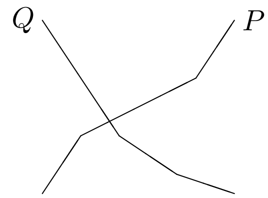
CS 498TC, Spring 2018
Timothy Chan

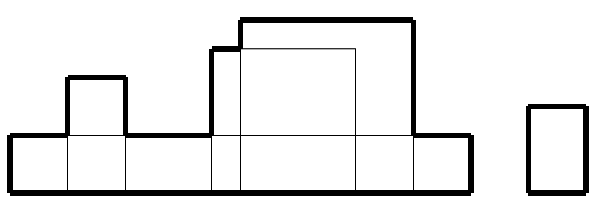
Assignment 1 (due Feb 9 Friday 2pm (in class))

You may work in a group of at most 3 students. Carefully read `https://courses.engr.illinois.edu/cs498tc3/policies.html` and `https://courses.engr.illinois.edu/cs498tc3/integrity.html`.

1. [*12 pts*] An *increasing* chain is a (not necessarily convex) polygonal chain $p_1p_2\cdots p_n$ such that $p_1.x < p_2.x < \cdots < p_n.x$ and $p_1.y < p_2.y < \cdots < p_n.y$. A *decreasing* chain is a polygonal chain $q_1q_2\cdots q_n$ such that $q_1.x < q_2.x < \cdots < q_n.x$ and $q_1.y > q_2.y > \cdots > q_n.y$. Given an increasing chain P and a decreasing chain Q, describe an $O(\log n)$-time algorithm to compute the intersection of P and Q. (Note that if an intersection exists, it must be unique.)



2. [*25 pts*] We consider the following problem of computing a "skyline" of buildings each represented as a 2D rectangle. More precisely, we are given a set R of n (axis-aligned) rectangles in 2D where all the bottom edges lie on the x-axis. (Each rectangle R is of the form $[a_1, a_2] \times [0, b]$.) We want to compute the union U of these rectangles. (U may be a polygon or consist of multiple polygons.)



   (a) [*10 pts*] Give an $O(n \log n)$-time algorithm for this problem.

   (b) [*5 pts*] Prove that the problem requires $\Omega(n \log n)$ worst-case time for comparison-based algorithms. To be precise, assume the version of the problem where the vertices of each polygon in U must be outputted in left-to-right order, and the polygons themselves (if there are multiple ones) may be outputted in arbitrary order.

(c) [*10 pts*] Give an $O(nh)$-time algorithm where h is the output size.

(d) [*Bonus, up to 5 pts*] Give an $O(n \log h)$-time algorithm where h is the output size.
[Hint: you may use the following fact: given a set of m vertical line segments, one can build a data structure in $O(m \log m)$ time so that we can find the first point hit by any horizontal ray in $O(\log m)$ time.]

3. [*8 pts*]

(a) [*4 pts*] Consider the following problem: given a set P of n red points and a set Q of n blue points in 2D, find a line ℓ such that all red points of P are below ℓ and all blue points of Q are above ℓ. Convert this problem into an equivalent one in *dual* space.

(b) [*4 pts*] Consider the following problem in 2D: given a set P of n points and a point q in 2D, find a line ℓ that passes through q and is tangent to the upper hull of P (with the upper hull of P lying below ℓ). Convert this problem into an equivalent one in *dual* space.