1. (6 points) A complete binary tree has $n$ nodes. Determine the exact height of the tree (as a function of $n$). Prove your answer.

2. (9 points) Describe in elegant pseudocode the DECREASE-KEY($x$, newkey) operation in a heap. ($x$ is the name of a node; newkey is the value with which we replace key($x$). We assume newkey < key($x$).)

3. (3+12 points) (a) State the Knapsack Problem. Conciseness and clarity are paramount. (b) Assume there are $n$ items; the values are positive integers. Solve the problem in $O(nV)$ steps where $V$ is the combined value of all items. Describe the algorithm in pseudocode. Define your variables. A clear mathematical definition of your variables will account for half the credit.
4. (a) (2 points) For two sequences of real numbers, \( \{a_n\} \) and \( \{b_n\} \), define the relation \( a_n \sim b_n \) ("\( a_n \) is asymptotically equal to \( b_n \") as in the handout.

(b) (3 points) Explain why the statement "\((\forall n)(a_n \sim b_n)\)" does not make sense. Your explanation should be short and to the point.

(c) (3 points) For two sequences of real numbers, \( \{a_n\} \) and \( \{b_n\} \), define the relation \( a_n \gtrsim b_n \) ("\( a_n \) is greater than or asymptotically equal to \( b_n \")).

(d) (4 points) Give an example of two sequences of real numbers, \( \{a_n\} \) and \( \{b_n\} \), such that \( a_n \gtrsim b_n \) but infinitely often \( a_n < b_n \) and yet \( a_n \not\sim b_n \).

5. (4+8+3B points) Consider the recurrent inequality \( T(n) \leq 2T(n/2) + O(n) \). (a) Name two significantly different algorithms discussed in class of which the cost analysis lead to this recurrence. (b) Assuming \( T(1) = 0 \), prove that \( T(n) = O(n \log n) \). (Assume \( n \) is a power of 2.) (c) (BONUS) Prove the same conclusion assuming \( T(1) = 1 \).

6. (6+6B points) (Selection by rank) Given a list of \( n \) data (real numbers), we wish to find the \( k \)-th smallest among them (this is the element of rank \( k \)). We can only do comparisons with the data. (a) Describe a randomized algorithm which achieves this in an expected \( O(n) \) number of comparisons. Describe the algorithm in elegant pseudocode. (b) (BONUS) Let \( f(n) \) denote the expected number of comparisons performed by the algorithm. Prove that \( f(n) = O(n) \). (Use the reverse of this sheet.)