Computational Geometry

Further Reading

TOPICS

- Preliminaries
- Point in a Polygon
- Polygon Construction
- Convex Hulls

Geometric Algorithms

Geometric Algorithms find applications in such areas as

- **Computer Graphics**
- **Computer Aided Design**
- **VLSI Design**
- **GIS**
- **Robotics**

We will study algorithms dealing with

points, lines, line segments, and polygons

In particular, the algorithms will

- ***Determine whether a point is inside a Polygon***
- ***Construct a Polygon***
- ***Determine Convex Hulls***

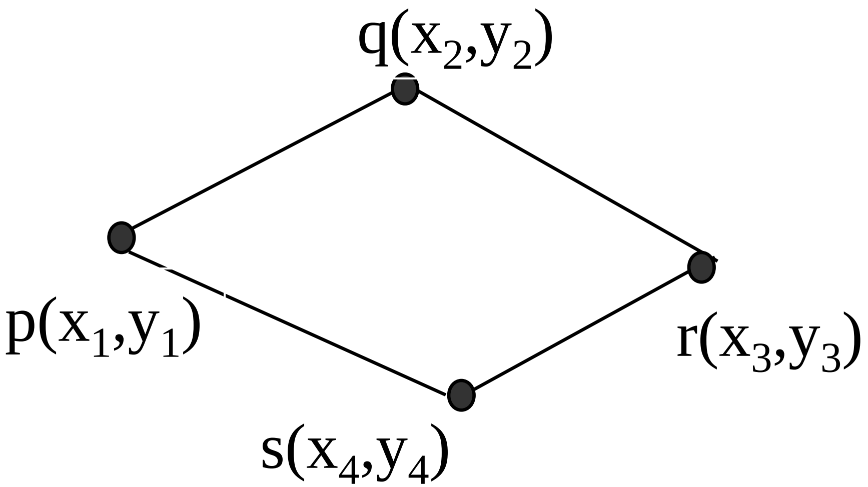
Preliminaries:

A point p is represented as a pair of coordinates (x,y)
A line is represented by a pair of points
A path is a sequence of points $p_1, p_2, \ldots p_n$ and the line segments connecting them,

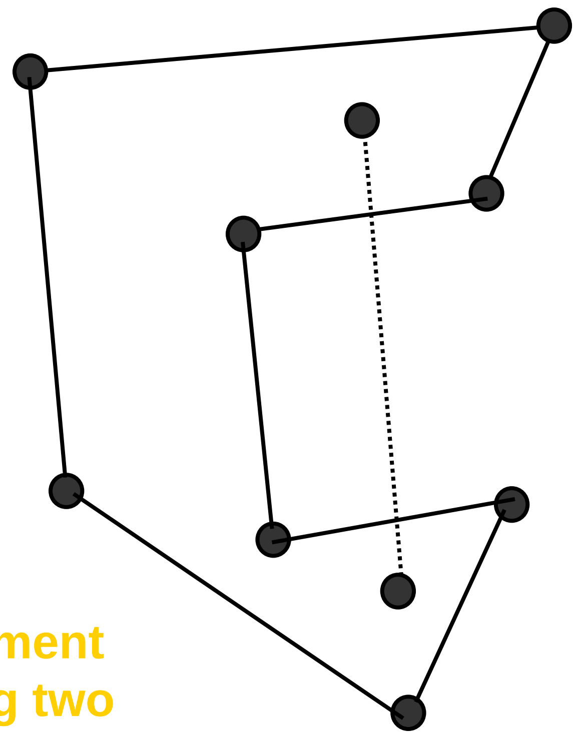
$$p_1\text{-}p_2,\ p_2\text{-}p_3,\ \ldots,\ p_{k-1}\text{-}p_k.$$

A closed path whose last point is the same as the first is a polygon.
A simple polygon is one whose corresponding path does not intersect itself. It encloses a region in the plane.

A convex Polygon is a polygon such that any line segment connecting two points inside the polygon is itself entirely in the polygon.
The convex hull of a set of points is defined as the smallest convex polygon enclosing all the given points.

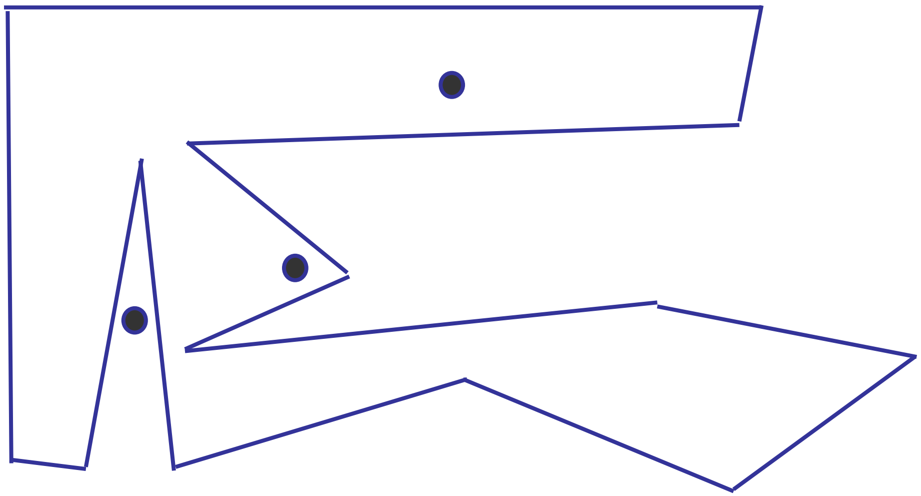


This is not a convex polygon



A line segment connecting two points:
The points are inside the polygon
The line segment is not entirely in the polygon

