Data Structures - Assignment no. 4

Date due: 20/8/02

Remarks:

- For every algorithmic problem, try to find the most efficient solution. Analyze the complexity of every algorithm you give, and in case it is not trivial, prove its correctness.
- Unless you are explicitly asked otherwise, please describe algorithms only to the extent which makes them clear to a reasonable reader. You may use plain language, or pseudo-code, whichever is more convenient.

1. Show that every non-negative integer has a *unique* representation as a sum of distinct powers of 2.
2. Describe an algorithm which, given a pointer to the root of a maximum heap and a number t, prints all the elements in the heap that are larger than t. The complexity of the algorithm should be $\Theta(k)$ where k denotes the number of elements printed.
3. Describe an algorithm which, given a pointer to the root of a maximum heap and a number t, prints the t largest elements in the heap. The complexity of the algorithm should be $\Theta(t \log t)$.
4. Given a heap implemented by a binary tree, except for its size which is not given, show an algorithm that finds the size of the heap in time $\Theta(\log^2 n)$, where n is the number of elements in the heap.
5. Show an algorithm which uses $O(1)$ space, that is given a pointer to the root of an almost totally balanced tree and performs a breadth first search on it. The running time of the algorithm should be linear in the number of nodes in it (and you should prove it!).