

# Advanced Programming Practice

## Cross Product and Convex Hull

2022 Fall

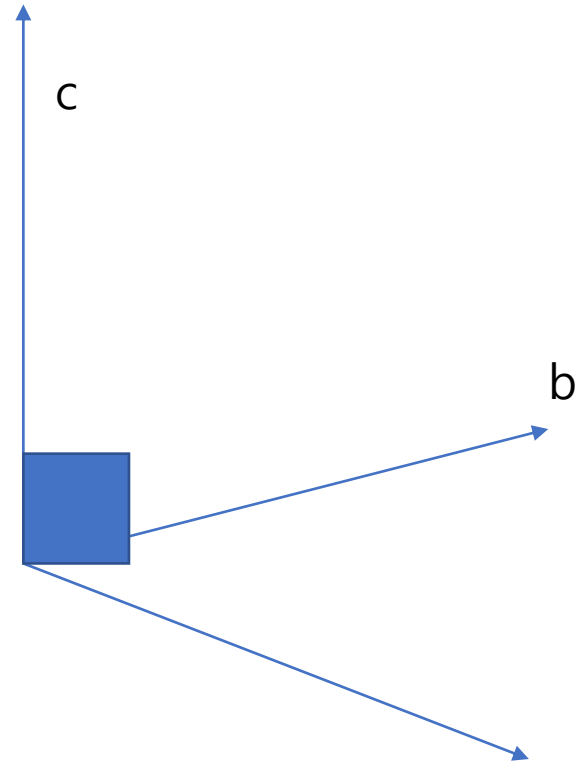
Sogang University



# Cross product

$$\mathbf{c} = \mathbf{a} \times \mathbf{b}$$

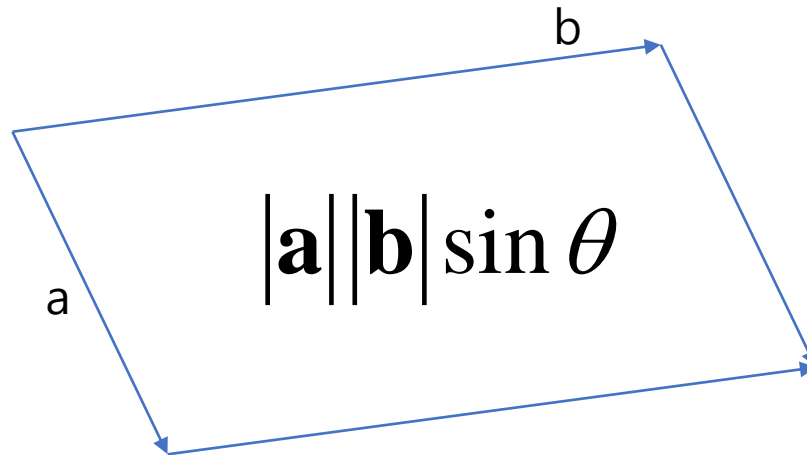
$$|\mathbf{c}| = |\mathbf{a}| |\mathbf{b}| \sin \theta$$



# Cross product

- An area of an triangle of two vectors  $\mathbf{a}$  and  $\mathbf{b}$

$$\frac{1}{2}|\mathbf{a}||\mathbf{b}|\sin\theta = \mathbf{a} \times \mathbf{b}$$



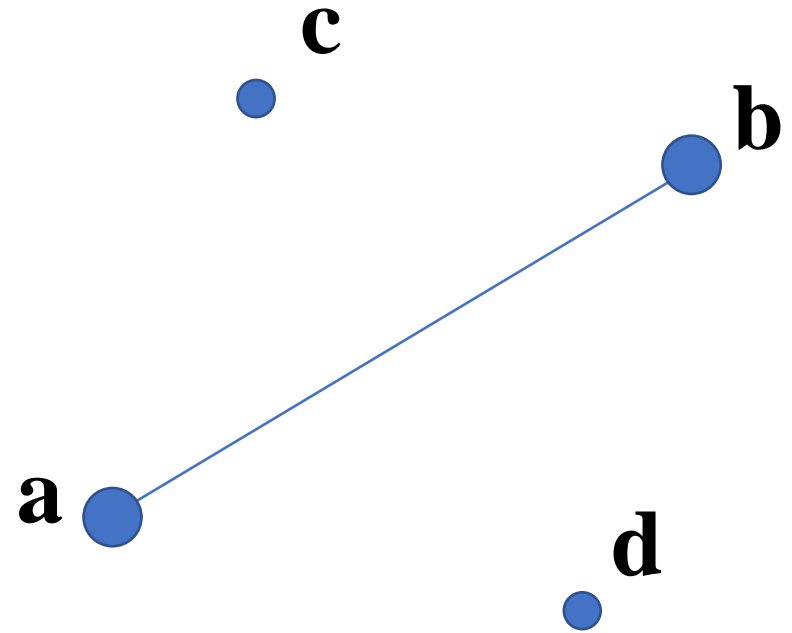
# Cross product

- Let vector **a** and **b** be on the **xy** plane.

