TCSS 342

Assignment 3
Due February 8

1. [**Induction**, 25%]

   a Suppose that you have a full binary tree (any node has exactly two children or zero children). Prove by induction that a tree of this type with n leaves has exactly $2n - 2$ edges. [7.5%]

   b (k-**ary trees**) Suppose that you have a tree such that any internal node has exactly k children, with $k \geq 2$. What is the maximum number of nodes that such a tree can have, if its height is h? Prove your answer by mathematical induction on h. [10%]

   c (k-**ary trees**) In such a tree of maximum number of nodes, what fraction of the nodes are leaves? Prove your answer by induction on h. [7.5%]

2. [**Heap**, 25%]

   a Add to the Heap class (minimum heap) provided to you a method for finding the maximum element from the heap in the most efficient manner possible, without changing the fact that each node is smaller than its children [1]. Your method should be called: public Object getMax() throws HeapEmptyException. [20%].

   b What is the big oh of your method? justify your answer. [5%]

[1] Maximum element means maximum value of the largest key

3.[**Binary Tree**, 30%]

a Write in Java, a method **buildHeap**() that takes an array of n integers and returns a heap (in array representation). Your program should runs in $O(n)$ in the worst-case. [15%]

b Consider the following problem: Given an unsorted array $A[0...n-1]$ of n distinct integers, find the k-th largest element. For example,

$$\textbf{findK}([9, 10, 6, 7, 4, 12], k = 0);\ \texttt{returns}\ 12$$

$$\textbf{findK}([9, 10, 6, 7, 4, 12], k = 2);\ \texttt{returns}\ 9$$

Write in java a method **findK** that runs in at most $O(n + k \log n)$. [15%]

Both methods should be in a class called **arrayHeap**(). This class should also have a main method. [2]

4. [**Linked List**, 20%] For the given single linked list implementation, add a removeBefore(Object o) and removeAfter(Object o) method. In both cases, you should remove from the list the element before or after the object passed in as a parameter. If o is not present in the list, or if there is no element before or after it, you should throw a NoSuchElementException.

[2] Note that when $k = 0$ **findK**() runs in $O(n)$ times hence sorting the array is not an option here.