Lab 07: Lists of Structures

Create a separate file for each question. Keep them in your "Labs" folder, with the name `liiqj` for Lab ii, Question j.

Download the headers for each function from the file `labinterface07.rkt` linked off the "Labs" page on the course Web site.

After you have completed a question (except class exercises), including creating tests for it, you can obtain feedback by submitting it and requesting a public test. Follow the instructions given in the Style Guide.

This lab makes use of the following structure and data definitions:

```
(define-struct event (type dur))
;; An event is a structure (make-event t d), where
;; * t is a symbol (the type of event) and
;; * d is a positive integer (the duration in minutes).

(define-struct card (value suit))
;; A card is a structure (make-card v s), where
;; * v is an integer in the range from 1 to 10 and
;; * s is a symbols from the set 'hearts, 'diamonds,
;;    'spades, and 'clubs.

(define-struct clock (hours mins))
;; A clock is a structure (make-clock h m), where
;; * h is an integer in the range from 0 to 23 and
;; * m is an integer in the range from 0 to 59.
```

After you have completed a question (except class exercises), including creating tests for it, you can obtain feedback by submitting it and requesting a public test. Follow the instructions given in the Style Guide.

Language level: Beginning Student.

1. *[Class exercise with lab instructor assistance]* Create a function *total-dur* that consumes a list of *event*s and produces the total duration in minutes.

2. Create a function *max-card* that consumes a nonempty list of *card*s and produces a *card* with the maximum value of any *card* in the list (if there is more than one, the one appearing closest to the end of the list).

3. Create a function *values* that consumes a symbol (a suit) and a list of distinct *card*s and produces a list of numbers, that is values of *card*s from the original list that have the specified suit.

4. Create a function *update-times* that consumes a list of *clock*s and a natural number *m* (a number of minutes) and produces a list of updated *clock*s, each *m* minutes later on the same day (you can assume none of the new times go past midnight to the next day).

5. Create a function *average-length* that consumes a list of *events* and a symbol (the type of the event) and produces the average duration of *event*s of that type. If a list is empty or there are no events of a particular type, the average will be zero. Be careful not to divide by zero!

6. *Optional open-ended questions* You now have enough tools to be able to use the teachpack `world.ss` to create animations. Create a structure for a shape (type, size1, size2, mode, colour) and a structure for a piece (shape, posn). You will make an animation in which a world is a list of pieces, using *place-image* to place the pieces, in order from last in the list to first, on a scene (initially empty, 400×400). Be sure to read the world.ss documentation available on the course Webpage, and read the section below "Loading the world.ss teachpack". Here are some ideas to get you started.

(a) Create a function *piece-list-to-scene* to consume a list of pieces and produce a scene.
(b) Create a function *advance-rainbow* to consume a list of pieces and produce a list of pieces in the next colour of the rainbow. (c) Use your functions to create animations.

Helpful tips

Loading the world.ss teachpack A teachpack is a file with Scheme definitions in it. Loading a teachpack means you can use any of the defnitions as if they were built-in definitions. Under “Language” on the main menu bar, select “Add Teachpack...” The left column in thewindow that pops up is labeled “Preinstalled HtDP Teachpacks”. Choose “world.ss” and click on “OK”. You will now be able to create images as described in our documentation.

Clearing all teachpacks Under “Language” on the menu bar, select “Clear All Teachpacks”.