CS5001 Assignment 7

Programming Language: ISL with Lambda

Due Dates: Wednesday 3/14 @ 10:00pm

Let's build some lego buildings out of lego bricks. Here are data definitions for lego bricks and lego buildings:

```
(define-struct lego (label color width))
;; A Lego is a structure:
;;    (make-lego Number Symbol Number)
;; interpretation: (make-lego l c w) is the lego brick
;; with label l, color c, and width w (in pixels).

(define-struct bigger (lego left right))
;; A LegoBldg (lego building) is one of:
;; - Lego
;; - (make-bigger Lego LegoBldg LegoBldg)
;; interpretation: (make-bigger lg lft rgt) makes a bigger
;; lego building by putting a lego brick lg on top of two lego
;; buildings lft (left) and rgt (right).
```

Problem 1 Design a function, `count-bricks`, that takes a lego building and produces the total number of lego bricks in that building.

Problem 2 Each lego brick is 10 pixels tall. Design a function, `how-high`, that takes a lego building and produces the total height of the lego building (in pixels).

Problem 3 Design a function, `contains-colored-brick?`, that takes a lego building and a color, and determines whether the building contains a lego brick of the given color.

Problem 4 Design a function, `find-colored-brick?`, that takes a lego building and a color and finds any lego with the given color in the building, or returns false if there are no such legos.

Here is the data definition for the type of data this function returns:

```
;; A MaybeLego is one of:
;; - false
;; - Lego
```

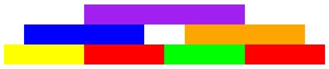
Your function should not use `contains-colored-brick?`, it should not traverse/examine parts of the building more than once, and it should stop searching once any brick of the given color is found.

Problem 5 Design a function, `lb->image`, that takes a lego building and produces an image of the building.

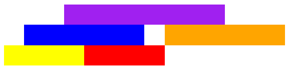
Hints: You may want to look up `above` and `beside/align` in Help Desk. Also, you may want to design a helper function, `lego->image`, that takes a lego and produces an image of the lego. All legos are rectangular and 10 pixels tall.

Here are some examples:

```
(make-bigger (make-lego 4 'purple 80)
             (make-bigger (make-lego 2 'blue 60)
                          (make-lego 1 'yellow 40)
                          (make-lego 3 'red 40))
             (make-bigger (make-lego 6 'orange 60)
                          (make-lego 5 'green 40)
                          (make-lego 7 'red 40)))
```



```
(make-bigger (make-lego 4 'purple 80)
             (make-bigger (make-lego 2 'blue 60)
                          (make-lego 1 'yellow 40)
                          (make-lego 3 'red 40))
             (make-lego 6 'orange 60))
```



Problem 6 Given the data definition for a binary tree below:

```
;; A BT is one of:
;; - 'leaf
;; - (make-node Number BT BT)
(define-struct node (data left right))
```

design a program to determine whether a `BT` is foldable.

A `BT` is foldable if the *structures* of the left and right subtrees of the tree are mirror images of each other. A leaf is considered foldable. For example, `btA` and `btB` are foldable and `btC` and `btD` are not. (Note: leaves are not shown in the drawings below.)

```
btA:        btB:         btC:         btD:
     3            3            2             3
   /   \        /   \        /   \         /   \
  2     4      2     4      1     4       1     4
 /       \      \   /            /         \     \
1         5      1 3            5           2     5
```