Carleton University

COMP 2402

Convex Hulls

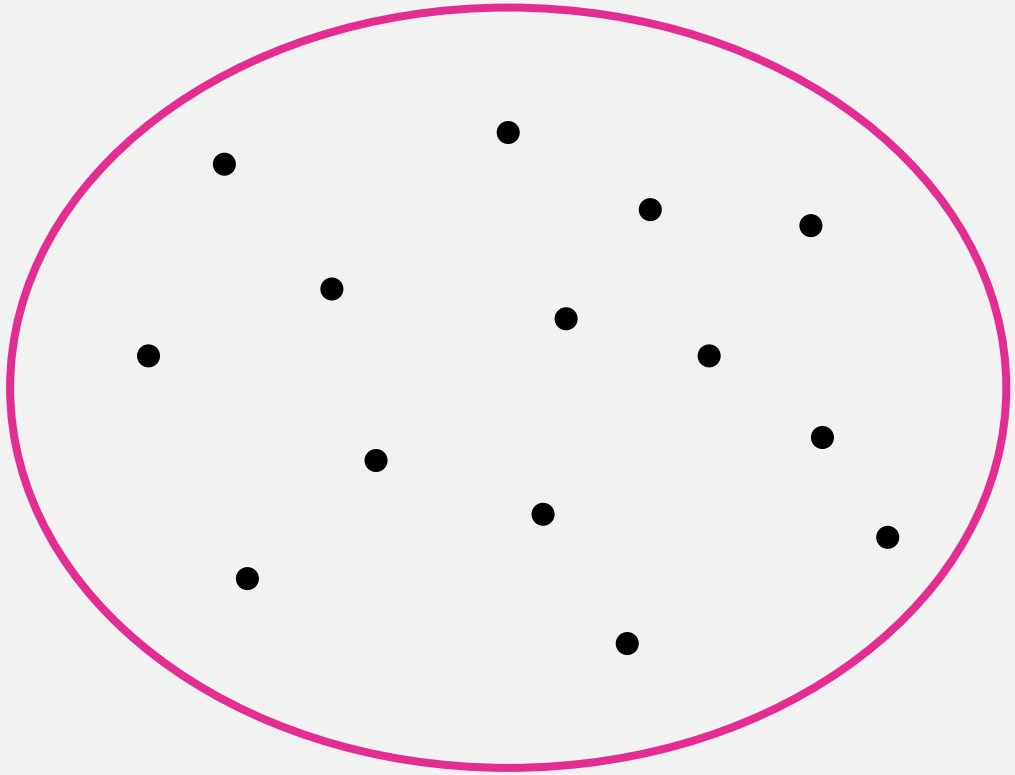
An Application of Stacks and Deques

Alina Shaikhet

Convex Hull

Let $P = \{p_0, p_1, \ldots, p_{n-1}\}$ be a set of points in the plane.

The **convex hull** of P is the **smallest** convex region (polygon) that contains P.

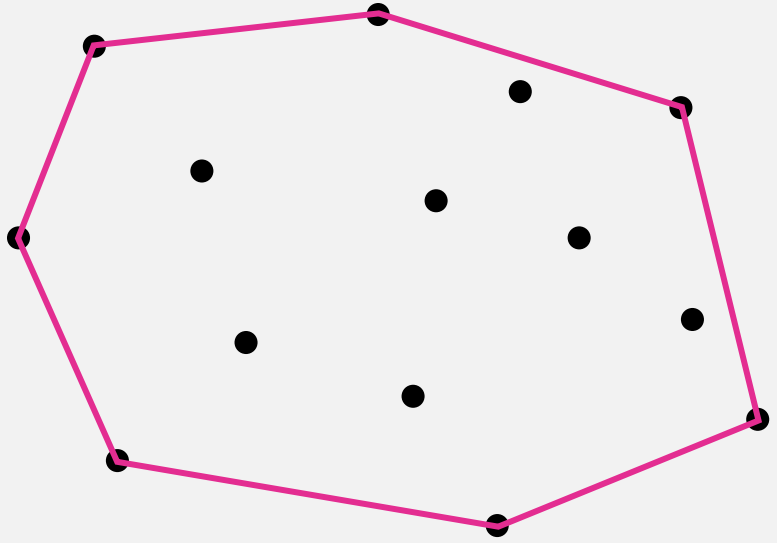


stretch a rubber band around P and let it go

Convex Hull

Let $P = \{p_0, p_1, \ldots, p_{n-1}\}$ be a set of points in the plane.

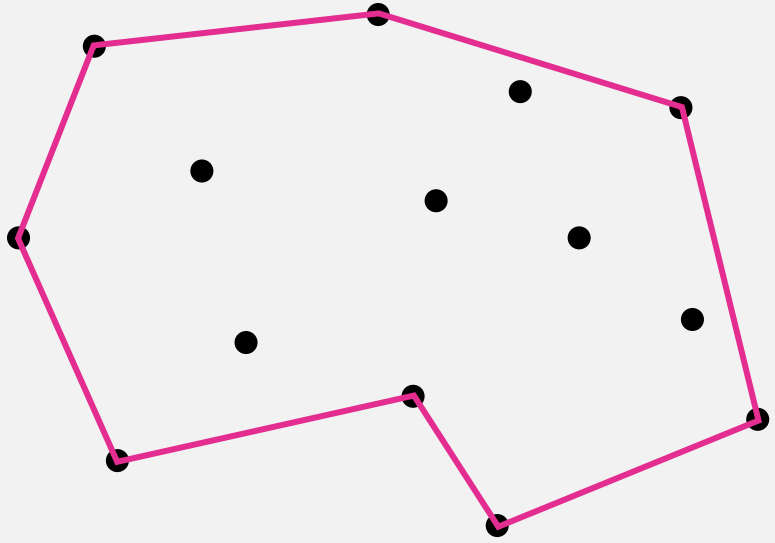
The **convex hull** of P is the **smallest** convex region (polygon) that contains P.

