CMSC 15100 Introduction to Computer Science I
Homework 6

Important: Please add to (require "../include/cs151-core.rkt") at top of your file. Your answers should be saved in a single .rkt file and submitted via Canvas before the start of the next lecture. At the top of the file for this assignment and all future assignments include a comment,

```
;; Homework 6
;;
;; your name
;; your CNET id
```

If your code doesn't compile, it may not be graded. Please double check to make sure your code runs before submitting!

Problems

1. (7 points) In Racket we can represent a currently-running TV by a function which takes in a channel number and outputs the name of the show on that channel,

   ```
   (define-type TV (Integer -> String))
   ```

   (a) Create a variable `friday-night` of type `TV` that lists "NBC Evening News" airing on Channel 5, "Game of Thrones" airing on Channel 7, "ESPN SportsCenter" running on Channel 33, and raises an error on any other channel.

   (b) Make a function `change-program` which takes three inputs and produces a TV,

   ```
   (: change-program : TV Integer String -> TV)
   (define (change-program tv chan program) ...)
   ```

   and returns "tv" with channel "chan" updated to "program".

2. (6 points) One function that is cool but not that useful is the `duplicate-string` function, which takes in two arguments, a `String` and an `Integer` n, and returns the string duplicated and appended to itself n times. For example,

   ```
   (duplicate-string "ha" 3) ==> "hahaha"
   ```

   Implement the `duplicate-string` function without recursion (partial credit will be given for a recursive implementation).

3. (12 points) Back on Homework 2 we wrote a function `my-expt` that could get the value of x^2, x^3, x^4, and so forth, but how can we compute the square root $x^{1/2}$? For this part of the problem, in order to not worry about inexact representation of real numbers, we'll write a function `my-sqrt` with type `Integer -> Integer` so that `(my-sqrt x)` outputs the largest integer less than or equal to $\sqrt{x}$. To see a couple of test cases, we should have

```
(my-sqrt 0) ==> 0
(my-sqrt 1) ==> 1
(my-sqrt 2) ==> 1
(my-sqrt 3) ==> 1
(my-sqrt 4) ==> 2
```

Note that you shouldn't use the `expt` function for this problem.

(a) A useful helper function for this problem is the function `list-0-to-n` that takes a single input n and computes a `(Listof Integer)` containing the numbers 0 through n, inclusive. Use the built-in function `build-list` to write this function.

(b) Implement the function `my-sqrt` using your helper function from part (a). If the input is negative, your code should raise an error.

Hint: the easiest way to implement this function is without recursion. Use a filter, then a fold.

(c) In addition to square roots, it's also a challenge to compute cube roots, four roots, fifth roots, etc. Modify your function from part (b) so that it takes a second input, k, and computes a k-th root instead of a square root. Raise an error if k is negative.

Hint: copy in your `my-expt` function from HW2

(d) (2 points extra credit) Improve the approximation of your square root function. Specifically, change the type of the function to be `Integer Real -> Exact-Rational`. On input x and t, the output should be an `Exact-Rational` that's within t of the actual real number $\sqrt{x}$.

By calling your function with very tiny numbers for t, for example 0.0001, it should be possible to get a very good `Exact-Rational` approximation for $\sqrt{x}$.