Determining whether a *point* is inside a polygon



Given a simple polygon polygon P, and a point q, determine whether the point is inside or outside the polygon. (a non-convex polygon)

Procedure **Point_in_a_Polygon(P,q)**

Input : **P (a simple polygon with vertices p_1,p_2,p_3, and edges e_1,e_2,e_3, … e_n and q (x_0,y_0) a point.**

Output: **INSIDE (a Boolean variable, True if q is inside P, and false otherwise)**

Count $\leftarrow$ 0;

for **all edges e_i of the polygon do**

if **the line $x = x_0$ intersects e_i then**

$y_i \leftarrow$ y coordinate of the intersection between lines e_i and $x=x_0$;

if **$y_i > y_0$** **then**

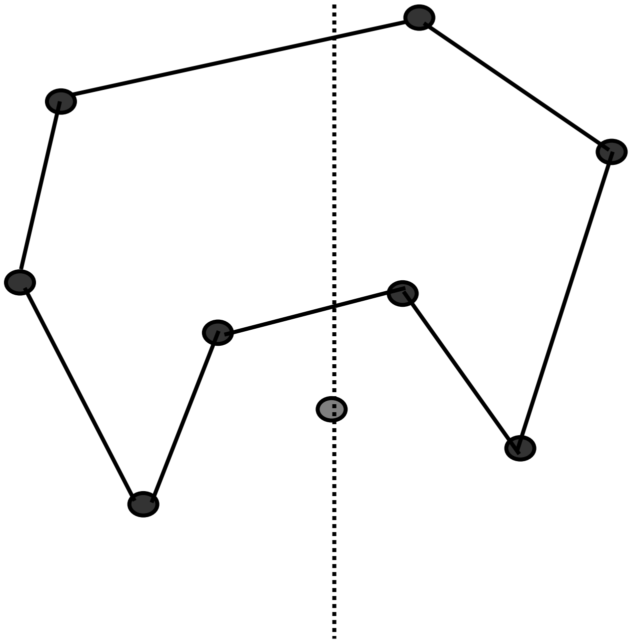
Count $\leftarrow$ Count +1;

if **count is odd** **then** **INSIDE $\leftarrow$ TRUE;**

else **INSIDE $\leftarrow$ FALSE**

This does not work if the line passes through terminal points of edges

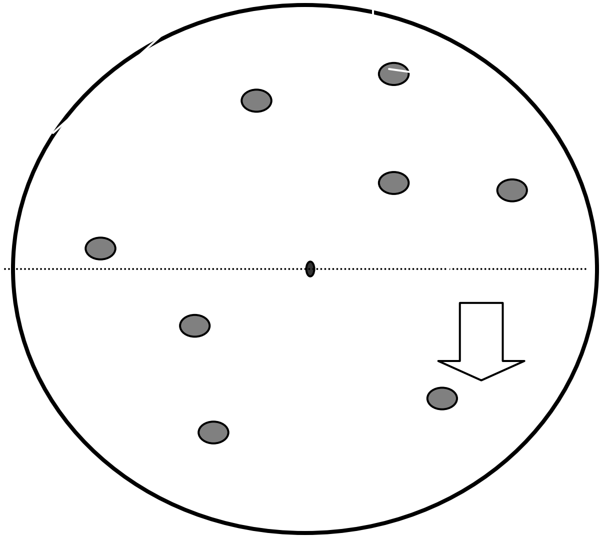
It takes constant time to perform an intersection between two line segments.
The algorithm computes n such intersections, where n is the size on the polygon.
Total running time of the algorithm, O(n).



Constructing a Simple Polygon

Given a set of points in the plane, connect them in a simple closed path.

> **Consider a large circle that contains all the points.**
> **Scan the area of C by a rotating line.**
> **Connect the points in the order they are encountered in the scan.**



Procedure Simple_Polygon

Input : $p_1, p_2, . . . p_n$ (points in the polygon)
Output : P (a simple polygon whose vertices $p_1, p_2, . . . p_n$ are in some order)

p1 $\leftarrow$ the point with the max 'x' value.

1. **for** i $\leftarrow$ 2 **to** n
2. $\alpha_i \leftarrow$ angle between line p_1-p_i and the x-axis;
3. sort the points according to the angles
   (use the corresponding priority for the point and do a heapsort)
4. P is the polygon defined by the list of points in the sorted order.

Complexity : Complexity of the sorting algorithm.

Convex Hulls

The convex hull of a set of points is defined as the smallest convex polygon enclosing all the points in the set.

