COMP 3649 Exercise Set 4

Given: Thursday, February 13
Complete by: Monday, February 24
Quiz: Tuesday, February 25 (in-class at 8:30 a.m.)

For questions 1-3, attempt to develop written answers by yourself. Once you have done your best, you may discuss your answers with two-to-three other students. Then, by yourself (without looking at notes from the discussion), revise your own written answers as you believe appropriate.

1. Using “dot dot” notation, write Haskell expressions for generating each of the following:

    a) A list of uppercase characters from ‘H’ to ‘W’
    b) A list of uppercase characters from ‘W’ down to ‘H’
    c) An infinite list of integers incrementing by 1 and starting at 0
    d) As for (c) but multiples of 3 only

2. Explain why “dot dot” notation cannot be used to produce the infinite list of non-negative integer powers of two:

    1, 2, 4, 8, 16, 32, 64, 128, 256, …

    Then, using recursion, define a function named `pows` that produces this list (do *not* use a list comprehension)

3. Using the lazy evaluation strategy, perform a reduction of the following. Follow the format used in class, showing one reduction step per line and the justification for each.

    ```
    pows !! 3
    ```

    Hint: the lazy evaluation strategy applies the “leftmost outermost” reduction at each step.

    Assume the following:

    ```
    (x:xs) !! 0 = x                   -- (!!).1
    (x:xs) !! n = xs !! (n-1)         -- (!!).2[1]
    ```

[1] Case 2 of this definition is not entirely correct because it leads to infinite recursion when n < 0. The correct definition is as shown below, but, for this question, the guard has been omitted to simplify the reduction.

```
(x:xs) !! n | n > 0 = xs !! (n-1)
```

Develop complete answers to questions 4-6 working in a group of two-to-four students (but ensure you can recreate a correct answer on your own).

4. The following list comprehension is an attempt to produce an infinite list of non-negative integer pairs, such that any given pair is eventually generated (in finite time):

   ```
   [ (x,y) | x <- [0..], y <- [0..] ]
   ```

   However, it does *not* actually work as hoped! Explain why.

5. Write a list comprehension that correctly accomplishes the previous goal. Hint: start by producing all pairs whose elements sum to 0. Then, move on to all pairs whose elements sum to 1. Then, move on to all pairs whose elements sum to 2. Etc.

6. Modify the previous list comprehension to exclude pairs where the elements are equal.

For questions 7-8, attempt to develop written answers by yourself. Once you have done your best, you may discuss your answers with two-to-three other students. Then, by yourself (without looking at notes from the discussion), revise your own written answers as you believe appropriate.

7. Are the following two λ-terms equivalent up to α-conversion? If yes, write a sequence of α-conversions (one per line) that show that one can be derived from the other. If no, explain why.

   λ y . (λ x . y z (λ y . x y))   vs.   λ x . (λ y . x z (λ x . y x))

8. β-reduce the following λ-term to normal form:

   (λ a . (λ b . a b)) (λ c d . b c) d