SỰ KỲ DIỆU
của
HÌNH HỌC TÍNH TOÁN

Đỗ Phan Thuận

Email/Facebook: thuandp.sinhvien@gmail.com
Bộ môn Khoa Học Máy Tính, Viện CNTT & TT,
Trường Đại Học Bách Khoa Hà Nội.

Ngày 14 tháng 6 năm 2018

Nội dung

- Hình học có đáng sợ?
- Hình học cơ bản
- Hình học tính toán

Hình học

Hình học có đáng sợ?

- Hình học là một chủ đề thường xuyên xuất hiện trong các cuộc thi lập trình.

Hình học có đáng sợ?

- Hình học là một chủ đề thường xuyên xuất hiện trong các cuộc thi lập trình.
- Dạng bài toán hình học:

Hình học có đáng sợ?

- Hình học là một chủ đề thường xuyên xuất hiện trong các cuộc thi lập trình.
- Dạng bài toán hình học:
  - Thuật toán đơn giản, chỉ cần vẽ những đối tượng hình học ra và áp dụng những công thức hình học cơ bản.

Hình học có đáng sợ?

- Hình học là một chủ đề thường xuyên xuất hiện trong các cuộc thi lập trình.
- Dạng bài toán hình học:
  - Thuật toán đơn giản, chỉ cần vẽ những đối tượng hình học ra và áp dụng những công thức hình học cơ bản.
  - Thuật toán phức tạp hơn + Công thức hình học cơ bản và quản lý các đối tượng hình học.

Hình học có đáng sợ?

- Hình học là một chủ đề thường xuyên xuất hiện trong các cuộc thi lập trình.
- Dạng bài toán hình học:
  - Thuật toán đơn giản, chỉ cần vẽ những đối tượng hình học ra và áp dụng những công thức hình học cơ bản.
  - Thuật toán phức tạp hơn + Công thức hình học cơ bản và quản lý các đối tượng hình học.
  - Tính chính xác hoặc tính gần đúng (chấp nhận sai số).

- Đa phần sợ các bài toán hình học vì những lý do sau:

- Đa phần sợ các bài toán hình học vì những lý do sau:
  - Dễ bỏ sót các trường hợp đặc biệt như :
    - ★ trường hợp một đường thẳng là thẳng đứng (có hệ số góc là vô hạn),
    - ★ trường hợp các điểm là thẳng hàng với nhau,
    - ★ trường hợp đa giác là lõm ,...

- Đa phần sợ các bài toán hình học vì những lý do sau:
  - Dễ bỏ sót các trường hợp đặc biệt như :
    - ★ trường hợp một đường thẳng là thẳng đứng (có hệ số góc là vô hạn),
    - ★ trường hợp các điểm là thẳng hàng với nhau,
    - ★ trường hợp đa giác là lõm ,...
  - Có thể gây ra lỗi về độ chính xác sau dấu phẩy động nên dù thuật toán đúng nhưng quá trình tính toán vẫn dẫn đến KẾT QUẢ SAI.
    - ★ Cần nắm vững nguyên lý lưu trữ biến và tính toán trong máy tính để có thể ước lượng được sai số tính toán.

LƯU Ý

- Sử dụng đơn vị radian với góc
- Quy đổi góc α độ $\rightarrow \alpha * \frac{PI}{180.0}$
- Thủ thuật: $PI = acos(-1)$

- Hai phần chính :
  - Phần thứ nhất : Hình học cơ bản, Điểm qua về các đối tượng hình học 0D, 1D, 2D, 3D.
  - Phần thứ hai : Một số phương pháp tính toán hình học.

- Hai phần chính :
  - Phần thứ nhất : Hình học cơ bản, Điểm qua về các đối tượng hình học 0D, 1D, 2D, 3D.
  - Phần thứ hai : Một số phương pháp tính toán hình học.

HÌNH HỌC CÓ THÚ VỊ?

HÌNH HỌC TÍNH TOÁN và THIÊN VĂN HỌC

HÌNH HỌC TÍNH TOÁN và THIÊN VĂN HỌC

Click here

HÌNH HỌC TÍNH TOÁN và THIÊN VĂN HỌC

Click here

Là sự NGẪU NHIÊN hay là sự SẮP ĐẶT TRƯỚC?

"Chúng ta chỉ mới ở vào buổi bình minh của khoa học, nhưng mỗi khám phá mới, mỗi kiến thức mới, đều đem lại cho chúng ta một bằng chứng rằng, vũ trụ này là công trình của một đấng tạo hoá. Hãy lấy một thí dụ toán học cho dễ hiểu. Nếu ta bỏ vào túi 10 thẻ nhỏ, mỗi thẻ có ghi từ số 1 đến số 10, và tuần tự rút ra từng cái một. Sau khi rút xong ta lại bỏ thẻ vào túi, trộn đều và rút ra lần nữa. Làm sao ta có thể rút tuần tự từ số 1 đến số 10? Theo toán học, ta phải rút mười lần, mới có một lần rút được thẻ mang số 1. Phải rút 100 lần mới có một lần rút được số 1 và 2. Phải rút 1000 lần mới được số 1, 2, 3 liên tiếp. Nếu muốn rút theo thứ tự từ 1 đến 10, thì trường hợp đặc biệt này chỉ có thể xảy ra một lần trong mười tỷ lần, có đúng không ? Nếu áp dụng toán học vào các điều kiện tạo đời sống ở quả đất này, thì ta thấy nguyên lý ngẫu nhiên không sao hội đủ các điều kiện cần thiết. Vậy thì ai đã tạo ra nó ? . . .

. . . Trái đất quay quanh trục của nó với vận tốc 1600 cây số một giờ ở giữa đường xích đạo. Nếu nó quay chậm 10 lần thì ngày sẽ dài gấp 10 và dĩ nhiên sức nóng của mặt trời cũng gia tăng gấp 10 lần. Thế thì cây cối, sinh vật đều bị thiêu sống hết còn gì. Nếu cái gì chống được sức nóng cũng chết lạnh vì đêm cũng dài ra gấp 10 và sức lạnh cũng tăng lên gấp 10 lần kia mà. Ai đã làm trái đất quay trong một điều kiện tốt đẹp như thế ? Mặt trời là nguồn sống của quả đất phải không ? Mặt trời nóng khoảng 5500 độ bách phân. Quả địa cầu ở đúng một vị trí tốt đẹp không xa quá mà cũng không gần quá. Vừa vặn đủ để đón nhận sức nóng của mặt trời. Nếu sức nóng mặt trời gia tăng một chút, ta sẽ chết thiêu, và ngược lại nếu sức nóng mặt trời giảm đi một chút, ta sẽ chết rét. Tại sao trái đất nằm ở điều kiện thuận lợi như vậy ? Trục trái đất nghiêng theo một toa độ là 23 độ. Nếu trái đất đứng thẳng, không nghiêng theo bên nào thì sẽ không có thời tiết bốn mùa. Nước sẽ bốc hơi hết về hai cực và đồng thành băng giá cả. Mặt trăng là một vệ tinh của trái đất, điều khiển thuỷ triều biển cả. Nếu nó không cách xa trái đất 380 000 cây số mà xích lại gần hơn 80 000 cây số thì một cuộc hồng thuỷ sẽ xảy ra. Nước sẽ bị sức hút dâng lên ngập tất cả các lục địa mỗi ngày hai lần. . . .

. . . Tóm lại tất cả mọi đời sống trên mặt địa cầu sẽ biến mất, nếu các điều kiện sai lệch đi một lý Nếu nói rằng đời sống chỉ là một sự ngẫu nhiên thì trong tỷ tỷ lần may ra mới có một điều kiện tốt đẹp hoàn toàn để có được sự sống như thế. "

HÀNH TRÌNH VỀ PHƯƠNG ĐÔNG. Gs. Baird T. Spalding

Cuốn sách do giáo sư Baird T. Spalding ghi chép lại lại cuộc du khảo cùng đoàn khoa học gia Hoàng gia Anh – những người giỏi và danh giá nhất Đại học Oxford danh tiếng đến vùng đất Ấn Độ và các vùng lân cận để khám phá thế giới huyền bí không giải thích hay chứng minh được bằng khoa học thực nghiệm. Sách dịch bởi giáo sư John Vu (tên thật là Vũ Văn Du, tựa Nguyên Phong).

Click here *Click here*

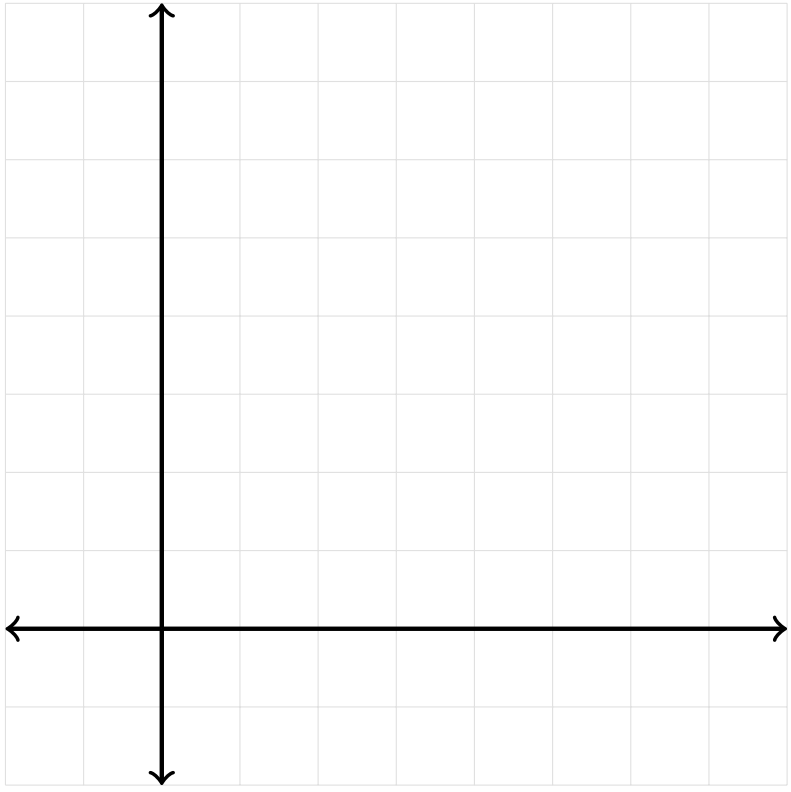
Click here *Click here*

"Dường như Đấng Tạo Hóa đang tiết lộ với chúng ta về bí mật của bản thiết kế mà Ngài đưa vào trong mỗi phần tử của vũ trụ!"

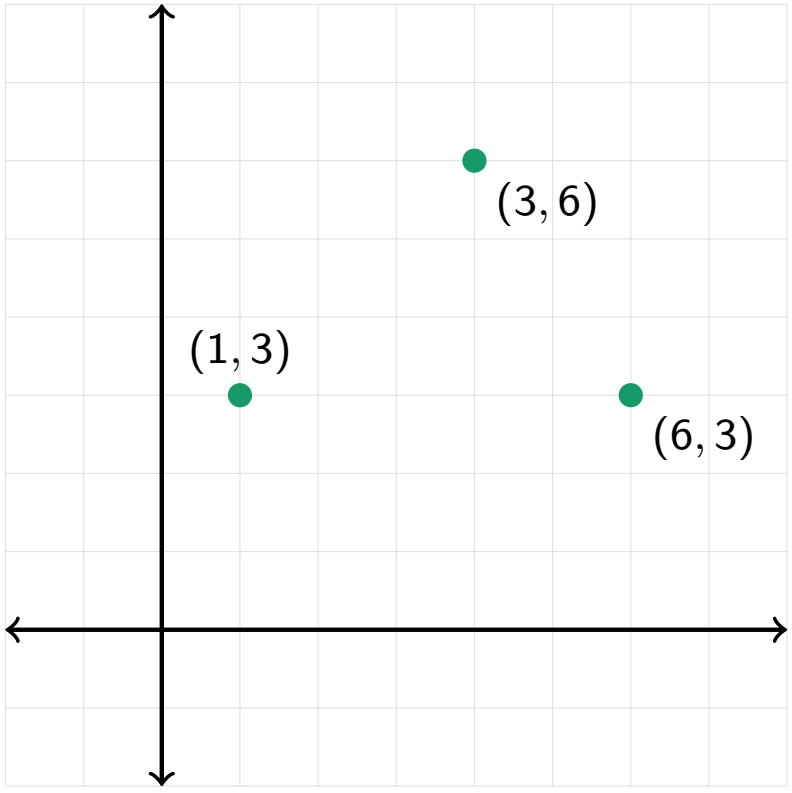
The Golden Ratio: The Story of Phi. *Mario Livio*

Hình học

Đối tượng 0D: Điểm và vec-tơ

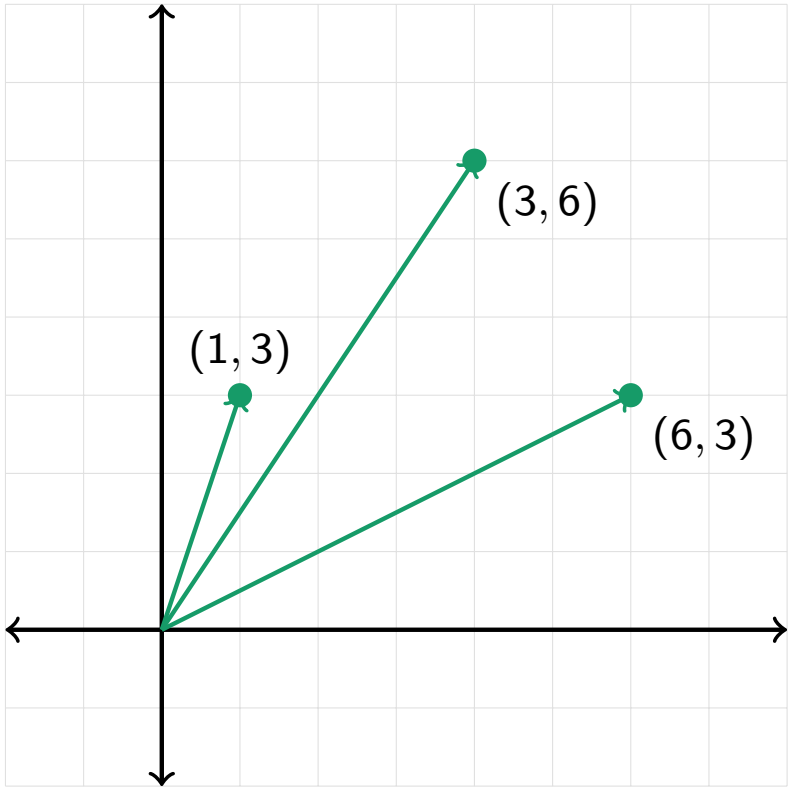


Đối tượng 0D: Điểm và vec-tơ

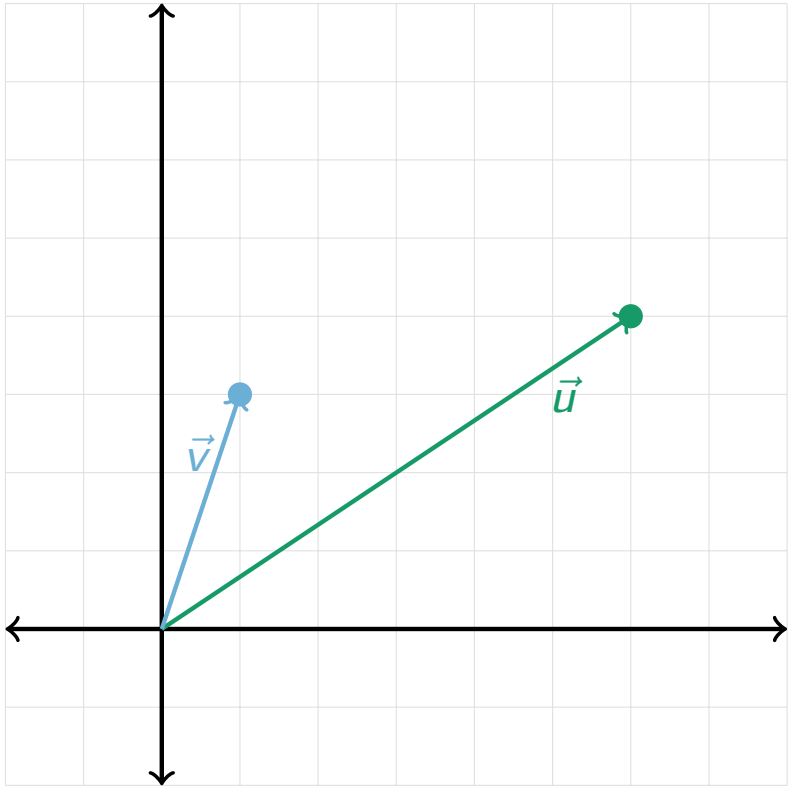


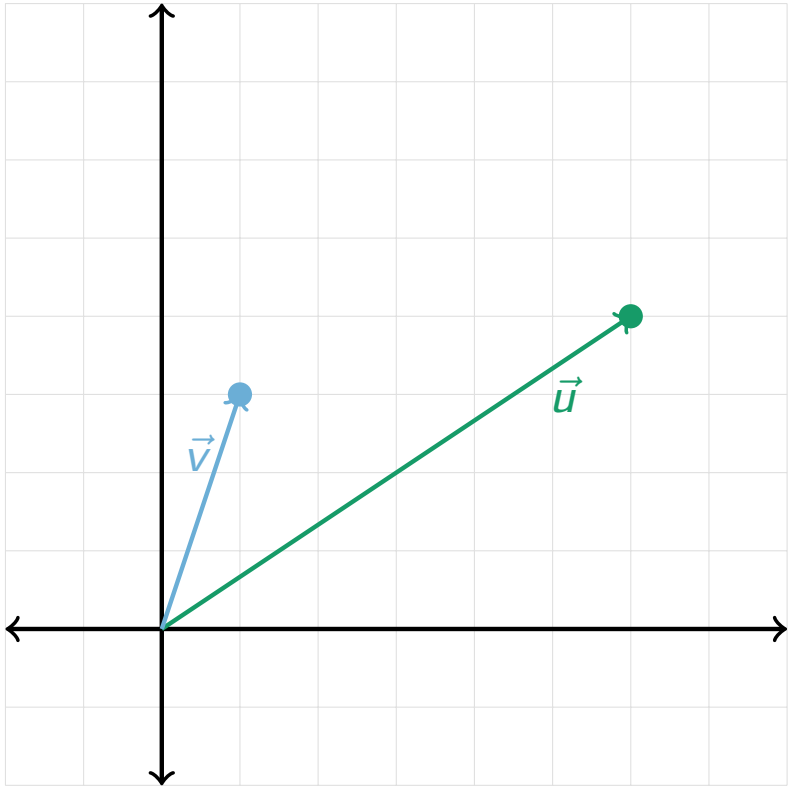
- Điểm được biểu diễn bởi một cặp số tọa độ (x, y).

Đối tượng 0D: Điểm và vec-tơ



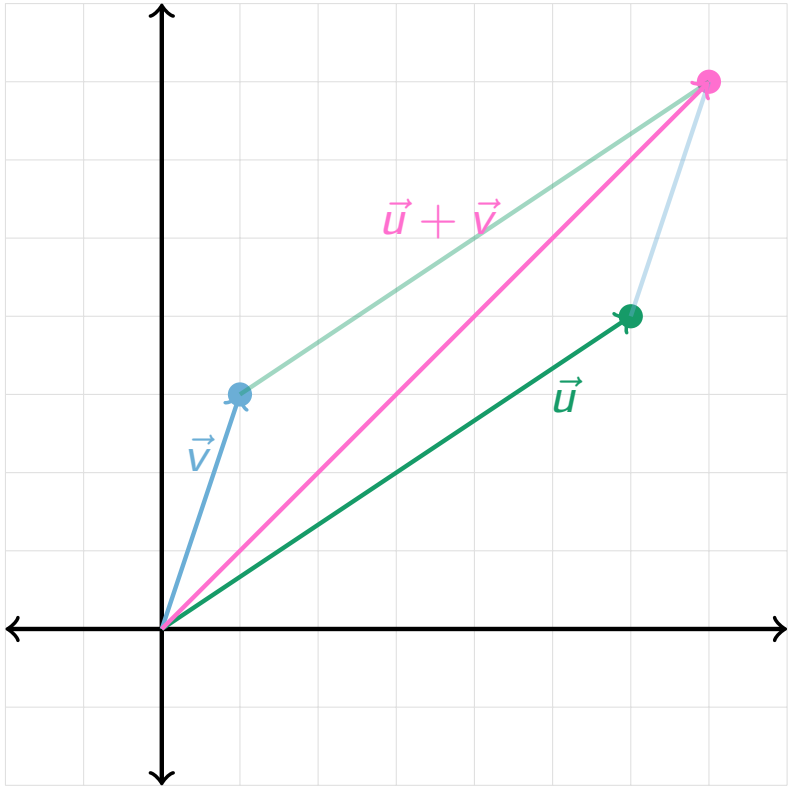
- Điểm được biểu diễn bởi một cặp số tọa độ (x, y).
- Vec-tơ được biểu diễn theo cách tương tự với gốc là gốc tọa độ.
- Vì vậy coi điểm như vec-tơ trong nhiều trường hợp giúp việc xử lý dễ dàng hơn.


