In-Class Problems — Week 7, Wed

Example Substitution Model Evaluations

Problem. Identify the control-parse and exact Substitution Model rewrite rule being applied at each step in the evaluations of `(factorial 3)` and `(iter-fact 3)` and `(y-factorial 3)` below.

```
submodel-eval>>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1)))))))
  (factorial 3))
==(1, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1)))))))
  ((lambda (n) (if (<= n 0) 1 (* n (factorial (- n 1))))) 3))
==(2, lambda)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 3))
  ((lambda () (if (<= n 0) 1 (* n (factorial (- n 1)))))))
==(3, lambda)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 3))
  (if (<= n 0)
      1
      (* n (factorial (- n 1)))))
==(4, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 3))
```

```
  (if (<= 3 0)
      1
      (* n (factorial (- n 1)))))
==(5, builtin)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 3))
  (if ()
      1
      (* n (factorial (- n 1)))))
==(6, if)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 3))
  (* n (factorial (- n 1))))
==(7, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 3))
  (* 3 (factorial (- n 1))))
==(8, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 3))
  (* 3 ((lambda (n) (if (<= n 0) 1 (* n (factorial (- n 1))))) (- n 1))))
==(9, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1)))))))
  (* 3 ((lambda (n) (if (<= n 0) 1 (* n (factorial (- n 1))))) (- 3 1))))
==(10, builtin)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1)))))))
  (* 3 ((lambda (n) (if (<= n 0) 1 (* n (factorial (- n 1))))) 2)))
==(11, lambda)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 2))
  (* 3 ((lambda () (if (<= n 0) 1 (* n (factorial (- n 1)))))))
==(12, lambda)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 2))
  (* 3 (if (<= n 0) 1 (* n (factorial (- n 1))))))
```

```
==(13, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 2))
  (* 3 (if (<= 2 0) 1 (* n (factorial (- n 1))))))
==(14, builtin)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 2))
  (* 3 (if () 1 (* n (factorial (- n 1))))))
==(15, if)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 2))
  (* 3 (* n (factorial (- n 1)))))
==(16, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 2))
  (* 3 (* 2 (factorial (- n 1)))))
==(17, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 2))
  (*
   3
   (* 2
      ((lambda (n) (if (<= n 0) 1 (* n (factorial (- n 1))))) (- n 1)))))
==(18, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1)))))))
  (*
   3
   (* 2
      ((lambda (n) (if (<= n 0) 1 (* n (factorial (- n 1))))) (- 2 1)))))
==(19, builtin)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1)))))))
  (* 3 (* 2 ((lambda (n) (if (<= n 0) 1 (* n (factorial (- n 1))))) 1))))
==(20, lambda)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 1))
```

```
  (* 3 (* 2 ((lambda () (if (<= n 0) 1 (* n (factorial (- n 1)))))))))
==(21, lambda)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 1))
  (* 3 (* 2 (if (<= n 0) 1 (* n (factorial (- n 1)))))))
==(22, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 1))
  (* 3 (* 2 (if (<= 1 0) 1 (* n (factorial (- n 1)))))))
==(23, builtin)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 1))
  (* 3 (* 2 (if () 1 (* n (factorial (- n 1)))))))
==(24, if)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 1))
  (* 3 (* 2 (* n (factorial (- n 1))))))
==(25, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 1))
  (* 3 (* 2 (* 1 (factorial (- n 1))))))
==(26, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 1))
  (*
   3
   (*
    2
    (*
     1
     ((lambda (n) (if (<= n 0) 1 (* n (factorial (- n 1))))) (- n 1))))))
==(27, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1)))))))
  (*
   3
   (*
    2
    (*
     1
```

```
     ((lambda (n) (if (<= n 0) 1 (* n (factorial (- n 1))))) (- 1 1))))))
==(28, builtin)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1)))))))
  (*
   3
   (* 2
      (* 1 ((lambda (n) (if (<= n 0) 1 (* n (factorial (- n 1))))) 0)))))
==(29, lambda)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 0))
  (*
   3
   (* 2 (* 1 ((lambda () (if (<= n 0) 1 (* n (factorial (- n 1)))))))))
==(30, lambda)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 0))
  (* 3 (* 2 (* 1 (if (<= n 0) 1 (* n (factorial (- n 1))))))))
==(31, instantiate)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 0))
