**CS 4700 Haskell Programming Assignment 5**

**40 Points**

1. (4 points) Write a Haskell function to determine the maximum subsequence of a list of integers. The maximum subsequence is the contiguous subsequence with the largest sum. You do not need to return the subsequence itself--only the sum of its elements. For convenience, assume that if all integers in the list are negative the sum is 0. Make your function as efficient as possible.
2. (4 points) Define a function to replace one substring with another

subst:: [Char] ->[Char]->[Char]->[Char]

The result of

subst "Fish & Chips and Vinegar" "Chip" "Boat" yields "Fish &Boats and Vinegar"

If there is no occurrence of the replacement string, the original string should remain unchanged.

What happens if you enter

subst "Fish & Chips and Vinegar" "" "Boat"

1. Define a Haskell type for a binary search tree,  given the following Haskell type for Integer binary trees

data Tree = Empty | Node Int Tree Tree

deriving (Show)

Define the following functions on binary search trees.

a)  (3 points)    Insert an item into a binary search tree, discarding duplicates. Call the function insertNode.

b) (3 points) Create a tree from a list of items using foldr. Call the function makeTree.

c) (6 points) Write a function createBal in which a completely balanced tree is generated from an ordered list of values. A tree is considered completely balanced if the number of nodes in the left subtree and the number of nodes in the right subtree differ by at most 1.

e)    (4 points) Collect the nodes at a given level in the tree in a list.

f) (6 points) Remove an element from the tree. Call the function deleteNode.

Hint: I found it helpful to have functions which returned a constant tree.

myTree = makeTree [4,5,24,1,52,3,26]

mytree1 = makeTree [11,15,24,1,14,3,5,9,8,41]

1. (5 points) Write a function **perms** to generate a list of all possible permutations of a list. perms [1,2,3] yields something like [[1,2,3], [1,3,2],[2,1,3][2,3,1], [3,1,2],[3,2,1]]
2. (5 points) Let us call a binary tree symmetric if you can draw a vertical line through the root node and then the right subtree is the mirror image of the left subtree (disregarding the actual values in the tree).

Generate all possible permutations of the list [1,2,3,4,5,6,7]. For each permutation, create a binary search tree by inserting the elements of the list in order. Create a list of all such trees which are symmetric.