$\vec{u}$
$\vec{v}$



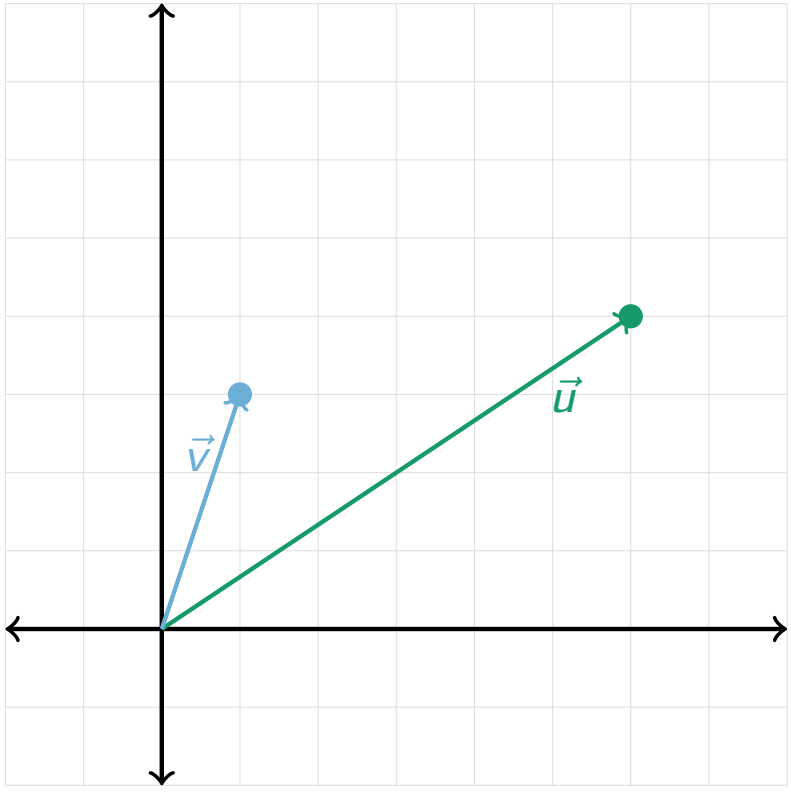
- Toán tử đơn giản nhất: phép cộng được định nghĩa như sau:

$$\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 + x_1 \\ y_0 + y_1 \end{pmatrix}$$



- Toán tử đơn giản nhất: phép cộng được định nghĩa như sau:

$$\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 + x_1 \\ y_0 + y_1 \end{pmatrix}$$

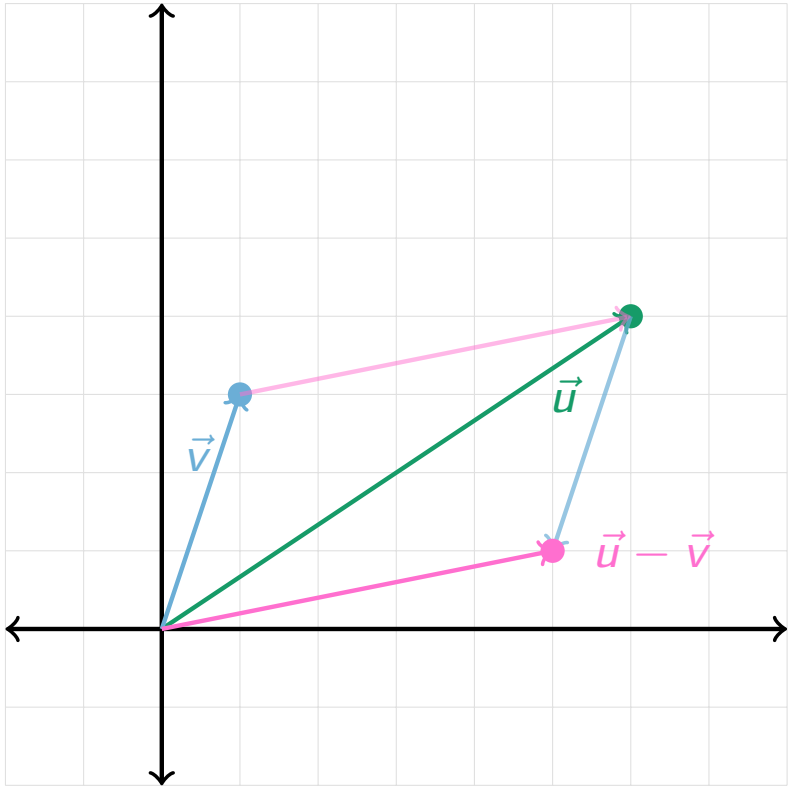


- Toán tử đơn giản nhất: phép cộng được định nghĩa như sau:

$$\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 + x_1 \\ y_0 + y_1 \end{pmatrix}$$

- Phép trừ được định nghĩa theo cách tương tự:

$$\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} - \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 - x_1 \\ y_0 - y_1 \end{pmatrix}$$



- Toán tử đơn giản nhất: phép cộng được định nghĩa như sau:

$$\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 + x_1 \\ y_0 + y_1 \end{pmatrix}$$

- Phép trừ được định nghĩa theo cách tương tự:

$$\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} - \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 - x_1 \\ y_0 - y_1 \end{pmatrix}$$

```
struct point {
    double x, y;
    point(double _x, double _y) {
        x = _x, y = _y;
    }

    point operator+(const point &oth){
        return point(x + oth.x, y + oth.y);
    }

    point operator-(const point &oth){
        return point(x - oth.x, y - oth.y);
    }
};
```

. . . hoặc ta có thể sử dụng lớp số phức complex<double>.

```
typedef complex<double> point;
```

. . . hoặc ta có thể sử dụng lớp số phức `complex<double>`.

```
typedef complex<double> point;
```

Lớp `complex` trong `C++` and `Java` có các phương thức được định nghĩa cho

- Phép cộng
- Phép trừ
- Phép nhân bởi một đại lượng vô hướng
- Chiều dài
- Hàm lượng giác

Số phức có phần thực và phần ảo. Ta có thể coi như vec-tơ hoặc điểm trên mặt phẳng phức.

Số phức có phần thực và phần ảo. Ta có thể coi như vec-tơ hoặc điểm trên mặt phẳng phức.

- `double real(p)` trả về phần thực, hay giá trị x của `p`.

Số phức có phần thực và phần ảo. Ta có thể coi như vec-tơ hoặc điểm trên mặt phẳng phức.

- `double real(p)` trả về phần thực, hay giá trị x của `p`.
- `double imag(p)` trả về phần ảo, hay giá trị y của `p`.

Số phức có phần thực và phần ảo. Ta có thể coi như vec-tơ hoặc điểm trên mặt phẳng phức.

- `double real(p)` trả về phần thực, hay giá trị x của `p`.
- `double imag(p)` trả về phần ảo, hay giá trị y của `p`.
- `double abs(p)` trả về giá trị tuyệt đối của số phức, hay chiều dài của vectơ.

Số phức có phần thực và phần ảo. Ta có thể coi như vec-tơ hoặc điểm trên mặt phẳng phức.

- `double real(p)` trả về phần thực, hay giá trị x của `p`.
- `double imag(p)` trả về phần ảo, hay giá trị y của `p`.
- `double abs(p)` trả về giá trị tuyệt đối của số phức, hay chiều dài của vectơ.
- `double sin(p)`, `double cos(p)`, `double tan(p)`, hàm lượng giác.

Kiểm tra hai điểm trùng nhau

- Ví dụ về hàm kiểm tra hai điểm có trùng nhau hay không:

Kiểm tra hai điểm trùng nhau

- Ví dụ về hàm kiểm tra hai điểm có trùng nhau hay không:

```
10 bool equalPoints(point a,point b){
11     return a.x == b.x && a.y == b.y;
12 }
```

Kiểm tra hai điểm trùng nhau

- Ví dụ về hàm kiểm tra hai điểm có trùng nhau hay không:

```
19 bool equalPoints(point a,point b){
20     return a.x == b.x && a.y == b.y;
21 }
```

CÓ THỂ GẶP LỖI SỐ THỰC DẤU PHẨY ĐỘNG!!!

Kiểm tra hai điểm trùng nhau

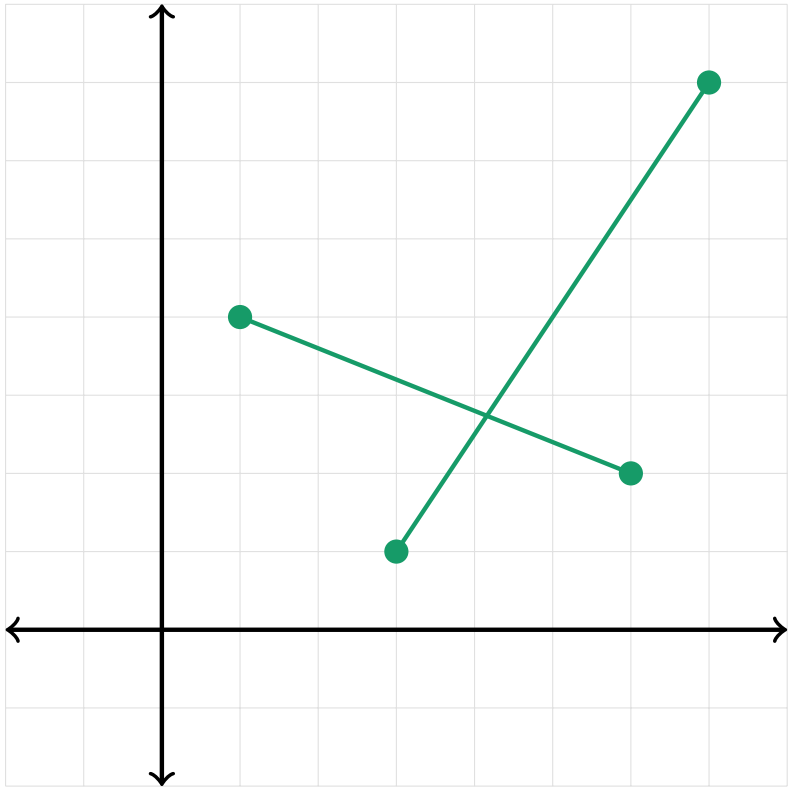
- Ví dụ về hàm kiểm tra hai điểm có trùng nhau hay không:

```
28 bool equalPoints(point a,point b){
29     return a.x == b.x && a.y == b.y;
30 }
```

CÓ THỂ GẶP LỖI SỐ THỰC DẤU PHẨY ĐỘNG!!!

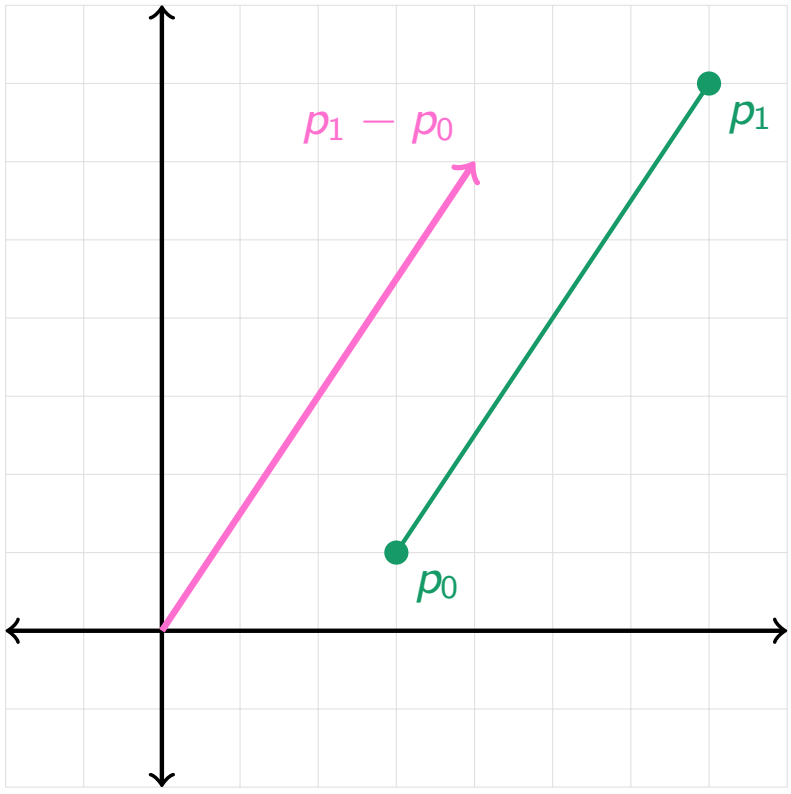
```
31 bool equalPoints(point a,point b){
32     double EPS = 1e-9;
33     return (fabs(a.x - b.x) < EPS &&
34     (fabs(a.y - b.y) < EPS));
35 }
```

Đối tượng 1D: Đường thẳng và đoạn thẳng



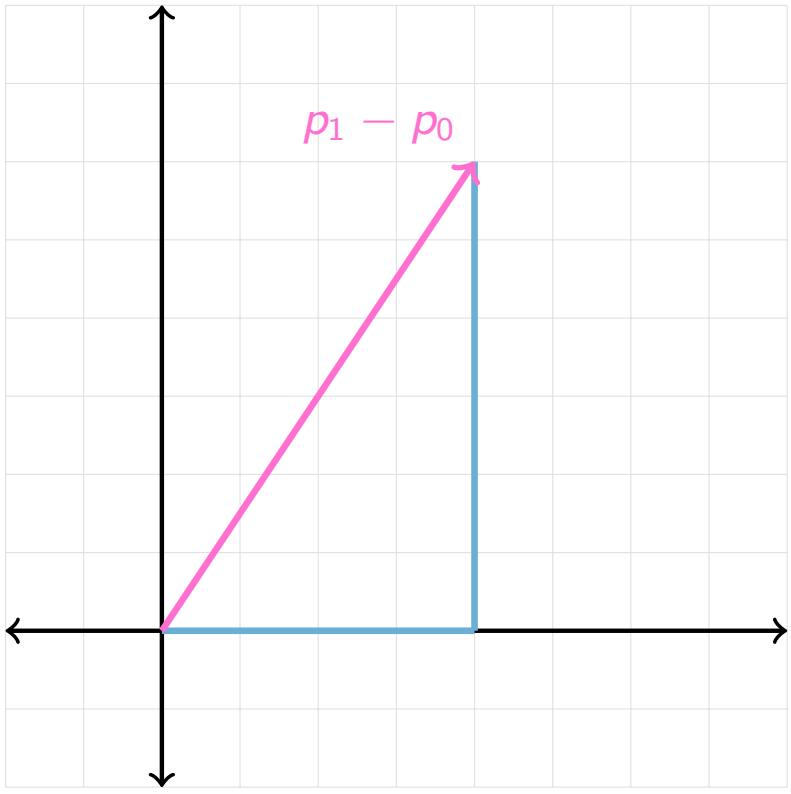
- Các đoạn thẳng được biểu diễn bởi một cặp điểm $((x_0, y_0), (x_1, y_1))$.

Đối tượng 1D: Đường thẳng và đoạn thẳng



- Các đoạn thẳng được biểu diễn bởi một cặp điểm $((x_0, y_0), (x_1, y_1))$.
- Khoảng cách giữa hai điểm là chiều dài của đoạn thẳng hoặc chiều dài của vectơ giữa hai điểm đó.

Đối tượng 1D: Đường thẳng và đoạn thẳng



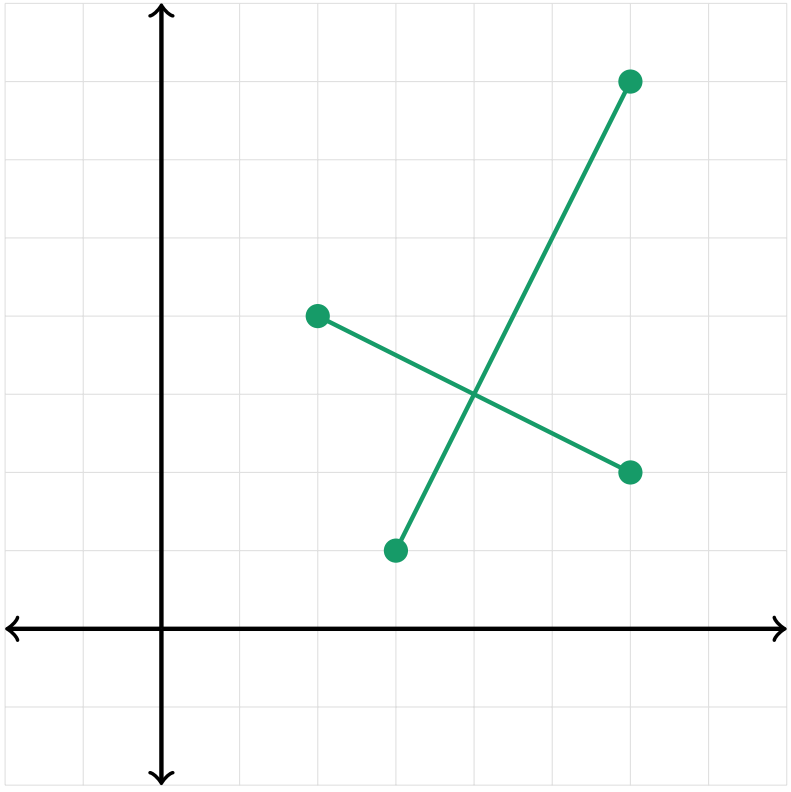
- Các đoạn thẳng được biểu diễn bởi một cặp điểm $((x_0, y_0), (x_1, y_1))$.
- Khoảng cách giữa hai điểm là chiều dài của đoạn thẳng hoặc chiều dài của vectơ giữa hai điểm đó.

$$\begin{aligned} d((x_0,y_0),(x_1,y_1)) &= |(x_1 - x_0, y_1 - y_0)| \\ &= \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} \end{aligned}$$

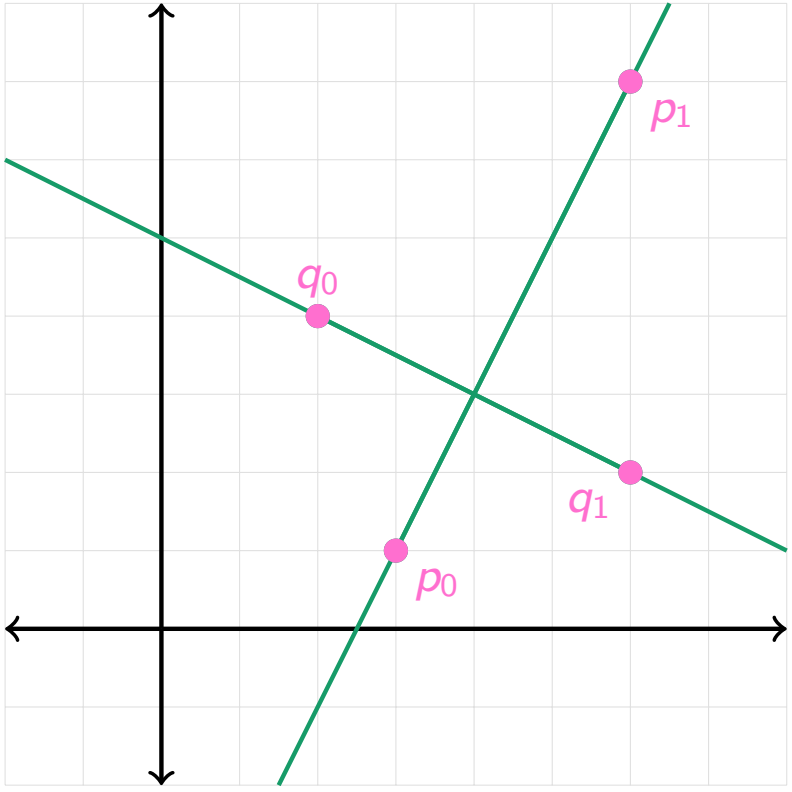
```
struct point {
    ...
    double distance(point oth = point(0,0)) const {
        return sqrt(pow(x - oth.x, 2.0)
                    + pow(y - oth.y, 2.0));
    }
    ...
}
```

```
struct point {
    ...
    double distance(point oth = point(0,0)) const {
        return sqrt(pow(x - oth.x, 2.0)
                    + pow(y - oth.y, 2.0));
    }
    ...
}
```

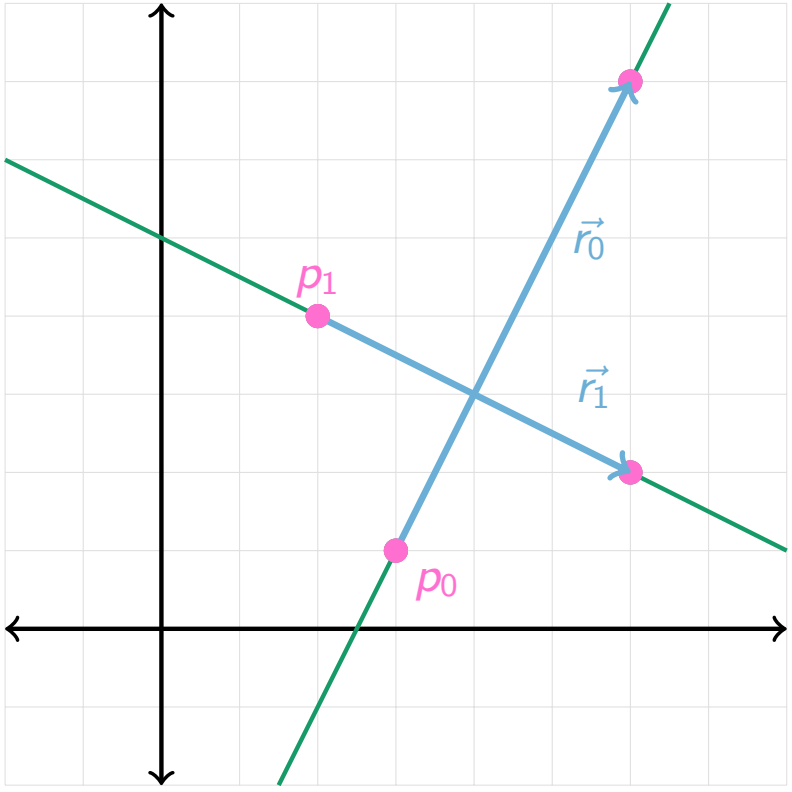
Hoặc sử dụng hàm abs với kiểu complex<double>.



- Biểu diễn đường thẳng giống như đối với đoạn thẳng.