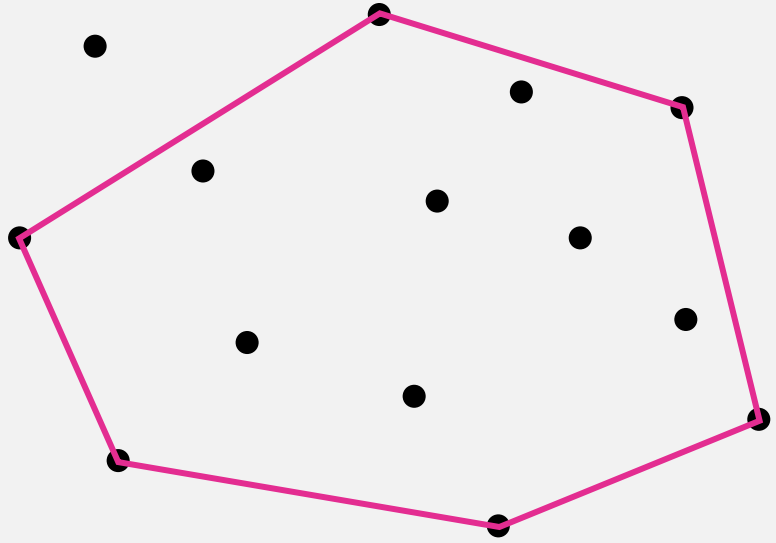


stretch a rubber band around P and let it go

– the region it snaps to is a convex hull of P

Is this a Convex Hull?





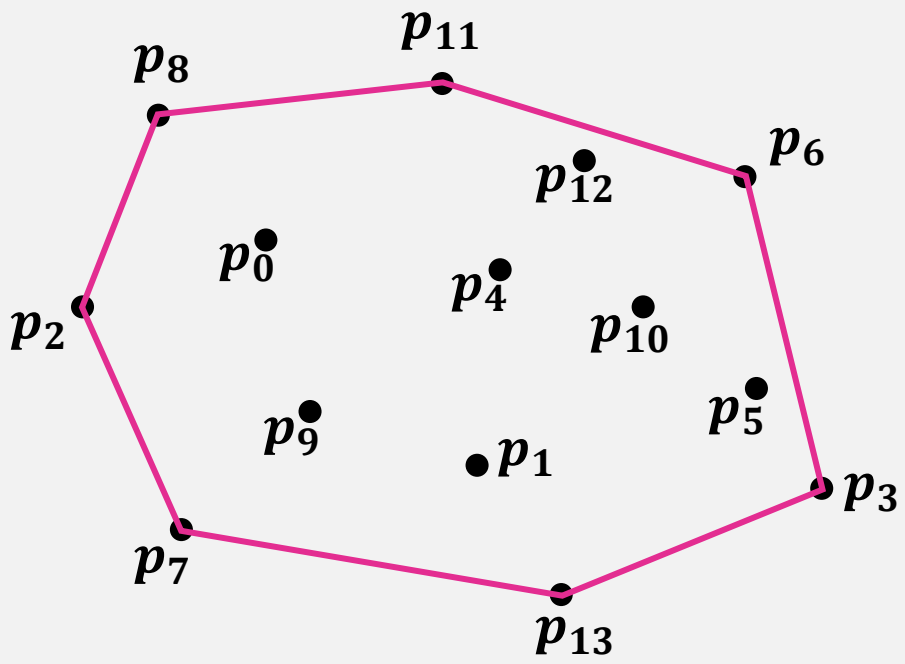
Point:

float x;
float y;

How can computer solve it?

Input: A set $P = \{p_0, p_1, \ldots, p_{n-1}\}$ of n points (unordered)