  (* 3 (* 2 (* 1 (if (<= 0 0) 1 (* n (factorial (- n 1))))))))
==(32, builtin)==>
(letrec ((factorial
          (lambda (n)
            (if (<= n 0)
                1
                (* n (factorial (- n 1))))))
         (n 0))
  (* 3 (* 2 (* 1 (if #t 1 (* n (factorial (- n 1))))))))
==(33, if)==>
(letrec ()
  (* 3 (* 2 (* 1 1))))
==(34, builtin)==>
(letrec ()
  (* 3 (* 2 1)))
==(35, builtin)==>
(letrec ()
  (* 3 2))
==(36, builtin)==>
(letrec ()
  6)
Final value after 37 steps:
6
```

```
submodel-eval>>
(letrec ((iter-fact
          (lambda (n)
            (letrec ((helper
                      (lambda (n p)
                        (if (<= n 0)
                            p
                            (helper (- n 1) (* n p))))))
              (helper n 1)))))
  (iter-fact 3))
==(1, instantiate)==>
(letrec ()
  ((lambda (n)
     (letrec ((helper
               (lambda (n p)
                 (if (<= n 0)
                     p
                     (helper (- n 1) (* n p))))))
       (helper n 1)))
   3))
==(2, lambda)==>
(letrec ((n 3))
  ((lambda ()
     (letrec ((helper
               (lambda (n p)
                 (if (<= n 0)
                     p
                     (helper (- n 1) (* n p))))))
       (helper n 1)))))
==(3, lambda)==>
(letrec ((n 3))
  (letrec ((helper
            (lambda (n p)
              (if (<= n 0)
                  p
                  (helper (- n 1) (* n p))))))
    (helper n 1)))
==(4, lets)==>
(letrec ((n 3)
         (helper
          (lambda (n p)
            (if (<= n 0)
                p
                (helper (- n 1) (* n p))))))
  (helper n 1))
==(5, instantiate)==>
(letrec ((n 3)
         (helper
          (lambda (n p)
            (if (<= n 0)
                p
                (helper (- n 1) (* n p))))))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) n 1))
==(6, instantiate)==>
(letrec ((helper
          (lambda (n p)
            (if (<= n 0)
                p
                (helper (- n 1) (* n p))))))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 3 1))
==(7, lambda)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 3))
  ((lambda (p) (if (<= n 0) p (helper (- n 1) (* n p)))) 1))
```

```
==(8, lambda)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 3)
         (p 1))
  ((lambda () (if (<= n 0) p (helper (- n 1) (* n p))))))
==(9, lambda)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 3)
         (p 1))
  (if (<= n 0)
      p
      (helper (- n 1) (* n p))))
==(10, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 3)
         (p 1))
  (if (<= 3 0)
      p
      (helper (- n 1) (* n p))))
==(11, builtin)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 3)
         (p 1))
  (if ()
      p
      (helper (- n 1) (* n p))))
==(12, if)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 3)
         (p 1))
  (helper (- n 1) (* n p)))
==(13, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 3)
         (p 1))
  ((lambda (n p) (if (<= n 0) p (helper (- n 1) (* n p)))) (- n 1)
                                                           (* n p)))
==(14, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 3)
         (p 1))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) (- 3 1)
                                               (* n p)))
==(15, builtin)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 3)
         (p 1))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 2 (* n p)))
==(16, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (p 1))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 2 (* 3 p)))
==(17, instantiate)==>
(letrec ((helper
          (lambda (n p)
```

```
           (if (<= n 0)
               p
               (helper (- n 1) (* n p))))))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 2 (* 3 1)))
==(18, builtin)==>
(letrec ((helper
          (lambda (n p)
            (if (<= n 0)
                p
                (helper (- n 1) (* n p))))))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 2 3))
==(19, lambda)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 2))
  ((lambda (p) (if (<= n 0) p (helper (- n 1) (* n p)))) 3))
==(20, lambda)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 2)
         (p 3))
  ((lambda () (if (<= n 0) p (helper (- n 1) (* n p))))))
==(21, lambda)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 2)
         (p 3))
  (if (<= n 0)
      p
      (helper (- n 1) (* n p))))
==(22, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 2)