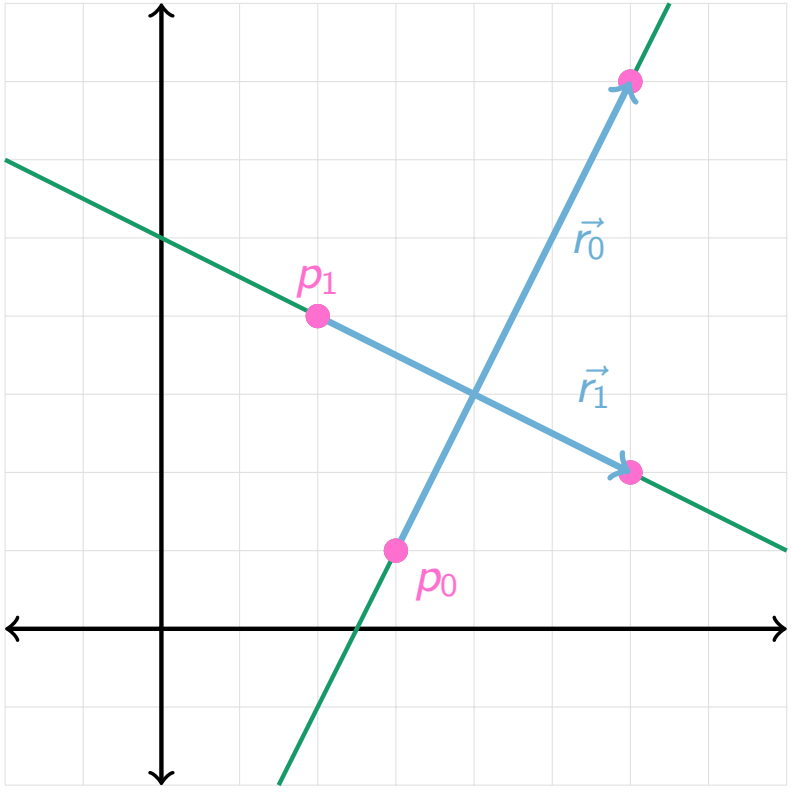


- Biểu diễn đường thẳng giống như đối với đoạn thẳng.
- Với hai điểm ta xác định được duy nhất một đường thẳng đi qua chúng.



- Biểu diễn đường thẳng giống như đối với đoạn thẳng.
- Với hai điểm ta xác định được duy nhất một đường thẳng đi qua chúng.
- Biểu diễn bởi một điểm và hướng một vectơ,

$$p + t \cdot \vec{r}$$



- Biểu diễn đường thẳng giống như đối với đoạn thẳng.
- Với hai điểm ta xác định được duy nhất một đường thẳng đi qua chúng.
- Biểu diễn bởi một điểm và hướng một vectơ,

$$p + t \cdot \vec{r}$$

- Hoặc

```
pair<point,point>
```

- Một đường thẳng cũng có thể được mô tả bởi các hệ số a, b, c của phương trình đường thẳng $Ax + By + C = 0$

- Một đường thẳng cũng có thể được mô tả bởi các hệ số a, b, c của phương trình đường thẳng $Ax + By + C = 0$

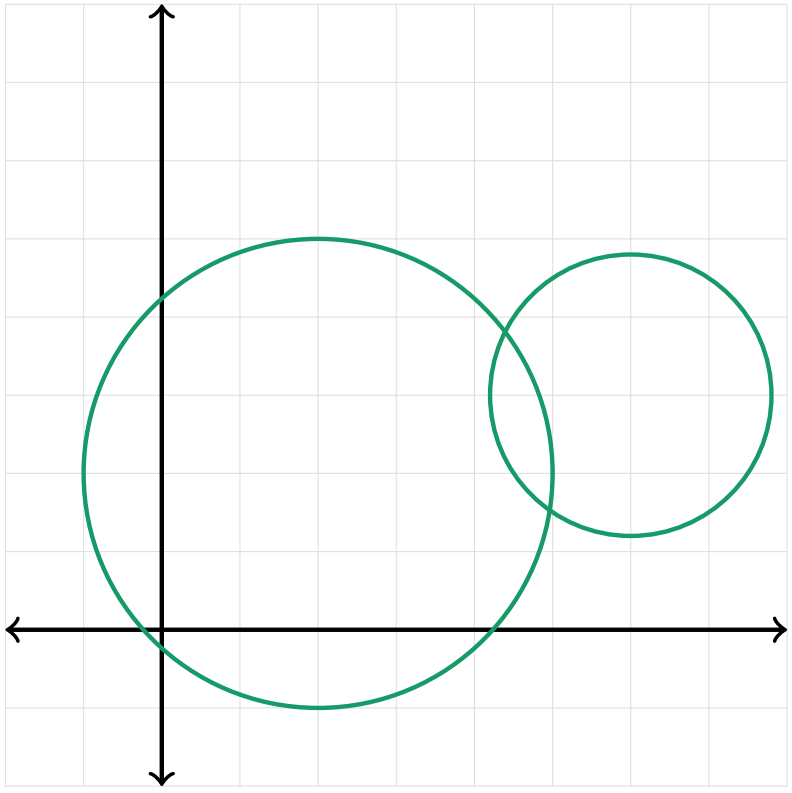
```
38 struct line { double A, B, C; };
```

- Một đường thẳng cũng có thể được mô tả bởi các hệ số a, b, c của phương trình đường thẳng $Ax + By + C = 0$

```
39 struct line { double A, B, C; };
```

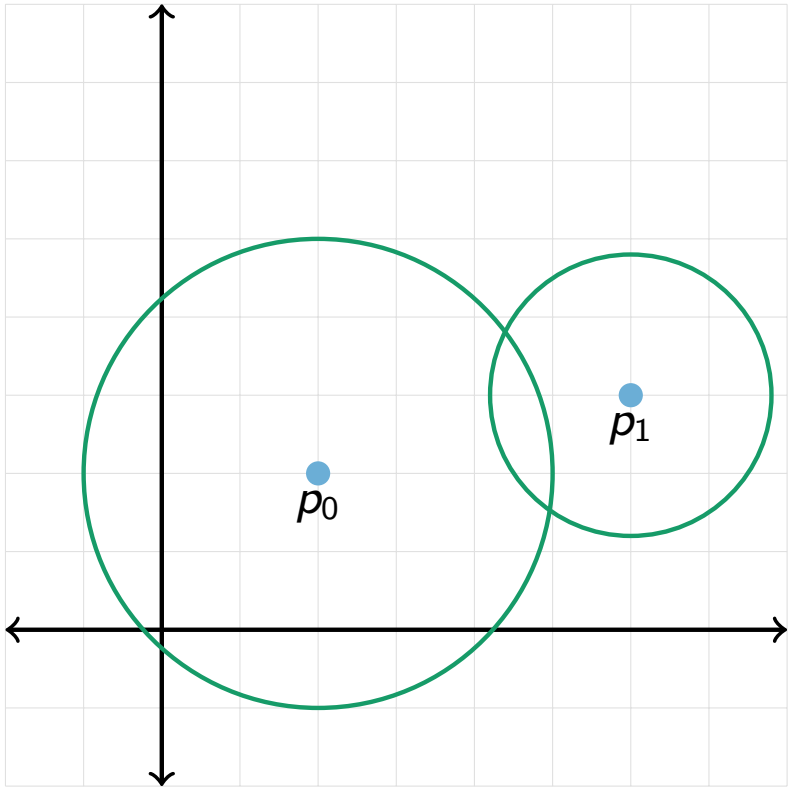
- Cẩn thận $B = 0$, khi đó đường thẳng có dạng thẳng đứng, và có hệ số góc là vô cùng.

Đối tượng 2D: Đường tròn



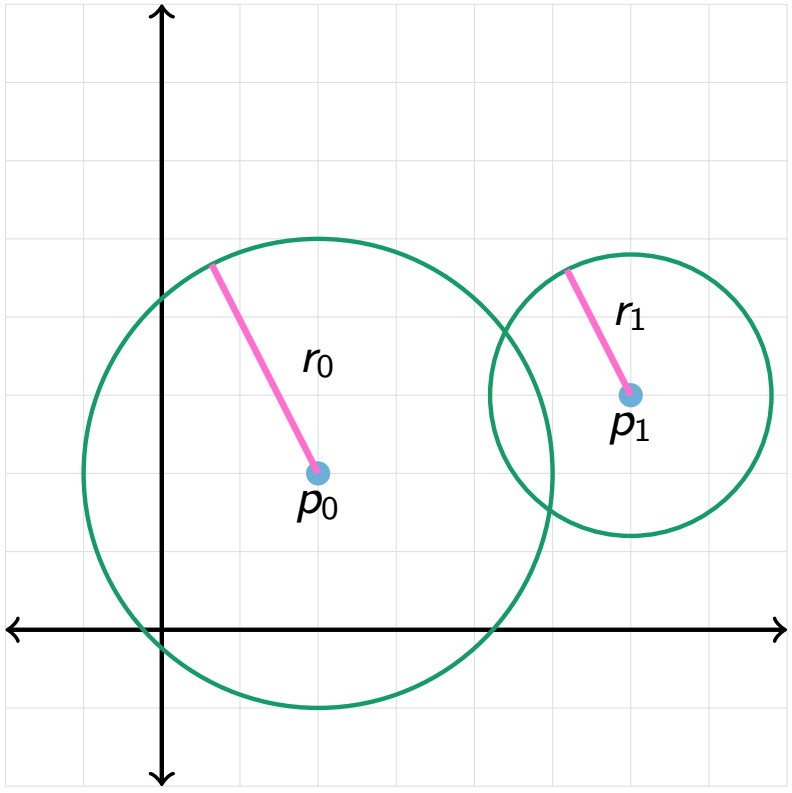
- Biểu diễn đường tròn rất đơn giản.

Đối tượng 2D: Đường tròn



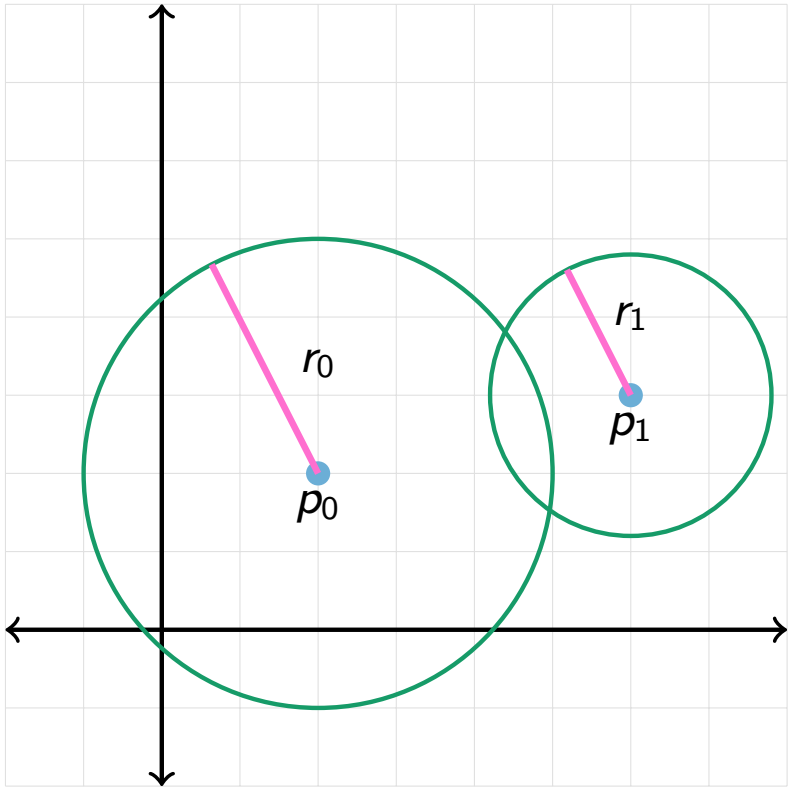
- Biểu diễn đường tròn rất đơn giản.
- Tâm $p = (x, y)$.

Đối tượng 2D: Đường tròn



- Biểu diễn đường tròn rất đơn giản.
- Tâm $p = (x, y)$.
- Và bán kính r.

Đối tượng 2D: Đường tròn



- Biểu diễn đường tròn rất đơn giản.
- Tâm $p = (x, y)$.
- Và bán kính r.

```
1 pair<point,double>
```

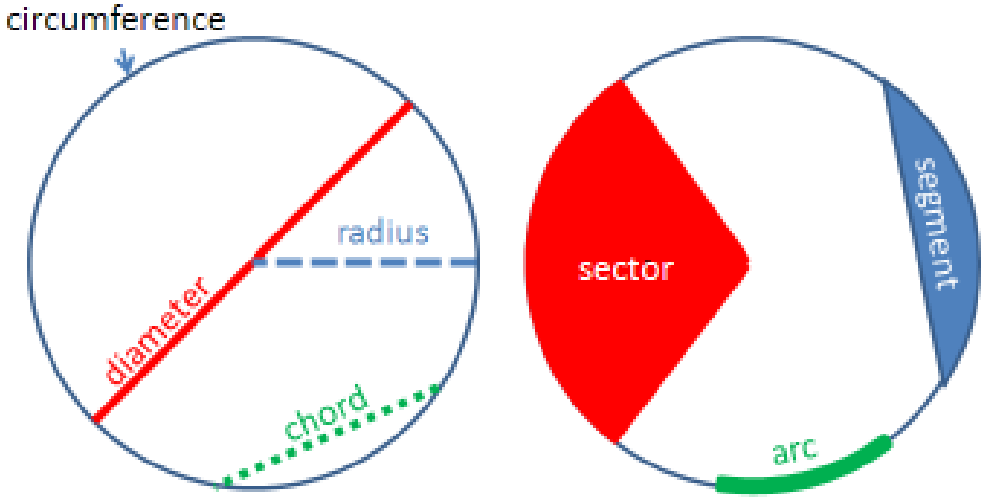
Đối tượng 2D: Đường tròn

- Trong hệ trục tọa độ đề-các 2D, một đường tròn tâm tại (a, b) với bán kính r là tập các điểm (x, y) thỏa mãn: $(x - a)^2 + (y - b)^2 = r^2$

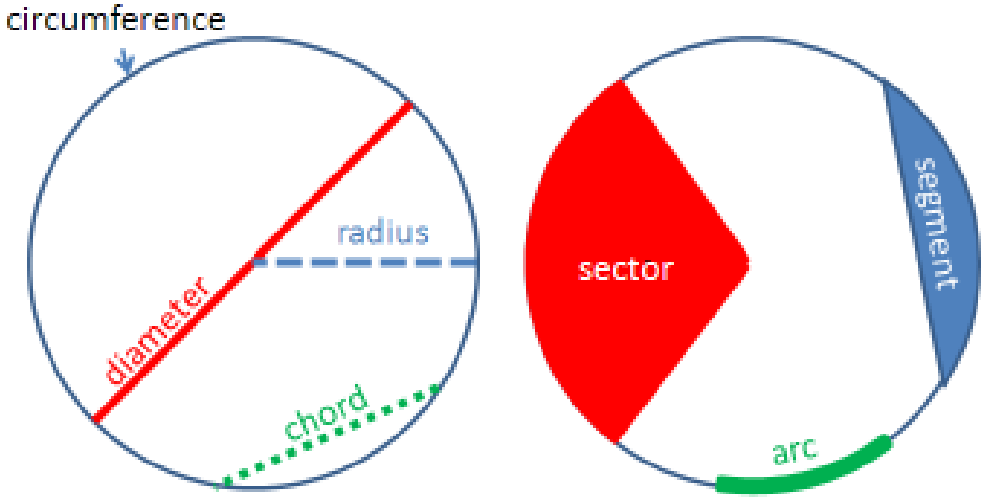
Đối tượng 2D: Đường tròn

- Trong hệ trục tọa độ đề-các 2D, một đường tròn tâm tại (a, b) với bán kính r là tập các điểm (x, y) thỏa mãn: $(x - a)^2 + (y - b)^2 = r^2$
- Chu vi đường tròn c với đường kính d là $c = \pi \times d$, với $d = 2 \times r$.

Đối tượng 2D: Đường tròn

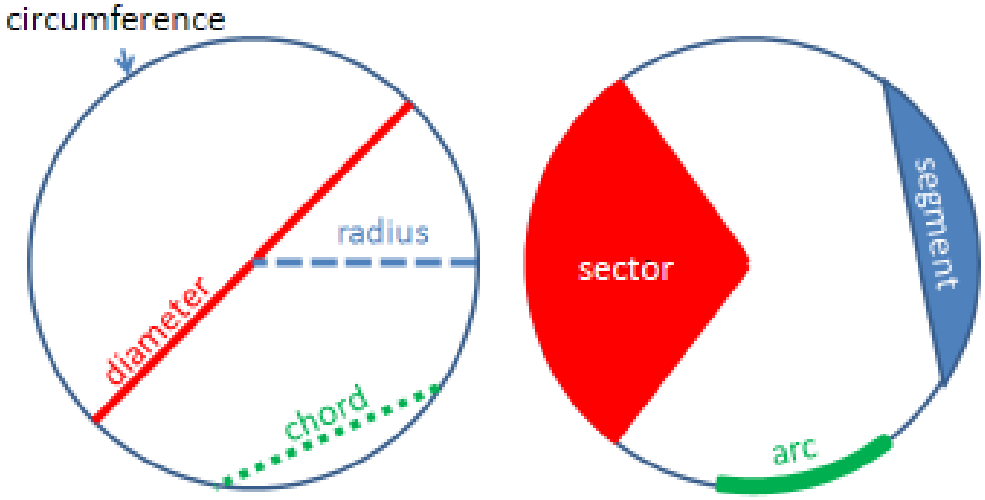


Đối tượng 2D: Đường tròn



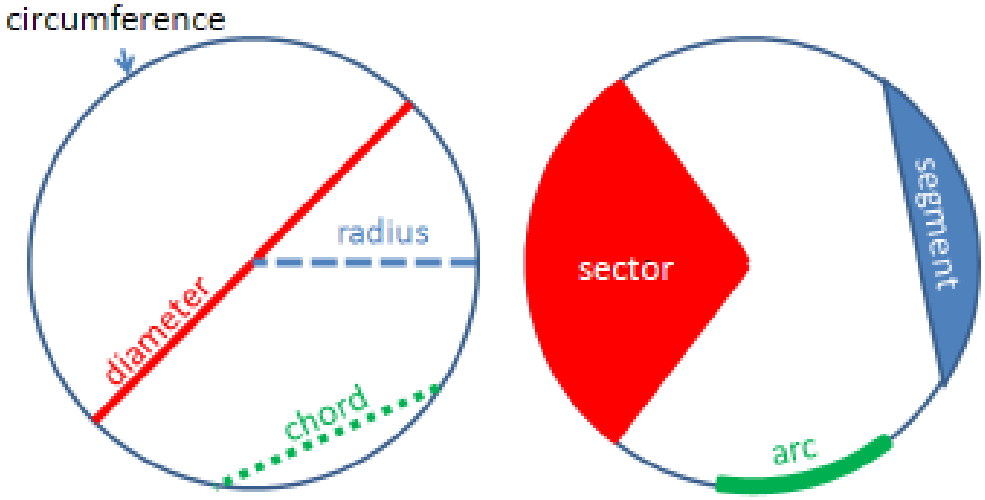
- Độ dài cung của đường tròn chu vi c và có góc α(độ):
  $arc = \frac{\alpha}{360.0} \times c$

Đối tượng 2D: Đường tròn



- Độ dài cung của đường tròn chu vi c và có góc α(độ): $arc = \frac{\alpha}{360.0} \times c$
- Độ dài dây cung của đường tròn bán kính r và có góc α(độ): chord= $2r^2 \times ((1 - cos(\alpha)))$

Đối tượng 2D: Đường tròn



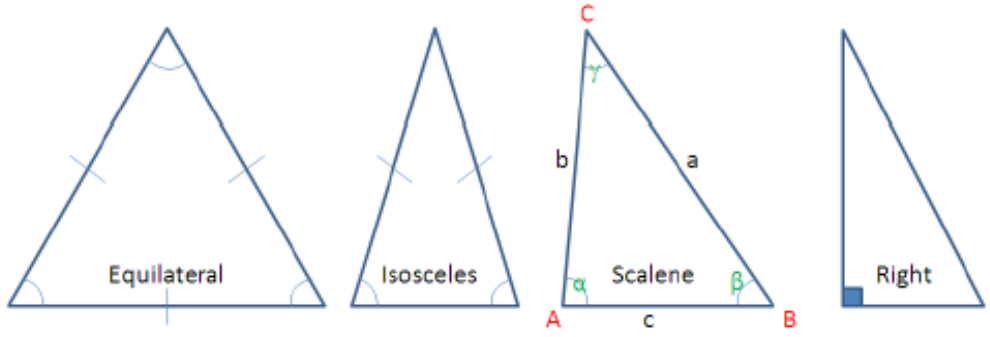
- Độ dài cung của đường tròn chu vi c và có góc α(độ): $arc = \frac{\alpha}{360.0} \times c$
- Độ dài dây cung của đường tròn bán kính r và có góc α(độ): chord= $2r^2 \times ((1 - cos(\alpha)))$
- Diện tich của một hình tròn với bán kính r: $area = \pi \times r^2$

Đối tượng 2D: Tam giác

- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.

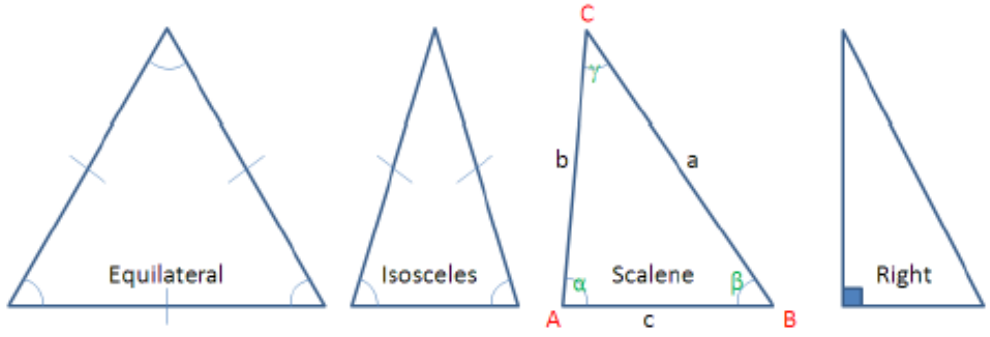
Đối tượng 2D: Tam giác

- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.
- Một vài kiểu tam giác: đều, cân, lệch, vuông .



Đối tượng 2D: Tam giác

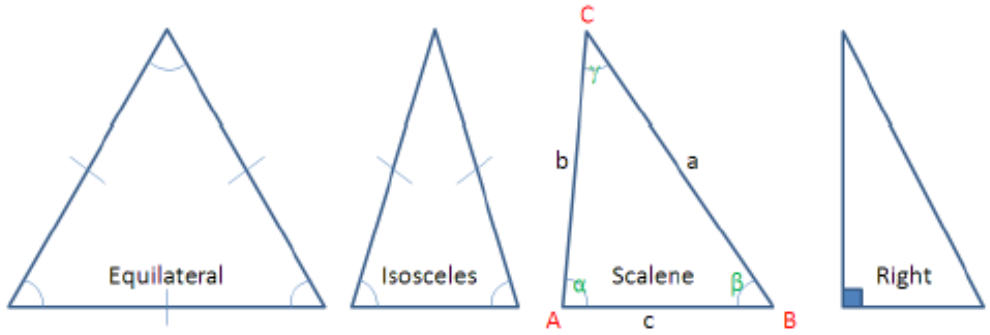
- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.
- Một vài kiểu tam giác: đều, cân, lệch, vuông .