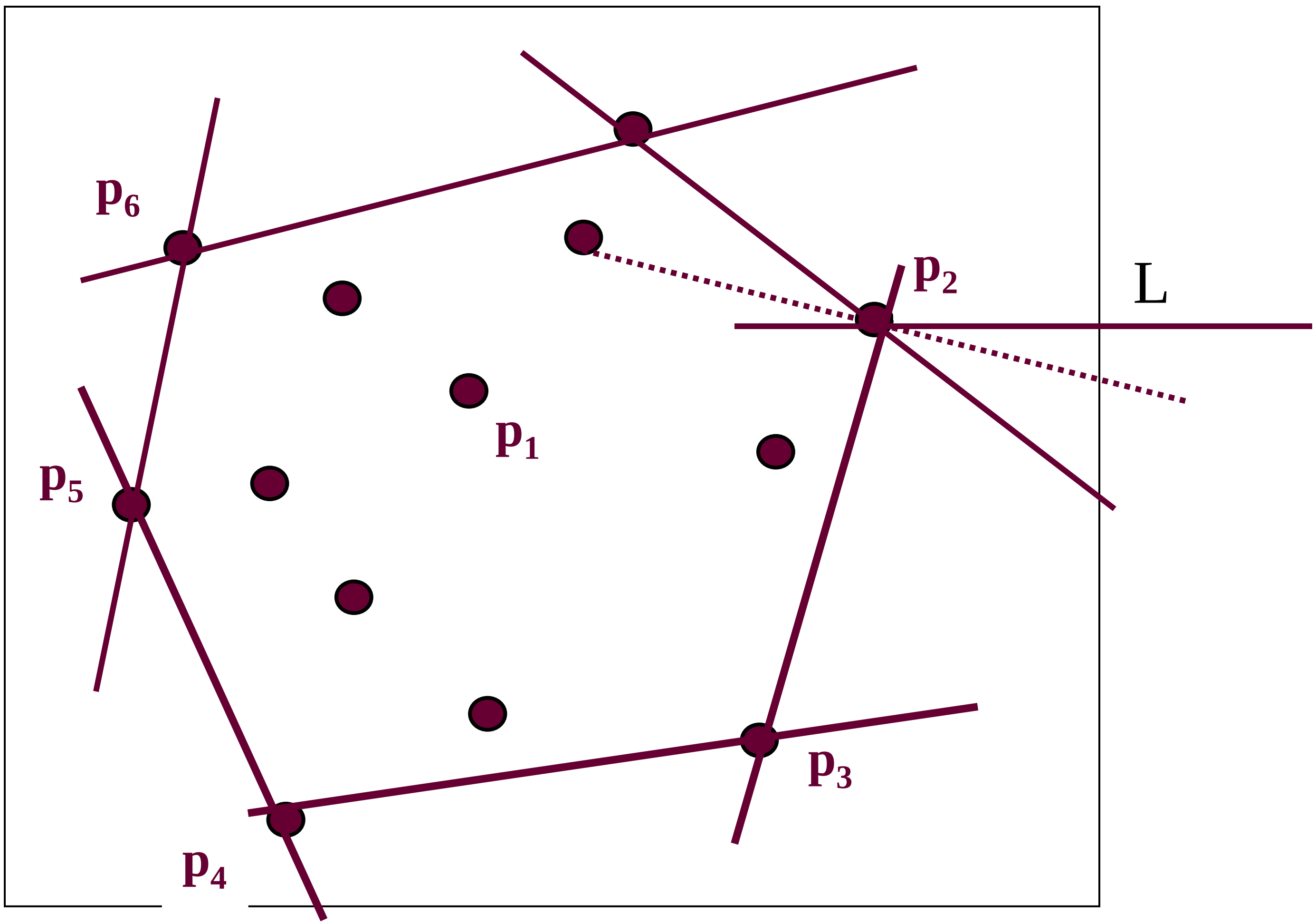
The convex hull is the smallest region encompassing a set of points.
A convex hull can contain as little as three and as many as all the points as vertices.

Problem Statement : Compute the convex hull of n given points in the plane.

There are two algorithms

Gift Wrapping O(n^2)

Graham's Scan O(nlogn)



