example, if we were to calculate $2^{1000} \bmod 5$, the original implementation will only deal with numbers $x \leq 5$, while Alyssa's version has to deal with the 2^{1000} (a number with 300 zeroes) first, and only *then* takes the remainder, leaving a measly 1 as the result instead of having to work with a 1 KB big number.