Collaboration policy in effect for this assignment: SEE last page before starting the assignment.

CSCI 320, Fall 2011, Homework 2

DUE: Tuesday, September 20, 3pm

1. Read sections 2.3 Lists, Iteration, and Recursion and 2.4 Pairs, Lists, and Racket Syntax, in *The Racket Guide* < http://docs.racket-lang.org/guide/>.
    - Complete the following programming exercises, from your text: Chapter 15, pages 698, 699, numbers: 5, 6, 7, 8, 9, 11, 12, 14, 16
    - Note: you may use existing basic scheme functions. If, however, there is already a scheme function that solves a given problem, say #5 for example, you can't use it. You must write it yourself. In solving the problems above, you may write other worker functions.
    - All code requiring repetition, should accomplish it through recursion.

2. Functional programming characteristics include:
    - *first class functions* – this means that the language supports passing functions as arguments to other functions, returning them as the values from other functions, and assigning them to variables or storing them in data structures. (Extracted from Wikipedia, "first class functions" on September 13, 2011).
    - *higher-order functions* – the term used by Sebesta means the same thing as "first class functions". Sebesta also refers to these as "functional forms." (See page 659.)

Higher-order functions or functional forms include:

Composition: a function that takes two functional parameters and yields a function whose value is the first actual parameter function applied to the result of the second. Composition is denoted by:

$h \equiv f \circ g$

For example, if $\begin{array}{l} f(x) \equiv x+2 \\ g(x) \equiv 3 * x \end{array}$ then $h(x) \equiv f(g(x))$, or $h(x) \equiv (3 * x) + 2$

Construction: a function that takes a list of functions as parameters. When applied to an argument, a construction applies each of its functional parameters to that argument and collects the results in a list or sequence.

Let $\begin{array}{l} g(x) \equiv x * x \\ h(x) \equiv 2 * x \\ i(x) \equiv x/2 \end{array}$ then $[g,h,i](4)$ yields $(16,8,2)$

Apply-to-all: a function that takes a single function as a parameter. If applied to a list of arguments, apply-to-all applies its functional parameter to each of the values in the list argument and collects the results in a list or sequence.

Let $h(x) \equiv x * x$ then $\alpha(h,(2,3,4))$ yields $(4,9,16)$

Assignment: Write functional, (referentially transparent, no side-effects and hence loop-less) Scheme code to implement the function forms composition, construction and apply-to-all:

- `composition(f, g)` – named composition, its parameters, f and g, are any two functions – will be called with different functions different times

Collaboration policy in effect for this assignment: SEE last page before starting the assignment.

- `construction(lstOfFuncts, a)` – takes as arguments, an arbitrarily long list of functions and a single argument
- `applyToAll(f, lstOfArgs)` – takes as arguments a function, f, and an arbitrarily long list of arguments

WARNING: Solutions (and the next assignment) will be posted immediately after the due date/time, so late assignments will not be accepted.

EARLY submissions are always accepted.

What to submit: Two files one called

- Part 1, which contains the solutions to problem 1 above, with calls to the functions embedded.
- Part 2, which contains the solutions to problem 2 above, with calls to the functions embedded.

Collaboration policy in effect for this assignment: SEE last page before starting the assignment.

- You may only submit solutions generated by your own work.
- You may discuss DrRacket with classmates and instructors.
- You may discuss Racket with classmates and instructors.
- You may discuss these problems with classmates and instructors. If it helps you to learn from others how to solve the problem, that is fine. The objective is that you CAN and DO solve the problem. You MUST acknowledge these discussions.
- You MAY NOT copy the work of others.
- You MAY NOT give your work to others.
- **You should not look for solutions on the internet, you should craft your own. I can't give you a grade for someone else's work!!!! If you do run into a solution that you use or that influences your solution, you must acknowledge that.**
- Your name must be in every file you submit.