COMP 3711H – Fall 2016
Tutorial 3

1. **Heapify**
   In class, we learned how to maintain a min-heap implicitly in an array.
   Given that $A[i \ldots (j-1)]$ represents an implicit min-heap, we saw how, in $O(\log j)$ time to add $A[j]$ to the min-heap. This led to an $O(n \log n)$ algorithm for constructing a min-heap from array $A[1, \ldots, n]$.

   For this problem show how to construct a min-heap from an array $A[1 \ldots n]$ in $O(n)$ time.

   It might help to visualize the min-heap as a binary tree and not an array.

   For simplification, you may assume that $n = 2^k + 1$ for some n, i.e., the tree is complete.

   Hint: Consider heapifying the nodes in the order from bottom to top.

2. **Randomized Binary Search Trees**

   - Consider a Binary search tree T on n keys.
     The *depth*, $d(v)$, of v in T is the length of the path from the root of T to v. Note that the depth of the root is 0. The *Path Length of* T, $PL(T)$, is the sum of the depths of all of the nodes of T; $PL(T) = \sum_{v \in T} d(v)$.

     Note that $\frac{1}{n} PL(T)$ is the average depth of a node in the tree. This is also the average time to search for a randomly chosen node in the tree.
   - Suppose that every key K_i in a set of n keys has real weight w_i associated with it, with the weights being unique.
   - There is a unique binary search tree that can be built on the n keys that also satisfies min-heap order by the weights (Why?).
   - Suppose n weights $w_1, w_2, \ldots, w_n$ are chosen independently at random from the unit interval $[0, 1]$ and then sorted. The resulting order is a random permutation of the n items.

   A *Treap* or *Randomized Binary Search Tree* on n keys K_i is constructed by choosing n weights w_i independently at random from the unit interval $[0, 1]$ and associating w_i with K_i. The Treap is the unique BST built on the n keys that also satisfies min-heap order on the weights.

   (a) If T is the Treap built, prove that the average value of $PL(T)$ is $O(n \log n)$
   Hint: consider quicksort

   (b) Describe how to build T in time $O(n \log n + PL(T))$

3. **AVL Trees**

   (a) Construct an AVL tree by inserting the items 134625 in that order.
   Next construct another AVL tree on those items by inserting in the order 123456.
   Do they have the same height?

   (b) Let T be a tree on n keys that satsifies the AVL balance condition. Is there always an insertion order that of the keys that builds T?
   If yes, what is it?

(c) What are the minimum and maximum heights for an AVL tree with 88 nodes labelled $1, 2, 3 \ldots, 88$?