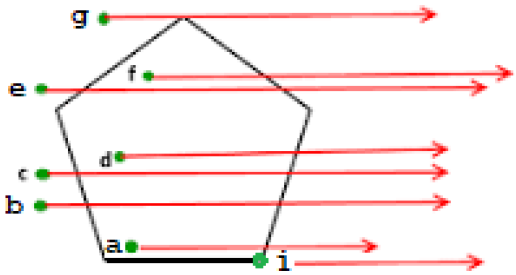
Inside or Outside

The motivation for this problem comes from this problem asked by Nvidia.

> You are given a list of `N` points `(x1, y1)`, `(x2, y2)`, ..., `(xN, yN)` representing a convex polygon. You can assume these points are given in order; that is, you can construct the polygon by connecting point `1` to point `2`, point `2` to point `3`, and so on, finally looping around to connect point `N` to point `1`.
> Determine if a new point `p` lies inside this polygon.
> Note: If `p` is on the boundary of the polygon, return false.

For example, in the figure to the right, the points a, d, and f all lie inside the polygon, the remaining points (including point i which lies on an edge) are all considered to lie outside the polygon.



My first step in solving this problem was to Google it, and according to:
https://www.geeksforgeeks.org/how-to-check-if-a-given-point-lies-inside-a-polygon/

1. *Draw a horizontal line to the right of the point and extend it to infinity.*
2. *Count the number of times the line intersects with polygon edges.*
3. *If either the count of intersections is even (remember, zero is even), or if the point lies on an edge of polygon, the point lies outside.*
4. *A point is inside the polygon if the count of intersections is odd and does NOT lie on an edge.*

To help with this problem, you have been given the completed `Point` class. The constructor for this class has two parameters, the first parameter is the x-coordinate and the second is the y-coordinate. It has accessor methods `getX()` that returns the x-value and `getY()` that returns the y-value.

The first method to implement is the static method `getSlopeIntercept(Point p1, Point p2)`, which given two points `p1` and `p2`, returns a `double[]` containing two values. The value in index 0 is the slope of the line containing the points `p1` and `p2`. The value in index 1 is the y-intercept of the line containing the points `p1` and `p2`. You may assume the line determined by `p1` and `p2` is not a vertical line.

The following code shows the results of the `getSlopeIntercept` method.

The following code	Returns
`double[] eq = InsideOrOutside.getSlopeIntercept(` `  new Point(1., 2.), new Point(3., 7.));`	
`eq[0]`	`2.5`
`eq[1]`	`-0.5`

Inside or Outside

The second method to implement is the static method `intersectsLineSegmentToRight(Point p, Point t1, Point t2)`, which returns a `boolean`. The method returns true if:

- The ray drawn horizontally to the right from the parameter `p` intersects the (interior of the) line segment connecting points `t1` and `t2`. That is, if the ray interests at the endpoints, the method returns false.
- And the Point `p` is not on the line segment connecting points `t1` and `t2`.

The following code shows the results of the `intersectsLineSegmentToRight` method.

The following code	Returns
`boolean doesInter = InsideOrOutside.intersectsLineSegmentToRight (new Point(1., 1.), new Point(0., 0.), new Point(6., 3.));`	
`doesInter`	`true`
`boolean doesInter = InsideOrOutside.intersectsLineSegmentToRight (new Point(3., 2.), new Point(1., 4.), new Point(2., 7.));`	
`doesInter`	`false`
`doesInter = InsideOrOutside.intersectsLineSegmentToRight (new Point(-2., 4.), new Point(1., 4.), new Point(2., 7.));`	
`doesInter`	`false`
`doesInter = InsideOrOutside.intersectsLineSegmentToRight(new Point(2., 5.), new Point(1., 4.), new Point(4., 7.));`	
`doesInter`	`false`

The third method is the (non static) method `numberEdgesIntersection(Point p)`, which returns the number of Polygon edges the Horizontal ray from `Point p` to the right intersects.

The following code shows the results of the `numberEdgesIntersection` method.

The following code	Returns
`Point[] poly1 = { new Point(1,1), new Point(2, 4), new Point(6, 6), new Point(7,5), new Point(3, 2)};`	
`InsideOrOutside ioo = new InsideOrOutside(poly1);`	
`ioo.numberEdgesIntersection(new Point(0, 2.5))`	2
`ioo.numberEdgesIntersection(new Point(3.0, 3.0))`	1
`ioo.numberEdgesIntersection(new Point(8.0, 5.0))`	0

Inside or Outside

The fourth (and final) method is the (non static) method `insidePolygon(Point p)`, which returns `true` if `Point p` is inside the polygon, otherwise, return `false`. Remember, if `Point p` lies on the boundary, return `false`.

The following code shows the results of the `insidePolygon` method.

The following code	Returns
`Point[] poly1 = { new Point(1,1), new Point(2, 4), new Point(6, 6), new Point(7,5), new Point(3, 2)};`	
`InsideOrOutside ioo = new InsideOrOutside(poly1);`	
`ioo.insidePolygon(new Point(0, 2.5));`	`false`
`ioo.insidePolygon(new Point(3.0, 3.0));`	`true`
`ioo.insidePolygon(new Point(8.0, 5.0));`	`false`

Inside or Outside

This page intentionally left blank