p_6
p_2
L
p_1
p_5
p_3
p_4

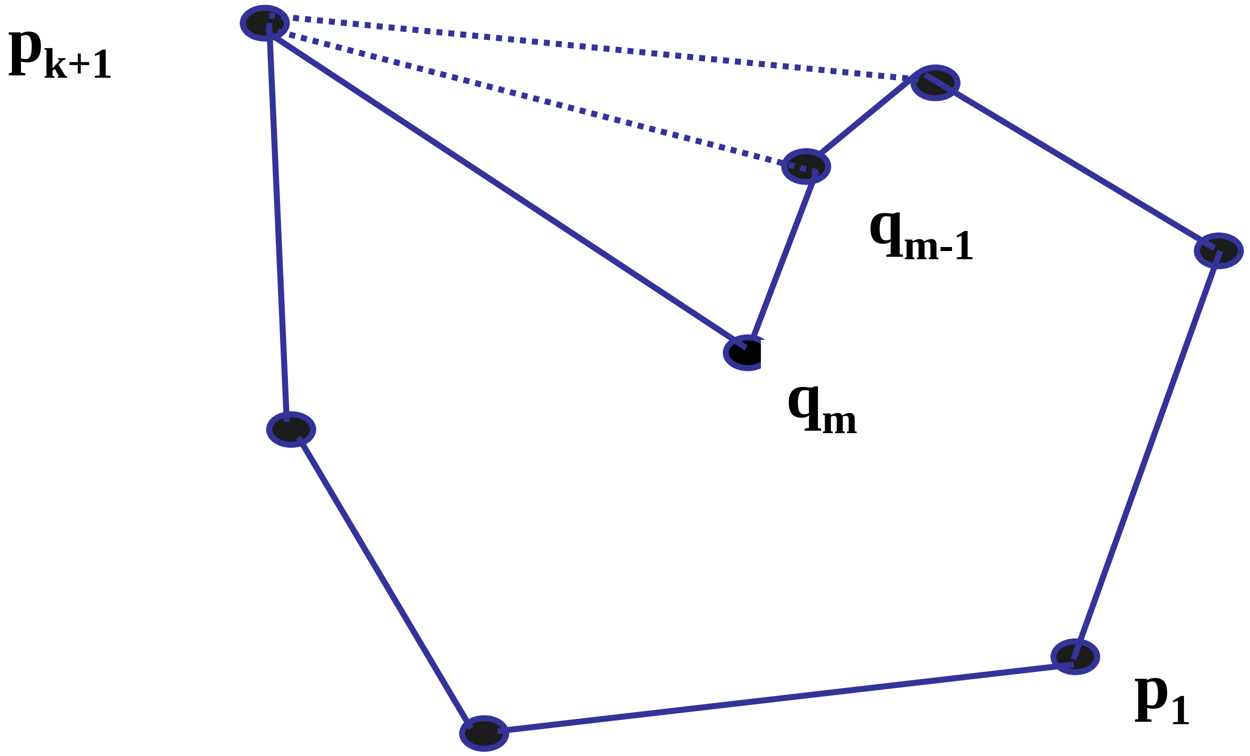
Procedure Gift_Wrapping($p_1, p_2, \ldots p_n$)

Input : $p_1, p_2, \ldots p_n$ (a set of points in the plane)
Output : P (the convex hull of $p_1, p_2, \ldots p_n$)

1. P $\leftarrow$ {0} or ε;
2. p $\leftarrow$ a point in the set with the largest x-coordinate;
3. Add p to P;
4. L $\leftarrow$ line containing p and parallel to the x-axis;
5. **while** |P| < n **do**
6. q $\leftarrow$ point such that the angle between the line -p-q- and L is minimal among all points;
7. add q to P;
8. L $\leftarrow$ line -p-q-;
9. p$\leftarrow$q;

Graham's Scan:

Given a set of n points in the plane, ordered according to the algorithm Simple Polygon, we can find a convex path among the first k points whose corresponding convex polygon encloses the first k points.



Procedure Graham's Scan($p_1, p_2, \ldots p_n$)

Input : $p_1, p_2, \ldots p_n$ (a set of points in the plane)
Output : $q_1, q_2, \ldots q_n$ (the convex hull of $p_1, p_2, \ldots p_n$)

p1 $\leftarrow$ the point in the set with the largest x-coordinate (and smallest y-coordinate if there are more than one point with the same x-coordinate)

Construct Simple Polygon and arrange points in order

Let order be $p_1, p_2, \ldots p_n$

$q_1 \leftarrow p_1$;

$q_2 \leftarrow p_2$;

$q_3 \leftarrow p_3$; (initially P consists of p_1, p_2, and p_3)

$m \leftarrow 3$;

for $k \leftarrow 4$ **to** n **do**

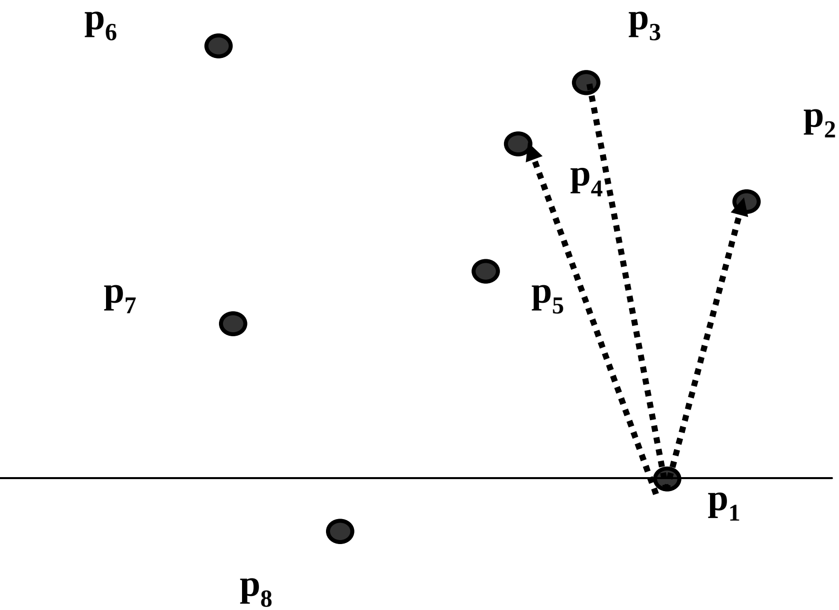
  while the angle between lines $-q_{m-1}-q_m-$ and $-q_m-p_k- \geq 180^o$ **do**

    $m \leftarrow m-1$;

  $m \leftarrow m+1$;

  $q_m \leftarrow p_k$;