- Diện tích *area* của tam giác với đáy b và chiều cao h: $area = 0.5 \times b \times h$.

Đối tượng 2D: Tam giác

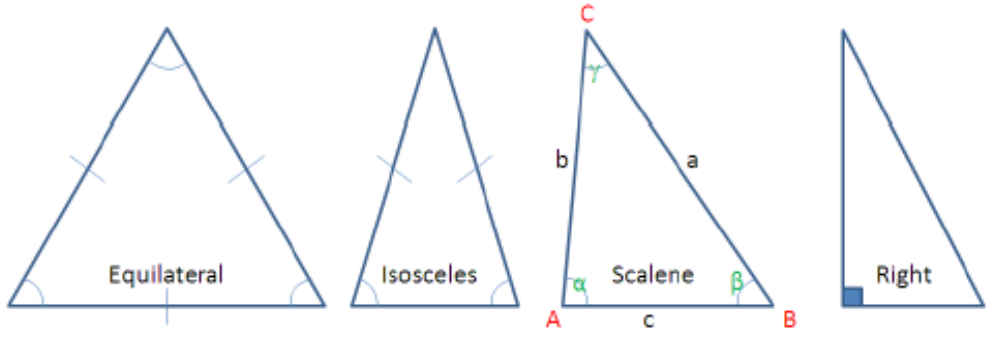
- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.
- Một vài kiểu tam giác: đều, cân, lệch, vuông .



- Diện tích *area* của tam giác với đáy b và chiều cao h: $area = 0.5 \times b \times h$.
- Chu vi p của tam giác có ba cạnh a, b, c: $p = a + b + c$.

Đối tượng 2D: Tam giác

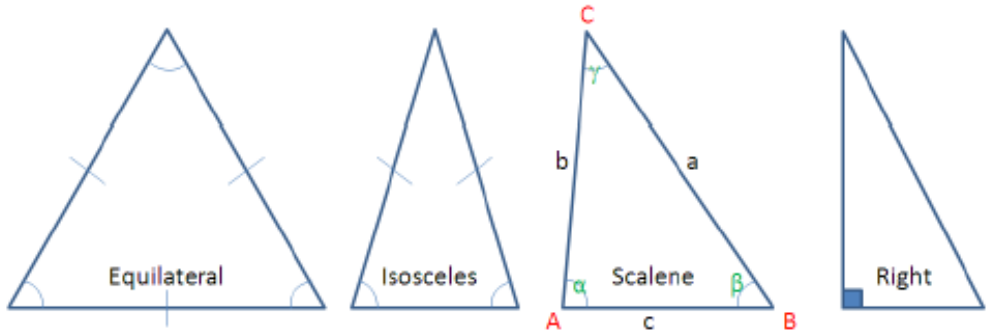
- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.
- Một vài kiểu tam giác: đều, cân, lệch, vuông .



- Diện tích *area* của tam giác với đáy b và chiều cao h: $area = 0.5 \times b \times h$.
- Chu vi p của tam giác có ba cạnh a, b, c: $p = a + b + c$.
- Công thức Heron: Diện tích *area* của tam giác với 3 cạnh a, b, c là: $area = \sqrt{(s-a) \times (s-b) \times (s-c)}$ với $s = \frac{p}{2}$.

Đối tượng 2D: Tam giác

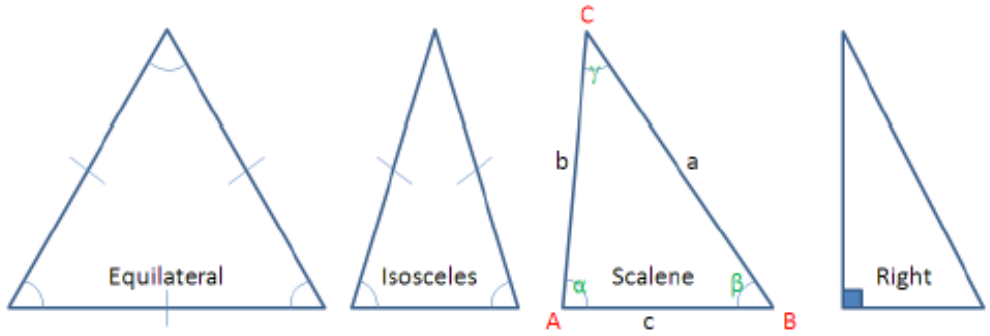
- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.



- Bán kính r của đường tròn nội tiếp tam giác có diện tích A và nửa chu vi s là : $r = \frac{A}{s}$.

Đối tượng 2D: Tam giác

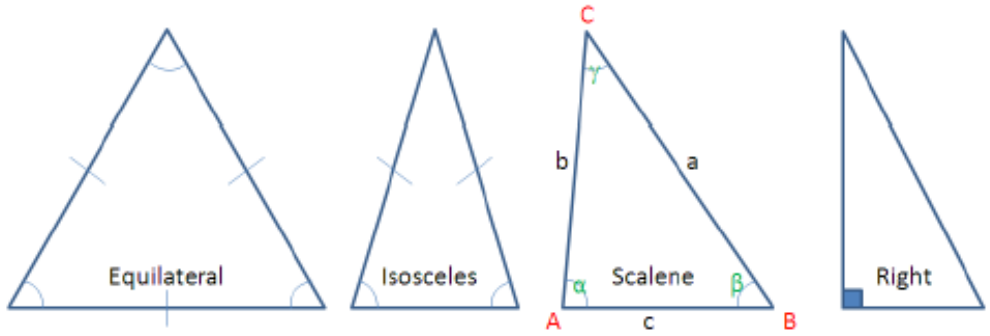
- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.



- Bán kính r của đường tròn nội tiếp tam giác có diện tích A và nửa chu vi s là : $r = \frac{A}{s}$.

Đối tượng 2D: Tam giác

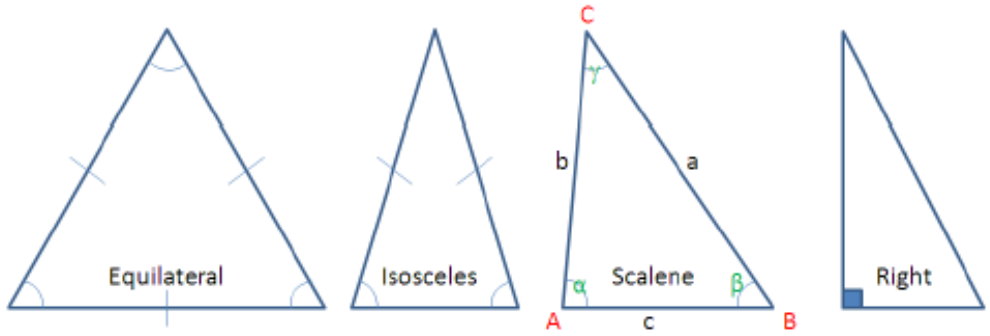
- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.



- Bán kính r của đường tròn nội tiếp tam giác có diện tích A và nửa chu vi s là : $r = \frac{A}{s}$.

Đối tượng 2D: Tam giác

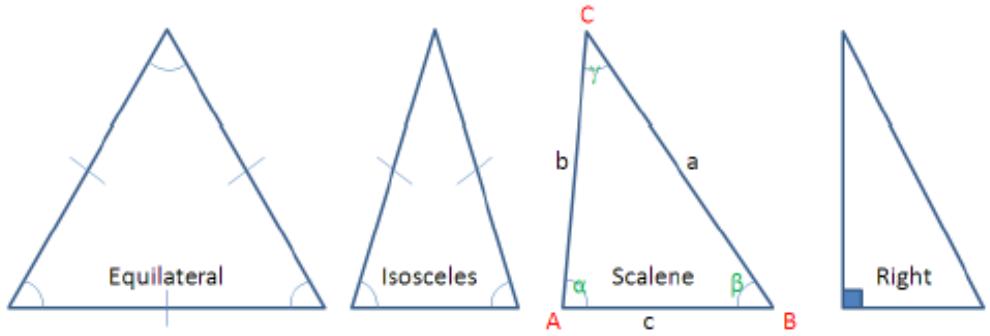
- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.



- Bán kính r của đường tròn nội tiếp tam giác có diện tích A và nửa chu vi s là : $r = \frac{A}{s}$.

Đối tượng 2D: Tam giác

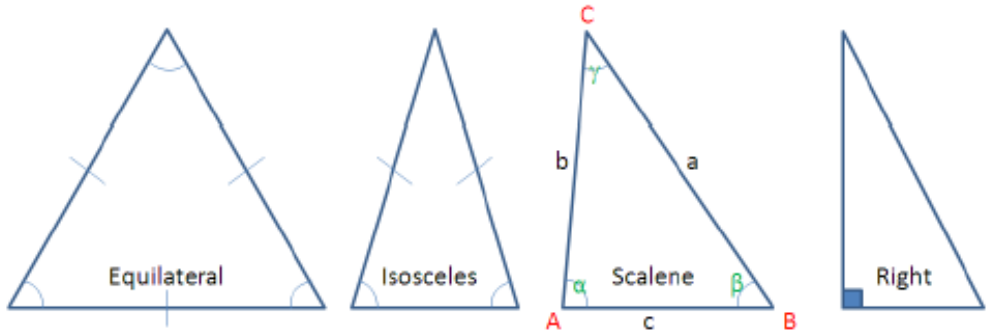
- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.



- Bán kính r của đường tròn nội tiếp tam giác có diện tích A và nửa chu vi s là : $r = \frac{A}{s}$.

Đối tượng 2D: Tam giác

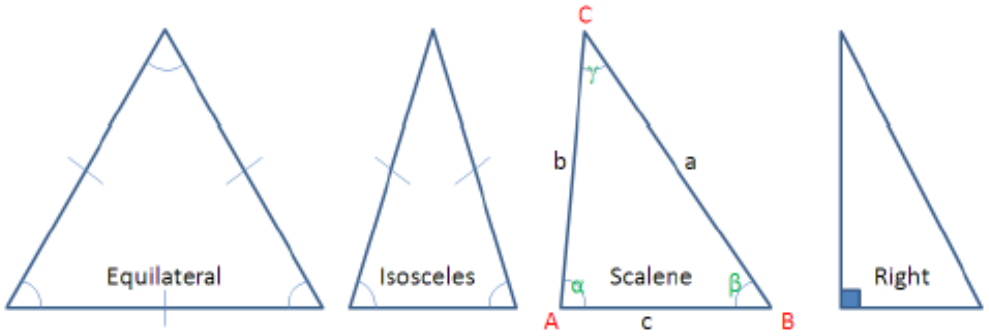
- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.



- Bán kính r của đường tròn nội tiếp tam giác có diện tích A và nửa chu vi s là : $r = \frac{A}{s}$.

Đối tượng 2D: Tam giác

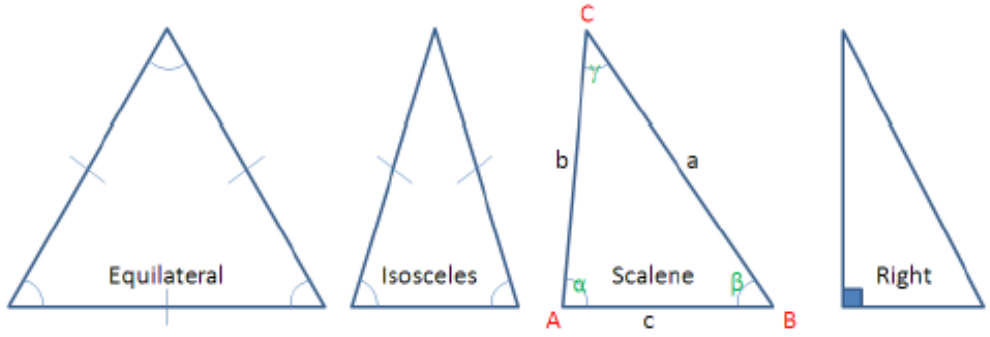
- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.



- Bán kính r của đường tròn nội tiếp tam giác có diện tích A và nửa chu vi s là : $r = \frac{A}{s}$.
- Bán kính R của đường tròn ngoại tiếp tam giác có 3 cạnh a, b, c và diện tích $area$ là : $R = \frac{a \times b \times c}{4 \times area}$

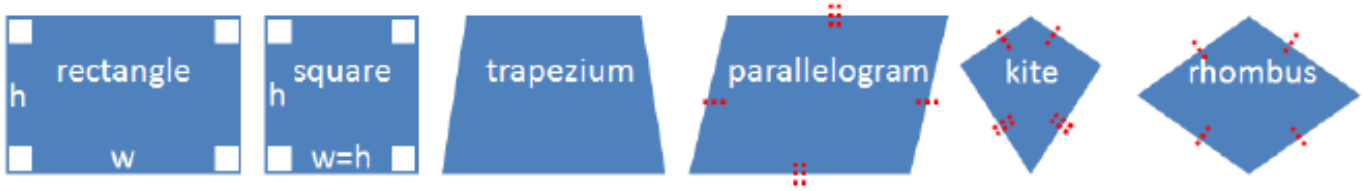
Đối tượng 2D: Tam giác

- Tam giác là một đa giác có 3 đỉnh A, B, C và 3 cạnh độ dài a, b, c.

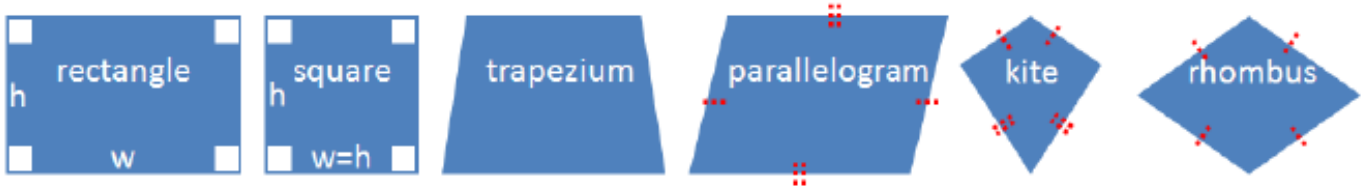


- Bán kính r của đường tròn nội tiếp tam giác có diện tích A và nửa chu vi s là : $r = \frac{A}{s}$.
- Bán kính R của đường tròn ngoại tiếp tam giác có 3 cạnh a, b, c và diện tích $area$ là : $R = \frac{a \times b \times c}{4 \times area}$
- Định lý Cô-sin: $c^2 = a^2 + b^2 - 2 \times a \times b \times cos(\gamma)$ với γlà góc đối của cạnh c.
- Định lý Sin: $\frac{a}{sin(\alpha)} = \frac{b}{sin(\beta)} = \frac{c}{sin(\gamma)}$

Đối tượng 2D: Tứ giác

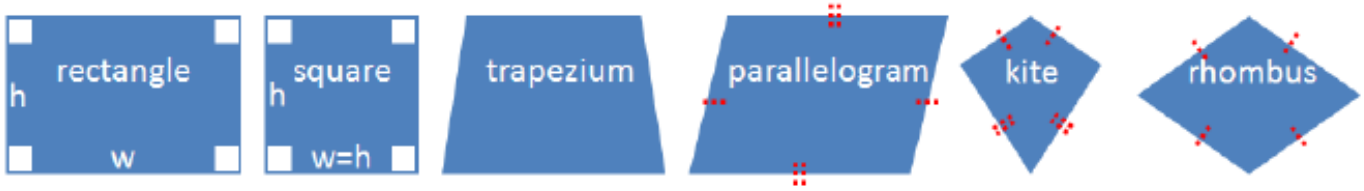


Đối tượng 2D: Tứ giác



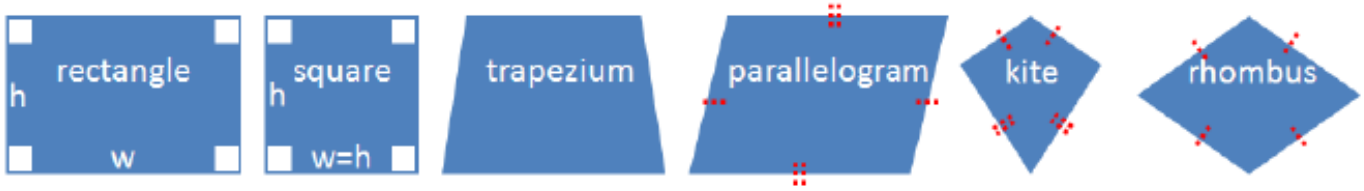
- Tứ giác là một đa giác có 4 đỉnh, và 4 cạnh.

Đối tượng 2D: Tứ giác



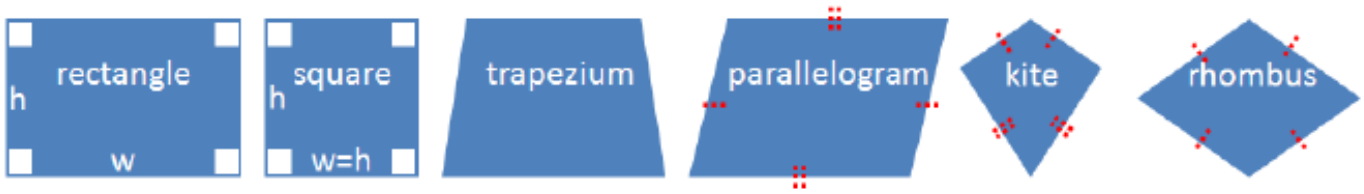
- Tứ giác là một đa giác có 4 đỉnh, và 4 cạnh.
- Hình chữ nhật là một tứ giác có bốn góc vuông.

Đối tượng 2D: Tứ giác



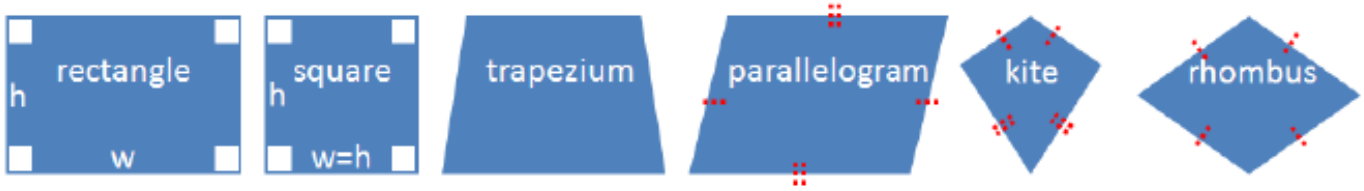
- Tứ giác là một đa giác có 4 đỉnh, và 4 cạnh.
- Hình chữ nhật là một tứ giác có bốn góc vuông.
- Một hình chữ nhật có chiều rộng w và chiều cao h có diện tích $A = w \times h$ và có chu vi $2 \times (w + h)$.

Đối tượng 2D: Tứ giác



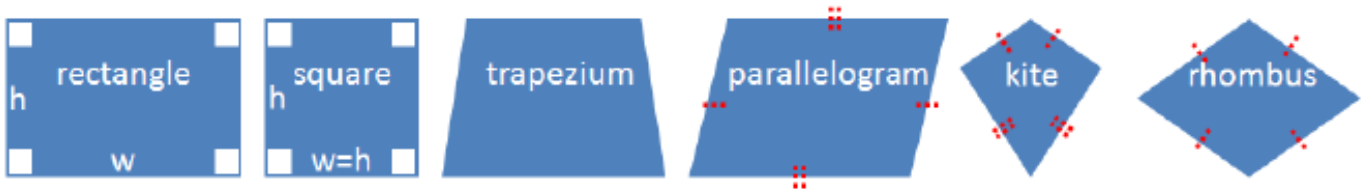
- Tứ giác là một đa giác có 4 đỉnh, và 4 cạnh.
- Hình chữ nhật là một tứ giác có bốn góc vuông.
- Một hình chữ nhật có chiều rộng w và chiều cao h có diện tích $A = w \times h$ và có chu vi $2 \times (w + h)$.
- Hình vuông là một trường hợp đặc biệt của hình chữ nhật với $w = h$.

Đối tượng 2D: Tứ giác



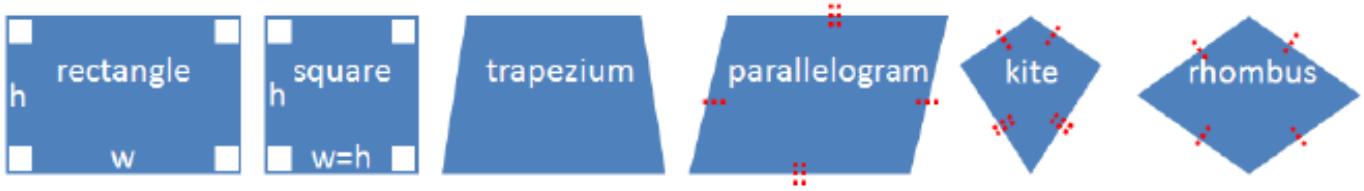
- Tứ giác là một đa giác có 4 đỉnh, và 4 cạnh.
- Hình chữ nhật là một tứ giác có bốn góc vuông.
- Một hình chữ nhật có chiều rộng w và chiều cao h có diện tích $A = w \times h$ và có chu vi $2 \times (w + h)$.
- Hình vuông là một trường hợp đặc biệt của hình chữ nhật với $w = h$.
- Hình thang là một tứ giác có một cặp cạnh song song. Nếu hai cạnh không song song có cùng chiều dài thì là một hình thang cân.

Đối tượng 2D: Tứ giác



- Tứ giác là một đa giác có 4 đỉnh, và 4 cạnh.
- Hình chữ nhật là một tứ giác có bốn góc vuông.
- Một hình chữ nhật có chiều rộng w và chiều cao h có diện tích $A = w \times h$ và có chu vi $2 \times (w + h)$.
- Hình vuông là một trường hợp đặc biệt của hình chữ nhật với $w = h$.
- Hình thang là một tứ giác có một cặp cạnh song song. Nếu hai cạnh không song song có cùng chiều dài thì là một hình thang cân.
- Hình thang với cặp cạnh song song có độ dài w_1 và w_2 , và chiều cao h, thì có diện tích là $A = 0.5 \times (w_1 + w_2) \times h$.

