Fundamentals of Computer Science I (CS151.01 2006F)

Homework 12: Tallying

Assigned: Friday, 3 November 2006
Due: Tuesday, 7 November 2006
No extensions!

Summary: In this assignment, you will write your own higher-order procedure.

Purposes: To help you think about higher-order procedures and how they might help with control.

Expected Time: One to two hours.

Collaboration: You may work in a group of any size between one and four, inclusive. You may consult others outside your group, provided you cite those others. You need only submit one assignment per group.

Submitting: Email me your work, using a subject of *CSC151 Homework 12*.

Warning: So that this exercise is a learning assignment for everyone, I may spend class time publicly critiquing your work.

Background

As you've seen in our discussions this past week, one of the key ideas in functional programming is that you can *factor out* common control structures. We've seen it possible to factor out the process of building a new list by recursing over the list with `map` and to factor out the process of checking all the values in a list with `list-of?`.

Here's another common task: Counting values that match some predicate. We've written procedures that count the number of symbols in a list and that count the number of odd numbers in a list of numbers.

```
(define tally-symbols
  (lambda (lst)
    (cond
      ((null? lst) 0)
      ((symbol? (car lst)) (+ 1 (tally-symbols (cdr lst))))
      (else (tally-symbols (cdr lst))))))
(define tally-odds
  (lambda (lst)
    (cond
      ((null? lst) 0)
      ((odd? (car lst)) (+ 1 (tally-odds (cdr lst))))
      (else (tally-odds (cdr lst))))))
```

Assignment

a. Write a procedure, `(tally pred? lst)`, that counts the number of values in `lst` for which `pred?` holds.

b. Rewrite `tally-symbols` and `tally-odds` using `tally`.

c. Write a procedure, `tally-as`, which takes a list of integers (representing grades) as a parameter and returns the number of values that are 90 and above. You should not verify the preconditions of the procedure (that is, do not check that it's a list of integers).