3a: (HKK 2.12 and 3.2) Your goal is to write two procedures, one iterative and one recursive, for finding the number of 2's in the prime factorization of a positive integer. In other words, if the integer n can be written $2^i \times k$ for k odd, your procedure would return i.

(a) Write a procedure which generates a recursive process.

(b) Write a procedure which generates an iterative process.

(c) Show how your two procedures work when given the argument 24 using the substitution model. Explain why your first procedure is iterative and the second is recursive.

3b: [HKK 3.15] Consider the following two procedures:

```
(define f
  (lambda (n)
    (if (= n 0)
        0
        (g (- n 1)))))

(define g
  (lambda (n)
    (if (= n 0)
        1
        (f (- n 1)))))
```

(a) Use the substitution model to evaluate each of `(f 1)`, `(f 2)`, and `(f 3)`.

(b) Can you predict `(f 4)`? `(f 5)`? In general, which arguments cause `f` to return 0 and which cause it to return 1? (You need only consider nonnegative integers.)

(c) Is the process generated by `f` iterative or recursive? Explain.

3c: [HKK 3.17] *Falling factorial powers* are similar to normal powers and also similar to factorials. We write them as $n^{\underline{k}}$ and say "n to the k falling." This means that k consecutive numbers should be multiplied together, starting with n and working downward. For example, $7^{\underline{3}} = 7 \times 6 \times 5$ (i.e., three consecutive numbers from 7 downward multiplied together).

Write a procedure for calculating falling factorial powers that generates an iterative process.