Đối tượng 2D: Tứ giác



- Tứ giác là một đa giác có 4 đỉnh, và 4 cạnh.
- Hình chữ nhật là một tứ giác có bốn góc vuông.
- Một hình chữ nhật có chiều rộng w và chiều cao h có diện tích $A = w \times h$ và có chu vi $2 \times (w + h)$.
- Hình vuông là một trường hợp đặc biệt của hình chữ nhật với $w = h$.
- Hình thang là một tứ giác có một cặp cạnh song song. Nếu hai cạnh không song song có cùng chiều dài thì là một hình thang cân.
- Hình thang với cặp cạnh song song có độ dài w_1 và w_2 , và chiều cao h, thì có diện tích là $A = 0.5 \times (w_1 + w_2) \times h$.
- Hình bình hành là một tứ giác có 2 cạnh đối diện luôn song song với nhau.

Đối tượng 2D: Đa giác

Đối tượng 2D: Đa giác

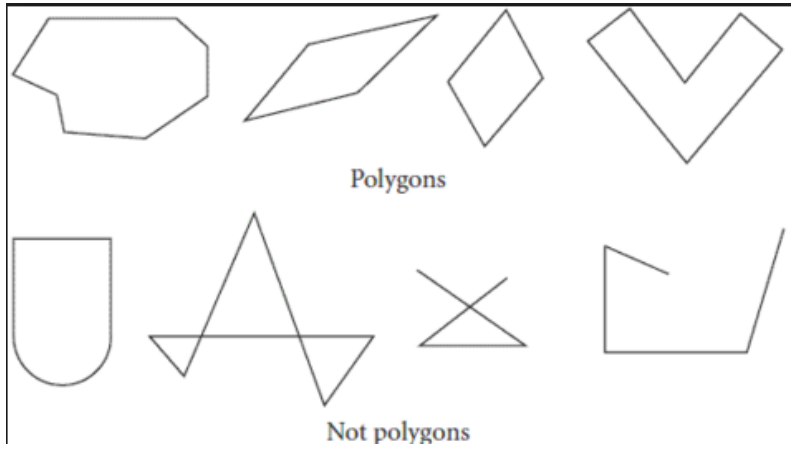
- Đa giác là một đường gấp khúc phẳng khép kín, nghĩa là gồm những đoạn thẳng nối tiếp nhau (mỗi điểm nối là đầu mút của vừa đúng hai đoạn thẳng) cùng nằm trên một mặt phẳng và khép kín (điểm nối đầu trùng với điểm nối cuối). Phần mặt phẳng giới hạn bởi các cạnh đa giác được gọi là hình đa giác.

Đối tượng 2D: Đa giác

- Đa giác là một đường gấp khúc phẳng khép kín, nghĩa là gồm những đoạn thẳng nối tiếp nhau (mỗi điểm nối là đầu mút của vừa đúng hai đoạn thẳng) cùng nằm trên một mặt phẳng và khép kín (điểm nối đầu trùng với điểm nối cuối). Phần mặt phẳng giới hạn bởi các cạnh đa giác được gọi là hình đa giác.
- Ta định nghĩa đa giác bao gồm các cạnh (ranh giới) và cả miền trong của nó.

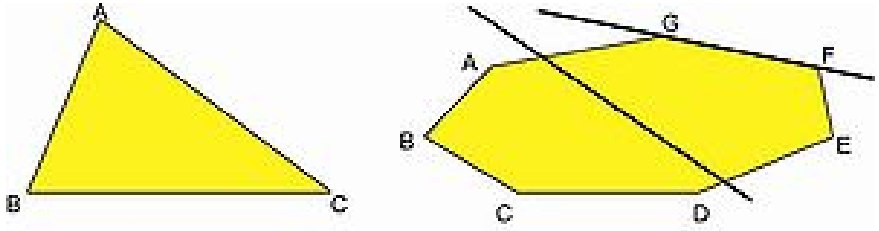
Đối tượng 2D: Đa giác

- Đa giác là một đường gấp khúc phẳng khép kín, nghĩa là gồm những đoạn thẳng nối tiếp nhau (mỗi điểm nối là đầu mút của vừa đúng hai đoạn thẳng) cùng nằm trên một mặt phẳng và khép kín (điểm nối đầu trùng với điểm nối cuối). Phần mặt phẳng giới hạn bởi các cạnh đa giác được gọi là hình đa giác.
- Ta định nghĩa đa giác bao gồm các cạnh (ranh giới) và cả miền trong của nó. Và khi nói một điểm thuộc đa giác thì có nghĩa là nó nằm trên cạnh của đa giác hoặc nằm ở miền trong của đa giác.



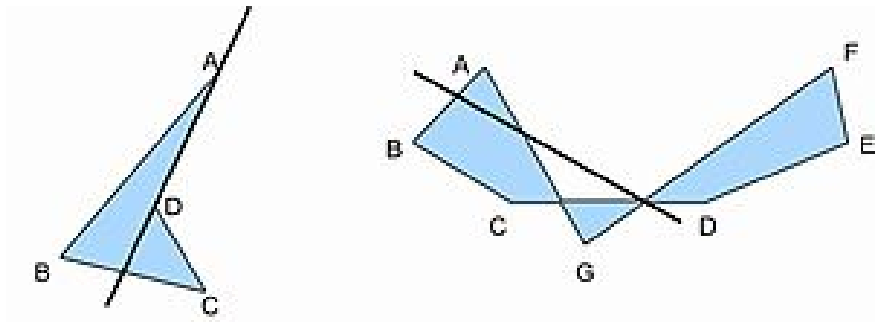
Khái niệm Đa giác

- Đa giác lồi : toàn bộ đa giác nằm về một phía của đường thẳng chứa một cạnh bất kỳ của đa giác.



Đa giác lồi-*Convex polygon*

- Đa giác lõm : đa giác nằm về hai phía của ít nhất một đường thẳng chứa một cạnh bất kỳ.



Đa giác lõm đơn
Simple concave polygon

Đa giác lõm phức
Simple complex polygon

Đa giác lõm-*Concave polygon*

Biểu diễn đa giác

- Đa giác đơn giản được biểu diễn bằng cách liệt kê các đỉnh (theo chiều kim đồng hồ hoặc theo chiều ngược kim đồng hồ):

```
40 // da giac voi 6 diem
41 vector<point> P;
42 P.push_back(point(1, 1)); // P0
43 P.push_back(point(3, 3)); // P1
44 P.push_back(point(9, 1)); // P2
45 P.push_back(point(12, 4)); // P3
46 P.push_back(point(9, 7)); // P4
47 P.push_back(point(1, 7)); // P5
```

Hình học

Tích vô hướng hai vec-tơ

Cho hai vec-tơ:

$$\vec{u} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \qquad \vec{v} = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$$

tích vô hướng của $\vec{u}$ và $\vec{v}$ được định nghĩa như sau:

$$\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = x_0 \cdot x_1 + y_0 \cdot y_1$$

Tích vô hướng hai vec-tơ

Cho hai vec-tơ:

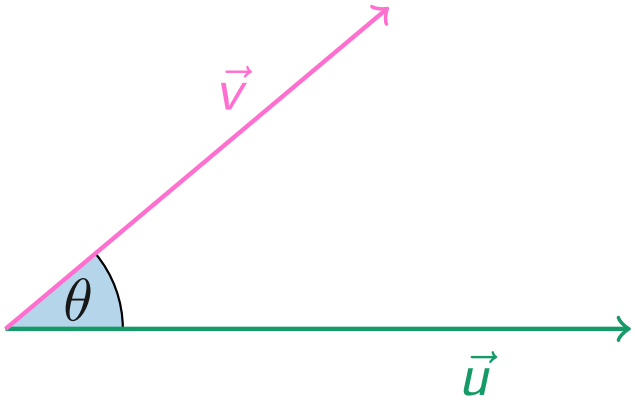
$$\vec{u} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \qquad \vec{v} = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$$

tích vô hướng của $\vec{u}$ và $\vec{v}$ được định nghĩa như sau:

$$\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = x_0 \cdot x_1 + y_0 \cdot y_1$$

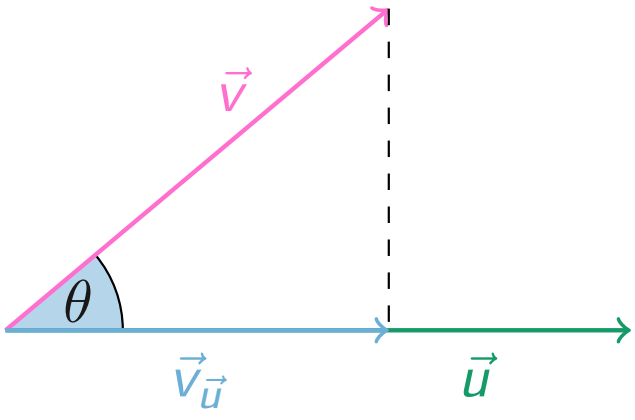
Trong thuật ngữ hình học thì:

$$\vec{u} \cdot \vec{v} = |\vec{u}||\vec{v}| \cos\theta$$



- Tính góc giữa $\vec{u}$ và $\vec{v}$:

$$\theta = \arccos\left(\frac{\vec{u}\cdot\vec{v}}{|\vec{u}||\vec{v}|}\right)$$

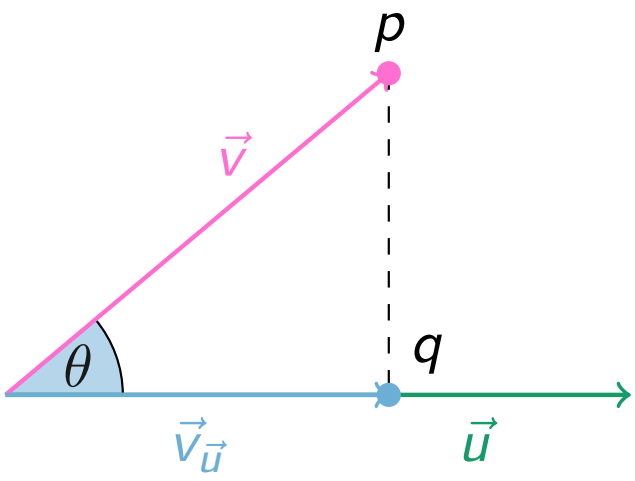


- Tính góc giữa $\vec{u}$ và $\vec{v}$:

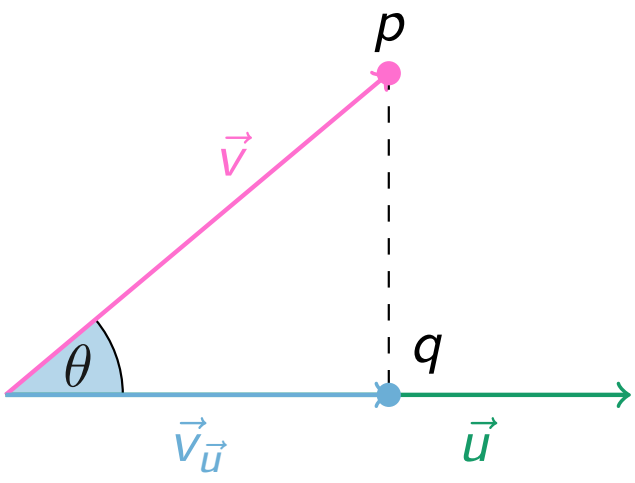
$$\theta = \arccos\left(\frac{\vec{u}\cdot\vec{v}}{|\vec{u}||\vec{v}|}\right)$$

- và hình chiếu của $\vec{v}$ lên $\vec{u}$:

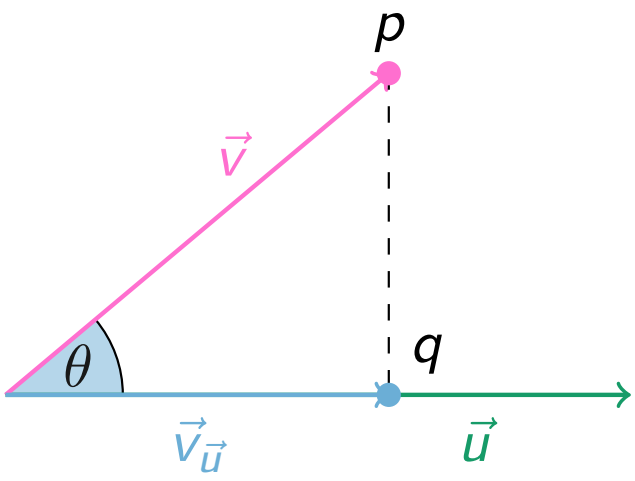
$$\vec{v}_{\vec{u}} = \left(\frac{\vec{u}\cdot\vec{v}}{|u|}\right)\vec{u}$$



- Điểm gần nhất trên $\vec{u}$ với p là q.



- Điểm gần nhất trên $\vec{u}$ với p là q.
- Khoảng cách giữa p tới $\vec{u}$ là khoảng cách từ p tới q.



- Điểm gần nhất trên $\vec{u}$ với p là q.
- Khoảng cách giữa p tới $\vec{u}$ là khoảng cách từ p tới q.
- Trừ khi q nằm ngoài $\vec{u}$, điểm gần nhất là một trong hai đầu mút.

Cài đặt với lớp complex.

```
#define P(p) const point &p
#define L(p0, p1) P(p0), P(p1)
double dot(P(a), P(b)) {
    return real(a) * real(b) + imag(a) * imag(b);
}
double angle(P(a), P(b), P(c)) {
    return acos(dot(b - a, c - b) / abs(b - a) / abs(c - b));
}
point closest_point(L(a, b), P(c), bool segment = false) {
    if (segment) {
        if (dot(b - a, c - b) > 0) return b;
        if (dot(a - b, c - a) > 0) return a;
    }
    double t = dot(c - a, b - a) / norm(b - a);
    return a + t * (b - a);
}
```

Tích có hướng hai vec-tơ

Cho hai vec-tơ:

$$\vec{u} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \qquad \vec{v} = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$$

tích có hướng của $\vec{u}$ và $\vec{v}$ là một "giả" vec-tơ vuông góc với mặt phẳng chứa $\vec{u}, \vec{v}$ có tọa độ trục Z là:

$$\left| \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \times \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} \right| = x_0 \cdot y_1 - y_0 \cdot x_1$$

Tích có hướng hai vec-tơ

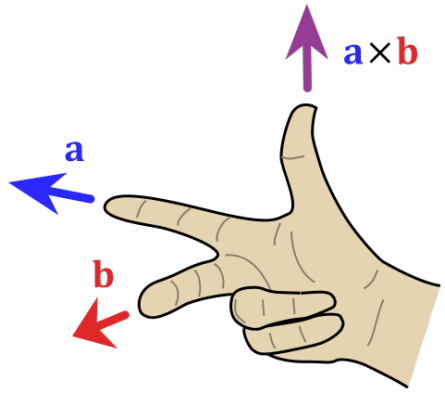
Cho hai vec-tơ:

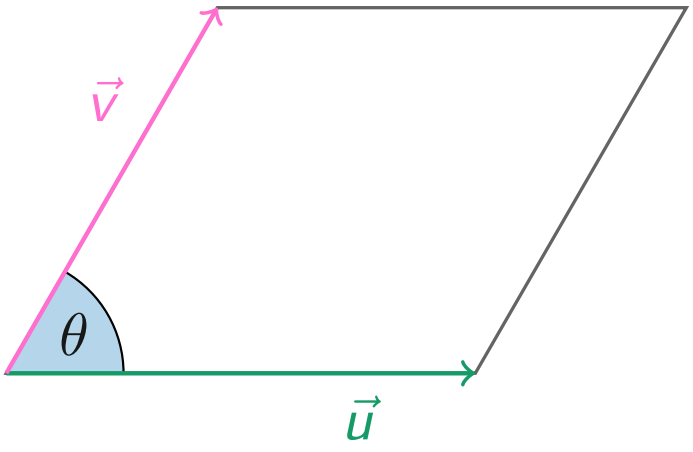
$$\vec{u} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \qquad \vec{v} = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$$

tích có hướng của $\vec{u}$ và $\vec{v}$ là một "giả" vec-tơ vuông góc với mặt phẳng chứa $\vec{u}, \vec{v}$ có tọa độ trục Z là:

$$\left| \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \times \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} \right| = x_0 \cdot y_1 - y_0 \cdot x_1$$

với hướng tuân theo quy tắc bàn tay phải:





- Theo thuật ngữ hình học thì là diện tích hình bình hành tương ứng:

$$|\vec{u} \times \vec{v}| = |\vec{u}||\vec{v}| \sin\theta$$

- Tính diện tích tam giác tạo bởi $\vec{u}$ và $\vec{v}$

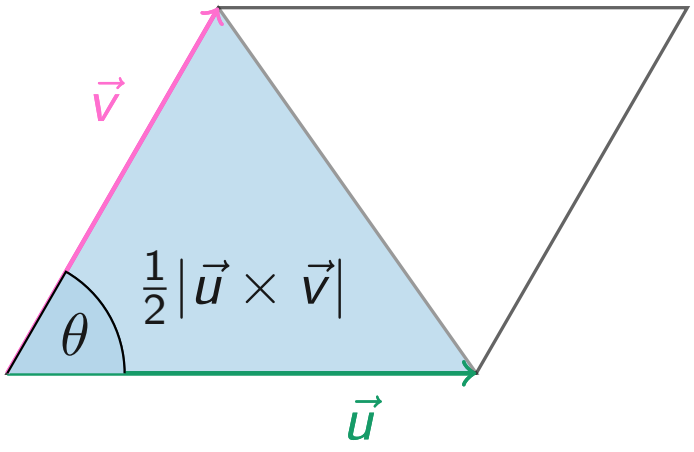
$$\frac{|\vec{u} \times \vec{v}|}{2}$$

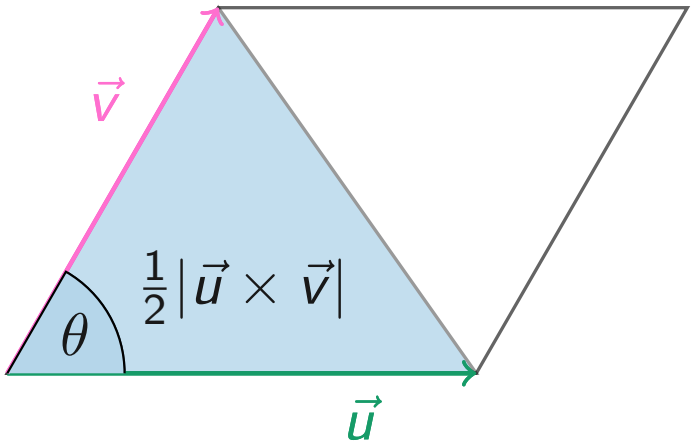
- Theo thuật ngữ hình học thì là diện tích hình bình hành tương ứng:

$$|\vec{u} \times \vec{v}| = |\vec{u}||\vec{v}| \sin\theta$$

- Tính diện tích tam giác tạo bởi $\vec{u}$ và $\vec{v}$

$$\frac{|\vec{u} \times \vec{v}|}{2}$$





- Theo thuật ngữ hình học thì là diện tích hình bình hành tương ứng:

$$|\vec{u} \times \vec{v}| = |\vec{u}||\vec{v}| \sin\theta$$

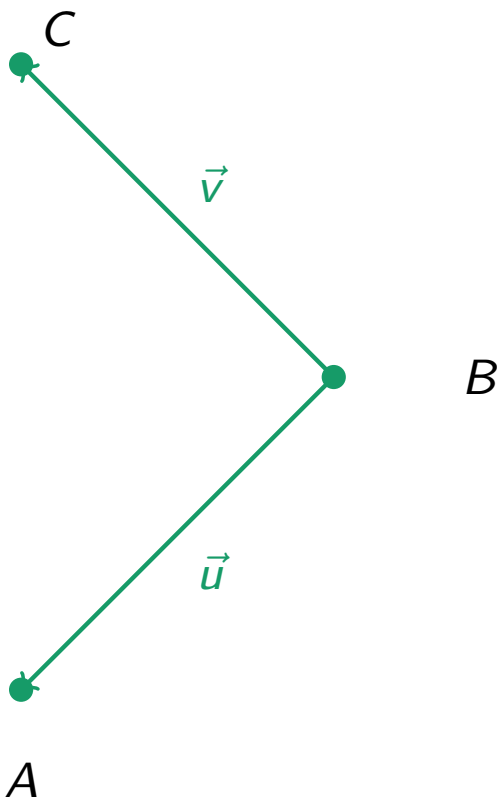
- Tính diện tích tam giác tạo bởi $\vec{u}$ và $\vec{v}$

$$\frac{|\vec{u} \times \vec{v}|}{2}$$

- và cho biết góc giữa $\vec{u}$ và $\vec{v}$ là dương hay âm:

$$\begin{aligned} |\vec{u} \times \vec{v}| < 0 \quad &\text{khi và chỉ khi} \quad \theta < \pi \\ |\vec{u} \times \vec{v}| = 0 \quad &\text{khi và chỉ khi} \quad \theta = \pi \\ |\vec{u} \times \vec{v}| > 0 \quad &\text{khi và chỉ khi} \quad \theta > \pi \end{aligned}$$

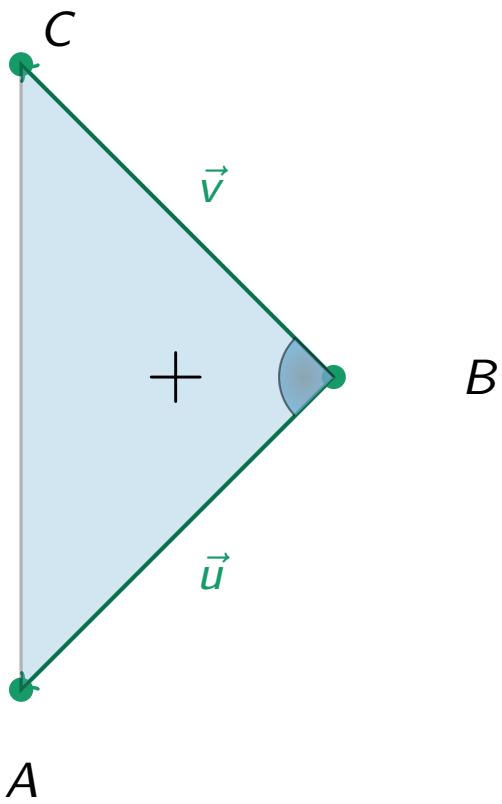
Kiểm tra ngược chiều kim đồng hồ CCW



- Với ba điểm A, B và C, làm thế nào để biết chúng đi theo một góc ngược chiều kim đồng hồ theo thứ tự đó hay không?