[Internal to the polygon]

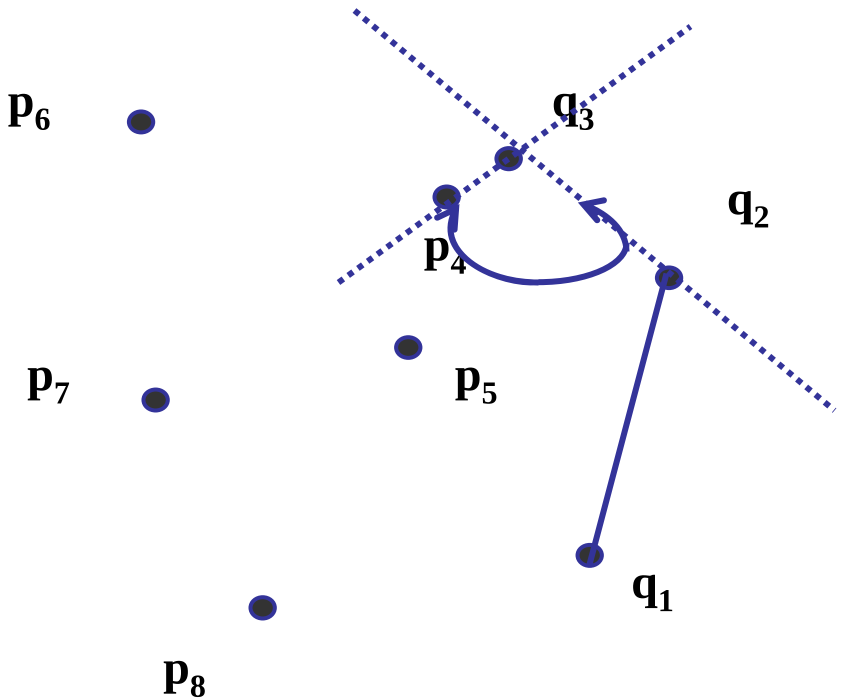


q1:p1

q2:p2

q3:p3

m:3

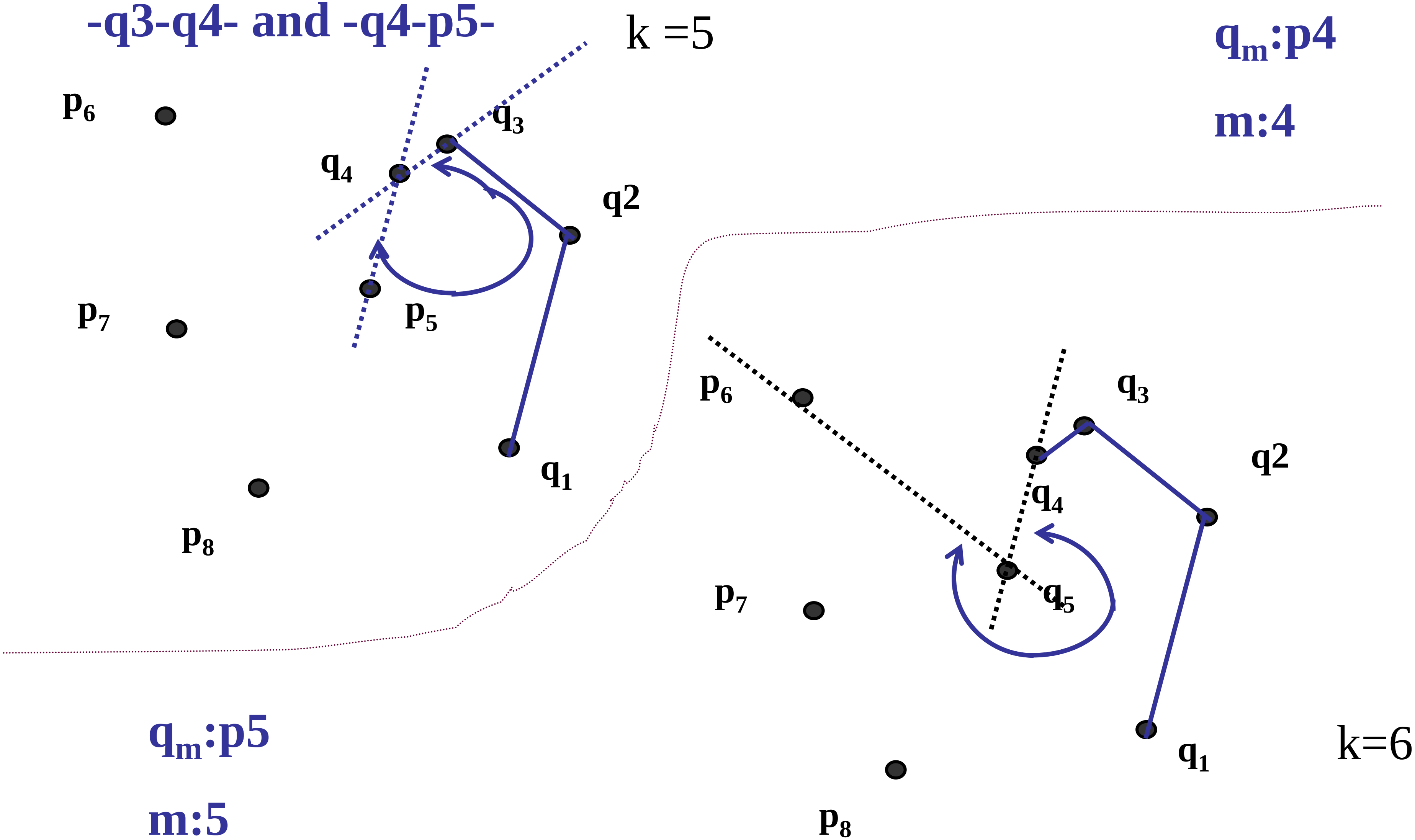


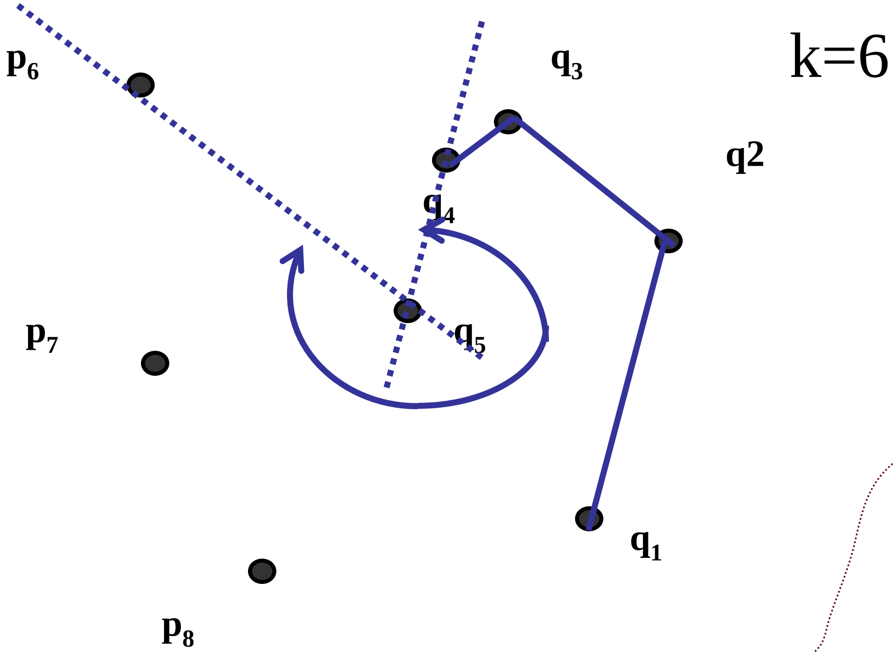
-q2-q3- and -q3-p4-

The angle is < 180

k =4

-q3-q4- and -q4-p5-



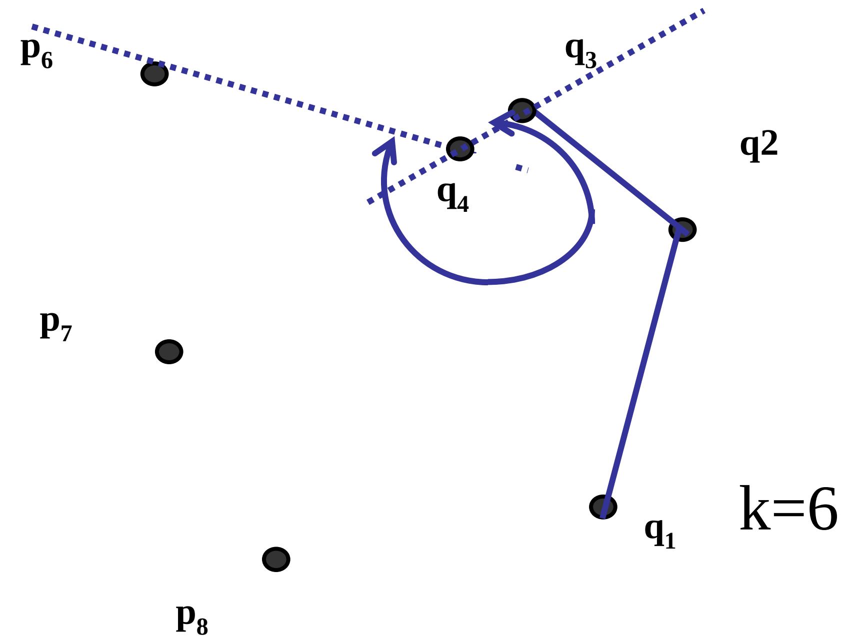


Angle between -q4-q5- and

-q5-p6- is greater than 180

Therefore   m = m-1 = 4

We skip p5



Angle between -q3-q4- and

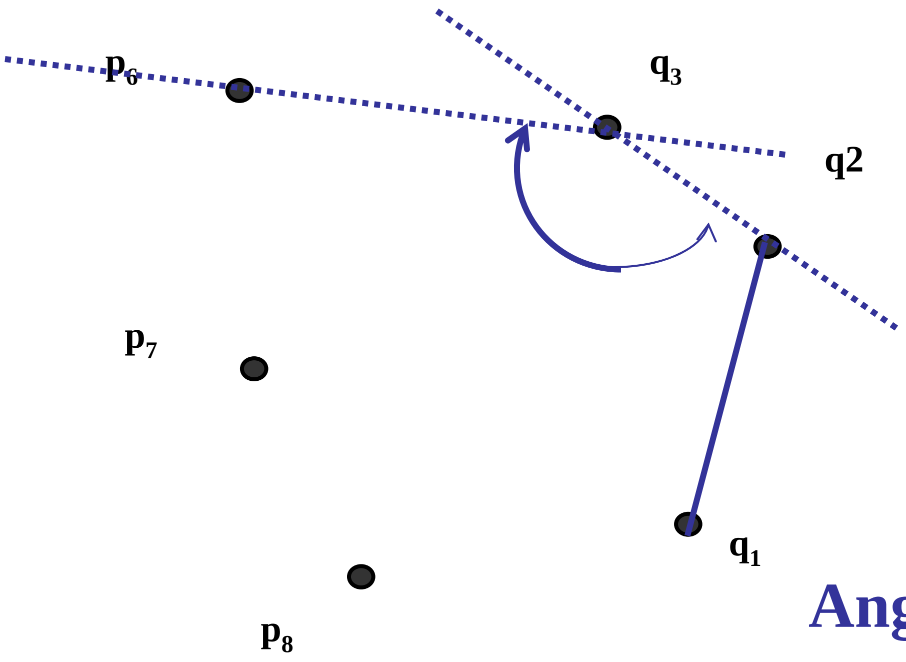
-q4-p6- is greater than 180

Therefore   m = m-1 = 3

We skip p4

-q3-q4- and -q4-p6-

k=6

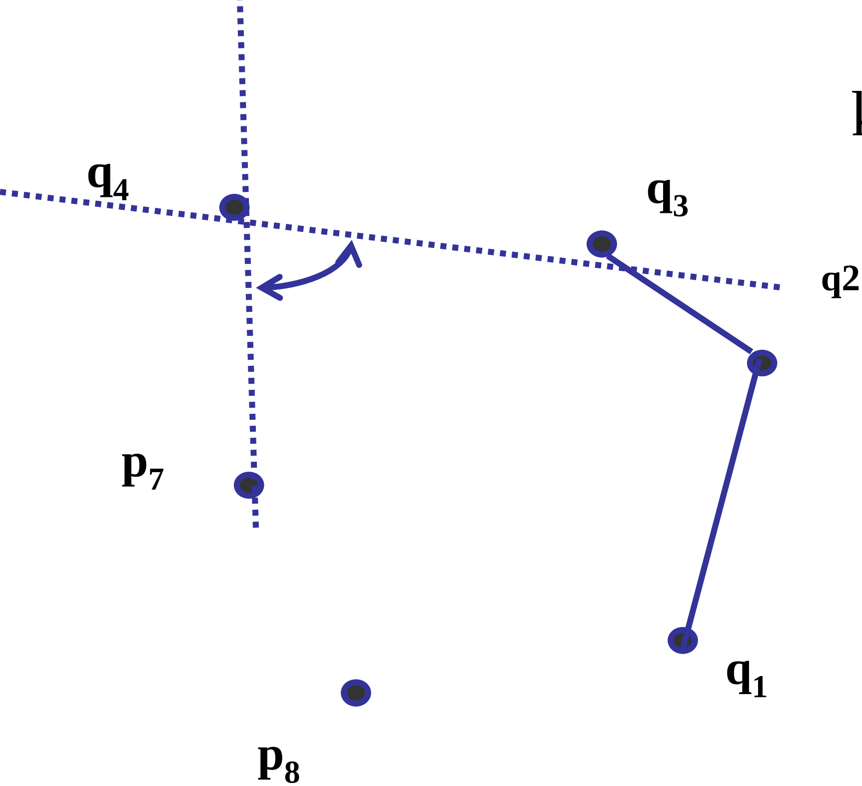