$$\mathbf{c} = \mathbf{a} \times \mathbf{b} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{vmatrix} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ a_1 & a_2 & 0 \\ b_1 & b_2 & 0 \end{vmatrix} = (a_1 b_2 - a_2 b_1) \vec{k}$$

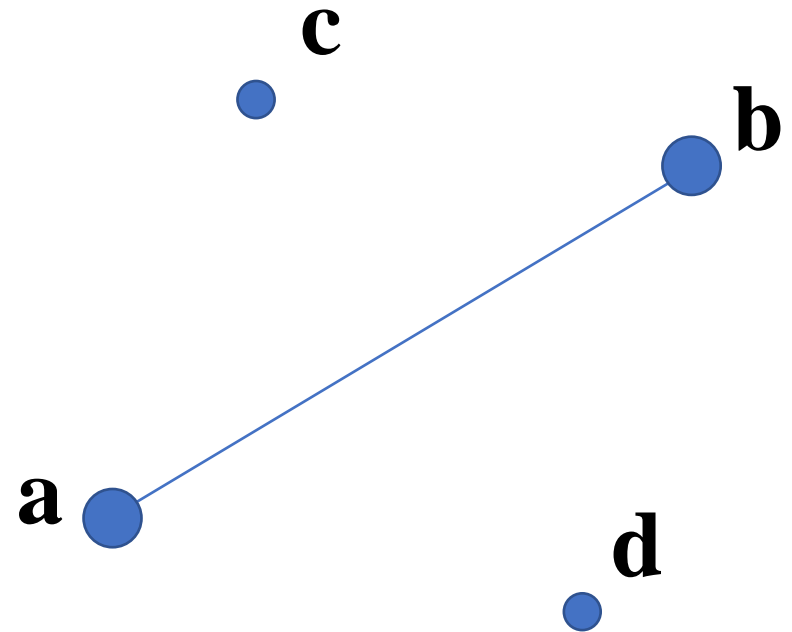
# Clockwise or Counter-clockwise

- Check whether points are on the clockwise or the counter-clockwise of line a-b with cross product.



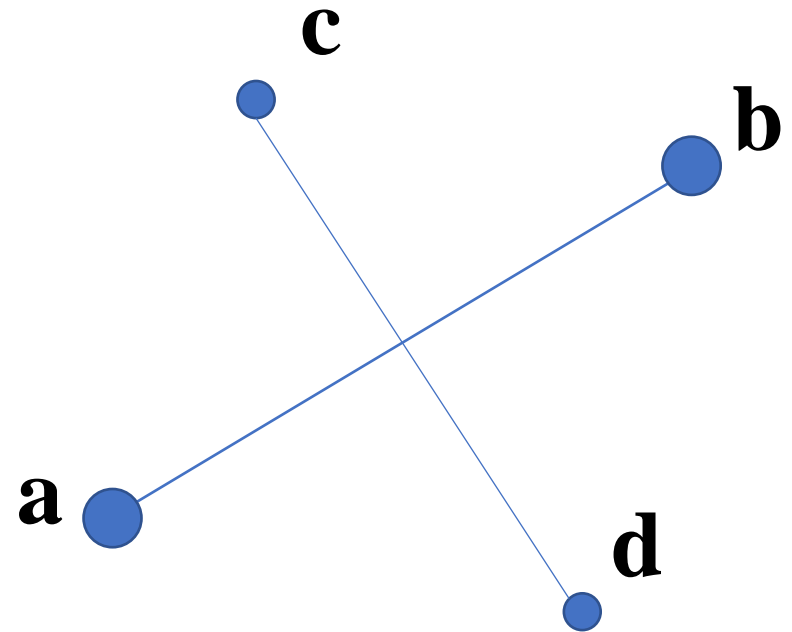
# Clockwise or Counter-clockwise

- Since the cross product  $(b-a) \times (c-a)$  directs to the forward of the display, point C is on the counter-clockwise.
- Since the cross product of  $(b-a) \times (d-a)$  goes through the backward of the display, point d is on the clockwise.



