COMP 3711 Design and Analysis of Algorithms (Fall 2015)
Assignment 2

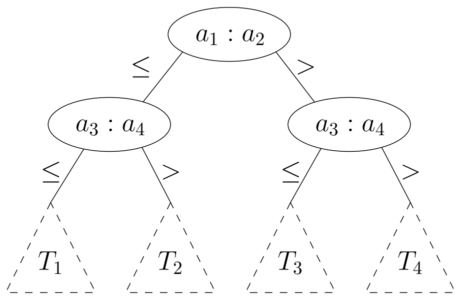
1. **Smallest k numbers in sorted order**

   Given a set of n numbers, we wish to find the k smallest numbers in sorted order using a comparison based algorithm. Below are some possible algorithms for this problem. For each algorithm, analyze its running times in terms of n and k. Note that all algorithms must be comparison-based. We assume that all numbers are distinct.

   (a) Sort all n numbers, and output the k smallest numbers in sorted order.
   (b) Build a heap on the n numbers, and call Extract-Min k times.
   (c) Build a heap on the n numbers by repeatedly inserting them into an initially empty heap, and call Extract-Min k times.
   (d) Can you design an algorithm better than all three above? [Hint: use the randomized linear-time selection algorithm.]

2. **Decision tree**

   The below figure shows part of the decision tree for mergesort operating on a list of 4 numbers, a_1, a_2, a_3, a_4. Please expand subtree T_3, i.e., show all the internal (comparison) nodes and leaves in subtree T_3.



3. **Sorting strings**

   Given an array A of m strings, where different strings may have different numbers of characters, but the total number of characters over all the strings in the array is n. Show how to sort the strings in $O(n)$ time. Note that the desired order here is the standard alphabetical order; for example, `a < ab < b`.

   More technically speaking, A is an array of pointers each pointing to a string (which is another array of characters); you can think about how strings are used in C. Also, we assume that each character can be viewed as an integer ranging from 0 to 255.

4. **AVL tree**

   Suppose we insert the following keys into an initially empty AVL-tree: 5, 3, 8, 2, 1, 4, 6, 7. Draw the tree after each insertion.

5. **Hashing**

   Suppose we insert n elements into a hash table of size m. The *collision number* is a measure of how good the hash table is, which is defined as the total number of pairs of elements that collide, i.e., $\sum_{i=1}^{m} \binom{n_i}{2}$ where n_i the number of elements hashed into location i in the hash table. For example, the hash table shown on slide 5 in the lecture notes has collision number $\binom{2}{2} + \binom{3}{2} + \binom{1}{2} + \binom{2}{2} = 1 + 3 + 0 + 1 = 5$. Assuming uniform hashing, derive the expected collision number in terms of m and n.