Angle between -q2-q3- and -q3-p6- is less than 180

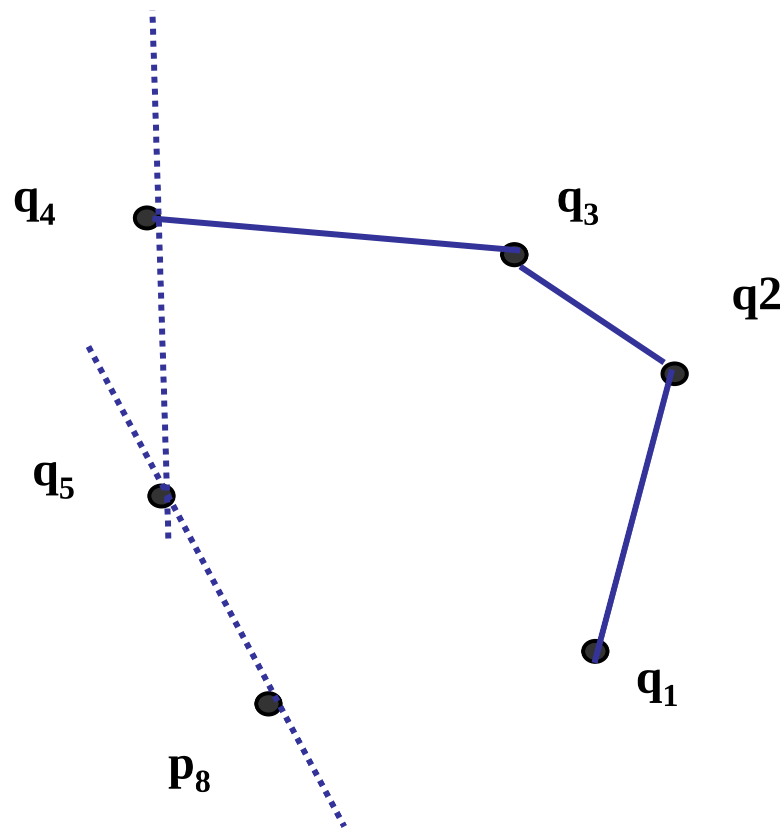
Therefore m = m+1 = 4

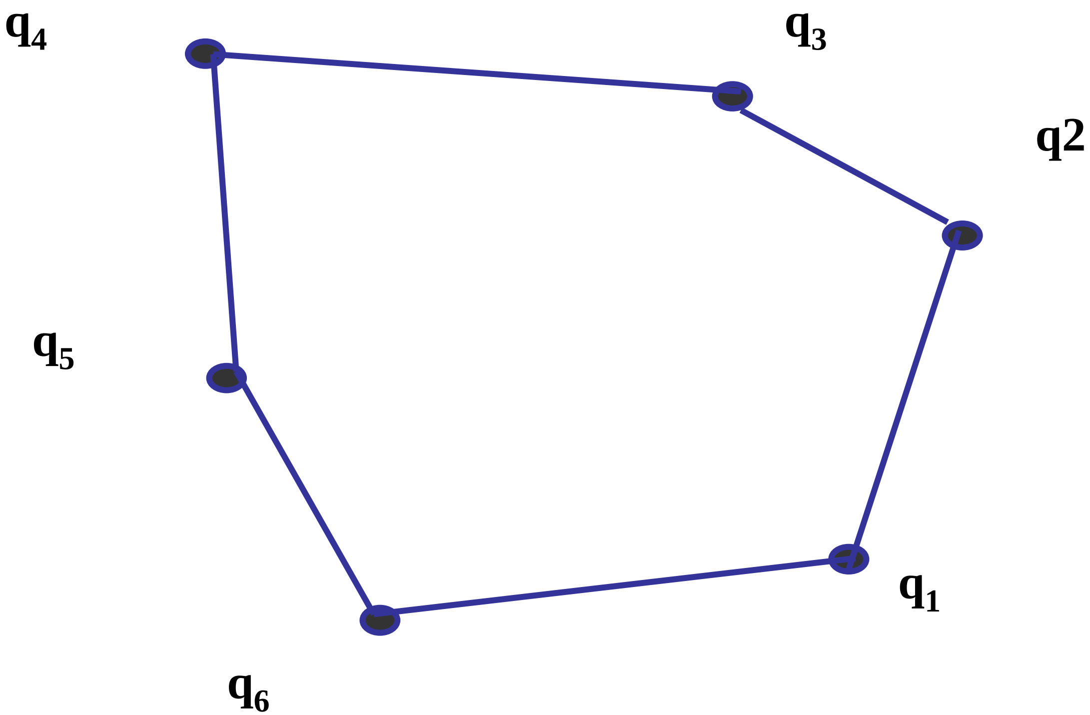
and q4 = p6

k=7



k=8





q4
q3
q2
q5
q1
q6

Procedure Graham's Scan($p_1, p_2, \ldots p_n$)

Input : $p_1, p_2, \ldots p_n$ (a set of points in the plane)

Output : $q_1, q_2, \ldots q_n$ (the convex hull of $p_1, p_2, \ldots p_n$)

p1 $\leftarrow$ the point in the set with the largest x-coordinate (and smallest y-coordinate if there are more than one point with the same x-coordinate)

Construct Simple Polygon and arrange points in order

Let order be $p_1, p_2, \ldots p_n$

