CS 2500, Spring 2013
Problem Set 4

Due date: Tuesday, February 5 @ 11:59pm

What to submit:
Using Blackboard, submit a single Racket file containing all of the code and documentation for this assignment. Place your name and husky email address in a comment at the beginning of your file.

Name your file: hw4-yourlastname.rkt

You **must** follow the design recipe. The graders will look for data definitions, contracts, purpose statements, examples/tests, and properly organized function definitions. For the latter, you **must** design templates, but make sure to comment them out.

Problem 1:

Evaluate the following expressions step by step and write down next to each step whether it is (1) "*arithmetic*" (of any form—not just on numeric data), (2) *function application* ("plugging in") or (3) a *conditional* step:

```
a)  (define (volume.v1 l w h)
            (* l w h))

     (volume.v1 10 5 20)

b)  (define-struct rprism (length width height))

     (define (volume.v2 rp)
          (* (rprism-length rp)
             (rprism-width rp)
             (rprism-height rp)))

     (volume.v2 (make-rprism 10 5 20))

c)  (define (step x)
          (cond [(< 1 x) (/ x 2)]
                [(< 0 x) (* 2 x)]
                [else    (sqr (+ x 1))]))

     (step 0)
```

You may want to write down each step inside of DrRacket. Since computation means calculating, you know that every step must produce the same answer. So if you're ever unsure whether your calculation is still on track, just run the whole program and watch the same answer pop up for as many steps as you have written down.

Problem 2:

A Book is one of:

- **hardcover**, which has the properties: title, author, number of pages, genre, and a Boolean determining whether this particular book is on the bestseller list;
- **ebook**, which has the properties: title, author, size in kilobytes, and a symbol designating the ebook reader as 'Kindle, 'Nook, or 'iPad;
- **audiobook**, which has the properties: title, author, length in minutes, and the name of the person whose voice is recorded.

a) Develop data and structure definitions for a Book
b) Develop the function `author-of-book` that, given a Book, produces the author of the book.
c) Develop the function `length-of-book` that, given a Book, produces length of the book. For a hardcover book it should produce the number of pages. For an audiobook it should produce the length in minutes. For an ebook, it should produce the size in kilobytes.

Problem 3:

Below is a data definition for a class of shapes (See the code at the bottom).

a) Add interpretations for the Square and Rectangle classes.
b) Design `shape-shift-x`. The function consumes a Shape, *sh*, and a number, *delta*. It produces a shape that is like *sh* but shifted by *delta* pixels along the x-axis.
c) Design `perimeter`. The function consumes a Shape, *sh*, and computes its perimeter.
d) Design `shape-draw`. The function consumes a Shape, *sh* and a Scene, *sc* and adds *sh* to *sc*.

```
;; Shape is one of:
;; -- Circle
;; -- Square
;; -- Rectangle

(define-struct circl (x y r outline c))
;; A Circle is a
;;  (make-circl Number Number Number Boolean Symbol)
;; interpretation: x and y determine the center of the circle,
;;   r the radius, outline whether it's outlined or solid,
;;   and c its color

(define-struct squar (x y size outline c))
;; A Square is a
;;  (make-squar Number Number Number Boolean Symbol)
;; interpretation: Supply a good interpretation of Square.

(define-struct recta (x y width height outline c))
;; A Rectangle is a
;;  (make-recta Number Number Number Number Boolean Symbol)
;; interpretation: Supply a good interpretation of Rectangle.
```

Problem 4:

a) Write a data definition for a list of images

b) Design a function called `height-of-all` which satisfies the following contract and purpose:

```
;; height-of-all: list-of-image -> number
;; consumes a list of images and returns
;; the sum of all of the image heights
```

Hint: you may use DrRacket's function `image-height` that returns the height of an image. The contract for `image-height` is

```
;; image-height: image -> number
```

c) Design a function called `overlay-all` which takes a list of images and overlays all of its members building a single image. Make sure to write the contract and purpose statement.

d) Design a function called `overlay/xy-all` which consumes a list of images and two numbers, x and y. The function overlays all of the images in the list producing a single image. Each image is moved by x pixels to the right and y down before overlaying them. Make sure to write the contract and purpose statement.