Exercise 1.44.

The idea of *smoothing* a function is an important concept in signal processing. If f is a function and dx is some small number, then the smoothed version of f is the function whose value at a point x is the average of $f(x - dx)$, $f(x)$ and $f(x + dx)$. Write a procedure `smooth` that takes as input a procedure that computes f and returns a procedure that computes the smoothed f. It is sometimes valuable to repeatedly smooth a function (that is, smooth the smoothed function, and so on) to obtained the *n-fold smoothed function*. Show how to generate the n-fold smoothed function of any given function using `smooth` and `repeated` from exercise 1.43.

Answer.

Given the discription of `smooth`, we can express it in Lisp fairly straightforward,

```
(define smooth
  (lambda (f)
    (lambda (x)
      (/ (+ (f (- x dx))
            (f x)
            (f (+ x dx)))
         3))))
```

Using `smooth` we've defined above and `repeated` from exercise 1.43, we can write the procedure `multi-smooth` to obtain the *n-fold smoothed function*,

```
(define multi-smooth
  (lambda (f n)
    (repeated (smooth f) n)))
```