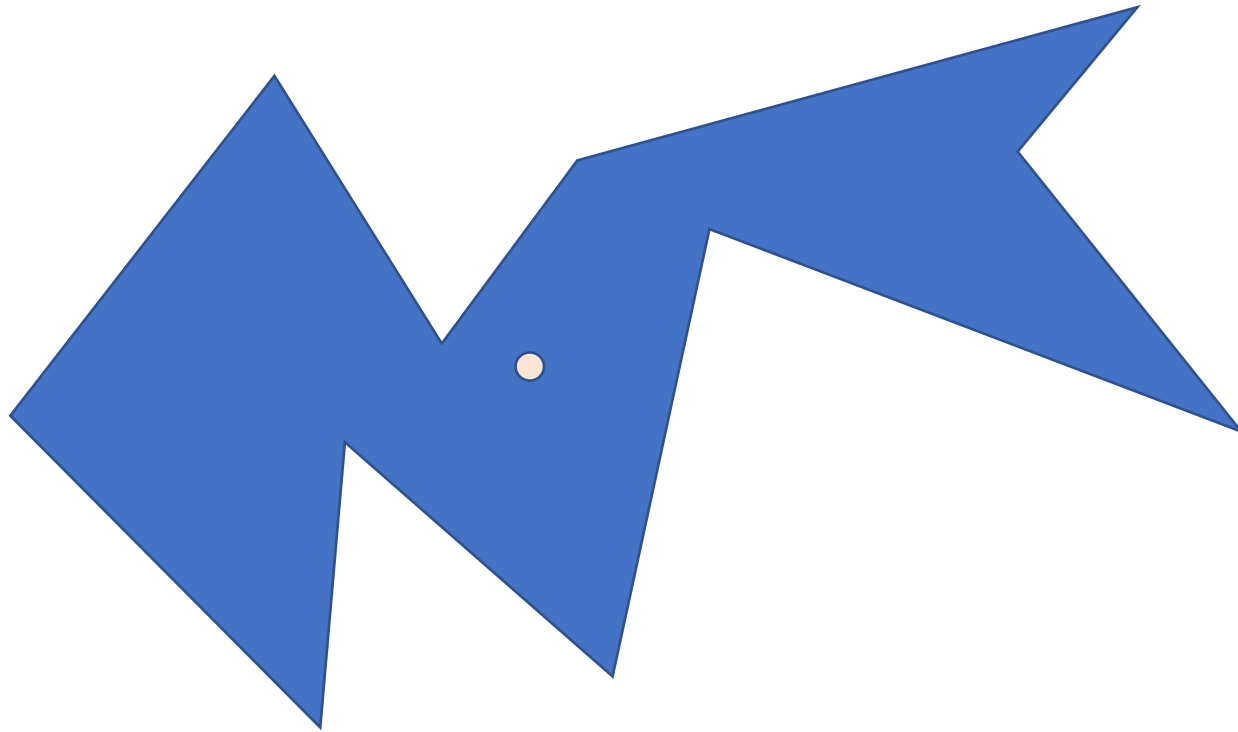
# Intersection

- Two lines are crossed when end points of each line are on different sides of the other line.



# Inner points

- How could we check whether a 2D point is in the polygon or not.



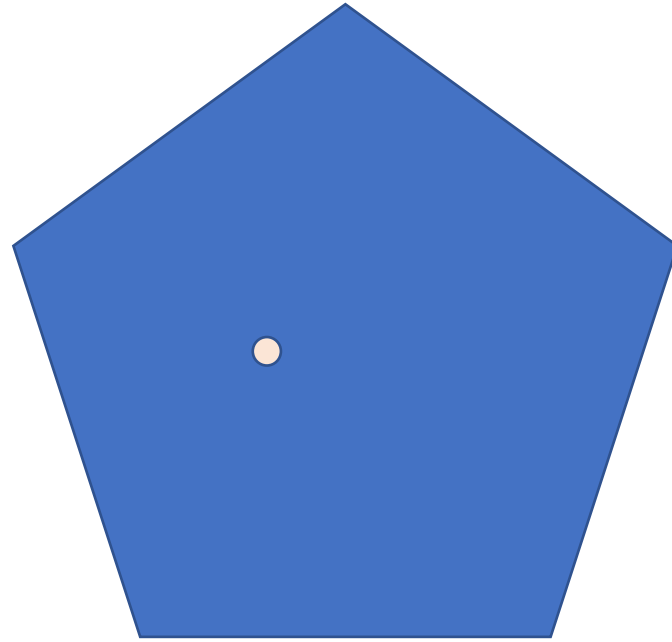


# Approach 1. only on the one side

- Approach 1: A point is inside of the polygon when the point is on one side (clockwise or counter-clockwise) of every polygon edges.

