CS 403/503 Exam 2 Fall 2009 Name _____

CS 403 students: Your grade is based on your best 8 answers to these 12 problems.
CS 503 students: Your grade is based on your best 10 answers to these 12 problems.

Problems 1 to 6: Write each Haskell function so that it calls map or foldr or foldl. You may also call other predefined functions, but do not write any named helper functions and do not use explicit recursion. Anonymous helper functions are ok.

1. display n constructs the list [[1],[1,2],[1,2,3],…,[1,2,3,…,n]]. Example: display 5 returns [[1],[1,2],[1,2,3],[1,2,3,4],[1,2,3,4,5]].

2. applyEvery fs xs applies every function in fs to every value in xs, and arranges the results into a list with sublists as follows. Example: applyEvery [(+1),(*2),(^2)] [3,4,5,6] returns [[4,5,6,7],[6,8,10,12],[9,16,25,36]].

3. concatReverse xs does the same thing as concat (reverse xs). However, do not call either concat or reverse. Example: concatReverse ["abc","defg","hij"] returns "hijdefgabc".

4. selectInRange low high xs returns a list of the values in xs that are between low and high, inclusively. Example: selectInRange 5 9 [1,9,3,7,11,5,13] returns [9,7,5].

5. sumOfProducts xss returns the sum of the products of the sublists of xss. However, do not call either sum or product. Example: sumOfProducts [[1,2,3,4],[5,6],[7,8],[9]] returns 1*2*3*4+5*6+7*8+9 = 119.

6. dotProduct f g id xs ys returns the dot product of lists xs and ys with respect to binary functions f and g. Assume that xs and ys have the same length. If xs and ys are empty, return the value id. Example: dotProduct (+) (*) 0 [2,3,4] [5,6,7] returns 2*5+3*6+4*7 = 56.

Problems 7 to 9: Construct these infinite lists. Use any features of Haskell.

7. facts = [1,1,2,6,24,120,720,5040,40320,362880,...] is the list of factorials. 0!=1, 1!=1, 2!=2, 3!=6, 4!=24, etc.

8. fibs = [0,1,1,2,3,5,8,13,21,34,55,...] is the list of Fibonacci numbers. Each value is the sum of the preceding two values, so for example, the next value not shown is 34+55 = 89.

9. table f xs ys applies binary function f to each value in xs paired with each value in ys, and arranges the results into a list with sublists as follows. Assume that xs and ys are infinite lists. Example: table (*) [1..] [1..] returns the infinite multiplication table [[1,2,3,4,...],[2,4,6,8,...],[3,6,9,12,...],[4,8,12,16,...],...].

Problems 10 to 12: Write each data type and function. Use any features of Haskell.

10. ExtendedFloat can hold any built-in Float value, or positive infinity or negative infinity. All comparison operators (==, /=, <, <=, >, >=) should work properly. Function negate :: ExtendedFloat -> ExtendedFloat extends the concept of the unary minus operator to ExtendedFloat values.

11. BinaryTree is a tree such that each node holds a value of arbitrary type and has at most two children (left and right). Function inorder :: BinaryTree a -> [a] returns the values of the tree as found via an inorder traversal. So for example, if the tree represents a binary search tree, inorder returns the values in ascending order.

12. MixedList is a mixed-type list such that each cell holds either a Float value or a Char value. Function combine :: MixedList -> (Float, String) returns a 2-tuple in which the first value is the sum of all the Float values in the list, and the second value is the string that consists of all the Char values in the list.