$q_1 \leftarrow p_1$;

$q_2 \leftarrow p_2$;

$q_3 \leftarrow p_3$; (initially P consists of p_1, p_2, and p_3)

$m \leftarrow 3$;

for $k \leftarrow 4$ **to** n **do**

while the angle between lines $-q_{m-1}-q_m-$ and $-q_m-p_k- \geq 180^o$ **do**

$m \leftarrow m-1$;

$m \leftarrow m+1$;

$q_m \leftarrow p_k$;

[Internal to the polygon]

Exercise Problems

1. **Let *P* be a simple (not necessarily convex) polygon enclosed in a given rectangle *R*, and *q* be an arbitrary point inside *R*. Design an efficient algorithm to find a line segment connecting *q* to any point outside *R* such that the number of edge of *P* that this line intersects is minimum.**
2. **Let *P* be a set of *n* points in a plane. We define the depth of a point *p* in *P* as the number of convex hulls that need to be 'peeled' (removed) for *p* to become a vertex of the convex hull. Design an $O(n^2)$ algorithm to find the depths of all points in *P*.**
3. **Given a set of n points in the plane *P*. A straight forward or brute force algorithm will take $O(n^2)$ to compute a pair of closest points. Give an $O(n \log^2 n)$ algorithm find a pair of closest points. You get a bonus if you can give an $O(n \log n)$ algorithm**