1. Consider the following dominance problem. Given a set $S$ of points in 3-dimensional space, report for every point $p \in S$, all points $q \in S \setminus \{p\}$ such that $x(q) \geq x(p)$, $y(q) \geq y(p)$, and $z(q) \geq z(p)$. More precisely, the problem asks to find all pairs $(p, q) \in S \times S$ such that $x(q) \geq x(p)$, $y(q) \geq y(p)$, $z(q) \geq z(p)$, and $q \neq p$.

(a) (5 pts) Consider the following algorithm to solve this problem using a 3-dimensional range tree: For each point $p \in S$, do a 3-dimensional range query to find the points that dominate $p$. How much space would this algorithm require and how long would this take to output $t$ pairs?

**ANSWER:** $d$-dimensional range trees use $O\left(\lg^{d-1} n + t\right)$ time per query with fractional cascading (but $O\left(\lg^d n + t\right)$ as shown in class), $O\left(n \lg^{d-1} n\right)$ space and can be built in $O\left(n \lg^{d-1} n\right)$ time. Creating a 3-dimensional range tree takes $O\left(n \lg^2 n\right)$ space and time. The $n$ 3-dimensional range queries take $O\left(\lg^2 n + t_i\right)$ time each for a total running time of $O\left(n \lg^2 n + t\right)$. This would be $O\left(n \lg^3 n + t\right)$ without fractional cascading.

(b) (15 pts) Develop an algorithm that solves this problem in $O\left(n \log n + t\right)$ time, where $t$ is the total number of reported pairs. (Hint: Use the sweep-line approach to reduce this problem to a set of two-dimensional problems and use the proper data structure to solve each two-dimensional problem.) As usual, give either a brief description of your approach along with pseudo-code or a detailed description that would be sufficient for someone to write the pseudocode.

**ANSWER:** Lets use a $yz$-plane sweep that moves over the $x$-coordinates of the points from $\infty$ to $-\infty$. Whenever we reach a new point $p$ we first add that point to a priority search tree using its $y$ and $z$ coordinates. We then do a 2-sided range query on this priority search tree to find all of the points that dominate $p$.

(c) (10 pts) Show that your algorithm is correct, that is it terminates (which should be trivial) and outputs every dominated pair and just the dominated pairs (induction or contradiction should both work well).

**ANSWER:** The algorithm terminates because we do a constant number of things (insertions, deletions, range-queries, etc) for each point. To prove that the algorithm is correct, lets assume that it is not. That is, the algorithm either outputs some pair $(p, q)$ such that $q$ does not dominate $p$, misses some pair $(p, q)$ such that $q$ does dominate $p$, or outputs some pair twice. The first case is impossible because we only output dominating pairs. In the second
case, consider the event point for $p$ during the algorithm. Since $q$ dominates $p$, we know that we have already added $q$ to the priority search tree. Then the range search on the priority search tree would have correctly found this pair. This range search is the only time we could output a pair $(p, x)$, for all $x \in S$, and it will not output any such pair twice, so the third case is also a contradiction. Therefore the algorithm is correct.

(d) (10 pts) Show that the running time of your algorithm is indeed $O(n \log n + t)$.

**ANSWER:** The running time is easy to see. We need to sort the pairs by $x$-coordinate and insert them into a queue which takes $O(n \log n)$ time. For each of the $n$ event points we do one priority search tree insertion and one range-query taking $O(\log n)$ amortized time and $O(\log n + t_i)$ time, respectively, where $t_i$ is the number of reported pairs. Thus the algorithm takes $O(n \log n + t)$ time to find $t$ pairs.