CS 403/503 Exam 2 **Fall 2015** **Name _____________________**

Your score will be based on your best 8 out of these 10 problems.
Extra credit will be awarded if you can solve additional problems.

Write each function below using Haskell, and show its inferred type unless it's provided.
Also, write any non-standard helper functions that you use.

1. Write a Haskell function count x ys that returns the number of occurrences of value x within list ys. Example: count 5 [2,4,5,7,3,5,8,5,0] returns 3.

   count ::

2. Write a Haskell function partition p xs that returns a pair of lists: the first list contains the values in xs that satisfy predicate p, and the second list contains the values in xs that do not satisfy predicate p. Example: partition (>4) [1,3,5,7,9,0,2,4,6,8] returns ([5,7,9,6,8],[1,3,0,2,4]).

   partition ::

3. First use lazy evaluation to define the short-circuited operators && and ||. Next define new operators &&& and ||| that are logically the same but they are not short-circuited. Each of these four operators &&, ||, &&&, ||| has type Bool -> Bool -> Bool. For this problem, you must use pattern matching, and do not use the if-then-else construct.

4. Suppose Haskell did not define any built-in integer type. The data type Natural defined below denotes the natural numbers {0,1,2,3,4,5,…}. For example, the number 3 could be represented as Successor (Successor (Successor Zero))). Define functions add, sub, and mult which perform addition, subtraction, and multiplication of natural numbers. Each of these functions add, sub, mult has type Natural -> Natural -> Natural.

   data Natural = Zero | Successor Natural

5. Write a Haskell function isPerfect n that returns True iff n is a positive integer that equals the sum of all its smaller divisors. Examples: isPerfect 28 returns True, because 1+2+4+7+14=28. isPerfect 32 returns False, because 1+2+4+8+16=31.

   isPerfect ::

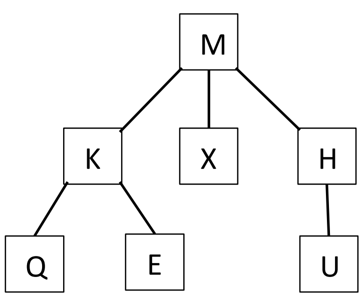
6. Write a Haskell function rotations xs that returns a list of all rotations of xs. Example: rotations "abcd" returns ["abcd", "bcda", "cdab", "dabc"].

   rotations ::

7. The polymorphic data type Tree defined below denotes general (non-binary) trees. Write Haskell functions preorder t and postorder t such that each function has type Tree a -> [a] and returns the appropriate list of node labels from tree t. Example: when t is the tree shown below, then preorder t returns "MKQEXHU" and postorder t returns "QEKXUHM".

   data Tree a = Node a [Tree a]

   t = Node 'M' [Node 'K' [Node 'Q' [ ], Node 'E' [ ]], Node 'X' [ ], Node 'H' [Node 'U' [ ]]]



8. Write a Haskell function transpose m that takes any matrix m, which is stored as a list of row lists, and returns the transpose of matrix m by interchanging its rows and columns. Example: transpose [[4,5,6], [7,8,9]] returns [[4,7], [5,8], [6,9]].

   transpose ::

9. Write a Haskell function intersect xs ys that returns the intersection of two infinite sets represented by ascending infinite lists xs and ys. Example: if xs = [1, 3, 6, 7, 9, 11, 14, 15, …] and ys = [2, 3, 5, 7, 10, 11, 13, 15, …] then intersect xs ys returns [3, 7, 11, 15, …].

   intersect ::

10. Write a Haskell function combine f g h xs ys that returns a list obtained by applying g to each element of xs and applying h to each element of ys, and then using f to unite the corresponding results. Assume that xs and ys have the same length. Example: combine (+) (\a -> a*a) head [2,3,5] [[1,2,3],[4,5,6],[7,8]] returns [2^2+1, 3^2+4, 5^2+7] = [5,13,32].

    combine ::