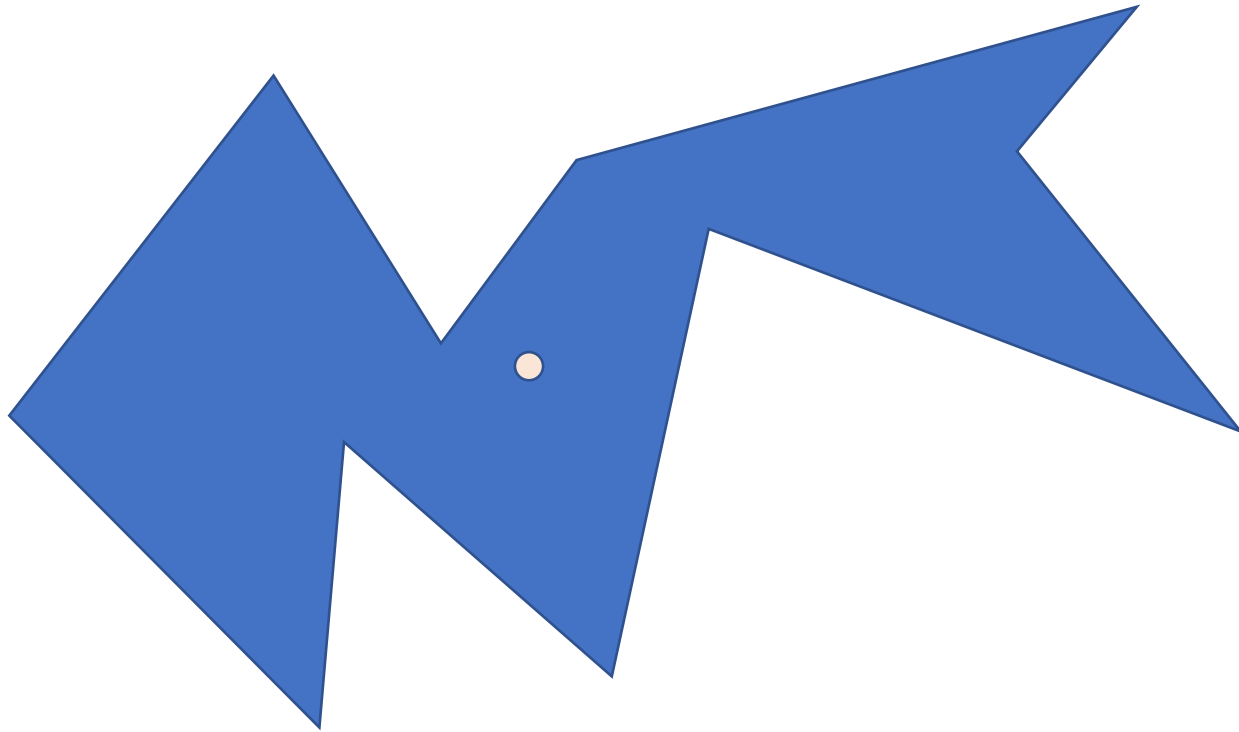
# Solving the problem

- Satisfied when the polygon is convex.