$$A \to B \to C$$

Kiểm tra ngược chiều kim đồng hồ CCW



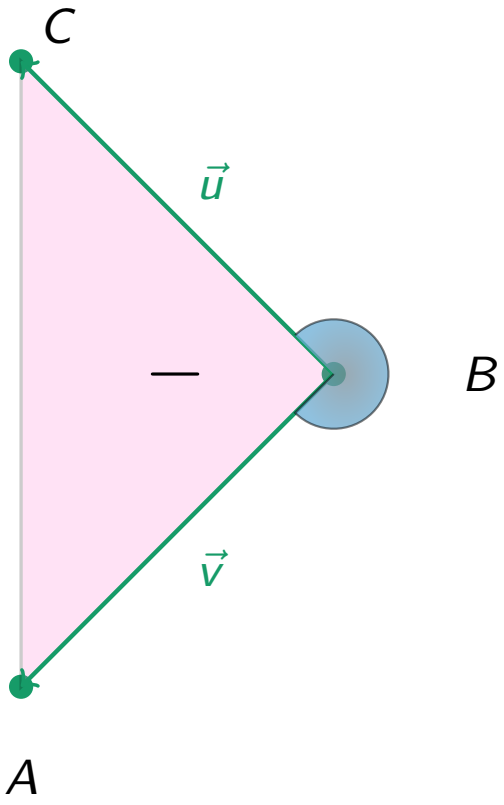
- Với ba điểm A, B và C, làm thế nào để biết chúng đi theo một góc ngược chiều kim đồng hồ theo thứ tự đó hay không?

$$A \to B \to C$$

- Hãy kiểm tra tích có hướng và diện tích tam giác tạo bởi hai vec-tơ:

$$\vec{u} = B - A \quad \vec{v} = B - C$$
$$|\vec{u} \times \vec{v}| > 0$$

Kiểm tra ngược chiều kim đồng hồ CCW



- Các điểm này theo thứ tự ngược lại không tạo thành góc ngược chiều kim đồng hồ:

$$C \to B \to A$$

- Theo thứ tự ngược lại, các vec-tơ thay đổi vị trí:

$$\vec{u} = B - C \quad \vec{v} = B - A$$
$$|\vec{u} \times \vec{v}| < 0$$

Kiểm tra ngược chiều kim đồng hồ CCW

- Các điểm này theo thứ tự ngược lại không tạo thành góc ngược chiều kim đồng hồ:

$$C \to B \to A$$

- Theo thứ tự ngược lại, các vec-tơ thay đổi vị trí:

$$\vec{u} = B - C \quad \vec{v} = B - A$$
$$|\vec{u} \times \vec{v}| < 0$$

- Nếu các điểm A, B và C trên cùng một dòng thì diện tích sẽ bằng 0.

```
double cross(P(a), P(b)) {
    return real(a)*imag(b) - imag(a)*real(b);
}
double ccw(P(a), P(b), P(c)) {
    return cross(b - a, c - b);
}
bool collinear(P(a), P(b), P(c)) {
    return abs(ccw(a, b, c)) < EPS;
}
```

Giao điểm đường thẳng, đoạn thẳng

Một bài toán rất hay dùng là tìm giao điểm của hai đường thẳng hoặc hai đoạn thẳng.

Giao điểm đường thẳng, đoạn thẳng

Một bài toán rất hay dùng là tìm giao điểm của hai đường thẳng hoặc hai đoạn thẳng.

- Tìm phương trình đường thẳng $Ax + By = C$ đi qua cặp điểm (x_0, y_0), (x_1, y_1).

Giao điểm đường thẳng, đoạn thẳng

Một bài toán rất hay dùng là tìm giao điểm của hai đường thẳng hoặc hai đoạn thẳng.

- Tìm phương trình đường thẳng $Ax + By = C$ đi qua cặp điểm (x_0, y_0), (x_1, y_1).
- Các tham số A, B, C sẽ là:

$$A = y_1 - y_0$$
$$B = x_0 - x_1$$
$$C = A \cdot x_0 + B \cdot y_1$$

Giao điểm đường thẳng, đoạn thẳng

Một bài toán rất hay dùng là tìm giao điểm của hai đường thẳng hoặc hai đoạn thẳng.

- Tìm phương trình đường thẳng $Ax + By = C$ đi qua cặp điểm (x_0, y_0), (x_1, y_1).
- Các tham số A, B, C sẽ là:

$$A = y_1 - y_0$$
$$B = x_0 - x_1$$
$$C = A \cdot x_0 + B \cdot y_1$$

- Với hai đường thẳng cho bởi dạng phương trình như trên, giao điểm của chúng là lời giải của hệ hai phương trình hai ẩn số x và y.

Với hai đường thẳng

$$A_0x + B_0y = C_0$$
$$A_1x + B_1y = C_1$$

giao điểm của chúng là

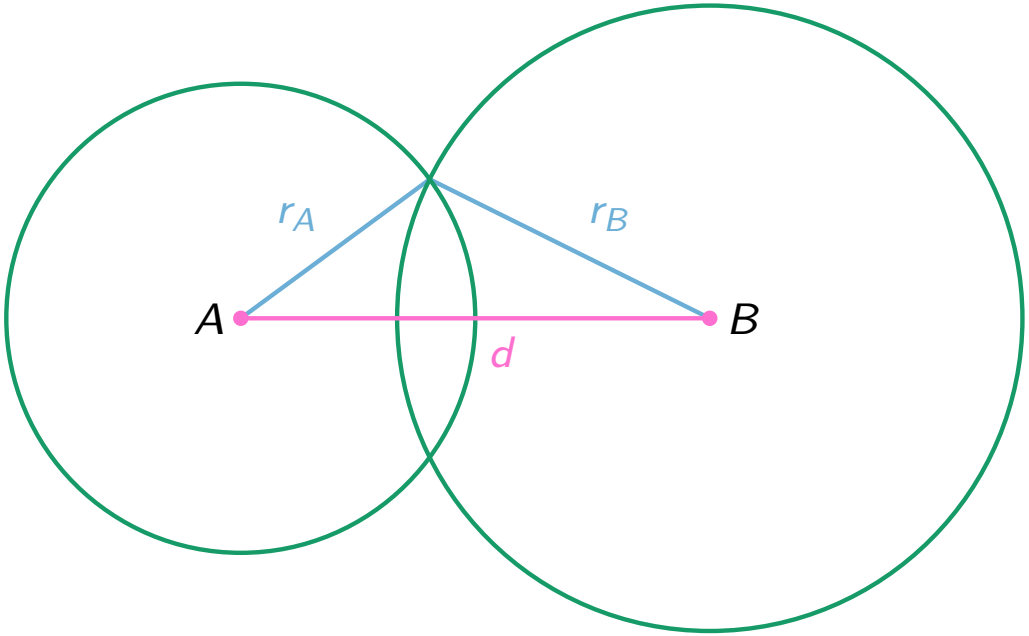
$$x = \frac{(B_1 \cdot C_0 - B_0 \cdot C_1)}{D}$$
$$y = \frac{(A_0 \cdot C_1 - A_1 \cdot C_0)}{D}$$

với

$$D = A_0 \cdot B_1 - A_1 \cdot B_0$$

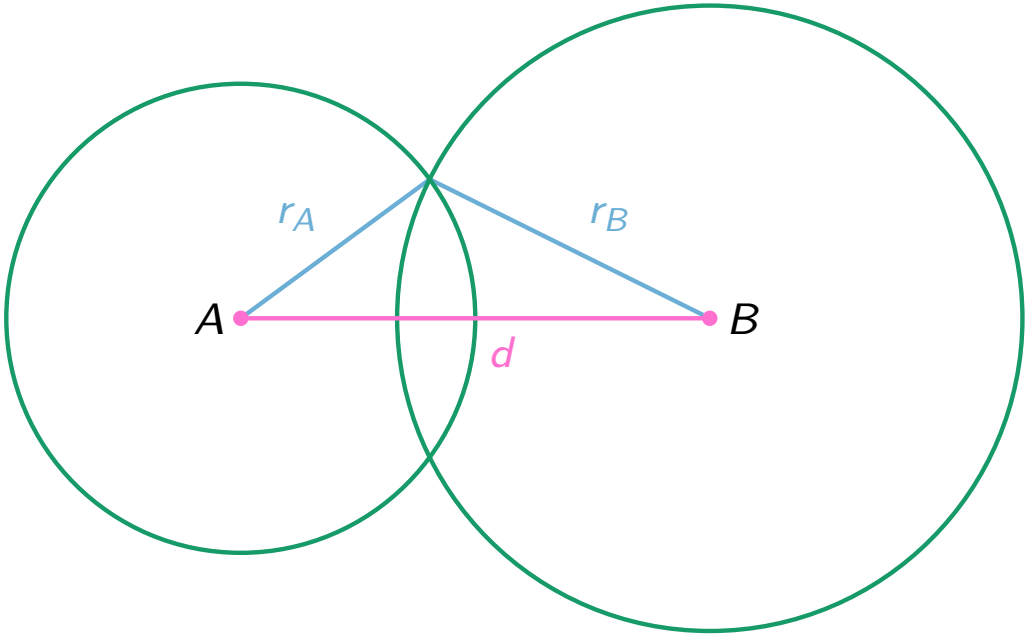
Giao điểm hai đường tròn

Bài toán tương tự là tìm các điểm giao nhau của hai đường tròn.



Giao điểm hai đường tròn

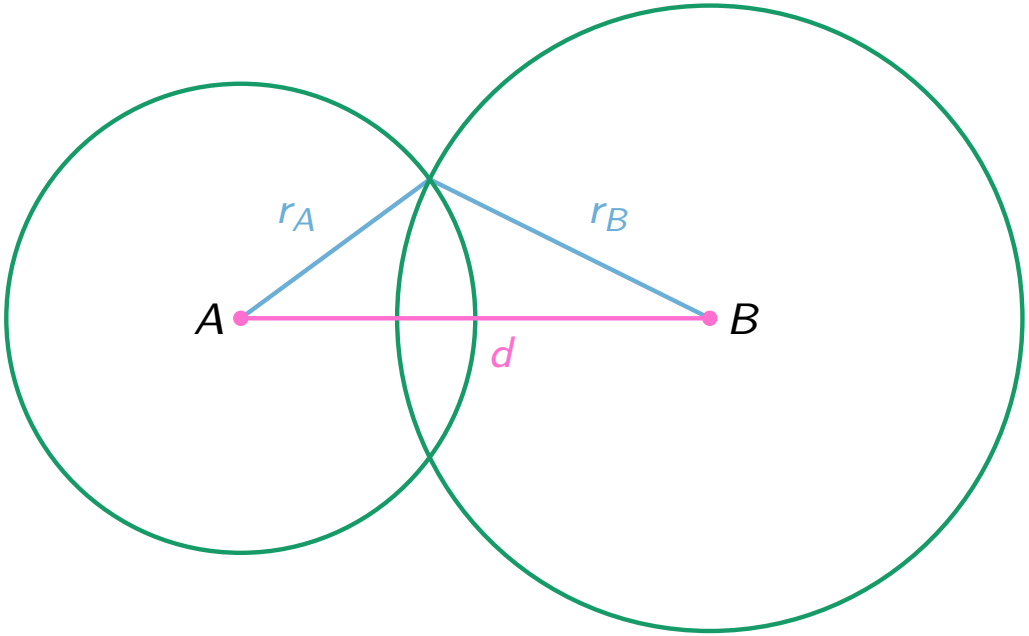
Bài toán tương tự là tìm các điểm giao nhau của hai đường tròn.



- Nếu $d > r_A + r_B$ hai đường tròn không giao nhau.

Giao điểm hai đường tròn

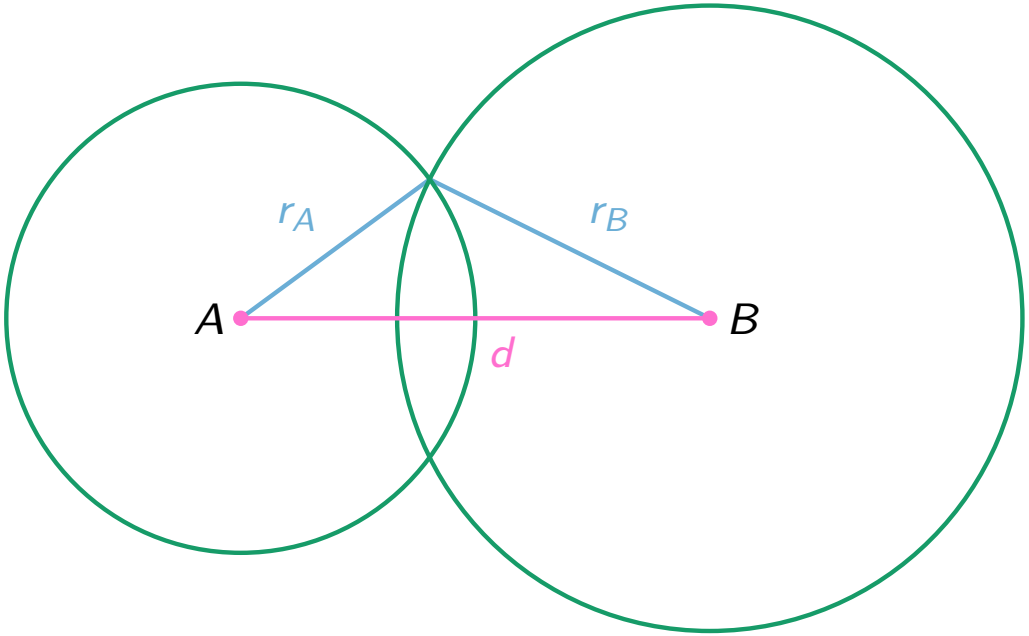
Bài toán tương tự là tìm các điểm giao nhau của hai đường tròn.



- Nếu $d > r_A + r_B$ hai đường tròn không giao nhau.
- Nếu $d < |r_A - r_B|$, một đường tròn nằm trong đường tròn còn lại.

Giao điểm hai đường tròn

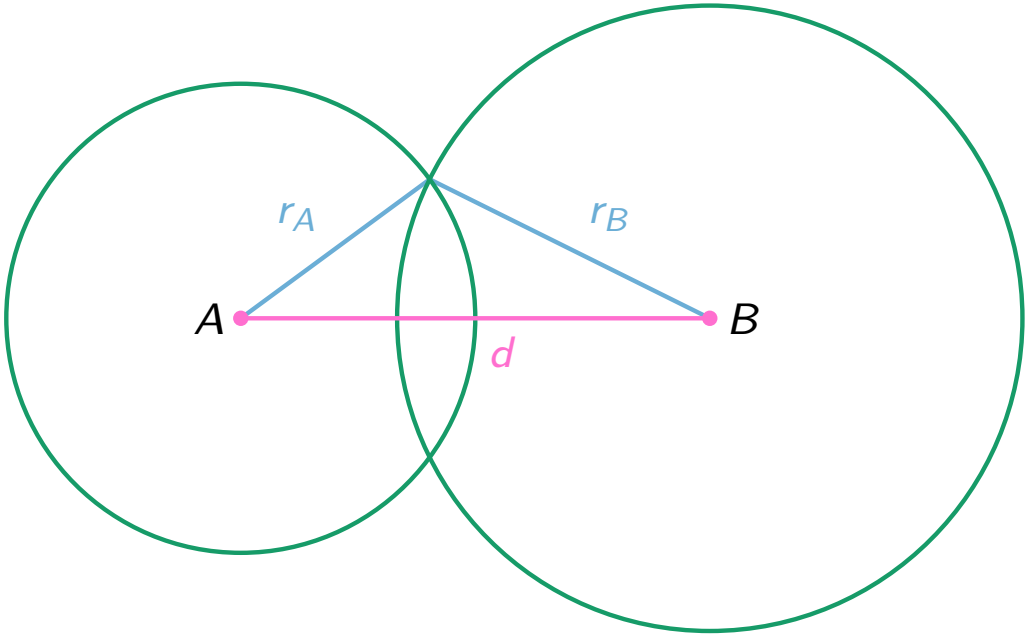
Bài toán tương tự là tìm các điểm giao nhau của hai đường tròn.



- Nếu $d > r_A + r_B$ hai đường tròn không giao nhau.
- Nếu $d < |r_A - r_B|$, một đường tròn nằm trong đường tròn còn lại.
- Nếu $d = 0$ và $r_A = r_B$, hai đường tròn là một.

Giao điểm hai đường tròn

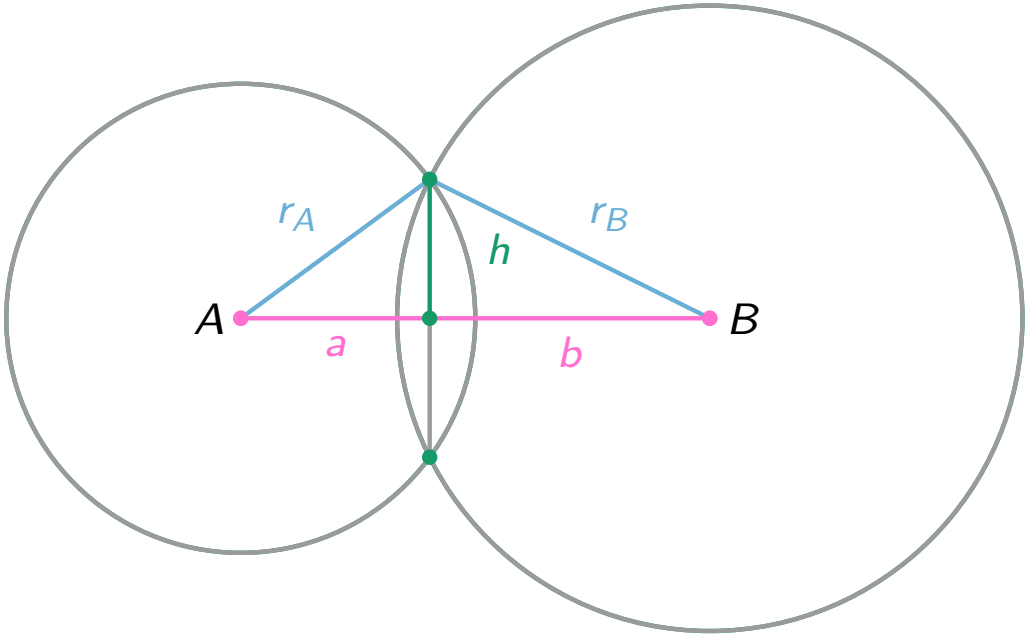
Bài toán tương tự là tìm các điểm giao nhau của hai đường tròn.



- Nếu $d > r_A + r_B$ hai đường tròn không giao nhau.
- Nếu $d < |r_A - r_B|$, một đường tròn nằm trong đường tròn còn lại.
- Nếu $d = 0$ và $r_A = r_B$, hai đường tròn là một.
- Trường hợp còn lại thì sao?

Giao điểm hai đường tròn

Bài toán tương tự là tìm các điểm giao nhau của hai đường tròn.

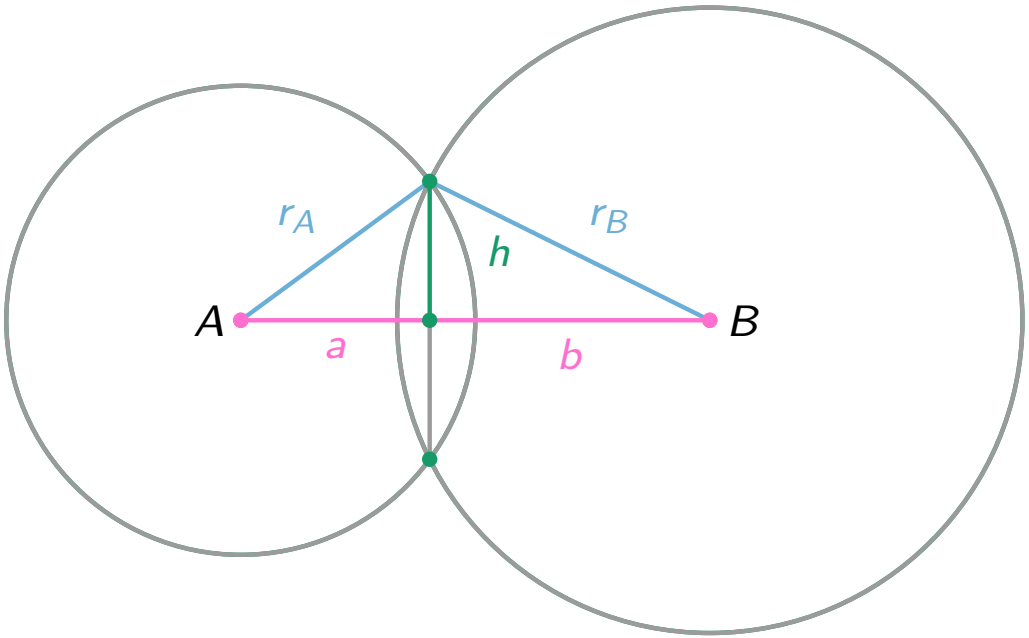


- Giải đối với các vec-tơ a và h từ phương trình

$$a^2 + h^2 = r_A^2 \quad b^2 + h^2 = r_B^2$$

Giao điểm hai đường tròn

Bài toán tương tự là tìm các điểm giao nhau của hai đường tròn.



