1. Suppose we are given a set H of n horizontal line segments in the plane, each specified by its left x-coordinate $h.l$, its right x-coordinate $h.r$, and its y-coordinate $h.y$. Describe a data structure for H that supports queries of the following form:

   - CrossCount(v): Given a vertical line segment v, specified by its x-coordinate $v.x$, its bottom y-coordinate $v.b$, and its top y-coordinate $v.t$, return the number of horizontal segments in H that intersect v.

   As usual, you should describe the data structure, analyze its space complexity, describe the query algorithm, briefly justify its correctness, and analyze its running time. *[Hint: Also as usual, don't reinvent the wheel. Also, remember that correctness is more important than achieving the optimality. I know of at least two different solutions.]*

The remaining problems are for you play with on your own.
Discussion in office hours or on Discord is welcome, but don't submit solutions!

---

2. Suppose we are given a set R of n axis-aligned rectangles in the plane, each with a priority $r.prior$. Describe a data structure for R that supports queries of the following form:

   - CLICK(q): Given a query point $q = (q.x, q.y)$, return the rectangle in R with minimum priority that contains q (or return ∞ if no rectangle in R contains q).

   This query models selecting a window in a standard graphical user interface.

3. The lecture notes describe range trees and segment trees as static data structures, but they can be modified to support insertions and deletions just like binary search trees.

   (a) Describe how to construct a one-dimensional range tree for n unsorted numbers in $O(n \log n)$ time.

   (b) Describe how to construct a two-dimensional range tree for n unsorted points in the plane in $O(n \log^2 n)$ time.

   (c) Suppose we decide to maintain a two-dimensional range tree using scapegoat trees, both for the primary range tree over the x-coordinates and for each secondary range tree over the y-coordinates. Whenever we rebuild a subtree of the primary tree, we must also rebuild all the secondary structures of nodes in that subtree. Even when we do not rebuild anything, inserting a point requires inserting its y-coordinate into $O(\log n)$ secondary structures.

   What is the amortized time to insert a single point into a 2D scapegoat tree? *[Hint: Describe the insertion algorithm in more detail first. Don't worry about deletions; we can handle those efficiently with tombstones.]*

   ★(d) Professor Janet Grundy doesn't like scapegoat trees, so she decides to use AVL trees to build a dynamic two-dimensional range tree instead. Each time we perform a rotation in the primary tree, we must rebuild the secondary structures at the primary nodes whose children change.

   What is the amortized time to insert a point into Professor Grundy's 2D AVL tree? *[Hint: Recall that each insertion into an AVL tree requires at most two rotations in the worst case. But where are those rotations, exactly? Again, don't worry about deletions; we can handle those efficiently with tombstones.]*