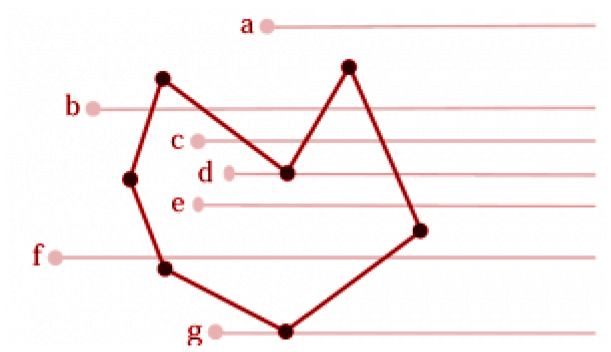
Check if a point lies in interior of a polygon

Problem Statement : You are given a polygon (not necessarily convex), and a point P is also given. We need to check whether the point is inside the polygon or outside the polygon.

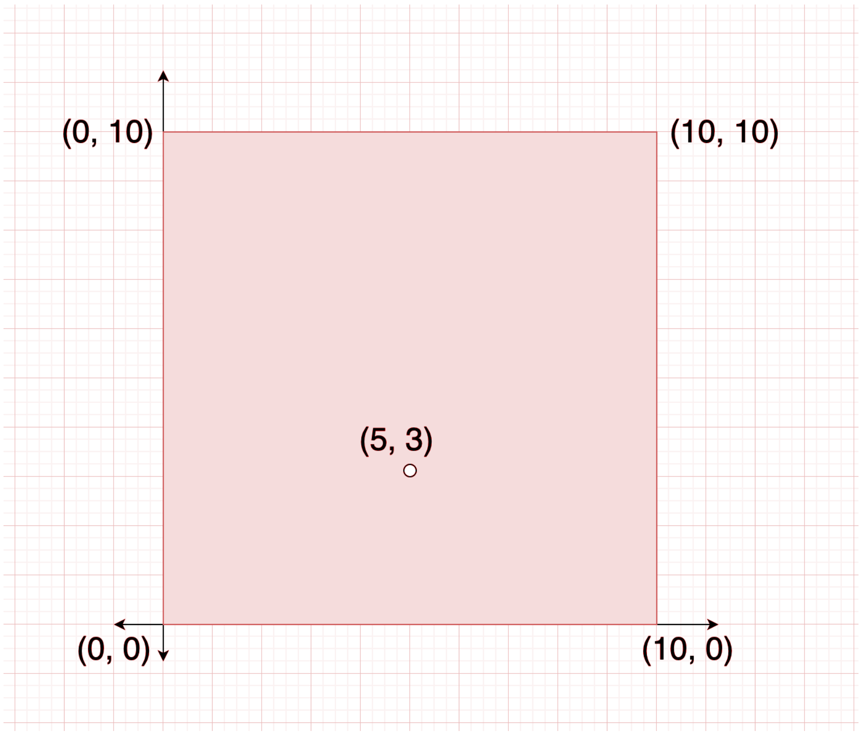


For solving it we will draw a straight line from the point P. It extends to the infinity. The line is horizontal, or it is parallel to the x-axis.
From that line, we will count how many times the line intersects the sides of a polygon.

- When the point is inside the polygon, it will intersect the sides an odd number of times.
- If P is placed on any side of the polygon, then it will cut an even number of times.
- If none of the conditions is true, then it is outside the polygon.

To visualize the above theorem you can look at the figure above.

Example :
Points of a polygon {(0, 0), (10, 0), (10, 10), (0, 10)}. And point P (5, 3) to check.
Output:
Point is inside.
This is because the given points form a square and all points with 0<=x<=10, 0<=y<=10 are inside the polygon. Also you can note that if we extend a line from (5, 3) horizontally to infinity then it intersects the right side of the square only, hence the intersections are odd (only 1) which proves the correctness of the given lemma.



So the Algorithm can be defined as below :

Algorithm

- If n < 3, return false because no polygon is possible
- Make a point at positive infinity that will be the other end of the horizontal ray from point p
- Now we count the total number of intersection points of the sides of the polygon with this horizontal ray.
- For each side of the polygon, check if the point lies on the segment: if it does, return true immediately, else check if the 2 lines intersect and increase the count.
- Finally, return true if the number of intersections is odd, else return false.

Pseudocode:

```
//   Returns true if the point p lies inside the polygon[] with n vertices
function isInside(Point polygon[], n, p)

        //   There must be at least 3 vertices in polygon[]
        if (n < 3) return false

        //   Create a point for line segment from p to infinite
        Point pInf(INF, p.y)

        //   count intersections of the above line with sides of polygon
```

```
int intersections = 0, i = 0
do

    int next = (i+1) % n

    /*
        Check if the line segment from 'p' to 'pInf' intersects
        with the line segment from 'polygon[i]' to 'polygon[next]'
    */

    if (Intersects(polygon[i], polygon[next], p, pInf))

        /*
            If the point 'p' lies on segment i -> next
            then return true
        */

        if (onSegment(polygon[i], p, polygon[next]))
            return true

        intersections++

    i = next

while (i != 0)

//   Return true if intersections is odd, false otherwise
return (intersections % 2)
```

TIme complexity : O(N), where N is the total number of points in the input array. We iterate on all N sides and check if they are intersecting in O(1) time each.

Space complexity : O(1), since we use constant additional space to keep count of intersections.