         (p 3))
  (if (<= 2 0) p (helper (- n 1) (* n p))))
==(23, builtin)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 2)
         (p 3))
  (if () p (helper (- n 1) (* n p))))
==(24, if)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 2)
         (p 3))
  (helper (- n 1) (* n p)))
==(25, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 2)
         (p 3))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) (- n 1)
     (* n p)))
==(26, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 2)
         (p 3))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) (- 2 1)
     (* n p)))
==(27, builtin)==>
```

```
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 2)
         (p 3))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 1 (* n p)))
==(28, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (p 3))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 1 (* 2 p)))
==(29, instantiate)==>
(letrec ((helper
          (lambda (n p)
            (if (<= n 0)
                p
                (helper (- n 1) (* n p))))))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 1 (* 2 3)))
==(30, builtin)==>
(letrec ((helper
          (lambda (n p)
            (if (<= n 0)
                p
                (helper (- n 1) (* n p))))))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 1 6))
==(31, lambda)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 1))
  ((lambda (p) (if (<= n 0) p (helper (- n 1) (* n p)))) 6))
==(32, lambda)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 1)
         (p 6))
  ((lambda () (if (<= n 0) p (helper (- n 1) (* n p))))))
==(33, lambda)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 1)
         (p 6))
  (if (<= n 0) p (helper (- n 1) (* n p))))
==(34, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 1)
         (p 6))
  (if (<= 1 0)
      p
      (helper (- n 1) (* n p))))
==(35, builtin)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 1)
         (p 6))
  (if ()
      p
      (helper (- n 1) (* n p))))
==(36, if)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 1)
         (p 6))
  (helper (- n 1) (* n p)))
```

```
==(37, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 1)
         (p 6))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) (- n 1)
     (* n p)))
==(38, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 1)
         (p 6))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) (- 1 1)
     (* n p)))
==(39, builtin)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 1)
         (p 6))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 0 (* n p)))
==(40, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (p 6))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 0 (* 1 p)))
==(41, instantiate)==>
(letrec ((helper
          (lambda (n p)
            (if (<= n 0)
                p
                (helper (- n 1) (* n p))))))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 0 (* 1 6)))
==(42, builtin)==>
(letrec ((helper
          (lambda (n p)
            (if (<= n 0)
                p
                (helper (- n 1) (* n p))))))
  ((lambda (n p)
     (if (<= n 0) p (helper (- n 1) (* n p)))) 0 6))
==(43, lambda)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 0))
  ((lambda (p) (if (<= n 0) p (helper (- n 1) (* n p)))) 6))
==(44, lambda)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 0)
         (p 6))
  ((lambda () (if (<= n 0) p (helper (- n 1) (* n p))))))
==(45, lambda)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 0)
         (p 6))
  (if (<= n 0) p (helper (- n 1) (* n p))))
==(46, instantiate)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
         (n 0)
         (p 6))
```

```
  (if (<= 0 0) p (helper (- n 1) (* n p))))
==(47, builtin)==>
(letrec ((helper (lambda (n p)
   (if (<= n 0) p (helper (- n 1) (* n p)))))
 (n 0)
         (p 6))
  (if #t p (helper (- n 1) (* n p))))
==(48, if)==>
(letrec ((p 6))
  p)
==(49, instantiate)==>
(letrec ()
  6)
Final value after 50 steps:
6

submodel-eval>>
```

```
submodel-eval>>
(letrec ((y-factorial
          (lambda (n)
            (letrec ((y
                      (lambda (f)
                        ((lambda (x)
                           (f (lambda (z) ((x x) z))))
                         (lambda (x)
                           (f (lambda (z) ((x x) z)))))))
                     (fact-def
                      (lambda (fact)
                        (lambda (n)
                          (if (<= n 0)