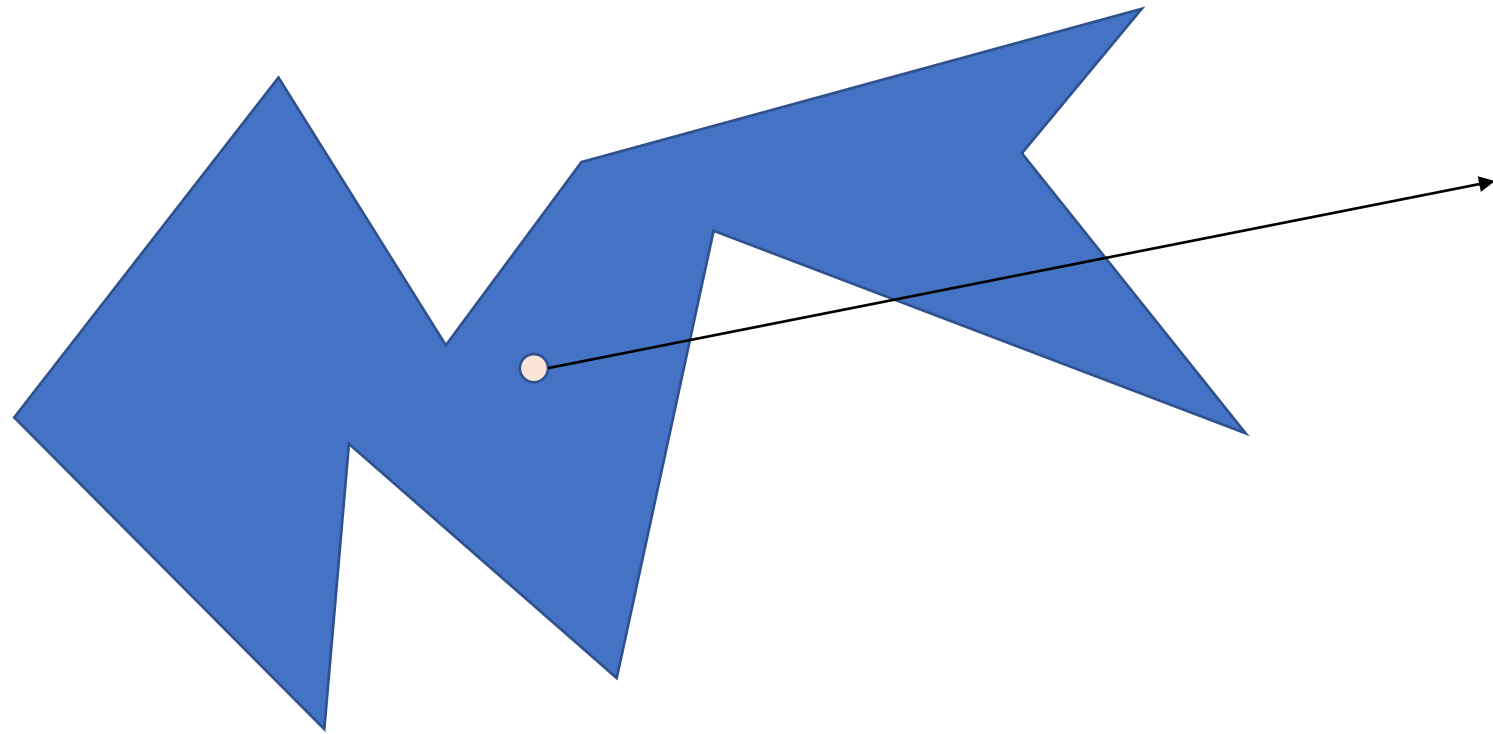
# Solving the problem

- Not satisfied for non-convex polygons.

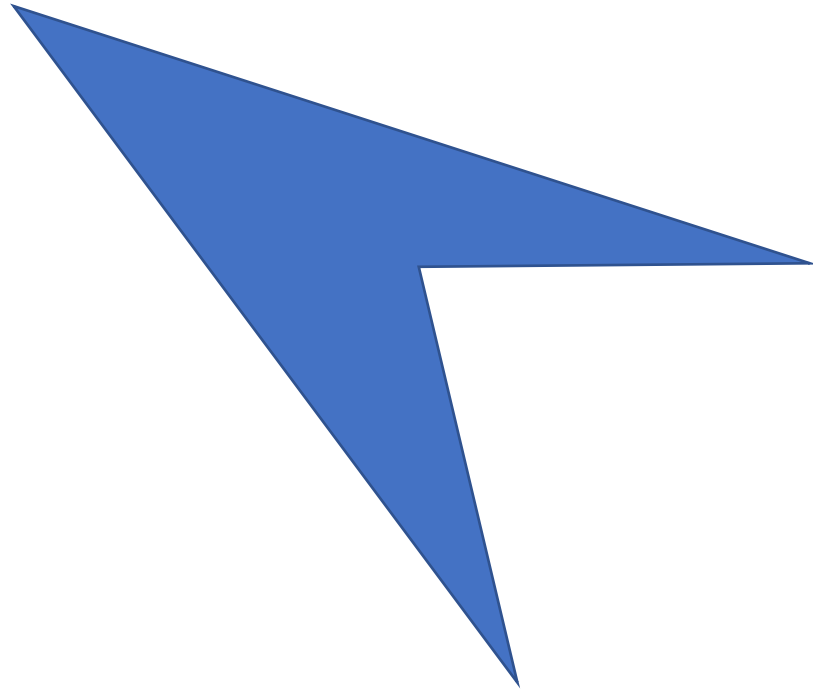


## Approach 2. draw a infinite line

- The infinite line starting from every inner point has the odd number of intersection points.

