CS 430 Introduction to Algorithms **Spring Semester, 2021**

Homework 2

Assigned: Feb. 15 *Due: March 1*

As we want to post partial solutions before the midterm (scheduled for March 8), we modify the penalty for late assignments: 10% if submitted by noon March 5, which will be the latest date that this homework is accepted.

Please respect the following guidelines for writing pseudocode:

1. C/Java/Python instructions are fine. But do not write object-oriented additions. Do not declare or use any class. Declare only procedures (if necessary) and explain in words what each procedure does, and what is the use of each parameter.
2. One instruction per line
3. Match the brackets with a horizontal line
4. Number your lines
5. Write down if your array is indexed $0 \dots n-1$ or $1 \dots n$.

Problem 1 Assume the priority queue also must support the update DELETE(A,i), where i gives you the location in the data strucure where element i is located. In a binary heap, that would be $A[i]$. Write pseudocode to achieve DELETE(A,i) in $O(\log n)$ time for binary heaps, where n is the size of the heap.

Justify the running time.

Problem 2 Give an algorithm with worst-case running time of $O(k \log k)$ to print the k biggest elements in a n-element max-heap (binary heap stored in an array) without modifying the max-heap (or if you do modify it, restore it to the input heap within the given time bound). For example, if the max-heap is $A = [100, 88, 44, 66, 77, 11, 22, 5, 55]$ and $k = 5$, the output could be $100, 88, 66, 77, 55$ (the order in which the algorithm prints does not matter). Do not assume $n = O(k)$. Here n can be much larger than k.

A correctness proof is not needed (but your pseudocode must be correct). You can use additional data structures. Argue the running time.

Problem 3 Using only the definition of a binary search tree, show that if a node in a binary search tree has two children, then its successor has no left child and that its predecessor has no right child.

Here all the keys in the binary search tree are distinct. The successor of a node j is defined as follows: if the node j has the biggest key, the successor is NIL; otherwise the successor is the node i that has a key that is bigger than the key of j and such that no other node has key between the key of i and the key of j.

Problem 4 Suppose that a node x is inserted into a red-black tree with RB-INSERT and then is immediately deleted with RB-DELETE. Is the resulting red-black tree the same as the initial red-black tree? Justify your answer.

Problem 5 Suppose you are given two n-element sorted sequences A and B, each representing a set (none has duplicate entries). Describe an $O(n)$-time method for computing a sequence representing the set $A \setminus B$ (with no duplicates; note: **the difference** of sets A and B consists of those elements in A and not in B).

You do not have to argue correctness (but, obviously, your method must be correct), but must justify the running time.