MA 151: Homework #7

due Tuesday November 8

Written problems

In each of these, simplify the expressions step-by-step to get the final value. If there is an error, say exactly what the problem is. If the function gives an infinite loop, explain in general terms what the output will be. You should show enough detail to make it clear that you know what is going on. In all cases, you should be able to check your answer by typing the expressions into GHCi.

1. `foldl (++) "" ["I", "love", "pizza"]`
2. `foldr ($) 3 [succ, (^2), pred]`
3. `foldr (\x y -> 2*x+3*y) 1 [1,2,3,4]`
4. `foldl (\x y -> 2*x+3*y) 1 [1,2,3,4]`
5. `foldr map [1,2,3] [(*2),pred, (^2), succ]`

Programming problems

- Write a function called `summa` using a fold which takes a list of `Int` and adds together the evens and subtracts the odds. For example `summa [1,4,5,6,2]` is $-1 + 4 - 5 + 6 + 2 = 6$. I suggest something ilke:

  ```
  summa xs = foldr f ??? ???
     where f a b = ???
  ```

  (For the definition of f, it should do $a + b$ when a is even, and something else when a is odd.)

- Use functions from the Data.String module to write a function called `wordCount` that takes a big string with spaces and counts the number of words in it. Use Hoogle to find the functions you need. Make your definition points-free.

- Use the same functions from the Data.String module to rewrite `exclaim` from Homework #5 without using recursion. (I suggest you use `map` and `concat`. It's OK if you end up with an extra space on the end of the answer.)

- Write a function called `ratMult` which takes two parameters of type `Rat` (use the type declaration from class on 11/1) and returns their product as a `Rat`.

- Write a function called `ratLegal` which takes a `Rat` and gives a `Bool` which says whether or not the `Rat` represents an actual fraction. (You need to check if the denominator is nonzero.)

- Write a function called `ratReduce` which takes one parameter of type `Rat` and reduces the fraction. The result should be a `Rat` where the numerator and denominator have no common factors. (Use `gcd`, which is a prelude function.)

- Write a function called `ratLeq` which takes two parameters of type `Rat` and returns a Boolean telling whether or not the first parameter is less than or equal to the second. Make sure you handle negative numbers correctly.

For the following functions, use the following data declaration for quadratic polynomials with decimal coefficients:

```
data Quad = Coeffs Float Float Float deriving Show
```

The idea for this type is that `Coeffs 3 2 (-7)` represents the polynomial $3x^2 + 2x - 7$.

- Write a function called `quadEval` which evaluates a polynomial at a particular x-value. For example: `quadEval (Coeffs 3 2 (-7)) 2` is $3 \cdot 2^2 + 2 \cdot 2 - 7 = 9$. (You'll see `9.0` because it's a `Float`.)
- Write a function called `quadAdd` which adds together two `Quad`s.
- Write a function called `quadDerivative` which takes a `Quad` and gives the derivative as another `Quad`. (Tell me if you don't know what the derivative is- it's complicated in general but very easy for a quadratic.)
- Write a function called `quadRoots` which takes a `Quad` and gives a list of `Float` containing all the polynomial's real roots. (These are the values of x which make $p(x) = 0$.)