## Range search

1. **Range tree space.** Prove or disprove: for any set of $n$ points in general position, the range tree uses $\Omega(n \log n)$ space.

2. **Priority Search Tree.** Show how to build a priority search tree in $O(n \log n)$ worst-case time. Note: in fact, $O(n)$ worst-case (using just CS240E material) is possible.

3. **kd-tree.** Create a set of $n$ points and a range-query such that doing the range-query on the kd-tree of the points requires $\Omega(\sqrt{n})$ boundary nodes.

4. **Quad-tree.**

   (a) For an arbitrary $n$, construct a set of points such that the quad-tree has at least $n$ nodes, and give a range-search query such that all nodes are visited, and not a single point gets returned.

   (b) Assume that $T$ is a quad-tree with at least two points such that during some range-search, there is at least one outside node and at least one inside-node (the example from Module 8, slide 11 satisfies this). What is the minimum possible height of $T$?

   The example has height 3, so the question is whether height 3 is always required, or whether this could also happen with height 2 or even height 1?

## String matching

5. **Cyclic shift.** Given two strings $w$ and $x$ of length $n$, determine if $w$ can be obtained by cyclically shifting the characters of $x$. For example, the algorithm should return `true` if the input is `alloy` and `loyal`, and `false` if the inputs are `tarot` and `otter`. Your algorithm should take $O(n)$ time for two strings of length $n$.