Algorithmics II (H)

Tutorial Exercises on Geometric Algorithms

1. Provide an implementation for the following function, assuming a suitable choice for the type `Line`.

```
/** returns true if the lines l1 and l2 intersect, and
  * returns false otherwise */
public boolean intersect(Line l1, Line l2)
```

2. Provide an implementation for the following function, assuming a suitable choice for the type `LineSegment`.

```
/** returns true if the rectangles containing line
  * segments l1 and l2 intersect, and returns false
  * otherwise */
private boolean boundingBox(LineSegment l1,
                            LineSegment l2);
```

3. Give an example to show why the `boundingBox` test is necessary in addition to the `onOppositeSides` tests, in order to correctly determine whether two line segments intersect.

4. Let S be a set of n horizontal line segments in the plane, all of which lie on the x-axis. Give an O($n$log n) algorithm to determine whether any two line segments intersect one another.

5. Suppose that we are given a set S of n rectangles, all of whose edges are parallel to the x and y-axes. Design an O(n) algorithm to find the intersection of all rectangles in S.

6. The algorithm described in lectures for constructing a simple polygon needs to be extended to take account of one additional special case. Describe what this special case is, indicate why the algorithm in its present form could give the incorrect answer in such a case, and describe a modification that will address this problem.

7. Is it strictly necessary for the Graham Scan algorithm to start with a point that is guaranteed to be on the convex hull? Explain why or why not.

8. Draw a set of points that makes the Graham Scan algorithm relatively inefficient.

9. Suppose that there are n points in the plane, $n/2$ of which are coloured red and $n/2$ of which are coloured blue. Consider whether the convex hull based $O(n \log n)$ algorithm to find a furthest pair of a set of n points can be amended to find:

    - a furthest pair of points of the same colour;
    - a furthest pair of points that have different colours.

    Explain how the algorithm can be easily amended for one of these problems, and why it may be more difficult to amend to solve the other.

10. Suppose P is a set of points in the plane, where the distances between every pair of points are distinct. Which of the following statements is/are *always* true for the Closest Pair algorithm? Explain your answer in each case.

    a) We can find an *actual* closest pair of points (rather than just the closest pair distance) in $O(n \log n)$ time.

    b) The line of code in at the top of Slide 9 of Lecture 3 could also have set d to be the largest y-coordinate of any point.

    c) We can find a second-closest pair of points in $O(n \log n)$ time.

11. A point P in the plane is said to *dominate* another point Q if both the x-coordinate and the y-coordinate of P are greater than or equal to those of Q. A point P is *maximal* in a given set S of points if no other point in S dominates it. Design an $O(n \log n)$ algorithm to find all the maximal points in a given set of n distinct points.

12. Describe an algorithm with $O(n^2 \log n)$ worst-case complexity that will take as input a set of n points in the plane and will determine whether the set contains any three points P, Q and R such that R is the mid point of the line segment P—Q. Justify the $O(n^2 \log n)$ complexity.

13. The input is a set of n points in the plane. Design an $O(n^2 \log n)$ algorithm to determine whether there exist four points in the set that are the corners of a square whose sides are horizontal and vertical.

14. Suppose we are given a set of line segments in the plane such that all segments are horizontal, vertical or have a 45° angle with the horizontal. Show how to extend the algorithm for reporting all intersections among a set of horizontal and vertical line segments to this case without increasing the asymptotic worst-case time complexity.