Procedures and Processes They Generate

The substitution model

To evaluate a combination (other than a special form):

1. Evaluate the subexpressions of the combination (in any order)

2. Apply the procedure that is the value of the leftmost subexpression to the arguments that are the values of the other subexpressions

To apply a compound procedure to arguments

1. Evaluate the body of the procedure with each formal parameter replaced by the corresponding argument

Orders of growth

We say that function $R$ of parameter $n$ has order of growth $\Theta(f(n))$, written $R(n) = \Theta(f(n))$ if there are positive constants $k_1$ and $k_2$ independent of $n$ and an $n_0$ such that for all $n > n_0$:

$$k_1 f(n) \leq R(n) \leq k_2 f(n).$$
Code

(define (times-1 a b)
  (if (= b 0)
      0
      (+ a (times-1 a (- b 1))))

(define (times-2 a b)
  (define (times-iter a b result)
    (if (= b 0)
        result
        (times-iter (- b 1) (+ result a))))
  (times-iter a b 0))

(define (move-tower size from to extra)
  (if (= size 1)
      (print-move from to)
      (begin (move-tower (- size 1) from extra to)
              (move-tower 1 from extra)
              (move-tower (- size 1) extra to from)))

(define (print-move from to)
  (begin (newline)
          (display "Move top disk from ")
          (display from)
          (display " to ")
          (display to)))

(define (fast-times-1 a b)
  (cond ((= b 0) 0)
        ((even? b) (fast-times-1 (double a) (halve b)))
        (else (+ a (fast-times-1 a (- b 1))))))

(define (fast-times-2 a b)
  (define (times-iter a b result)
    (cond ((= b 0) result)
          ((even? b) (times-iter (double a) (halve b) result))
          (else (times-iter a (- b 1) (+ a result))))
  (times-iter a b 0))

(define (slow-expmod b e m)
  (if (= e 0)
      1
      (modulo (* b (slow-expmod b (- e 1) m)) m)))

(define (expmod b e m)
  (cond ((= e 0) 1)
        ((even? e)
         (modulo (square (expmod b (/ e 2) m)) m))
        (else
         (modulo (* b (expmod b (- e 1) m)) m)))))