Please solve the following 3 problems, both of which have multiple parts.

**Some important homework policies**

- Groups of one or two students may work together. They should submit a single copy of their assignment using one of their eLearning accounts. Everybody in the group will receive the same grade.

- Each group must write their solutions in their own words. Clearly print your name(s), the homework number (Homework 1), and the problem number at the top of every page in case we print anything. Start each numbered homework problem on a new page.

- Unless the problem states otherwise, you must justify (prove) (argue) that your solution is correct.

- Any illegible solutions will be considered incorrect, so you might consider using \LaTeX to typeset your solutions. There is a template provided on the course website to help you get started.

- If you use outside sources or write solutions in close collaboration with others outside your group, then you may cite that source or person and still receive full credit for the solution. Material from the lecture, the textbook, lecture notes, or prerequisite courses need not be cited. Failure to cite other sources or failure to provide solutions in your own words, even if quoting a source, is considered an act of academic dishonesty.

- The homework is assigned to give you the opportunity to learn where your understanding is lacking and to practice what is taught in class. Its primary purpose is not for Kyle to grade how well you paid attention in class (although it does contribute substantially to your final grade). Read through the questions early. Do not expect to know the answers right away. Questions are not necessarily given in order of difficulty. Please, please, please attend office hours or email Kyle so he can help you better understand the questions and class material. Seriously, **Kyle enjoys busy office hours**.

- You may assume that any reasonable operation involving a constant number of objects of constant complexity can be done in $O(1)$ time. You may also assume inputs lie in general position (whatever that means for your particular problem such as no two points sharing an $x$-coordinate, no three points sharing a line, etc.). Clearly state your assumptions if they are not something we already used in lecture.

See [https://personal.utdallas.edu/~kyle.fox/courses/cs6319.001.22s/about/](https://personal.utdallas.edu/~kyle.fox/courses/cs6319.001.22s/about/) and [https://personal.utdallas.edu/~kyle.fox/courses/cs6319.001.22s/writing/](https://personal.utdallas.edu/~kyle.fox/courses/cs6319.001.22s/writing/) for more detailed policies before you begin. If you have any questions about these policies, please do not hesitate to ask during lecture, in office hours, or through email.
1. (a) Truthfully write the phrase “I have read and understand the policies on the course website.”

Let \( P = \{p_1, \ldots, p_n\} \) be a set of \( n \) points in the plane \((\mathbb{R}^2)\), and let \( p_i = (x_i, y_i) \) for each \( i \). The Pareto set \( \text{Pareto}(P) \subseteq P \) is the subset of \( P \) containing each point \( p_i \) such that there exists no point \( p_j \in (P \setminus \{p_i\}) \) such that both \( x_j \geq x_i \) and \( y_j \geq y_i \) are true. In other words, each point of \( \text{Pareto}(P) \) has no other point of \( P \) both above and to the right of it; you can visualize them as the turns on a staircase. See Figure 1.

(b) Describe and analyze an \( O(n \log n) \) time algorithm to compute \( \text{Pareto}(P) \).

[Advice: Write a modification of Graham’s scan.]

(c) Let \( h = |\text{Pareto}(P)| \). Describe and analyze an \( O(nh) \) time algorithm to compute \( \text{Pareto}(P) \).

[Advice: Write a modification of Jarvis’s march.]

(d) Describe and analyze an \( O(n \log h) \) time algorithm to compute \( \text{Pareto}(P) \). For simplicity, you may assume that the value \( h \) is known in advance.

[Advice: Write a modification of Chan’s \( O(n \log h) \) time convex hull algorithm.]

2. Let \( \mathcal{C} = \{C_1, \ldots, C_n\} \) be a set of \( n \) circles in \( \mathbb{R}^2 \) where each circle \( C_i \) is given as its center point \( q_i = (x_i, y_i) \) and radius \( r_i > 0 \). Describe and analyze an \( O(n \log n) \) time algorithm to determine whether or not any pair of circles intersect. Note that some circles may be completely nested in one-another without intersecting. See Figure 2.

[Advice: Describe a plane sweep algorithm. To describe one, it suffices to describe 1) what is stored for the sweep-line status and what data structure is used, 2) what are the event points and what data structure is used for the event queue, and 3) what operations and in particular what updates to the sweep-line status and event queue occur during events.]
3. A civil engineer has asked for your help in analyzing whether a dangerous portion of a river will flood. They present you with the following (admittedly rather unrealistic) model of the river. The portion of the river of interest is modeled as an $x$-monotone polygon $P$ that is bounded between two vertical lines at $x = x^-$ and $x = x^+$ (see Figure 3). The river is bounded on its left and right ends by two vertical line segments of lengths $w^-$ and $w^+$, respectively. Inside the polygon are some number of disjoint $x$-monotone polygons that represent islands in the river. Let $n$ denote the total number of vertices, including both the outer banks of the river and the islands.

![Figure 3. The portion of the river that may flood.](image)

The engineer tells you that in order to avoid a flood, the width of the river (not counting islands) at every vertical cut must be at least some minimum value $w_{\text{min}}$. For example, in the figure, the sum of the lengths of the two blue vertical segments at $x = x_0$ must be at least $w_{\text{min}}$ in order to avoid a flood.

Describe an $O(n \log n)$ time algorithm that, given the polygon $P$ and the value $w_{\text{min}}$, determines whether the river will flood. In other words, it should report whether or not there is a vertical cut whose total width is smaller than $w_{\text{min}}$.

[Advice:  
1) Prove that it suffices to check the width only at vertical cuts $x = x_0$ for which $x_0$ is the $x$-coordinate of a polygon vertex.  
2) Use a plane sweep algorithm. Argue that between polygon vertices, the width of the sweep line’s cut changes at a constant rate. How is this rate of change affected as the sweep line passes over a polygon vertex?]

---

1'We'll formally define monotone polygons on Tuesday, February 1st. See Mount page 33.