Exercises

Problem 1. Suppose that G is a regular SLPG (straight-line planar graph). The point location structure discussed in the lecture assumes a triangle ABC (the points A, B, C are not in G) that contains all the segments of G in its interior. Give an algorithm to find such a triangle in $O(n)$ time, where n is the number of vertices in G.

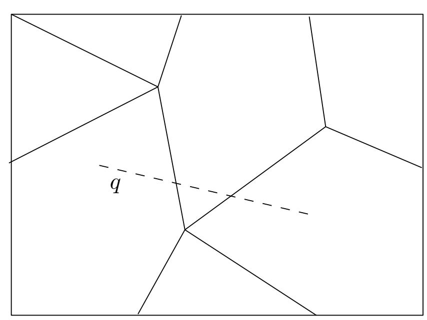
Problem 2. Let G be a regular SLPG with m edges, represented in the adjacency list format. Give an $O(m \log m)$ algorithm to find all the bounded faces of G. Your algorithm should produce each face's edges in clockwise order.

(Hint: after a face has been reported, mark all its edges. Each edge can be marked at most twice).

Problem 3. Prove: the point location structure discussed in the lecture can be constructed in $O(m \log m)$ time, where m is the number of edges of the input SLPG.

Problem 4. Let S be a set of disjoint concave polygons. Preprocess S into a data structure such that, given any point $q \in \mathbb{R}^2$, we can find the id of the polygon containing q (if such a polygon does not exist, output nothing). Your structure should consume $O(m)$ space and answer a query in $O(\log m)$ time, where m is the total number of edges of the polygons in S.

Problem 5. Given a regular SLPG G where every bounded face is a convex polygon, explain how to build a structure to answer queries of the following form: given a query segment q, find all the faces of G intersecting q. For example, in the figure below, q intersects with 3 faces. Your structure needs to consume $O(m)$ space where m is the number of edges in G; it must answer any query in $O(k \log n)$ time in expectation, where k is the number of faces reported.



Problem 6. Let G be any regular SLPG (not necessarily triangulated) with n vertices and m edges. Prove: $m = O(n)$.