                              1
                              (* n (fact (- n 1))))))))
              ((y fact-def) n)))))
  (y-factorial 3))
==(1, instantiate)==>
(letrec ()
  ((lambda (n)
     (letrec ((y
               (lambda (f)
                 ((lambda (x)
                    (f (lambda (z) ((x x) z))))
                  (lambda (x)
                    (f (lambda (z) ((x x) z)))))))
              (fact-def
               (lambda (fact)
                 (lambda (n)
                   (if (<= n 0)
                       1
                       (* n (fact (- n 1))))))))
       ((y fact-def) n)))
   3))
==(2, lambda)==>
(letrec ((n 3))
  ((lambda ()
     (letrec ((y
               (lambda (f)
                 ((lambda (x)
                    (f (lambda (z) ((x x) z))))
                  (lambda (x)
                    (f (lambda (z) ((x x) z)))))))
              (fact-def
               (lambda (fact)
                 (lambda (n)
                   (if (<= n 0)
                       1
                       (* n (fact (- n 1))))))))
       ((y fact-def) n)))))
==(3, lambda)==>
(letrec ((n 3))
  (letrec ((y
            (lambda (f)
              ((lambda (x)
                 (f (lambda (z) ((x x) z))))
               (lambda (x)
                 (f (lambda (z) ((x x) z)))))))
           (fact-def
            (lambda (fact)
              (lambda (n)
                (if (<= n 0)
                    1
                    (* n (fact (- n 1))))))))
    ((y fact-def) n)))
==(4, lets)==>
(letrec ((n 3)
```

```
         (y
          (lambda (f)
            ((lambda (x)
               (f (lambda (z) ((x x) z))))
             (lambda (x)
               (f (lambda (z) ((x x) z)))))))
         (fact-def
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1))))))))
  ((y fact-def) n))
==(5, instantiate)==>
(letrec ((n 3)
         (fact-def
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1))))))))
  (((lambda (f)
      ((lambda (x)
         (f (lambda (z) ((x x) z))))
       (lambda (x)
         (f (lambda (z) ((x x) z))))))
    fact-def)
   n))
==(6, instantiate)==>
(letrec ((n 3))
  (((lambda (f)
      ((lambda (x)
         (f (lambda (z) ((x x) z))))
       (lambda (x)
         (f (lambda (z) ((x x) z))))))
    (lambda (fact)
      (lambda (n)
        (if (<= n 0)
            1
            (* n (fact (- n 1))))))))
   n))
==(7, lambda)==>
(letrec ((n 3)
         (f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1))))))))
  (((lambda ()
      ((lambda (x)
         (f (lambda (z) ((x x) z))))
       (lambda (x)
         (f (lambda (z) ((x x) z))))))
   n))
==(8, lambda)==>
(letrec ((n 3)
         (f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1))))))))
  (((lambda (x)
      (f (lambda (z) ((x x) z))))
    (lambda (x)
      (f (lambda (z) ((x x) z)))))
```

```
   n))
==(9, lambda)==>
(letrec ((n 3)
         (f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (((lambda () (f (lambda (z) ((x x) z))))) n))
==(10, lambda)==>
(letrec ((n 3)
         (f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  ((f (lambda (z) ((x x) z))) n))
==(11, instantiate)==>
(letrec ((n 3)
         (f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (((lambda (fact)
      (lambda (n)
        (if (<= n 0)
            1
            (* n (fact (- n 1))))))
    (lambda (z)
      ((x x) z)))
   n))
==(12, lambda)==>
(letrec ((n 3)
         (f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z))))
  (((lambda () (lambda (n) (if (<= n 0) 1 (* n (fact (- n 1)))))) n))
==(13, lambda)==>
(letrec ((n 3)
         (f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z))))
  ((lambda (n) (if (<= n 0) 1 (* n (fact (- n 1))))) n))
==(14, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
```

```
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z))))
  ((lambda (n) (if (<= n 0) 1 (* n (fact (- n 1))))) 3))
==(15, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 3))
  ((lambda () (if (<= n 0) 1 (* n (fact (- n 1)))))))
==(16, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 3))
  (if (<= n 0)
      1
      (* n (fact (- n 1)))))
==(17, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 3))
  (if (<= 3 0)
      1
      (* n (fact (- n 1)))))
==(18, builtin)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 3))
  (if ()
      1
      (* n (fact (- n 1)))))
==(19, if)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 3))
  (* n (fact (- n 1))))