- Giải đối với các vec-tơ a và h từ phương trình

$$a^2 + h^2 = r_A^2 \quad b^2 + h^2 = r_B^2$$

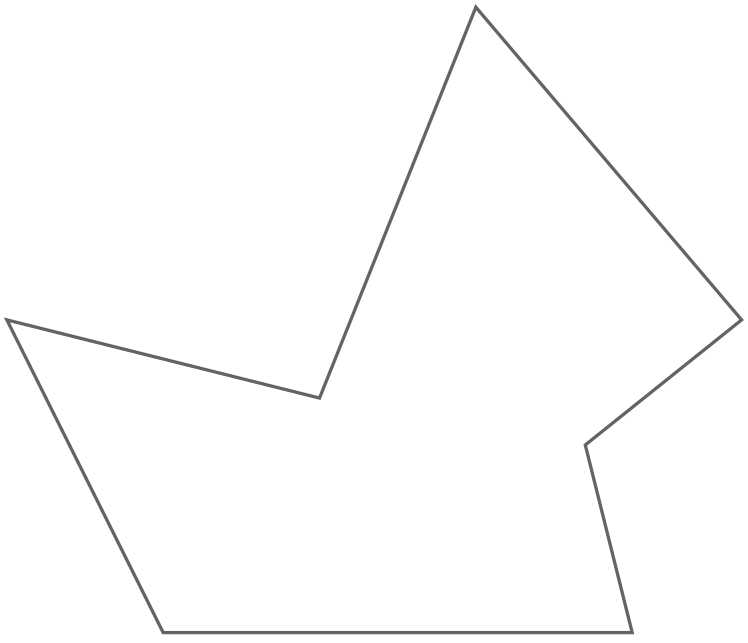
- Ta có

$$a = \frac{r_A^2 - r_B^2 + d^2}{2 \cdot d}$$

$$h^2 = r_A^2 - a^2$$

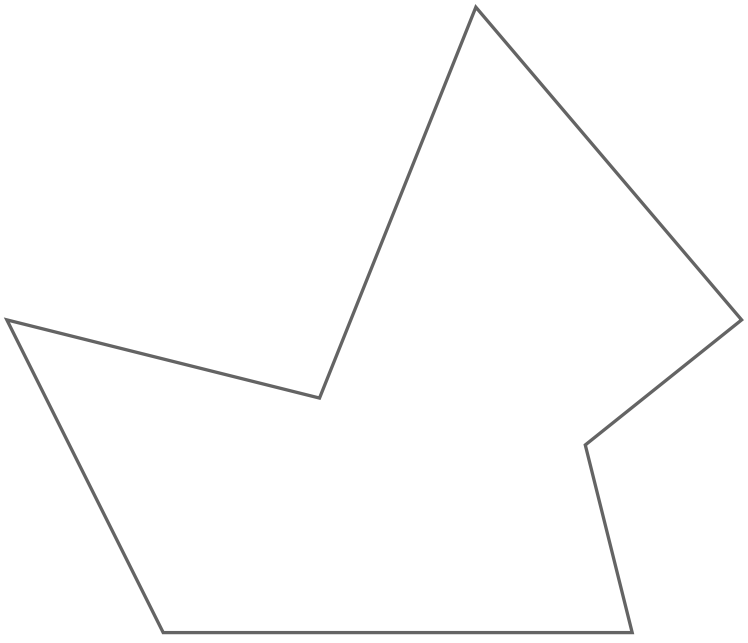
```
#define C(p, r) const point &p, double r
int intersect(C(A, rA), C(B, rB),
                  point & res1, point & res2) {
    double d = abs(B - A);
    if ( rA + rB <  d - EPS || d < abs(rA - rB) - EPS){
      return 0;
    }
    double a = (rA*rA - rB*rB + d*d) / 2*d;
    double h = sqrt(rA*rA - a*a);
    point v = normalize(B - A, a);
    u = normalize(rotate(B-A), h);
    res1 = A + v + u;
    res2 = A + v - u;
    if (abs(u) < EPS){
        return 1;
    }
    return 2;
}
```

Diện tích đa giác



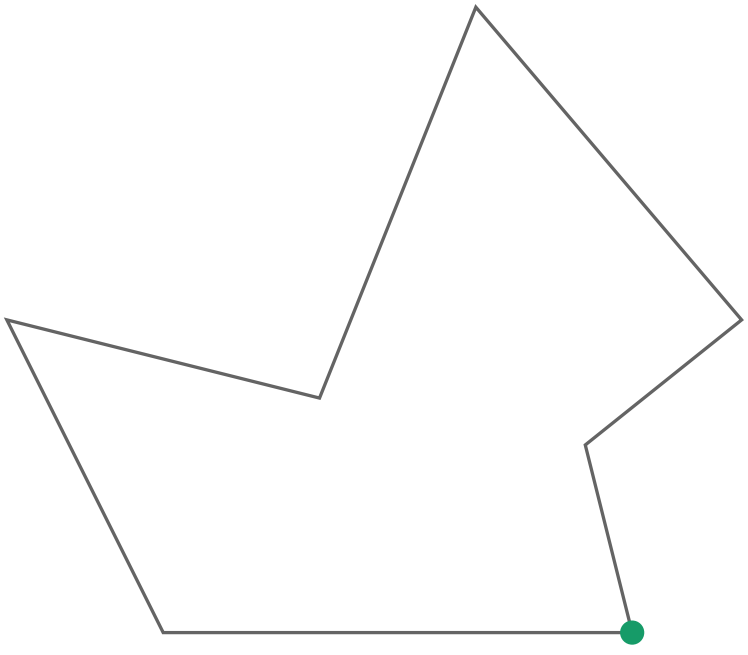
- Đa giác được biểu diễn bởi một danh sách các điểm theo thứ tự biểu diễn các cạnh.

Diện tích đa giác



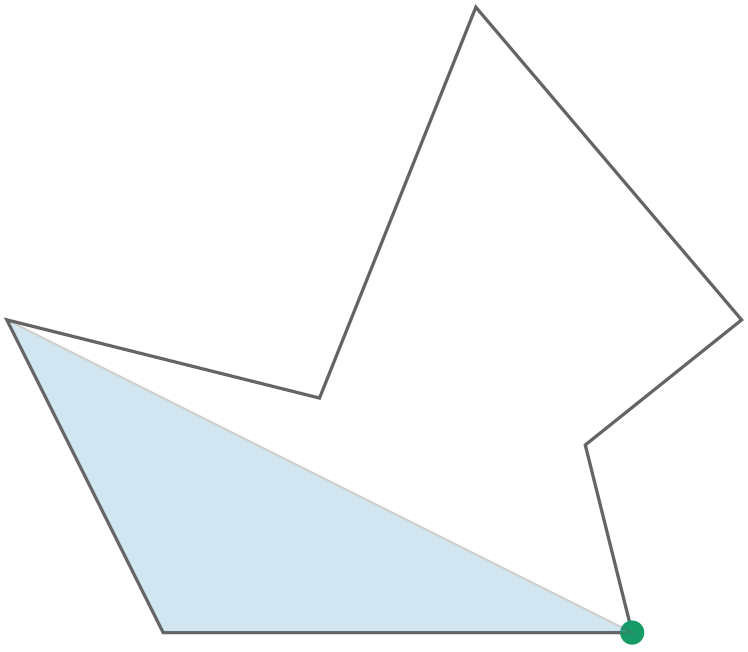
- Đa giác được biểu diễn bởi một danh sách các điểm theo thứ tự biểu diễn các cạnh.
- Để tính diện tích:

Diện tích đa giác



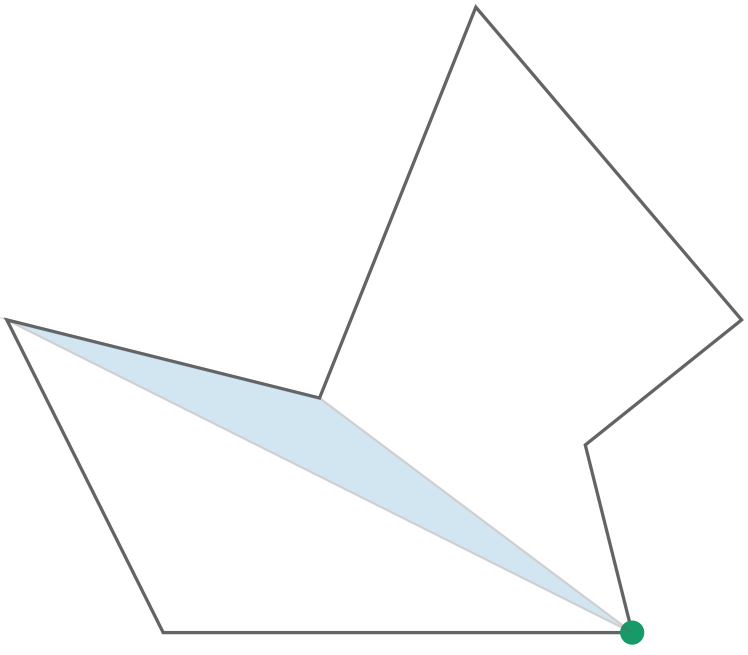
- Đa giác được biểu diễn bởi một danh sách các điểm theo thứ tự biểu diễn các cạnh.
- Để tính diện tích:
  - Chọn một điểm bắt đầu.

Diện tích đa giác



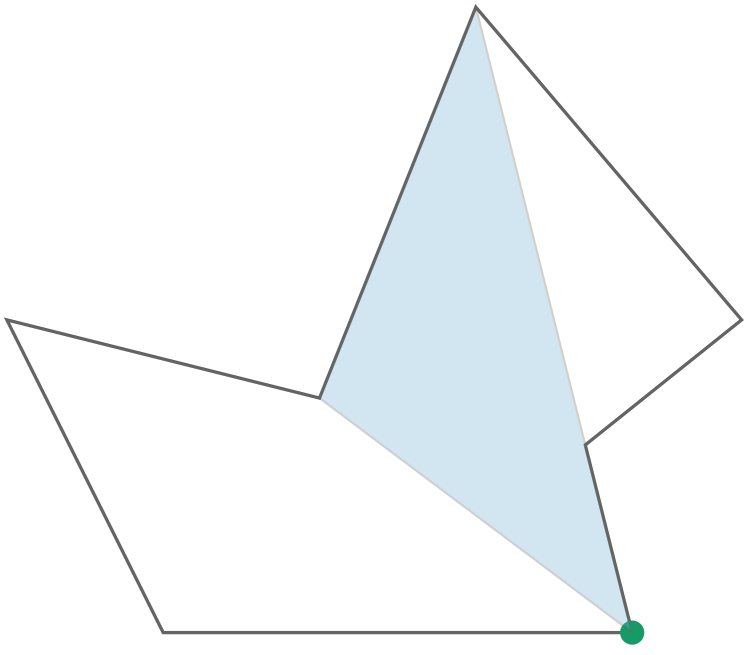
- Đa giác được biểu diễn bởi một danh sách các điểm theo thứ tự biểu diễn các cạnh.
- Để tính diện tích:
  - Chọn một điểm bắt đầu.
  - Duyệt qua tất cả các cặp điểm liền kề khác và tính tổng diện tích của tam giác.

Diện tích đa giác



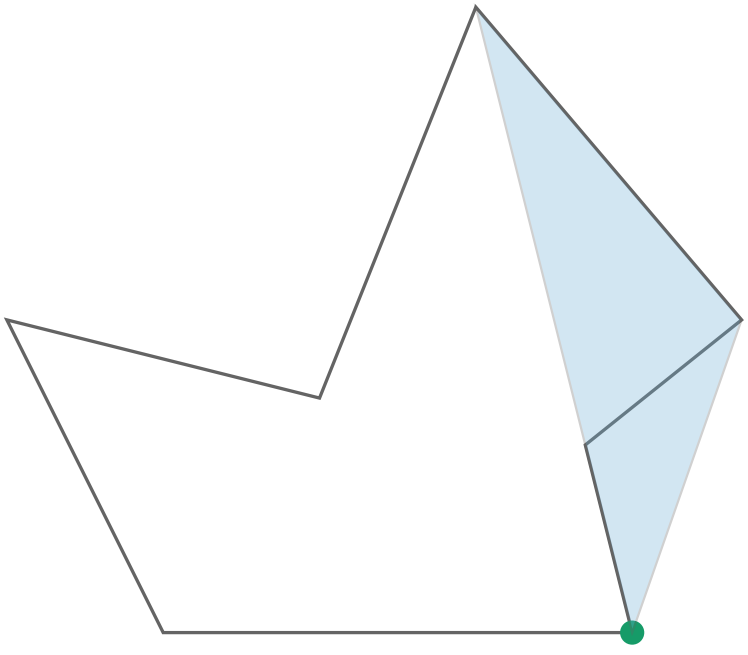
- Đa giác được biểu diễn bởi một danh sách các điểm theo thứ tự biểu diễn các cạnh.
- Để tính diện tích:
  - Chọn một điểm bắt đầu.
  - Duyệt qua tất cả các cặp điểm liền kề khác và tính tổng diện tích của tam giác.

Diện tích đa giác



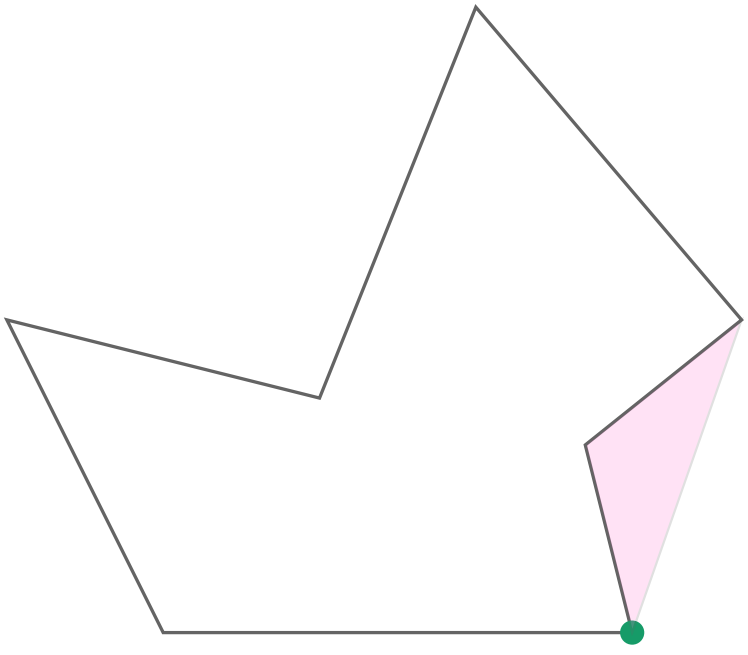
- Đa giác được biểu diễn bởi một danh sách các điểm theo thứ tự biểu diễn các cạnh.
- Để tính diện tích:
  - Chọn một điểm bắt đầu.
  - Duyệt qua tất cả các cặp điểm liền kề khác và tính tổng diện tích của tam giác.

Diện tích đa giác



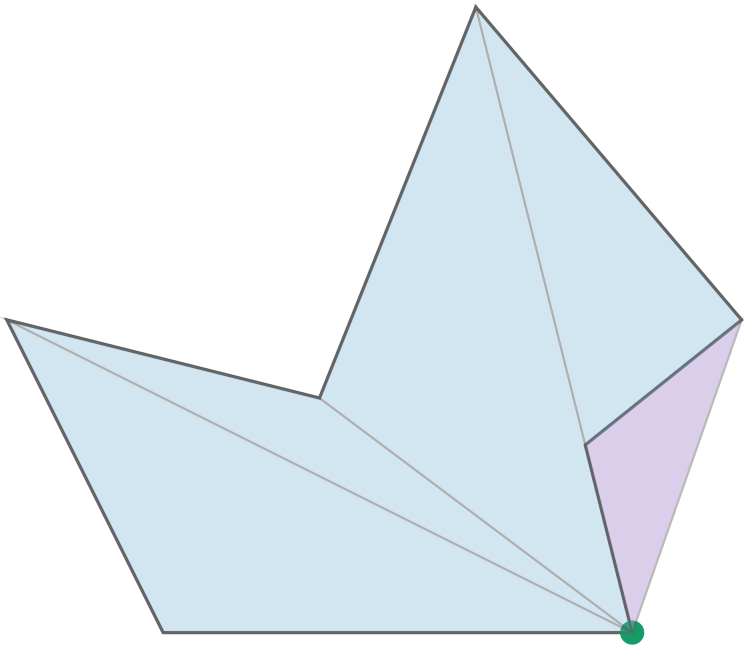
- Đa giác được biểu diễn bởi một danh sách các điểm theo thứ tự biểu diễn các cạnh.
- Để tính diện tích:
  - Chọn một điểm bắt đầu.
  - Duyệt qua tất cả các cặp điểm liền kề khác và tính tổng diện tích của tam giác.

Diện tích đa giác



- Đa giác được biểu diễn bởi một danh sách các điểm theo thứ tự biểu diễn các cạnh.
- Để tính diện tích:
  - Chọn một điểm bắt đầu.
  - Duyệt qua tất cả các cặp điểm liền kề khác và tính tổng diện tích của tam giác.
  - Ngay cả khi tổng tính cả diện tích bên ngoài đa giác, do tích có hướng nên sau phần bên ngoài đó sẽ được trừ đi.

Diện tích đa giác



- Đa giác được biểu diễn bởi một danh sách các điểm theo thứ tự biểu diễn các cạnh.
- Để tính diện tích:
  - Chọn một điểm bắt đầu.
  - Duyệt qua tất cả các cặp điểm liền kề khác và tính tổng diện tích của tam giác.
  - Ngay cả khi tổng tính cả diện tích bên ngoài đa giác, do tích có hướng nên sau phần bên ngoài đó sẽ được trừ đi.

```cpp
double polygon_area_signed(const vector<point> &p) {
    double area = 0;
    int cnt = size(p);
    for (int i = 1; i + 1 < cnt; i++){
        area += cross(p[i] - p[0], p[i + 1] - p[0])/2;
    }
    return area;
}
double polygon_area(vector<point> &p) {
    return abs(polygon_area_signed(p));
}
```

Cài đặt theo cấu trúc `struct point{double x,y};`

- Diện tích đa giác có thể tính bằng công thức :

$$A = \frac{1}{2} \times \begin{bmatrix} x_0 & y_0 \\ x_1 & y_1 \\ x_2 & y_2 \\ .. & .. \\ x_{n-1} & y_{n-1} \end{bmatrix} =$$

$$\frac{1}{2} \times (x_0 \times y_1 + x_1 \times y_2 + \cdots + x_{n-1} \times y_0 - x_1 \times y_0 - x_2 \times y_1 - \cdots - x_0 \times y_{n-1})$$

Cài đặt theo cấu trúc struct point{double x,y};

- Diện tích đa giác có thể tính bằng công thức :

$$A = \frac{1}{2} \times \begin{bmatrix} x_0 & y_0 \\ x_1 & y_1 \\ x_2 & y_2 \\ .. & .. \\ x_{n-1} & y_{n-1} \end{bmatrix} =$$

$$\frac{1}{2} \times (x_0 \times y_1 + x_1 \times y_2 + \cdots + x_{n-1} \times y_0 - x_1 \times y_0 - x_2 \times y_1 - \cdots - x_0 \times y_{n-1})$$

- Code:

```
1 double area(const vector<point> &P) {
2 double result = 0.0, x1, y1, x2, y2;
3 for (int i = 0; i < (int)P.size()-1; i++) {
4 x1 = P[i].x; x2 = P[i+1].x;
5 y1 = P[i].y; y2 = P[i+1].y;
6 result += (x1 * y2 - x2 * y1);
7 }
8 return fabs(result) / 2.0; }
```

Bao lồi

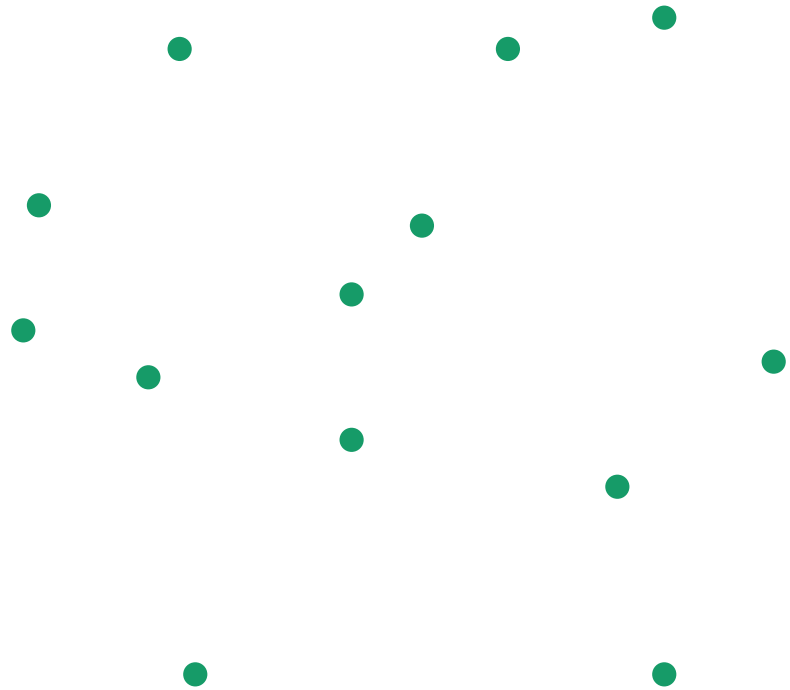
- Tìm bao lồi của một tập điểm cho trước?

Bao lồi

- Tìm bao lồi của một tập điểm cho trước?
- Bao lồi của một tập điểm có thể được hình dung như một đa giác tạo bởi một dải dây cao su căng bao quanh tập hợp các điểm.

