CSC 263 H5S **Homework Assignment # 2** **Spring 2010**

Worth: 11% **Due:** Friday February 12

For each question, please write up detailed answers carefully. Make sure that you use notation and terminology correctly, and that you explain and justify what you are doing. Marks may be deducted for incorrect/ambiguous use of notation and terminology, and for making incorrect, unjustified or vague claims in your solutions.

1. [10 marks]

Let D be an ordered Dictionary with n items implemented with an AVL tree. Show how to implement the following method for D in time $O(\log n)$:

countAllinRange(k_1, k_2): return the number of items in D with key k such that $k_1 \leqslant k \leqslant k_2$.

note: you may assume that the AVL tree supports a data field at each node x which stores the number of nodes in the subtree rooted at x.

2. [10 marks]

Given two AVL trees T_1 and T_2, where the largest key in T_1 is less than the smallest key in T_2, Join(T_1, T_2) returns an AVL tree containing the union of the elements in T_1 and T_2.
Give an algorithm (in pseudocode) for Join that runs in time $O(\log n)$, where n is the size of the resulting AVL tree. Justify the correctness and efficiency of your algorithm.

3. [10 marks]

Suppose we are using a priority queue Q to schedule jobs on a CPU. Job requests together with their priorities are inserted into the queue. Whenever the CPU is free, the next job to execute is found by extracting the highest priority job in Q. Using a heap implementation for Q the scheduler can both extract the next job from Q and add a new job to Q in time O(log n) where n is the number of pending jobs in the queue.

However, in practice, we want our scheduler to be able to do more than just insert jobs and extract the highest priority job. In particular, we want our scheduler to be able to remove a job x from the job queue (Delete(x)) as well as change the priority of a job x to some new value k (Change-Priority(x, k)). Show how to add the operations Delete(x) and Change-Priority(x, k) to the heap data structure so that (a) they both run in time O(log n) and (b) the resulting data structure after executing either these operations is still a heap. For both operations, you may assume that parameter x gives you the index corresponding to job x in the array representation of the heap.

4. [10 marks]

Consider two Binomial Heap H_1 and H_2. You are required to create the union H of H_1 and H_2 and also to find the minimum key in H. A standard way would be to perform

$H \leftarrow \mathrm{Union}(\mathrm{H}_1, \mathrm{H}_2)$
$m \leftarrow \mathrm{LookupMin(H)}$.

Alternatively, you may perform

$m_1 \leftarrow \mathrm{LookupMin}(\mathrm{H}_1)$
$m_2 \leftarrow \mathrm{LookupMin}(\mathrm{H}_2)$
$m \leftarrow \min(m_1, m_2)$
$H \leftarrow \mathrm{Union}(\mathrm{H}_1, \mathrm{H}_2)$.

Give an example of two heaps H_1 and H_2 in which the first sequence of operation is faster, and one in which the second is faster. Justify your answer.