Homework 2

Sam Windham *with Lucio Franco*

10/17/17

1.

```
PROCEDURE LeftTangent(Polygon p, Point q):
    // first point is left tangent
    if isLeft(q, p[n-1], p[0]) AND !isLeft(q, p[0], p[1]):
        return (q, p[0])

    let i = 0, offset = 1
    loop:
        // direction of current edge
        let edgeDir = isLeft(q, p[i], p[i+1])

        // current edge is the left tangent point
        if isLeft(q, p[i-1], p[i]) AND !edgeDir:
            return (q, p[i])

        // check directions and continue search

        let prevEdgeDir = isLeft(q, p[i - offset], p[i - offset + 1])

        // current edge is left
        if edgeDir:
            // previous edge is left AND current point left of previous point
            if prevEdgeDir AND !isLeft(q, p[i-offset], p[i]):
                i -= offset
                offset = 1
            // prev edge is right OR current edge left of previous edge
            else:
                offset *= 2
        else:
            // previous edge is left
            if prevEdgeDir:
                i -= offset
                offset = 1
            else:
                offset *= 2

        i = min(i + offset, n - 1)
```

The algorithm performs an exponential search on the points of P and returns the line segment between q and the tangent point. The `isLeft(a, b, c)` function returns the polarity of `cross(<a,b>, <b,c>)`. Exponential search runs in O(log n).

3.

It is possible to report all positive slope lines using a line-sweep in the dual plane. Each point can be transformed to the dual plane as lines. These lines will produce a total of $\binom{n}{2}$ intersections (or the induced lines in the primal). We want to report the total number of these dual-plane intersections that have x-coordinates greater than zero (which will be all induce lines with positive slope).

Starting at $x = 0$ in the dual plane, we add all lines to the status. Iterating from top to bottom on this status, we can check each pair of lines. Two adjacent lines, top with positive slope, and bottom with negative slope can be ignored, since they will not intersect to the right of the status position. All other adjacent pairs need to be checked and intersections added to the event queue. At each event, swap lines and check against their neighbors.

So for n points, we have k induced lines (or k intersections in the dual). Each event takes $\mathrm{O}(\log n)$ using a balanced binary tree. Initially adding all the points to the status takes $\mathrm{O}(n\log n)$. Therefore our total runtime will be $\mathrm{O}(n\log n + k\log n) = \mathrm{O}((n+k)\log n)$. This is an improvement on the naive brute-force method, checking all $\binom{n}{2}$ pairs in $\theta(n^2)$.

4.

5.