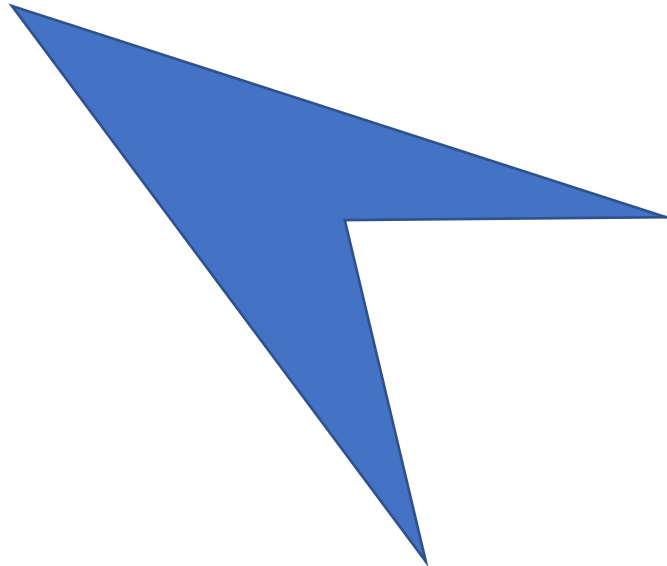


# Computing the area of a polygon



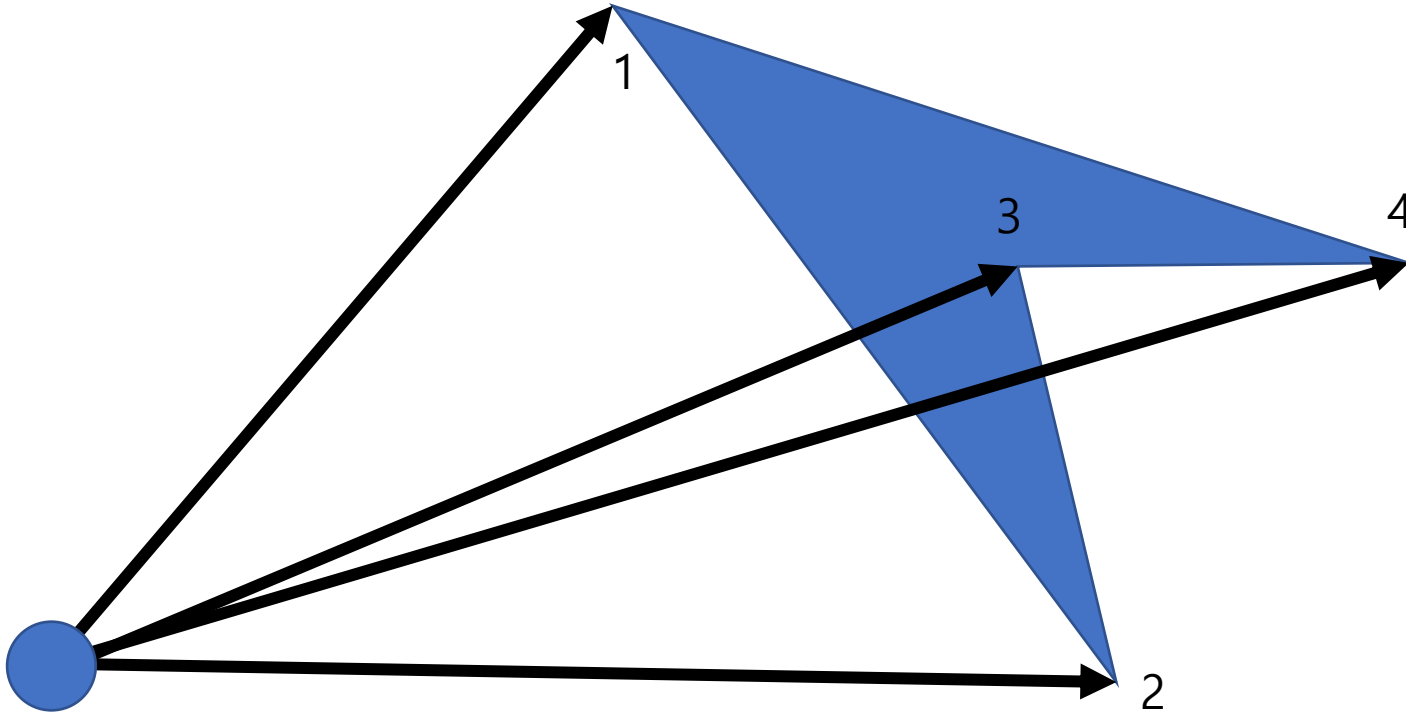
# Computing the area of a polygon (cont.)

- Sort vertices in counter-clock wise.
- Find triangles where  $(i+1)$  th vertex is on clockwise of the edge of the  $i$ -th node and  $i+1$ -th node and accumulate areas of triangles.
- Delete triangles and repeat this process again until no vertex is left.



