MA 151: Homework #6

due Tuesday October 31

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.

These ones use `even`, which is a prelude function which is True when the number is even, and False when it's odd.

1. `map (+5) (filter even [1..4])`
2. `map ($2) (map (.succ) [succ, pred, (*(-1)), (*2)])`
3. `map (take 4) ["haskell","curry","is","my","father"]`
4. `filter (==True) (map (even . length) ["haskell","curry","is","my","father"])`
5. `map (\x -> x*4 + 2) [1..5]`

Programming problems

For these functions, use higher order functions. **Don't use recursion or list comprehensions**. Always include a type signature.

- Here is a function defined by a list comprehension- rewrite it using `map` and `filter`.

  `compy y = [x^2 - 4*x | x <- [1..y], mod x 4 == 1]`

- Using functions from class on 10/13, make a function called `collatzN` which takes a number n and returns the smallest number whose Collatz sequence has length exactly n. If you want a bigger challenge, do this without a `where` or lambda.
- Rewrite `deVowel` from class on 10/17 without using recursion or list comprehensions.
- Write a polymorphic function called `excluder` which takes two lists, and returns the second list after deleting any elements which are also in the first list. So:

  `excluder [1,2,3] [5,2,6,7,3,1,0]` is `[5,6,7,0]`

- Write a polymorphic function called `intersection` which takes two lists and returns the list of all elements which appeared in both lists.
- Write a function called `maxxer` that takes a list of lists of numbers, and returns the biggest number appearing inside. Make your definition points free. You can assume none of the lists are empty. (Don't use `concat` for this one.) For example:

  `maxxer [[4,2,5],[0,1,2,3],[4,7,3],[0,3]]` is 7

- Write a points-free definition of a function called `startsWithOne` which takes a list of integers and gives True if the first element is 1, and False otherwise. (Use `head`- it's OK if the empty list gives an error.)
- Rewrite `slowTalk` from class on 10/13 without using recursion or list comprehension. (Hint: use `map` and the built-in function `concat`, which takes a list of lists and "flattens" it.)