MIDTERM EXAMINATION — COMPUTER SCIENCE 308-251B — WINTER 1992
INTRODUCTION TO COMPUTER SCIENCE

Examiner: Luc Devroye
Associate examiner: Clark Verbrugge
Date: February 20, 2:30 pm.
Time limit: 90 minutes.

Instructions. Calculators are allowed. Answer in the examination booklets. The weights of the questions are shown in brackets.

Question 1. (6%) In a **random binary search tree** on n nodes, what is the expected depth of the node of minimal value? (Give an exact formula if possible.)

Question 2. (6%) A **binary tree** with n nodes numbered 1 through n is to be stored in an array of n records in such a way that in constant time, we can determine whether node u is an ancestor of node v for any nodes u, v. The total storage should be $O(n)$. We are not interested in any other operations. Explain what you would store and why.

Question 3. (6%) Consider a **complete k-ary tree** with n nodes stored in the usual fashion in an array. What is the index of the parent of node i, and how many leaves are there?

Question 4. (6%) **Quicksort** was repeatedly timed on problems of size $1024 = 2^{10}$, and the average time was 1000 μs. What average time would you predict for problems of size $8192 = 2^{13}$?

Question 5. (6%) Consider a **red-black tree** with four full levels of nodes. (Plus, if you wish, a fifth level of external nodes.) The values stored at the nodes are the integers $\{1, \ldots, 15\}$. All the leaves (level 4 nodes) and all the level 2 nodes are red, while the others are black.

A. What is the rank of the root?

B. Draw the tree (showing values and colors of nodes) that results after the element with value 0 is inserted.

Question 6. (6%) A **game tree** is given in which each node has k children. It is a two-player game in which the root corresponds to a position you are confronted with. You want to maximize the return. Assume that the k^2 grandchildren of the root correspond to final game positions with values $1, 2, \ldots, k^2$ from left to right. What is the value of the root?

Question 7. (48%) The **timed binary search tree** (or **timbin** for short) is an ordinary binary search tree in which besides the standard fields label, left and right, we also store time, where "time" is an integer storing the time of insertion. The first element inserted has time = 1, the second one time = 2, and so forth. Give a PASCAL algorithm for the operation `deleteroot`, in which the root is deleted in such a way that the timbin is restored to what it would be if the root had never been inserted. Do not forget the type definition either. The restoration must be consistent with the "time" stamps stored at the nodes. `deleteroot` should be efficient, i.e., it should take expected time $O(\log n)$ on a random timbin. Hint: when designing your solution, it helps to draw the tree in a time versus value coordinate plane.

Question 8. (10%) The abstract data type **freetree** consists of a cloud of points connected by edges in such a way that between any pair of points there is one and only one connecting path. Any node could be made the root of a tree. The shape of the tree is determined by the following simple rule: if (u, v) is an edge, and v is closer to the root than u, then v is the parent of u. The operations for **freetree** are:

A. parent(v,r,t): returns the parent of v in a **freetree** t in which node r is the root.

A. kids(v,r,t): returns the children of v in a **freetree** t in which node r is the root.

Describe a data structure that you would use for this situation (type definition, drawing, explanation), describe the operations, and discuss the time and space efficiency of your method.

Question 9. (6%) In a **ternary tree** with n nodes, let n_0, n_1, n_2 and n_3 be the number of nodes with 0, 1, 2 and 3 children respectively. If $n_3 = 30000$ and $n_2 = 20000$, how large is n_0? (It would also help if you told me how you deduced this result.)