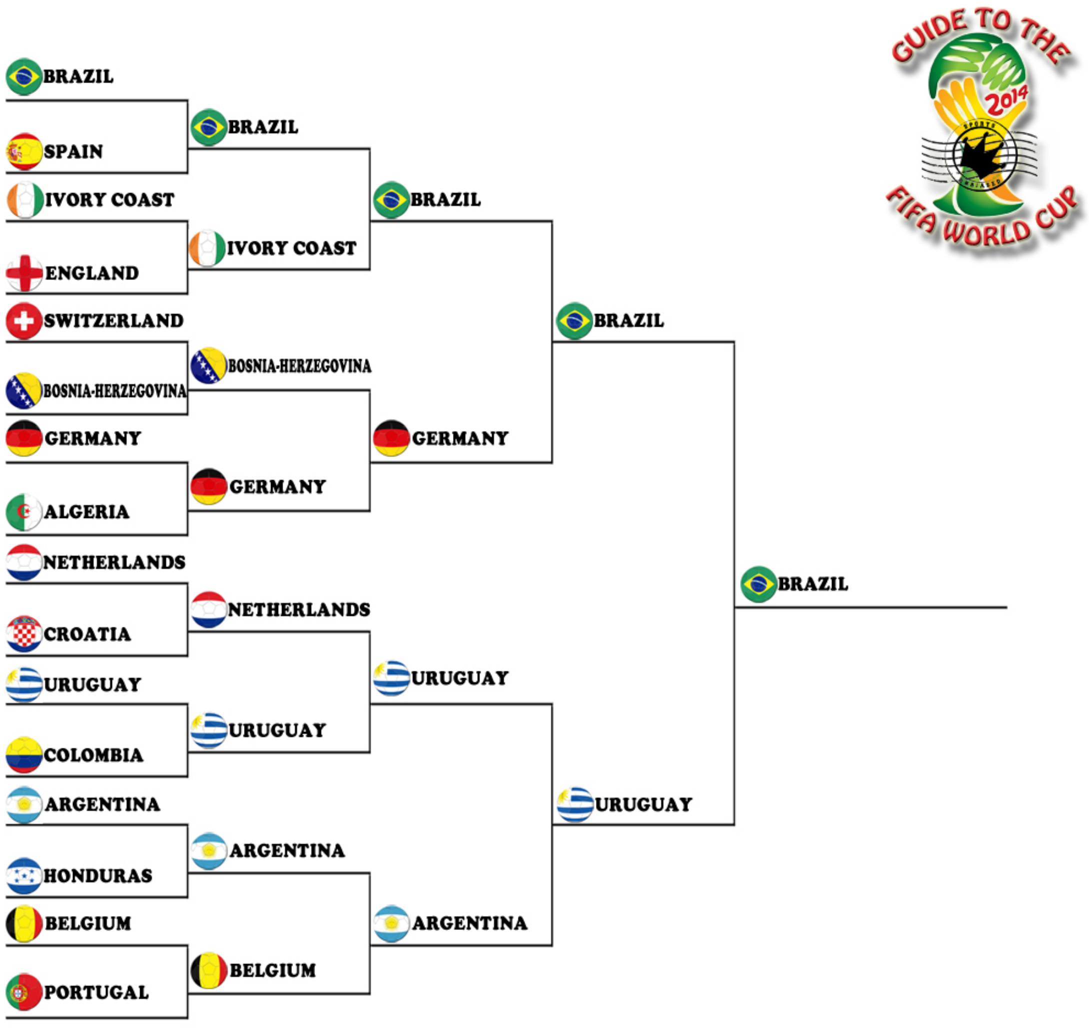
4. (11 points)

In world cup soccer teams face off in a bracket, with the winning teams advancing and the losing teams getting knocked out. For example, a bracket with someone's prediction for the winners of a past World Cup is shown on the next page.

In Racket, we can represent a bracket by a `(Tree String)`. There are two conditions for a `(Tree String)` to be a bracket:

- Each node of the tree must have either 0 or 2 children
- If a node has 2 children, one of its children must have the same `String` as it. For example, in the picture notice how the nodes labeled "Brazil" also have children labeled "Brazil" (except for the leaf at the top left). This corresponds to the winner advancing up the bracket.

Write a function `valid-bracket?` which takes in a `(Tree String)` and returns true if it represents a valid bracket.


GUIDE TO THE 2014 FIFA WORLD CUP
BRAZIL
SPAIN
IVORY COAST
ENGLAND
SWITZERLAND
BOSNIA-HERZEGOVINA
GERMANY
ALGERIA
NETHERLANDS
CROATIA
URUGUAY
COLOMBIA
ARGENTINA
HONDURAS
BELGIUM
PORTUGAL
BRAZIL
IVORY COAST
BOSNIA-HERZEGOVINA
GERMANY
NETHERLANDS
URUGUAY
ARGENTINA
BELGIUM
BRAZIL
GERMANY
URUGUAY
ARGENTINA
BRAZIL
URUGUAY
BRAZIL