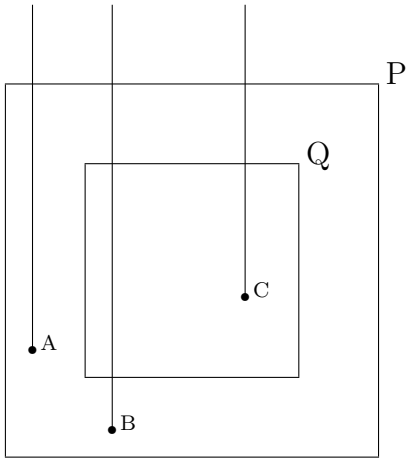
CLASSICAL AND QUANTUM COMPUTING
EXERCISE XIX

Suppose we have a polygon in 2 dimensions. The polygon can be represented by a sequence of points

$$(x_0, y_0) \to (x_1, y_1) \to \ldots \to (x_{n-1}, y_{n-1}) \to (x_0, y_0).$$

For a polygon it is not allowed that the lines connecting these point intersect. Given a point (x, y) how can we determine if the point is inside the polygon? A simple way is to draw a straight line from the point (x, y) outwards in any direction continuing to infinity. If the line intersects the polygon an odd number of times, the point is inside the polygon. For example



Using a vertical line results in a simple description of an algorithm to determine whether a point is in a polygon. Thus point A is in P but not in Q, point B is in P but not in Q and point C is in both P and Q. We could interpret Q as a "hole" in P in which case we would say C is not in P.

A *bounding box* of an edge is the rectangle determined by the endpoints of the edge.

The algorithm is as follows

1. $crossings := 0$

2. *edge* := first edge in polygon

3. A := point to test

4. If both endpoints of *edge* are left of A goto 9.

5. If both endpoints of *edge* are right of A goto 9.

6. If both endpoints of *edge* are below A goto 9.

7. If both endpoints of *edge* are above A and there is an endpoint on either side of A, $crossings := crossings + 1$.

8. If A is contained within the bounding box of *edge* Let $(x_L, y_L) \rightarrow (x_R, y_R) := edge$, where (x_L, y_L) is the leftmost point of *edge*.

$$(x_A, y_A) := A$$

$$y_C := y_L + \frac{y_R - y_L}{x_R - x_L}(x_A - x_L).$$

   If $y_C > y_A$ $crossings := crossings + 1$.

9. If edges from the polygon remain to be tested *edge* := next edge of polygon and goto 4.

10. If *crossings* is odd the point is in the polygon, otherwise the point is outside the polygon.

Write a C++ or Java program which determines whether a given point is in a given two dimensional polygon.