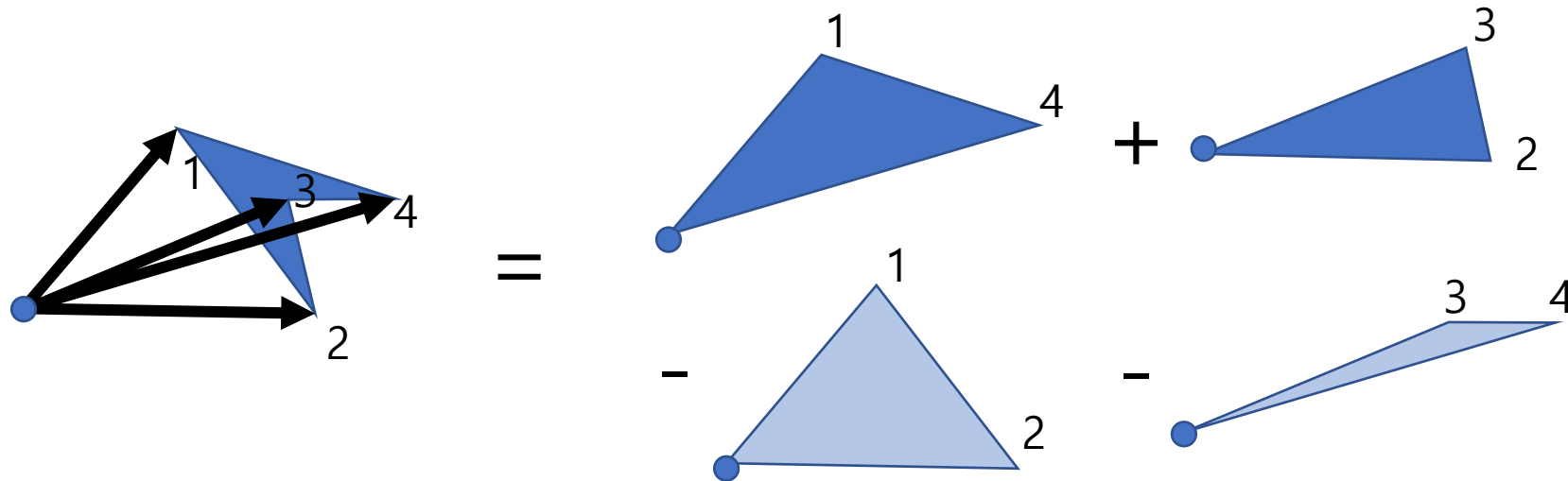
# Computing the area of a polygon (cont.)

- Add up all cross products of edges and divide it by 2.



# Computing the area of a polygon (cont.)

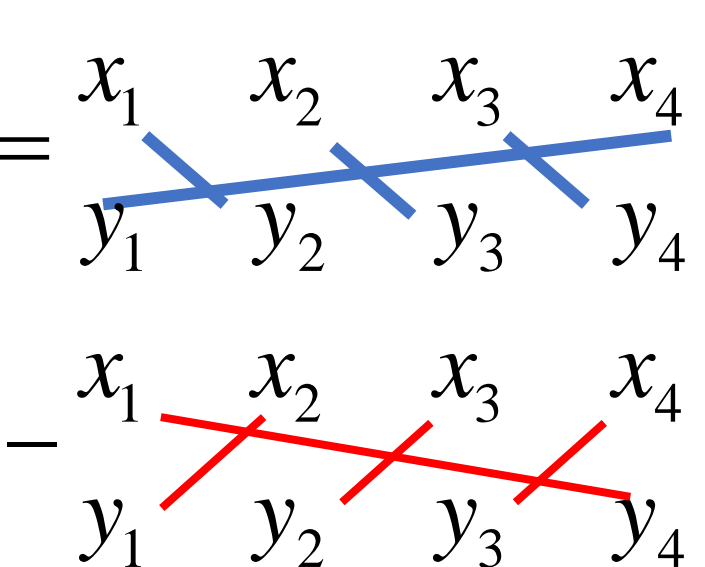
- Depending the direction of an edge, the area of the triangle is added or subtracted.
- Surprisingly, only the polygon area is left.





