Please do all problems; we will grade a subset of the assigned problems (same subset for everyone). Please follow all of the instructions given in the handout for Homework 1.

**Note:** This assignment includes an optional extra credit problem (#7) which will be graded if you choose to do it.

1. (12 points) Consider the Tree-Delete routine discussed in class (not the one in the text) for deleting items from a binary search tree $T$. Suppose that an external application holds a pointer to a node $y$ in $T$ and that Tree-Delete$(T, z)$ is called, where $z$ is the in-order predecessor of $y$. Explain carefully what problem might potentially arise. Rewrite Tree-Delete, by modifying the routine given in class slightly, to avoid this problem.

2. (10 points) Ex. 13.3-3, p. 322, and Ex. 13.4-5, p. 330.
   Show the requested information before and after each transformation given in Figs. 13.5–13.7 and verify what is asked for in the two questions. You may show the requested information in the figures themselves or in a table.

3. (16 points) Problem 13-2, p. 332-333. (Skip part (e) as it is symmetric to part (b).) Supplement your answer with short code fragments, as appropriate.

4. (12 points) Ex. 14.3-3, p. 353. Describe the main ideas underlying your solution (from which correctness should be evident), give pseudocode, and analyze the running time.

5. (12 points) Let $T$ be a red-black tree storing $n$ integer-valued keys. Each node, $x$, has the usual fields, i.e., key, color, and pointers to its parent and children. In addition, $x$ also stores a field, $d(x)$, which equals the number of odd-valued keys in $x$’s subtree minus the number of even-valued keys in $x$’s subtree. ($x$’s subtree includes $x$.)
   For example, if $x$’s subtree has 6 odd-valued keys and 4 even-valued keys, then $d(x) = 2$; if it has 3 odd-valued keys and 4 even-valued keys, then $d(x) = -1$.
   Use the General Augmentation Theorem (GAT) to show that the field $d(\cdot)$ can be maintained at all nodes of $T$ while keys are inserted or deleted in $T$ without affecting the $O(\log n)$ time bounds for these operations. Do not store any other information at the nodes. **You must use the GAT rather than proceed from first principles.**

6. (14 points) Let $R$ be a set of $n$ rectangles in the $xy$-plane, with sides parallel to the coordinate axes, and let $P$ be a set of $n$ points in the plane. Given $R$ and $P$, our goal is to decide for each point in $P$ whether or not the point lies inside at least one rectangle of $R$, i.e., the output should simply be “IN” if the point lies in the interior or on the boundary of at least one rectangle of $R$, and “OUT” otherwise. (Think of the rectangles as windows on a computer screen and the points as mouse-clicks, for instance.) Give a sweepline algorithm for this problem which runs in $O(n \log n)$ time.
   Your answer should include (i) a description of the main ideas and data structures underlying your solution (from which the correctness of your solution should be evident), (ii) pseudocode, and (iii) an analysis of the running time.
Assume that each rectangle $R_i \in \mathcal{R}$ is specified by its lower-left corner $(\ell_i, b_i)$ and upper-right corner $(r_i, t_i)$. For simplicity, assume that all $x$-coordinates and all $y$-coordinates in $\mathcal{R} \cup \mathcal{P}$ are distinct.

You may use any data structure studied in class as a black-box, so long as you do not modify it.


Your solution should use an order-statistic tree. Also, assume that you have a routine that can sort the endpoints of the chords around the circle. Use the order-statistic tree and the sorting routine as black boxes; i.e., you do not have to write pseudocode for these.

Your answer should include (i) a description of the main ideas underlying your solution (from which the correctness of your solution should be evident), (ii) pseudocode, and (iii) an analysis of the running time.

Note: This is an optional extra credit problem to help you improve your performance in the class, if you so desire. The points you earn on this problem will be added to the score you receive on the other problems that we grade, without changing the overall value of the assignment. So, for instance, suppose that the required part of the homework that we grade is worth (say) 50 points. If you get (say) 45 points on that and 8 points on this optional problem, then your score will be 53/50; thus, it is possible to earn more than 100% on this assignment.