CPSC 221: Summer 2015
Theory Assignment 2

History of Non-Trivial Changes:
- June 8, 2015 – Posted

Due Date: No later than Saturday, June 13 at 11:59pm. Submit your digital solutions online using handin (the name for this assignment is 'ta2'). Note, you *may* work with a partner on this assignment. In order to do so, you must turn in **one assignment only**, with *both partners'* identifying information clearly stated (as per question 0). You ***may not*** work in groups of three or more.

0. Make sure to clearly identify your name, student number, 4-or-5-character ugrad ID, and your lab section. (Do this for both students if two students are working together.) If you do not follow these instructions, you may lose marks

1. A number of different sized pancakes are stacked. A sorted pancake stack is defined as having the smallest pancake on top, the second smallest pancake under the smallest pancake, etc. The only tool provided is a spatula that will flip any top partition of the stack. There only ever exists *one* stack of pancakes, not multiple smaller, independent stacks. If every flip took one unit of time to complete, exactly how many flips or units of time are required in the worst-case (i.e., for the worst-case arrangement of pancakes) to sort the stack? Aim for a good algorithm. Express your answer, $T(n)$, as a function of the number of pancakes. Then, give a Big-Θ estimate. What is the minimum number of flips needed to sort a worst-case arrangement of 4 pancakes? (Note that your algorithm may not necessarily give the true minimum number of flips.)

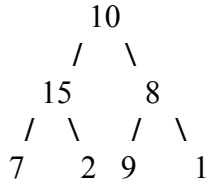
2. Let T be a full binary tree, and assume that the nodes of T are ordered by a *preorder* traversal. This traversal assigns the label 1 to all internal nodes, and the label 0 to each leaf (a node is internal if it is not a leaf). The sequence of 0's and 1's that results from the preorder traversal of T is called the tree's characteristic sequence.
    a) Determine the full binary trees for the characteristic sequences:
        i. 10110010100
        ii. 1011100100100

3. In class we employed a version of the Heapify algorithm in which an unordered sequence of elements was converted to a heap by the repeated insertion of the elements into a heap (beginning with the empty heap). After each insertion, the ReheapUp algorithm was applied to the new element to reheapify the tree structure.

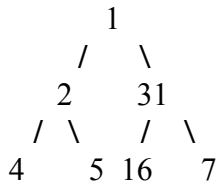
   Here's another way to Heapify a sequence of elements. First, just put all the elements into a complete binary tree, ignoring any notion of "heapness". Apply the ReheapDown algorithm to the roots of the smallest subtrees, then to the roots of the subtrees whose left and right subtrees have been heapified, and so on up the tree.

   Use this new version of Heapify to convert the following trees to heaps, and then sort them using Heapsort (either ascending or descending order for the results is fine). Clearly show each step.

   a) Create a *minimum* heap (then sort):



   b) Create a *maximum* heap (then sort):



4. The degree of a node in a tree refers to the number of edges *incident* on it (that is, the number of edges that connect to it). In the case of a binary tree, the maximum degree of any node is 3. The total degree refers to the sum total of every node's degree.

   For the following questions, either draw a tree with the given specification or explain why no such tree exists:

   a) Tree, 12 vertices, 15 edges
   b) Tree, 5 vertices, total degree 10
   c) Full binary tree, 5 internal (non-leaf) vertices
   d) Full binary tree, 8 internal (non-leaf) vertices, 7 leaves
   e) Complete binary tree, 4 leaves
   f) Nearly complete binary tree, 7 leaves

5. What is the height of the tallest possible nearly complete binary tree with 240 leaves? *(Do not draw the tree-- instead show your calculation.)*