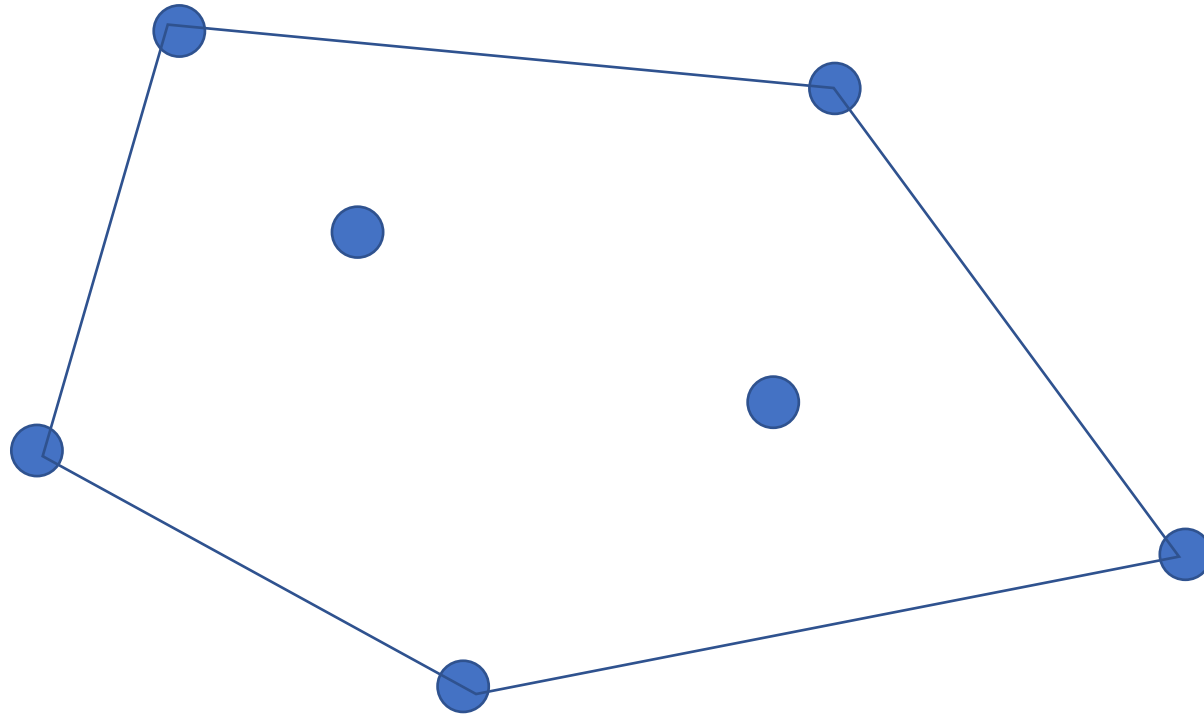
# Bootstrapping

- It is analogous to sum of cross products of edges.

$$\begin{aligned} & \mathbf{p}_1 \times \mathbf{p}_2 + \mathbf{p}_2 \times \mathbf{p}_3 + \mathbf{p}_3 \times \mathbf{p}_4 + \mathbf{p}_4 \times \mathbf{p}_1 \\ &= \begin{vmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{vmatrix} = \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{array} \\ & \quad \quad \quad - \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{array} \end{aligned}$$


# Computing a convex hull

- 2D points are given.



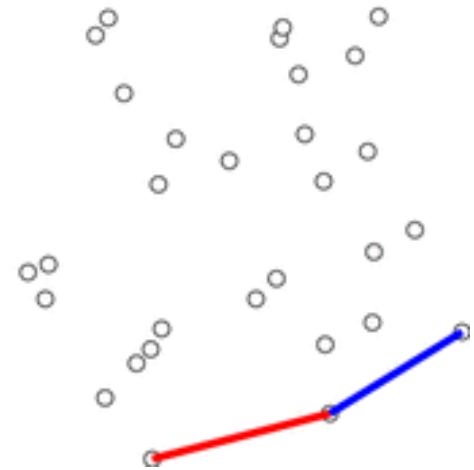
# Graham Scan

- Algorithm to find a convex hull.

```
let points be the list of points
let stack = empty_stack()

find the lowest y-coordinate and leftmost point, called P0
sort points by polar angle with P0

for point in points:
    # pop the last point from the stack if we turn clockwise to reach this point
    while count stack > 1 and ccw(next_to_top(stack), top(stack), point) <= 0:
        pop stack
    push point to stack
end
```



# Assignment

# Computing the area

For given N 2D points that composes a polygon, compute the area of the polygon.

Input

5 // N points

0 0 // (x1, y1)

2 0 // (x2, y2)

2 2

1 1

0 2

Output

3

# Checking whether inside or outside of the polygon

For given N 2D points that composes a polygon and M 2D points, please check whether each point is in the polygon or not.

Points on lines is considered as inside.

## Input

5 // N points

0 0 // (x1, y1)

2 0 // (x2, y2)

2 2

1 1

0 2

2

0 0

0.5 0.5

-1 -1

## Output

Inside

Inside

Outside

# Computing the area of the convex hull

For given N 2D points, compute a convex hull and its area.

Input

5 // N points

0 0 // (x1, y1)

2 0 // (x2, y2)

2 2

1 1

0 2

Output

4

(0, 0), (2, 0), (2, 2), and (0, 2) are points of convex hull.