==(20, instantiate)==>
(letrec ((f
          (lambda (fact)
```

```
           (lambda (n)
             (if (<= n 0)
                 1
                 (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 3))
  (* 3 (fact (- n 1))))
==(21, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (n 3))
  (* 3 ((lambda (z) ((x x) z)) (- n 1))))
==(22, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (* 3 ((lambda (z) ((x x) z)) (- 3 1))))
==(23, builtin)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (* 3 ((lambda (z) ((x x) z)) 2)))
==(24, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (z 2))
  (* 3 ((lambda () ((x x) z)))))
==(25, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (z 2))
  (* 3 ((x x) z)))
==(26, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (z 2))
  (* 3 (((lambda (x) (f (lambda (z) ((x x) z)))) x) z)))
==(27, instantiate)==>
```

```
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 2))
  (*
   3
   (((lambda (x)
       (f (lambda (z) ((x x) z))))
     (lambda (x)
       (f (lambda (z) ((x x) z)))))
    z)))
==(28, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 2)
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (* 3 (((lambda () (f (lambda (z) ((x x) z))))) z)))
==(29, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 2)
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (* 3 ((f (lambda (z) ((x x) z))) z)))
==(30, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 2)
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (*
   3
   (((lambda (fact)
       (lambda (n)
         (if (<= n 0)
             1
             (* n (fact (- n 1))))))
     (lambda (z)
       ((x x) z)))
    z)))
==(31, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 2)
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z))))
  (*
   3
   (((lambda ()
       (lambda (n) (if (<= n 0) 1 (* n (fact (- n 1)))))) z)))
```

```
==(32, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 2)
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z))))
  (* 3 ((lambda (n) (if (<= n 0) 1 (* n (fact (- n 1))))) z)))
==(33, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z))))
  (* 3 ((lambda (n) (if (<= n 0) 1 (* n (fact (- n 1))))) 2)))
==(34, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 2))
  (* 3 ((lambda () (if (<= n 0) 1 (* n (fact (- n 1)))))))
==(35, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 2))
  (* 3 (if (<= n 0) 1 (* n (fact (- n 1))))))
==(36, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 2))
  (* 3 (if (<= 2 0) 1 (* n (fact (- n 1))))))
==(37, builtin)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 2))
  (* 3 (if () 1 (* n (fact (- n 1))))))
==(38, if)==>
(letrec ((f
```

```
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 2))
  (* 3 (* n (fact (- n 1)))))
==(39, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 2))
  (* 3 (* 2 (fact (- n 1)))))
==(40, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (n 2))
  (* 3 (* 2 ((lambda (z) ((x x) z)) (- n 1)))))
==(41, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (* 3 (* 2 ((lambda (z) ((x x) z)) (- 2 1)))))
==(42, builtin)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (* 3 (* 2 ((lambda (z) ((x x) z)) 1))))
==(43, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (z 1))
  (* 3 (* 2 ((lambda () ((x x) z))))))
==(44, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (z 1))
```

```
  (* 3 (* 2 ((x x) z))))
==(45, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (z 1))
  (* 3 (* 2 (((lambda (x) (f (lambda (z) ((x x) z)))) x) z))))
==(46, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 1))
  (*
   3
   (*
    2
    (((lambda (x)
        (f (lambda (z) ((x x) z))))
      (lambda (x)
        (f (lambda (z) ((x x) z)))))
     z))))
==(47, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 1)
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (* 3 (* 2 (((lambda () (f (lambda (z) ((x x) z)))) z))))
==(48, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 1)
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (* 3 (* 2 ((f (lambda (z) ((x x) z))) z))))
==(49, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 1)
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (*
   3
   (*
    2
    (((lambda (fact)
        (lambda (n)
          (if (<= n 0)
              1
              (* n (fact (- n 1))))))
      (lambda (z)
```

```
       ((x x) z)))
     z))))
==(50, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 1)
