

```

1 #lang racket
2
3 (define (fac n)
4   (if (= 1 n)
5       1
6       (* n (fac (- n 1)))))
7
8 ;; Tail-recursive version
9
10 (define (tail-fac n)
11   (letrec ((helper (lambda (n acc)
12                     (if (= 1 n)
13                         acc
14                         (helper (- n 1) (* n acc))))))
15     (helper n 1)))
16
17 (tail-fac 4)
18
19 ;; Version from last week
20
21 (define (string-reverse str)
22   (letrec ((helper
23             (lambda (x n)
24               (if (= n 0)
25                   (string (string-ref x n))
26                   (string-append (string (string-ref x n))
27                                   (helper x (- n 1)))))))
28     (helper str (- (string-length str) 1))))
29
30 (string-reverse "elephant")
31
32 ;; Tail-recursive version
33
34 (define (tail-string-reverse str)
35   (letrec ((helper
36             (lambda (x n res)
37               (if (= n 0)
38                   (string-append res (string (string-ref x n)))
39                   (helper x (- n 1) (string-append res (string (string-ref
39 x n))))))))))
40     (helper str (- (string-length str) 1) "")))
41
42 (tail-string-reverse "elephant")

```