COMP163 Homework Assignment 1: Due Tuesday, September 18, 2018

Reading: Please read Chapter 1 of the Required Text, and also explore the "Lecture Notes" and/or other recommended texts.

Problems: Please remember to cite *ALL* your sources (TA, friends, faculty members, books, websites).

1. Left Turn – just persuade yourself but do *not* submit: Given points $A = (x_1, y_1)$, $B = (x_2, y_2)$, and $C = (x_3, y_3)$, the determinant $D =$

$$\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$$

   gives twice the signed area of triangle ΔABC where the sign is + if and only if A, B, C appear in counterclockwise order on the boundary of ΔABC. In other words, A, B, C forms a left turn if and only if $D > 0$. Use analytic geometry to derive and verify this fact to yourself.

   You do *NOT* need to hand-in a written solution to this problem!

2. Un-ordered Divide and Conquer Convex Hull

   (a) Let P_1 and P_2 be n_1- and n_2-vertex convex polygons, respectively, with no vertices in common. Prove that the number of lines of support common to P_1 and P_2 is at most $2 * min\{n_1, n_2\}$ and that this bound is achievable. [How many common lines of support can they have if P_1 and P_2 do not intersect? How many if P_1 is interior to P_2]

   (b) Given two arbitrary convex polygons P_1 and P_2 with n_1 and n_2 vertices each (their boundaries may intersect one, two, or more times; they may be disjoint; one may be contained within the other), specify as efficient an algorithm as you can to compute the convex hull of $P \cup Q$. Analyse its complexity.

   (c) Use this algorithm to generate a divide-and-conquer algorithm without presorting for finding the convex hull of an arbitrary set of n points. Analyse complexity.

3. Reprise of in-class exercise
   Given a set S of n planar points in general position and in arbitrary order, provide and analyse the most efficient algorithm that you can for constructing:

   - a simple monotone polygon R whose vertices are exactly the points in S and which contains the line segment between the point in S of smallest x-coordinate and the point in S of largest x-coordinate. [Can you do this in $\Theta(n \log n)$ time?]
   - a simple starshaped polygon P whose vertices are exactly the points in S for which the point of S of smallest x-coordinate "sees" every point of P. [Can you do this in $\Theta(n \log n)$ time?]

   Now assume that you are given a series of query points q that are NOT in S. For each of P and R, provide and analyze an algorithm that will test whether each query point q is interior or exterior to that polygon. [Can you do this in $\Theta(\log n)$ time?]