Output: A list of the points of P on the convex hull in **clockwise order**



$\{p_2, p_8, p_{11}, p_6, p_3, p_{13}, p_7\}$

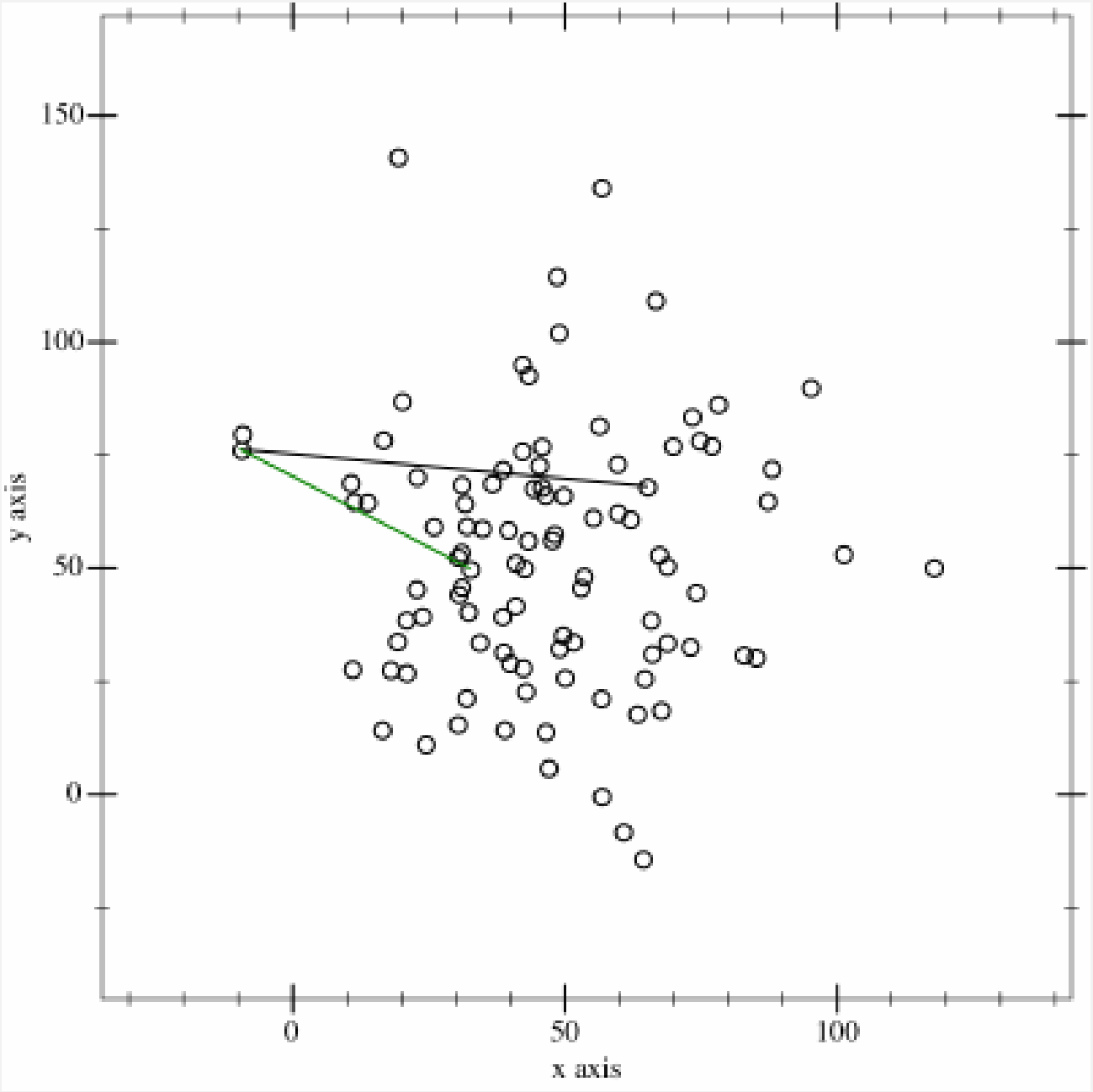
Jarvis march

Gift Wrapping Algorithm

1973

$$O(nh)$$

h is the number of points on the convex hull

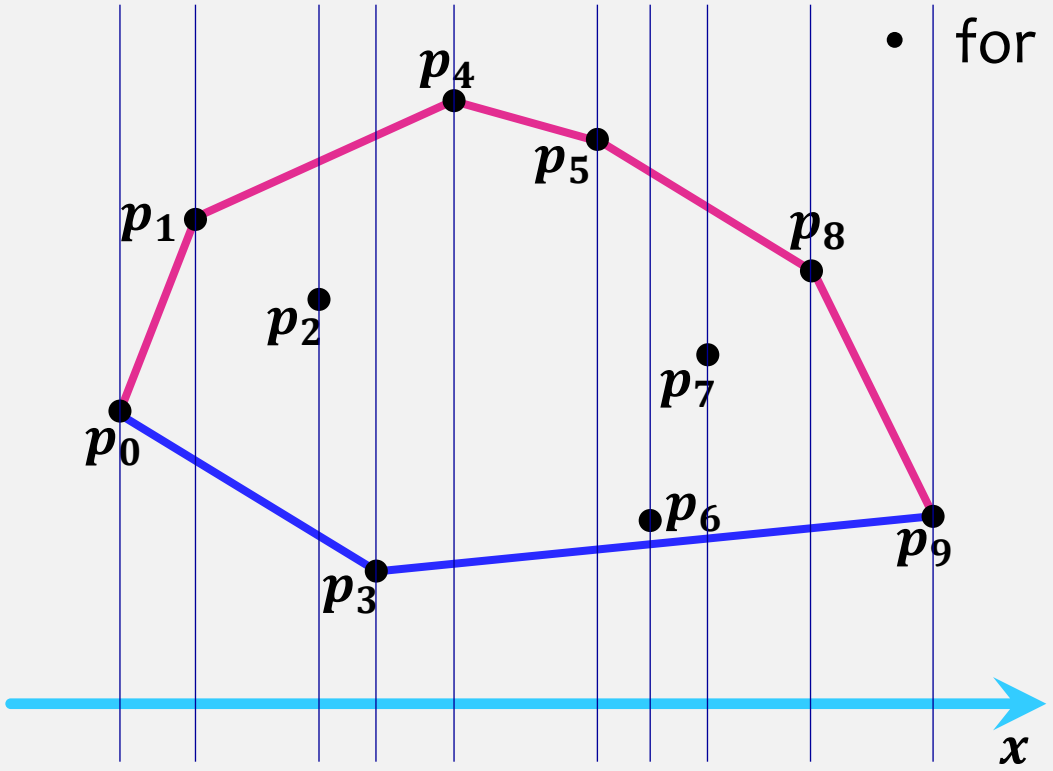


https://en.wikipedia.org/wiki/Gift_wrapping_algorithm

Graham's Scan
1972

p_0 is the point with the smallest x-coordinate (if there are ties – take the lowest one)

- Sort the points by x-coordinate
- Construct the **upper hull** incrementally using a **stack**:



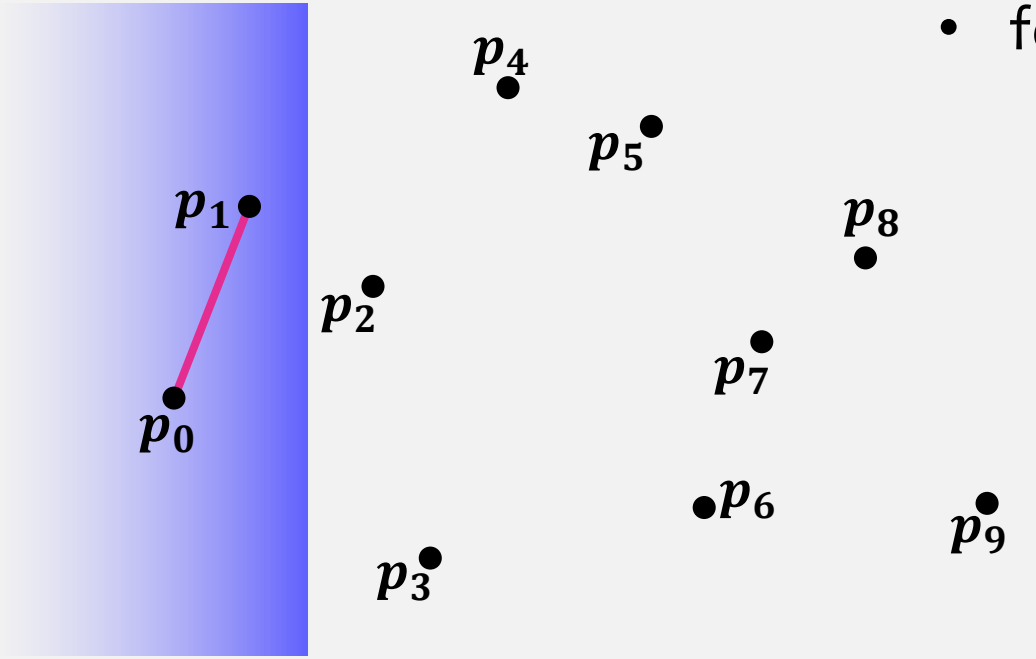
- Create a stack s containing p_0 and p_1
- for $(i = 2; i < n; i++)$
  add p_i to the convex hull of $\{p_0, p_1, \dots, p_{i-1}\}$:

```
while p_i is "above" the line (p_{i-2}, p_{i-1})
    s.pop();
s.push(p_i);
```

points with the largest and the smallest x-coordinate are always on the upper hall

Graham's Scan

- Sort the points by x-coordinate
- Construct the **upper hull** incrementally using a **stack**:



- Create a stack s containing p_0 and p_1
- for $(i = 2; i < n; i++)$
  add p_i to the convex hull of $\{p_0, p_1, \ldots, p_{i-1}\}$:

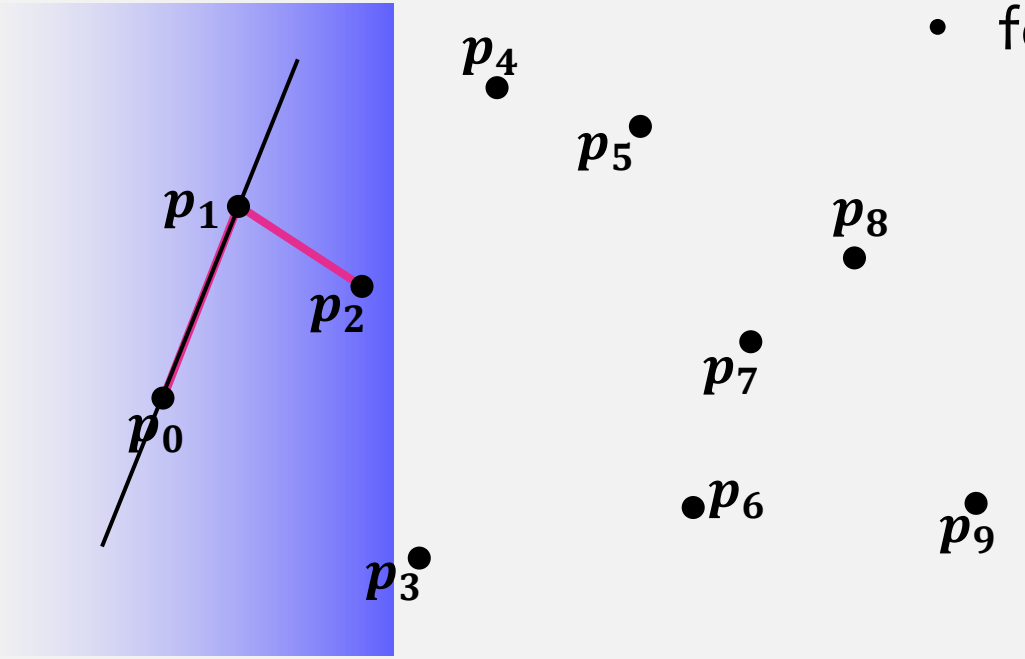
```
while p_i is "above" the line (p_{i-2}, p_{i-1})
    s.pop();
s.push(p_i);
```

points with the largest and the smallest x-coordinate are always on the upper hall

$$s = \langle p_0, p_1 \rangle$$

Graham's Scan

- Sort the points by x-coordinate
- Construct the **upper hull** incrementally using a **stack**:



- Create a stack s containing p_0 and p_1
- for $(i = 2; i < n; i++)$
  add p_i to the convex hull of $\{p_0, p_1, \ldots, p_{i-1}\}$:

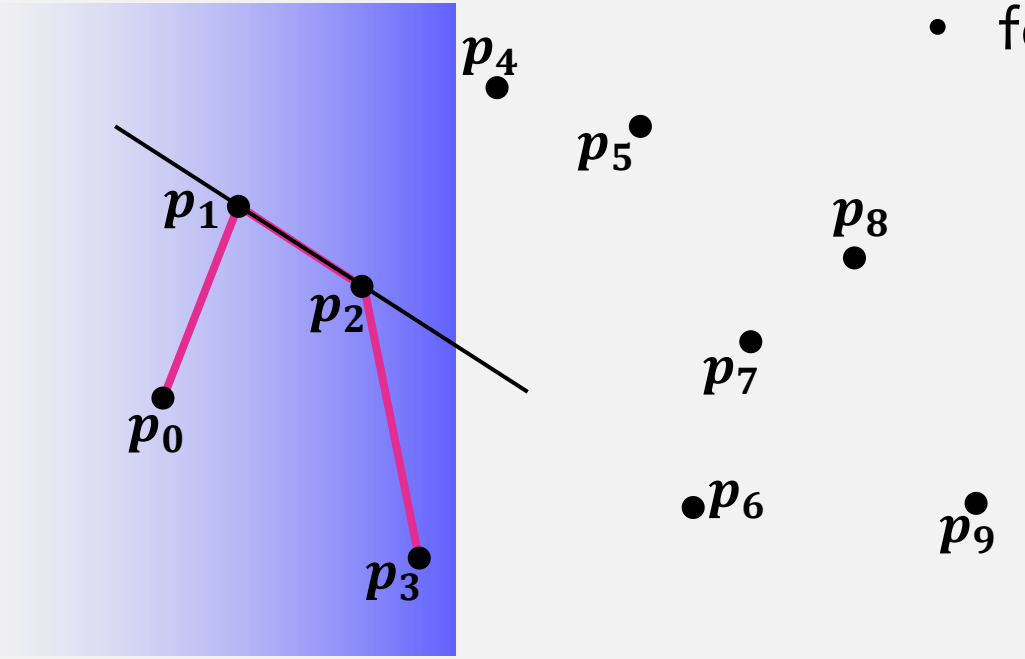
```
while p_i is "above" the line (p_{i-2}, p_{i-1})
    s.pop();
s.push(p_i);
```

points with the largest and the smallest x-coordinate are always on the upper hall

$$s = \langle p_0, p_1 \rangle$$

Graham's Scan

- Sort the points by x-coordinate
- Construct the **upper hull** incrementally using a **stack**:



- Create a stack s containing p_0 and p_1
- for $(i = 2; i < n; i++)$
  add p_i to the convex hull of $\{p_0, p_1, \ldots, p_{i-1}\}$:

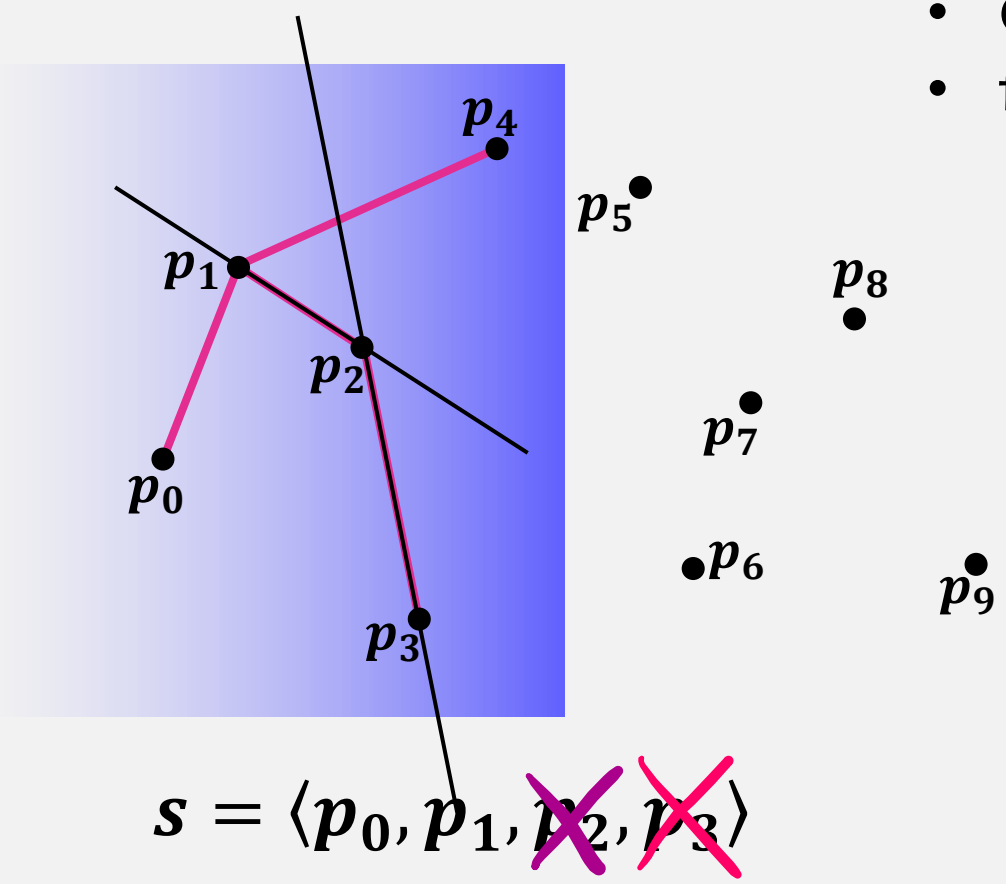
```
while p_i is "above" the line (p_{i-2}, p_{i-1})
    s.pop();
s.push(p_i);
```