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z))))
  (*
   3
   (*
    2
    (((lambda () (lambda (n) (if (<= n 0) 1 (* n (fact (- n 1)))))))
     z))))
==(51, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 1)
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z))))
  (* 3 (* 2 ((lambda (n) (if (<= n 0) 1 (* n (fact (- n 1))))) z))))
==(52, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z))))
  (* 3 (* 2 ((lambda (n) (if (<= n 0) 1 (* n (fact (- n 1))))) 1))))
==(53, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 1))
  (* 3 (* 2 ((lambda () (if (<= n 0) 1 (* n (fact (- n 1))))))))
==(54, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 1))
  (* 3 (* 2 (if (<= n 0) 1 (* n (fact (- n 1)))))))
==(55, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
```

```
                (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 1))
  (* 3 (* 2 (if (<= 1 0) 1 (* n (fact (- n 1))))))
==(56, builtin)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 1))
  (* 3 (* 2 (if () 1 (* n (fact (- n 1))))))
==(57, if)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 1))
  (* 3 (* 2 (* n (fact (- n 1)))))
==(58, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 1))
  (* 3 (* 2 (* 1 (fact (- n 1)))))
==(59, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (n 1))
  (* 3 (* 2 (* 1 ((lambda (z) ((x x) z)) (- n 1)))))
==(60, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (* 3 (* 2 (* 1 ((lambda (z) ((x x) z)) (- 1 1)))))
==(61, builtin)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (* 3 (* 2 (* 1 ((lambda (z) ((x x) z)) 0)))))
==(62, lambda)==>
```

```
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (z 0))
  (* 3 (* 2 (* 1 ((lambda () ((x x) z)))))))
==(63, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (z 0))
  (* 3 (* 2 (* 1 ((x x) z)))))
==(64, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (z 0))
  (* 3 (* 2 (* 1 (((lambda (x) (f (lambda (z) ((x x) z)))) x) z)))))
==(65, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 0))
  (*
   3
   (*
    2
    (*
     1
     (((lambda (x)
         (f (lambda (z) ((x x) z))))
       (lambda (x)
         (f (lambda (z) ((x x) z)))))
      z)))))
==(66, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 0)
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (* 3 (* 2 (* 1 (((lambda () (f (lambda (z) ((x x) z))))) z)))))
==(67, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 0)
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
```

```
  (* 3 (* 2 (* 1 ((f (lambda (z) ((x x) z))) z)))))
==(68, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 0)
         (x (lambda (x) (f (lambda (z) ((x x) z))))))
  (*
   3
   (*
    2
    (*
     1
     (((lambda (fact)
         (lambda (n)
           (if (<= n 0)
               1
               (* n (fact (- n 1))))))
       (lambda (z)
         ((x x) z)))
      z)))))
==(69, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 0)
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z))))
  (*
   3
   (*
    2
    (*
     1
     (((lambda () (lambda (n) (if (<= n 0) 1 (* n (fact (- n 1)))))))
      z)))))
==(70, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (z 0)
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z))))
  (* 3
     (* 2 (* 1 ((lambda (n) (if (<= n 0) 1 (* n (fact (- n 1))))) z)))))
==(71, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z))))
  (* 3
     (* 2 (* 1 ((lambda (n) (if (<= n 0) 1 (* n (fact (- n 1))))) 0)))))
==(72, lambda)==>
(letrec ((f
```

```
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 0))
  (* 3 (* 2 (* 1 ((lambda () (if (<= n 0) 1 (* n (fact (- n 1))))))))))
==(73, lambda)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 0))
  (* 3 (* 2 (* 1 (if (<= n 0) 1 (* n (fact (- n 1))))))))
==(74, instantiate)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 0))
  (* 3 (* 2 (* 1 (if (<= 0 0) 1 (* n (fact (- n 1))))))))
==(75, builtin)==>
(letrec ((f
          (lambda (fact)
            (lambda (n)
              (if (<= n 0)
                  1
                  (* n (fact (- n 1)))))))
         (x (lambda (x) (f (lambda (z) ((x x) z)))))
         (fact (lambda (z) ((x x) z)))
         (n 0))
  (* 3 (* 2 (* 1 (if #t 1 (* n (fact (- n 1))))))))
==(76, if)==>
(letrec ()
  (* 3 (* 2 (* 1 1))))
==(77, builtin)==>
(letrec ()
  (* 3 (* 2 1)))
==(78, builtin)==>
(letrec ()
  (* 3 2))
==(79, builtin)==>
(letrec ()
  6)
Final value after 80 steps:
6

submodel-eval>>
```