Due: 21st of April 2011 at 1pm.

COMP 5045 – Assignment 2

1. Let P be a set of n points in the plane, and let R be a set of n non-intersecting axis-parallel rectangles in the plane. A point in P may or may not lie in a rectangle in R. Design an algorithm that outputs all the points in P that do **not** lie in any of the rectangles in R. Argue on the correctness and run time of your algorithm. The faster running time and the less storage space your algorithm uses, the more points you get. For full points your algorithm should run in $O(n \log n)$ time. [10 points]

2. Imagine you have an application that involves orthogonal range queries, and you have to implement all necessary data structures, preprocessing and query algorithms. Assume you have to decide between range trees and kd-trees. What criteria would you base this decision on? Please discuss! [10 points]

3. Given a convex polygon P and a point w that is external to P, we say that a point q on the boundary of P is visible to w if the open line segment $\overline{wq}$ does not intersect P. Let P denote an n-sided convex polygon enclosed within a closed circular disk C. Present an $O(n)$ time algorithm that, given P and C, determines whether there exist two points w_1 and w_2 lying within C but outside of P, such that every point on the boundary of P is visible to either w_1 or w_2 (or both). If so your algorithm should output such a pair of points, see Fig. 1a. [10 points]

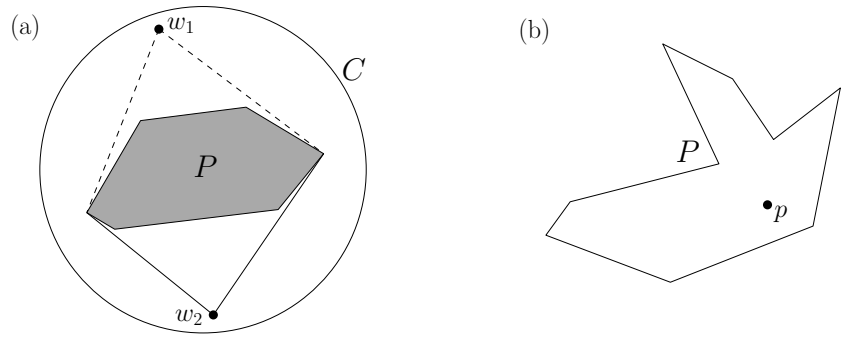

Figure 1: (a) Illustration of question 3. (b) The polygon is start-shaped since p can "see" all of P.

4. A simple (hole-free) polygon P is called *star-shaped* if it contains a point q such that for any point p in P the line segment $\overline{pq}$ is contained in P, see Fig. 1b. Give an $O(n \log n)$ time algorithm, or an $O(n)$ randomised expected time algorithm that decides if a given polygon is star-shaped or not. [10 points]

5. *Point queries among rectangles.* Let P be a set of n rectangles in the plane. Report all rectangles in P that intersects a query point q. Describe a data structure for this problem that uses $O(n \log n)$ preprocessing, $O(n)$ storage and $O(n^{3/4} + k)$ query time. *Hint:* How can a rectangle in 2D be described in 4D? [10 points]