points with the largest and the smallest x-coordinate are always on the upper hall

$$s = \langle p_0, p_1, p_2 \rangle$$

Graham's Scan

- Sort the points by x-coordinate
- Construct the **upper hull** incrementally using a **stack**:



$s = \langle p_0, p_1, p_2, p_3 \rangle$

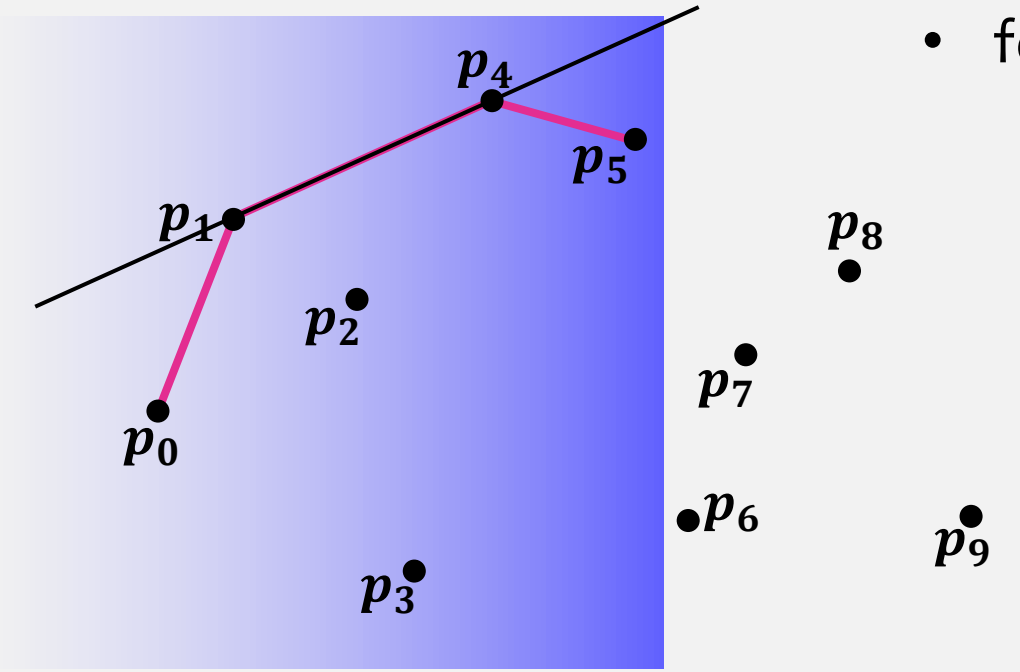
- Create a stack s containing p_0 and p_1
- for $(i = 2; i < n; i++)$
  add p_i to the convex hull of $\{p_0, p_1, \ldots, p_{i-1}\}$:

```
while p_i is "above" the line (p_{i-2}, p_{i-1})
    s.pop();
s.push(p_i);
```

points with the largest and the smallest x-coordinate are always on the upper hall

Graham's Scan

- Sort the points by x-coordinate
- Construct the **upper hull** incrementally using a **stack**:



- Create a stack s containing p_0 and p_1
- for $(i = 2; i < n; i++)$
  add p_i to the convex hull of $\{p_0, p_1, \ldots, p_{i-1}\}$:

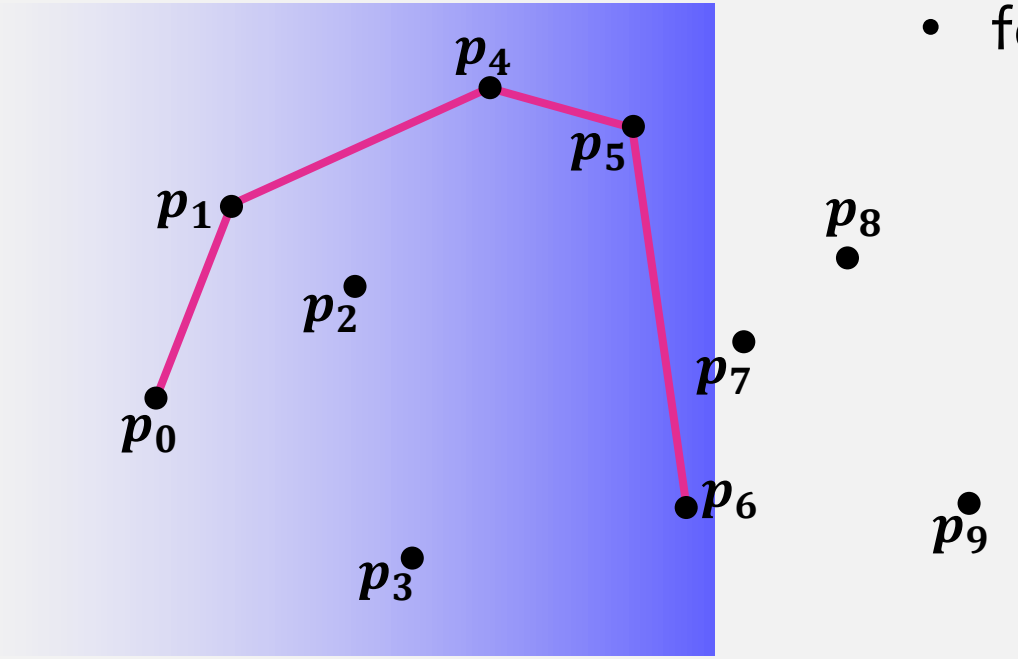
```
while p_i is "above" the line (p_{i-2}, p_{i-1})
    s.pop();
s.push(p_i);
```

points with the largest and the smallest x-coordinate are always on the upper hall

$$s = \langle p_0, p_1, p_4 \rangle$$

Graham's Scan

- Sort the points by x-coordinate
- Construct the **upper hull** incrementally using a **stack**:



- Create a stack s containing p_0 and p_1
- for $(i = 2; i < n; i++)$
  add p_i to the convex hull of $\{p_0, p_1, \ldots, p_{i-1}\}$:

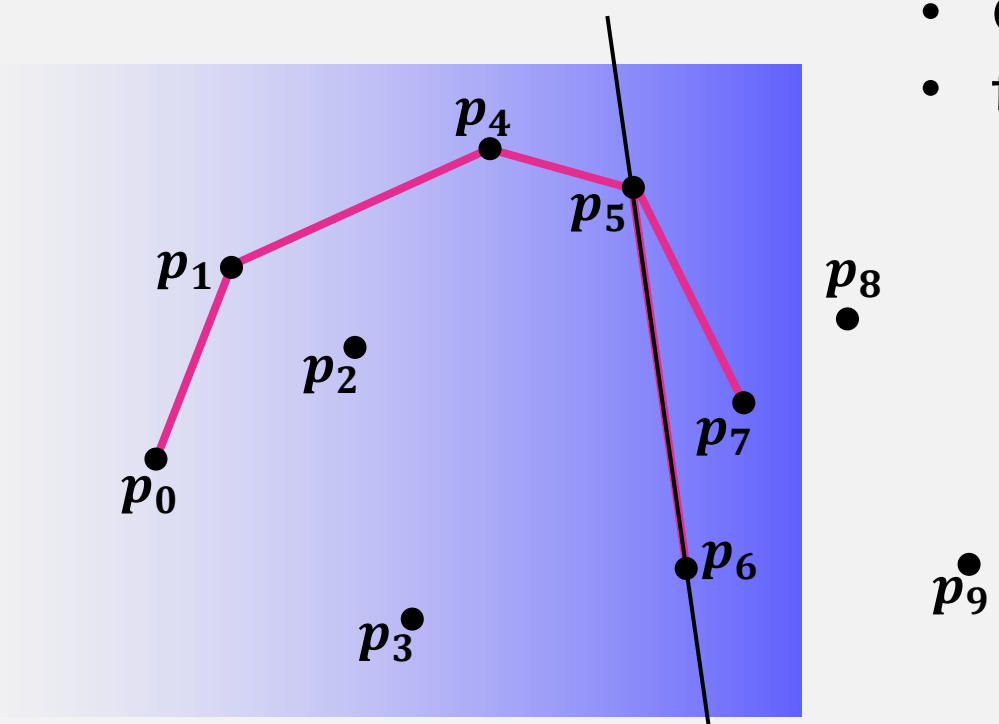
```
while p_i is "above" the line (p_{i-2}, p_{i-1})
    s.pop();
s.push(p_i);
```

points with the largest and the smallest x-coordinate are always on the upper hall

$$s = \langle p_0, p_1, p_4, p_5 \rangle$$

Graham's Scan

- Sort the points by x-coordinate
- Construct the **upper hull** incrementally using a **stack**:



$s = \langle p_0, p_1, p_4, p_5, p_6 \rangle$

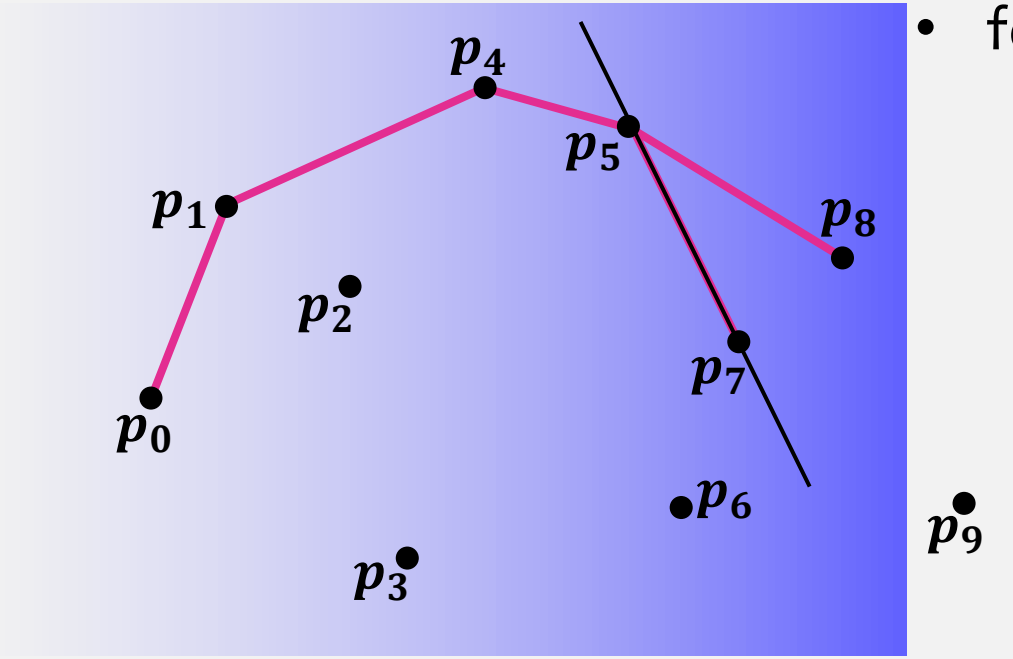
- Create a stack s containing p_0 and p_1
- for $(i = 2; i < n; i++)$
  add p_i to the convex hull of $\{p_0, p_1, \ldots, p_{i-1}\}$:

```
while p_i is "above" the line (p_{i-2}, p_{i-1})
    s.pop();
s.push(p_i);
```

points with the largest and the smallest x-coordinate are always on the upper hall

Graham's Scan

- Sort the points by x-coordinate
- Construct the **upper hull** incrementally using a **stack**:



- Create a stack s containing p_0 and p_1
- for $(i = 2; i < n; i++)$
  add p_i to the convex hull of $\{p_0, p_1, \ldots, p_{i-1}\}$:

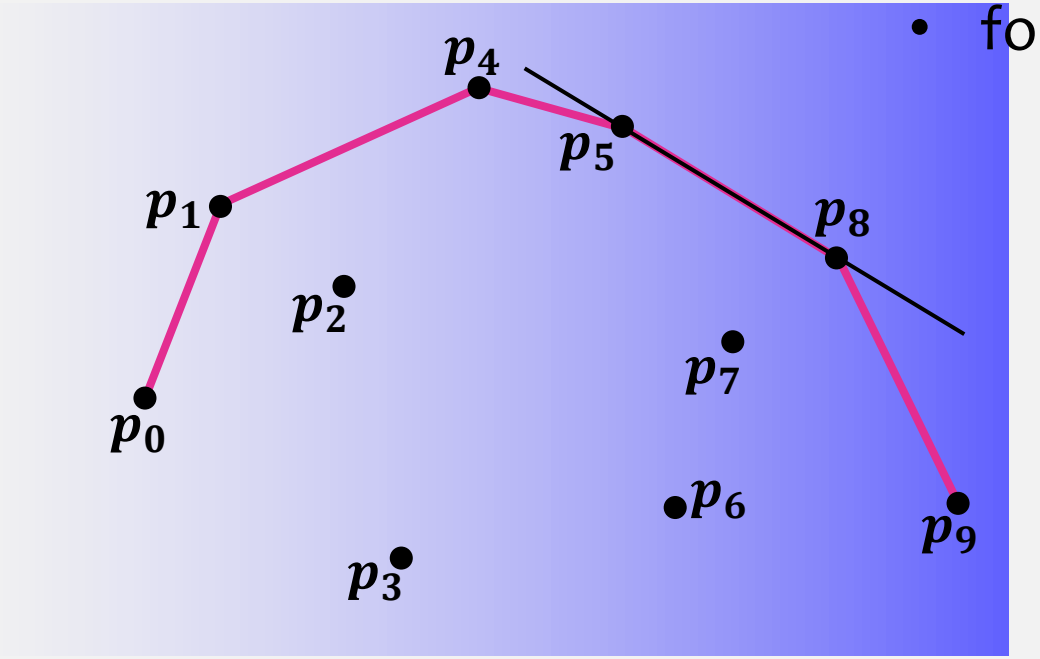
```
while p_i is "above" the line (p_{i-2}, p_{i-1})
    s.pop();
s.push(p_i);
```

points with the largest and the smallest x-coordinate are always on the upper hall

$$s = \langle p_0, p_1, p_4, p_5, p_7 \rangle$$

Graham's Scan

- Sort the points by x-coordinate
- Construct the **upper hull** incrementally using a **stack**:



- Create a stack s containing p_0 and p_1
- for $(i = 2; i < n; i++)$
  add p_i to the convex hull of $\{p_0, p_1, \ldots, p_{i-1}\}$:

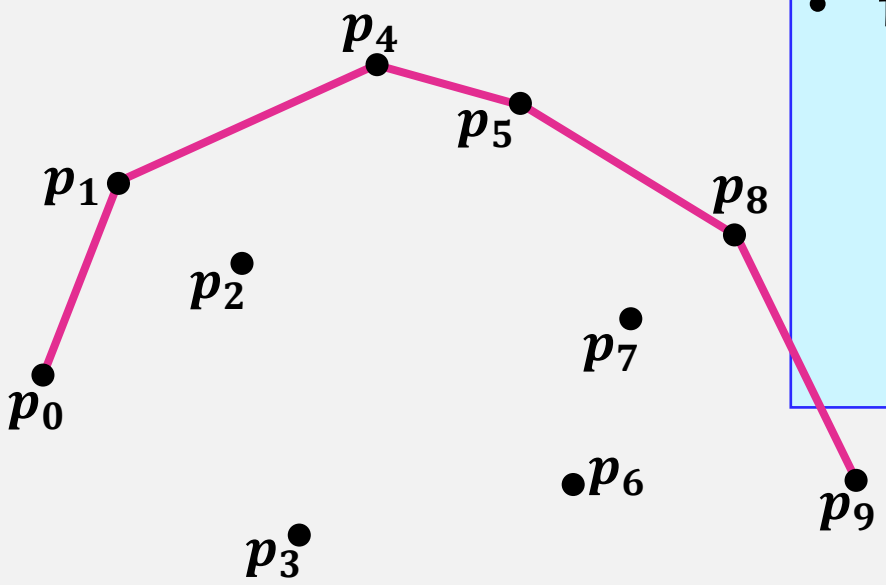
```
while p_i is "above" the line (p_{i-2}, p_{i-1})
    s.pop();
s.push(p_i);
```

points with the largest and the smallest x-coordinate are always on the upper hall

$$s = \langle p_0, p_1, p_4, p_5, p_8 \rangle$$

Graham's Scan

- Sort the points by x-coordinate $\boldsymbol{O(n \log n)}$
- Construct the **upper hull** incrementally using a **stack**:



$\boldsymbol{O(n)}$

- Create a stack s containing p_0 and p_1
- for $(i = 2; i < n; i++)$
  add p_i to the convex hull of $\{p_0, p_1, \ldots, p_{i-1}\}$:

  while p_i is "above" the line (p_{i-2}, p_{i-1})
    s.pop(); ← $\leq n - 2$ times
  s.push(p_i);

points with the largest and the smallest x-coordinate are always on the upper hall

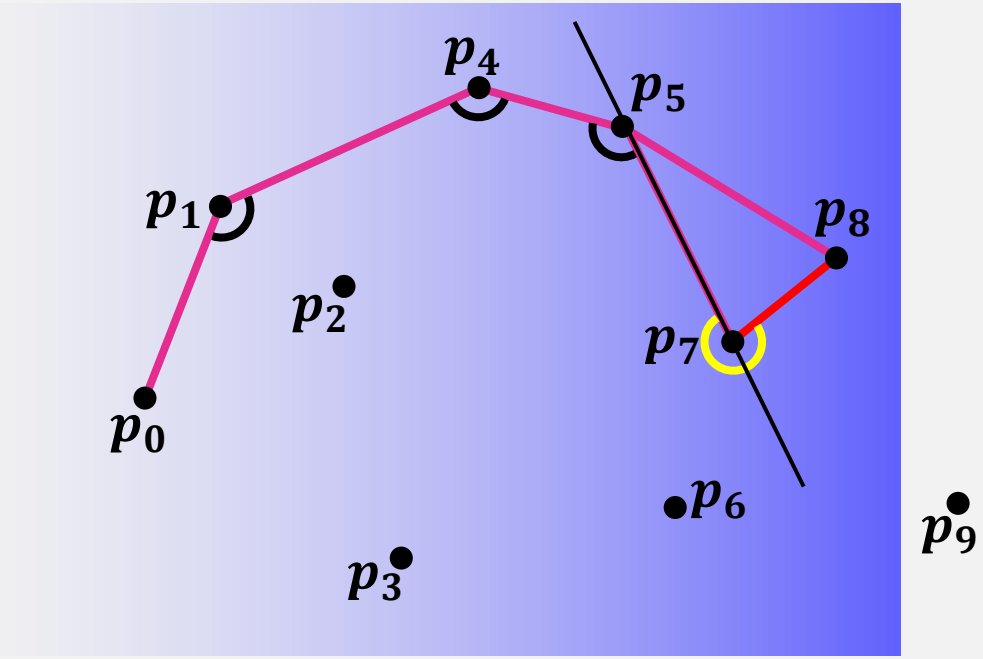
$s = \langle p_0, p_1, p_4, p_5, p_8, p_9 \rangle$

