Instructions: Please submit your solutions at the extra class, Wed. May 6, 5-7PM, or under my office door before or during that class. You should imagine that this is an in class, open book exam: in other words you can use books and notes, but you are expected to work COMPLETELY INDEPENDENTLY, not even to discuss the questions with other persons.

• Please write “these solutions are entirely my own work” on the first page of your solutions and then sign your name.

• Use sufficient detail to make answers and explanations convincing. When giving an algorithm for some task, always give the most efficient one you can find. In any case make sure you give SOME algorithm that can do the task. Always give the complexity of your alg.

• Be explicit if you employ algorithms we have already studied, and state which of their properties you are using (e.g. “... using Bentley-Ottman line sweep with $O(\log n)$ cost per vertex swept”).

• (NOTE:) If you explicitly use material from a book or a paper, or from the web, cite the reference.

• Please try to avoid pseudocode unless it really simplifies the description of an algorithm.

1. Let $P_1 = (0, 0)$, $P_2 = (4, 0)$, $P_3 = (1, 1)$, and $P_4 = (0, 3)$.

   (a) (5 pts) Carefully describe Voronoi diagram $\text{VOR}(P_1, \ldots, P_4)$ using Euclidian distance; i.e,
   \[ d_2(C, D) = \sqrt{(C_x - D_x)^2 + (C_y - D_y)^2} \]
   is the distance between $C = (C_x, C_y)$ and $D = (D_x, D_y)$.

   (b) (20 pts) Now carefully describe $\text{VOR}(P_1, P_2)$, $\text{VOR}(P_1, P_4)$, and $\text{VOR}(P_1, P_2, P_4)$ using both the ($L_1$) distance and the ($L_\infty$) distance: The $L_1$ distance is defined by
   \[ d_1(C, D) = |C_x - D_x| + |C_y - D_y|, \]
   and the $L_\infty$ distance is defined by
   \[ d_\infty(C, D) = \max(|C_x - D_x|, |C_y - D_y|). \]

   (c) (15 pts) And for something completely different, we are now given a set $S$ of $n$ points in the plane, and are asked to construct a circle $C$ that contains the greatest number of points of $S$ on its circumference, and satisfies property (*) namely that “C has NO points of $S$ in its interior”. Describe an algorithm for this task, explain why it works, and give its running time. EXTRACREDIT: What can be said if $C$ IS allowed points of $S$ in its interior?
2. (40 pts) We are given a set \( S = \{P_1, \ldots, P_n\} \) of \( n \) points in general position in the plane. For this problem we will enforce properties that would hold with probability ONE for \( n \) random points: (i) no two points with the same \( x \)-coordinates; (ii) no three points collinear; (iii) no two pairs of points determine parallel lines.

A “3-strip” for \( S \) is determined by a pair of parallel, non-vertical lines \( \ell_1, \ell_2 \) with equations \( y = mx + b \) and \( y = mx + c \), respectively, with the property that exactly three points of \( S \) lie ON or BETWEEN these parallel lines. The strip consists of the two lines, and the three points of \( S \) that are ON or BETWEEN them. The width of a 3-strip is the \textit{vertical} distance between the two lines, that is, \(|b - c|\).

(a) Using the duality \( T \) that maps \( P = (x, y) \) to \( TP \), the line with equation \( v = xu + y \) and the line \( \ell \) with equation \( y = Ax + D \) to \( T\ell = (\rightarrow A, D) \), carefully describe the dual of a 3-strip.

(b) Prove that if \( \ell_1 \) and \( \ell_2 \) define a 3-strip for \( S \) of minimal width, then NO points of \( S \) are interior to the strip; i.e., one of the lines \( \ell_1 \) or \( \ell_2 \) contains two points of \( S \), and the other contains one. How does this refine the answer to (a)?

(c) Devise an algorithm that finds a minimum width 3-strip for \( S \), explain why it works, and give its running time. Then discuss the complexity of this task.