Graham's Scan computes the convex hull of an n-point set in $\boldsymbol{O(n \log n)}$

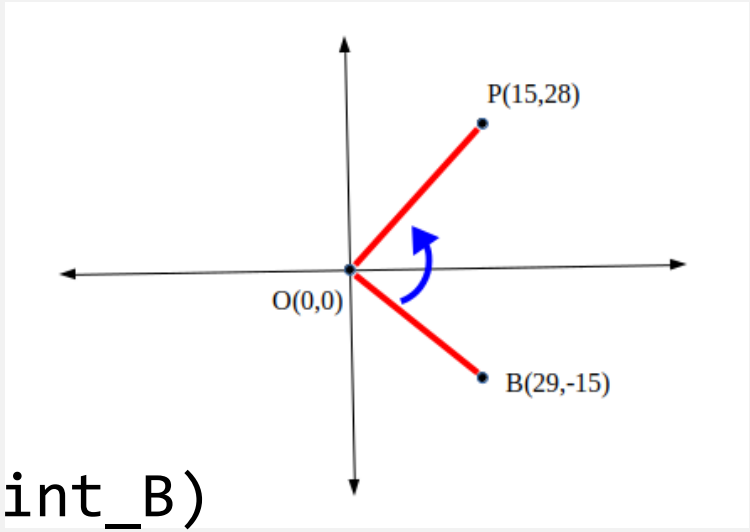
Coding Details

How can computer check if p_i is "above" the line (p_{i-2}, p_{i-1})?

Cross-Product of two points $A(A_x, A_y)$ and $B(B_x, B_y)$ is: $A_x * B_y - A_y * B_x$



The cross-product of two points is **positive** if and only if the angle of those points at origin $(0, 0)$ is in **counter-clockwise** direction.




```
boolean leftTurn(point_A, point_at_origin, point_B)
```

Coding Details

```
public static List<Point2D> grahamScan(List<Point2D> p) {
    Collections.sort(p, new XComparator());
    List<Point2D> s = new ArrayList<Point2D>();
    s.add(p.get(0));
    s.add(p.get(1));
    for (int i = 2; i < p.size(); i++) {
        Point2D pi = p.get(i);
        while (s.size() >= 2 && leftTurn(s.get(s.size()-2),
                                         s.get(s.size()-1), pi)) {
            s.remove(s.size()); // pop
        }
        s.add(pi);
    }
    return s;
}
```

Summary

If points are already sorted, then Graham's Scan takes $\boldsymbol{O(n)}$ time.

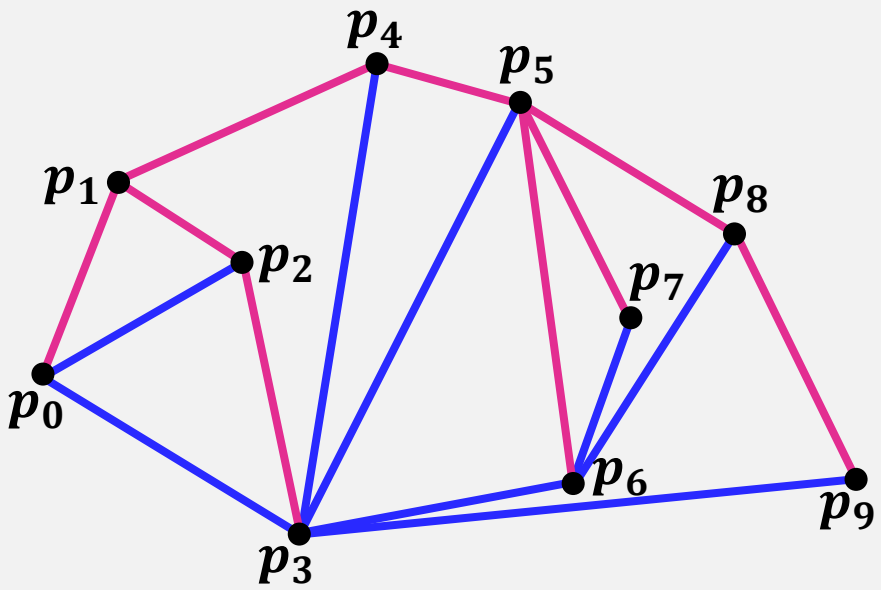
Theorem: Given a collection $\boldsymbol{P}$ of $\boldsymbol{n}$ points in the plane, Graham's Scan can compute their upper hull in $\boldsymbol{O(n \log n)}$ time.

With two passes, we can compute the upper hull and lower hull and attach them together to get the convex hull.
In this case we only need to sort the points once.

Theorem: Given a collection $\boldsymbol{P}$ of $\boldsymbol{n}$ points in the plane, two applications of Graham's Scan can compute their convex hull in $\boldsymbol{O(n \log n)}$ time.

By using a **Deque**, we only need one pass.

Graham's Scan with a Deque



$$s = \langle p_0, p_1 \rangle$$

$$i = 2: \quad s = \langle p_2, p_0, p_1, p_2 \rangle$$

$$i = 3: \quad s = \langle p_3, p_0, p_1, p_2, p_3 \rangle$$

$$i = 4: \quad s = \langle p_4, p_3, p_0, p_1, p_4 \rangle$$

$$i = 5: \quad s = \langle p_5, p_3, p_0, p_1, p_4, p_5 \rangle$$

$$i = 6: \quad s = \langle p_6, p_3, p_0, p_1, p_4, p_5, p_6 \rangle$$

$$i = 7: \quad s = \langle p_7, p_6, p_3, p_0, p_1, p_4, p_5, p_7 \rangle$$

$$i = 8: \quad s = \langle p_8, p_6, p_3, p_0, p_1, p_4, p_5, p_8 \rangle$$

$$i = 9: \quad s = \langle p_9, p_3, p_0, p_1, p_4, p_5, p_8, p_9 \rangle$$