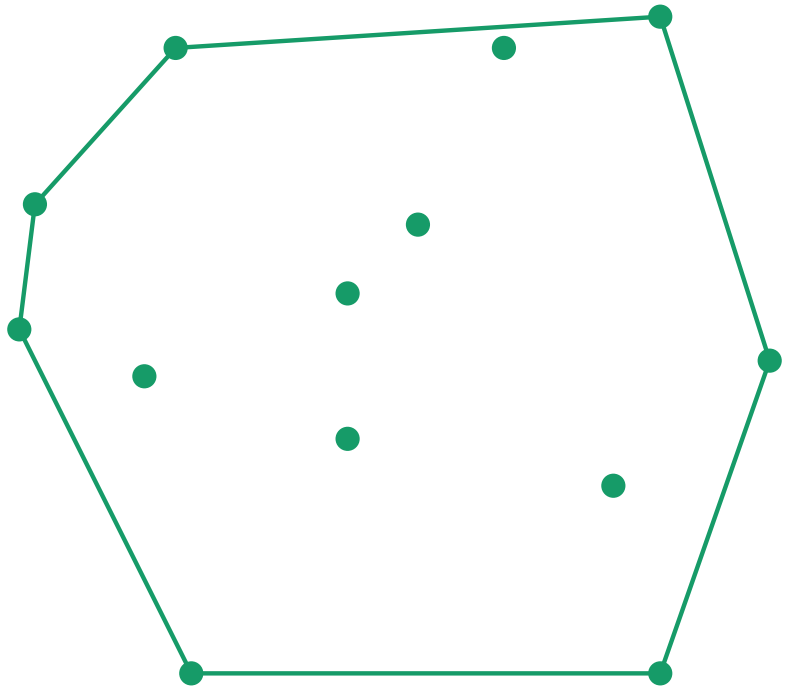
Bao lồi

- Tìm bao lồi của một tập điểm cho trước?
- Bao lồi của một tập điểm có thể được hình dung như một đa giác tạo bởi một dải dây cao su căng bao quanh tập hợp các điểm.



Bao lồi

- Tìm bao lồi của một tập điểm cho trước?
- Bao lồi của một tập điểm có thể được hình dung như một đa giác tạo bởi một dải dây cao su căng bao quanh tập hợp các điểm.



Thuật toán quét Graham:

Thuật toán quét Graham:

- Chọn điểm p_0 với tọa độ y nhỏ nhất. $\mathcal{O}(n)$

Thuật toán quét Graham:

- Chọn điểm p_0 với tọa độ y nhỏ nhất. $\mathcal{O}(n)$
- Sắp xếp tất cả các điểm theo góc tạo với p_0 và trục hoành. $\mathcal{O}(n \log n)$

Thuật toán quét Graham:

- Chọn điểm p_0 với tọa độ y nhỏ nhất. $\mathcal{O}(n)$
- Sắp xếp tất cả các điểm theo góc tạo với p_0 và trục hoành. $\mathcal{O}(n \log n)$
- Duyệt lần lượt qua tất cả các điểm ($\mathcal{O}(n)$)

Thuật toán quét Graham:

- Chọn điểm p_0 với tọa độ y nhỏ nhất. $\mathcal{O}(n)$
- Sắp xếp tất cả các điểm theo góc tạo với p_0 và trục hoành. $\mathcal{O}(n \log n)$
- Duyệt lần lượt qua tất cả các điểm ($\mathcal{O}(n)$)
  - Nếu điểm hiện tại tạo thành một góc theo chiều kim đồng hồ với hai điểm ngay trước đó, loại bỏ điểm cuối cùng ra khỏi bao lồi.

Thuật toán quét Graham:

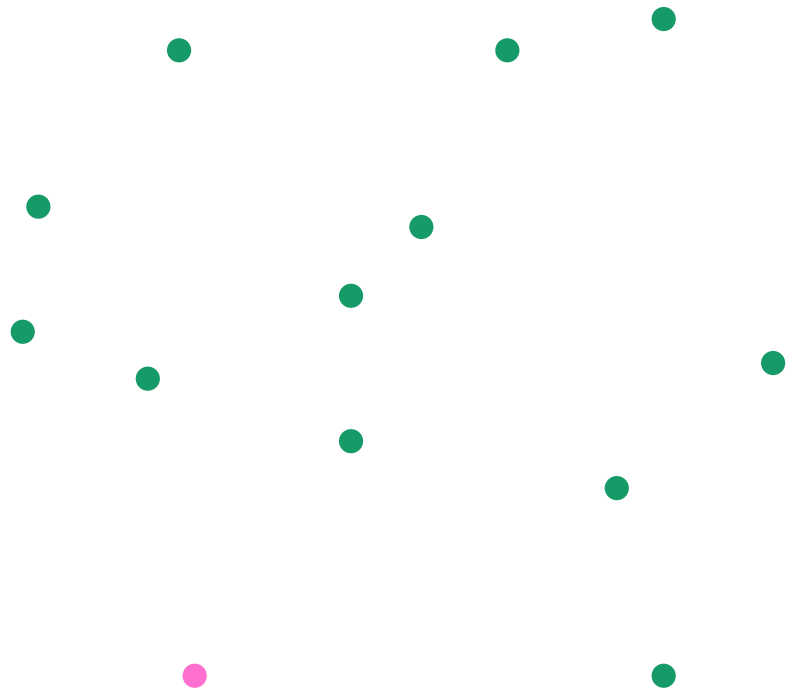
- Chọn điểm p_0 với tọa độ y nhỏ nhất. $\mathcal{O}(n)$
- Sắp xếp tất cả các điểm theo góc tạo với p_0 và trục hoành. $\mathcal{O}(n \log n)$
- Duyệt lần lượt qua tất cả các điểm ($\mathcal{O}(n)$)
  - Nếu điểm hiện tại tạo thành một góc theo chiều kim đồng hồ với hai điểm ngay trước đó, loại bỏ điểm cuối cùng ra khỏi bao lồi.
  - Nếu không, thêm điểm hiện tại vào bao lồi.

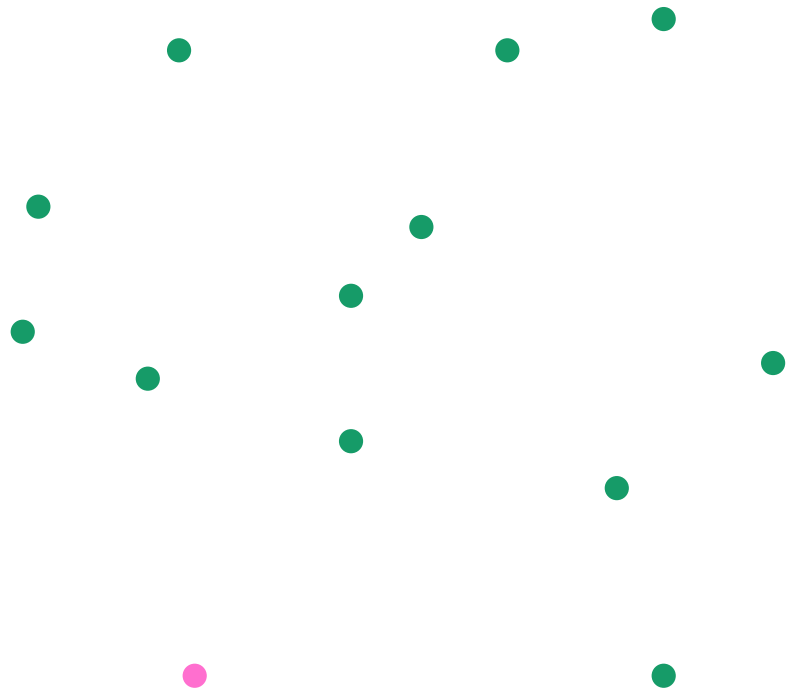
Thuật toán quét Graham:

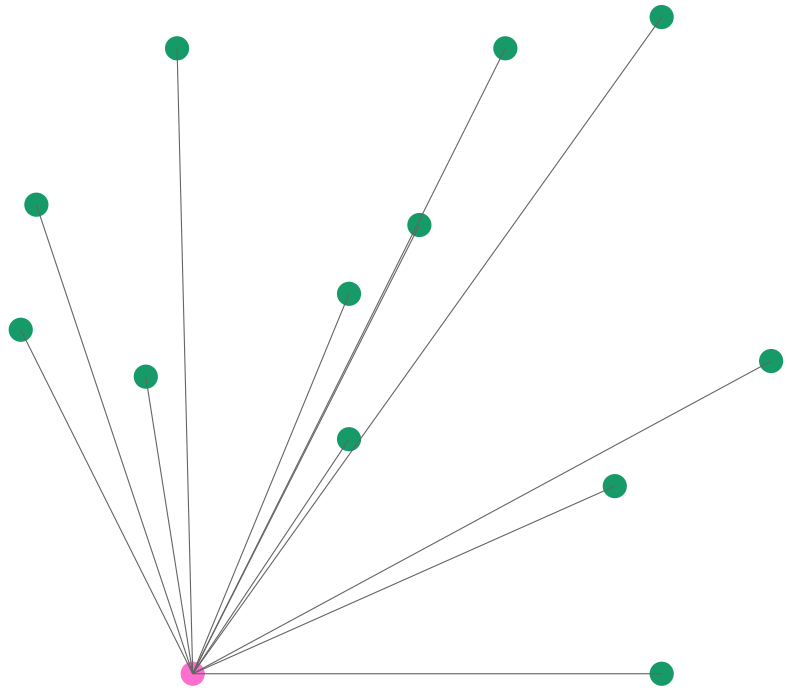
- Chọn điểm p_0 với tọa độ y nhỏ nhất. $\mathcal{O}(n)$
- Sắp xếp tất cả các điểm theo góc tạo với p_0 và trục hoành. $\mathcal{O}(n \log n)$
- Duyệt lần lượt qua tất cả các điểm ($\mathcal{O}(n)$)
  - Nếu điểm hiện tại tạo thành một góc theo chiều kim đồng hồ với hai điểm ngay trước đó, loại bỏ điểm cuối cùng ra khỏi bao lồi.
  - Nếu không, thêm điểm hiện tại vào bao lồi.

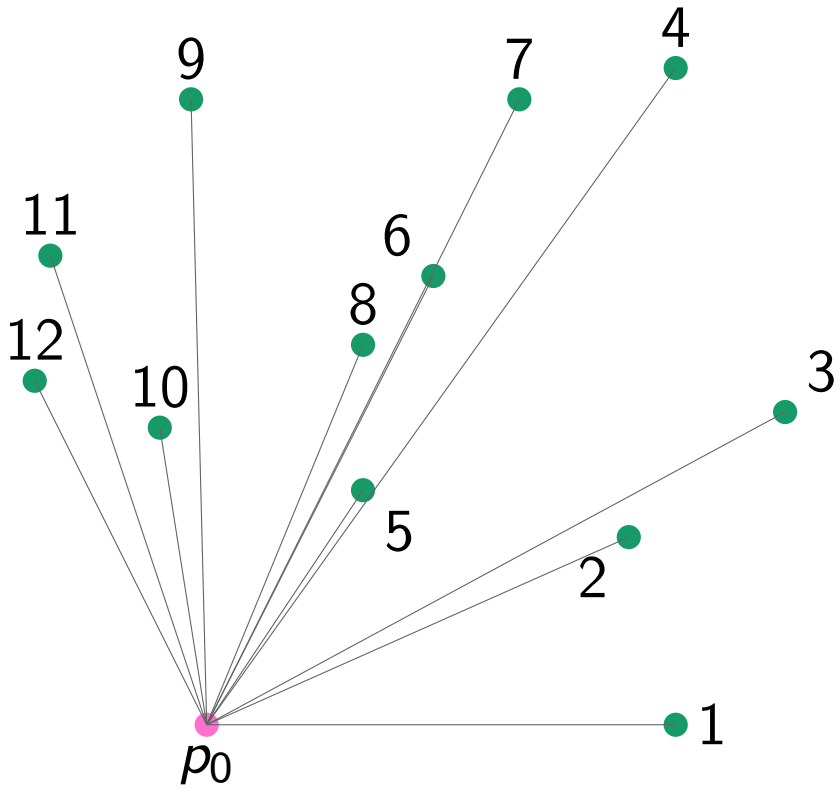
Độ phức tạp thuật toán: $\mathcal{O}(n \log n)$

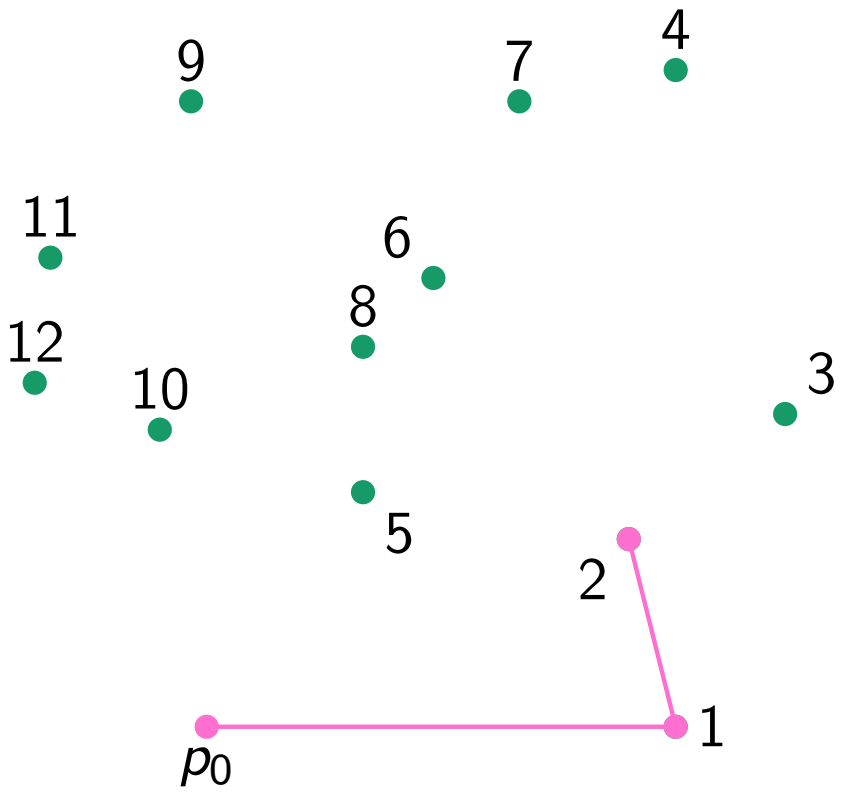
Lưu ý trường hợp các điểm đã cho là thẳng hàng
$\longrightarrow$ RUNTIME ERROR!!!

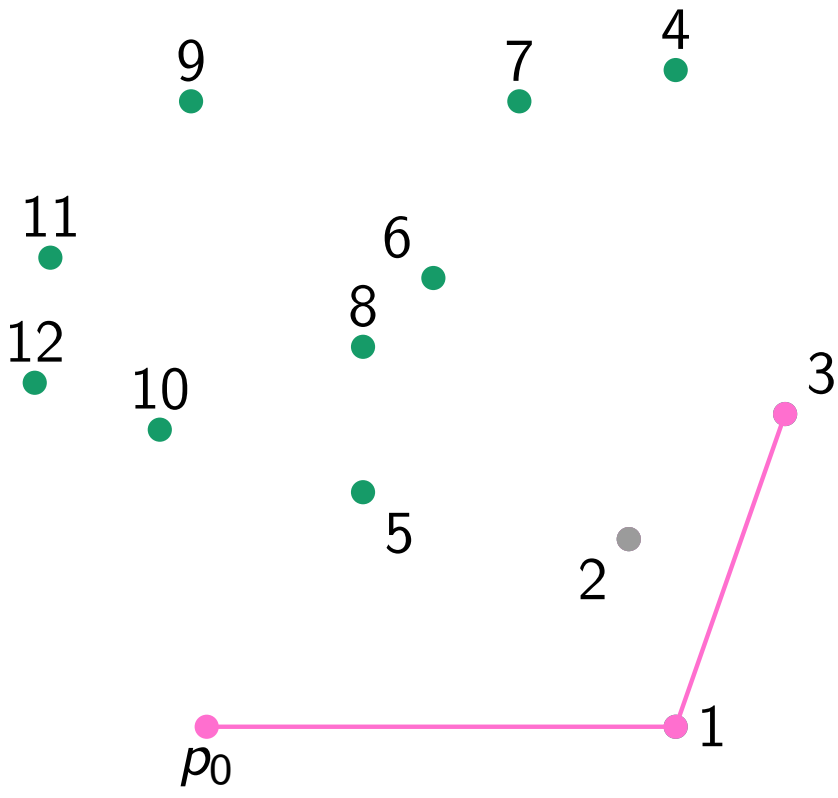


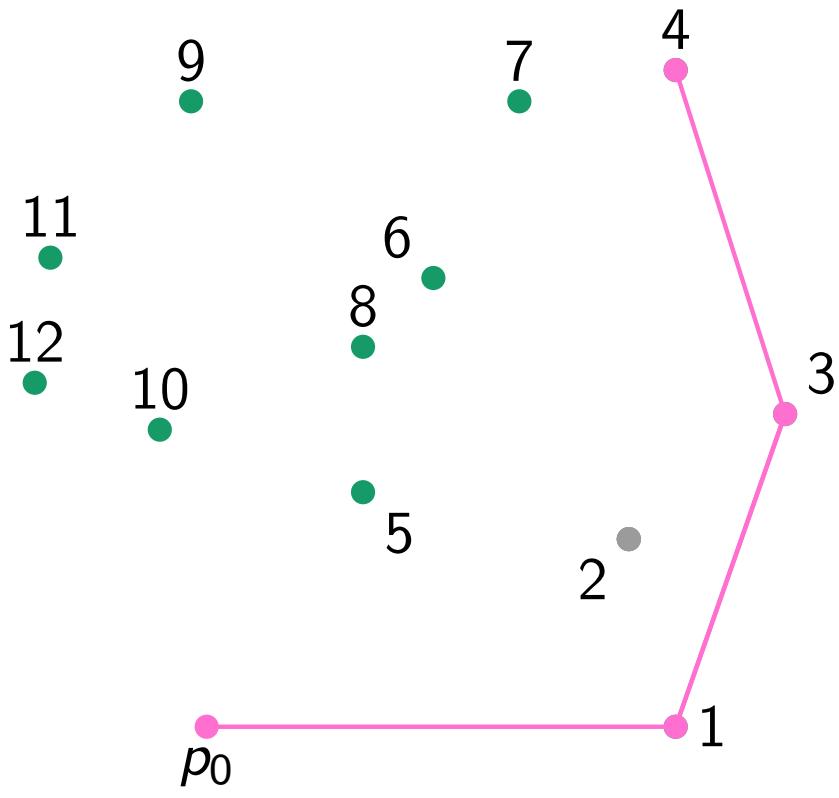


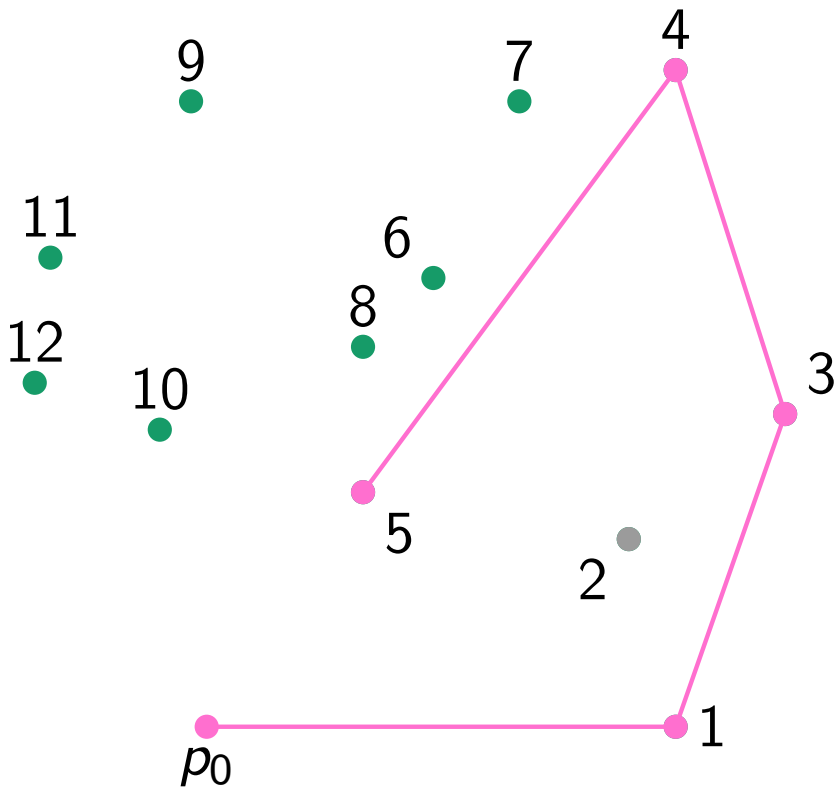


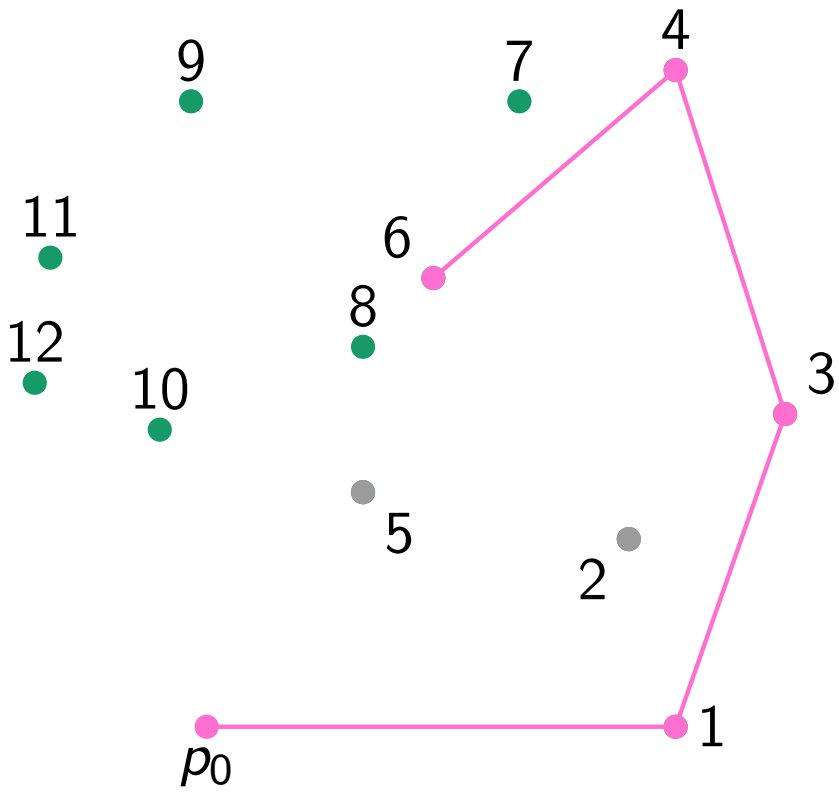

9
7
4
11
6
8
12
10
3
5
2
1
p_0

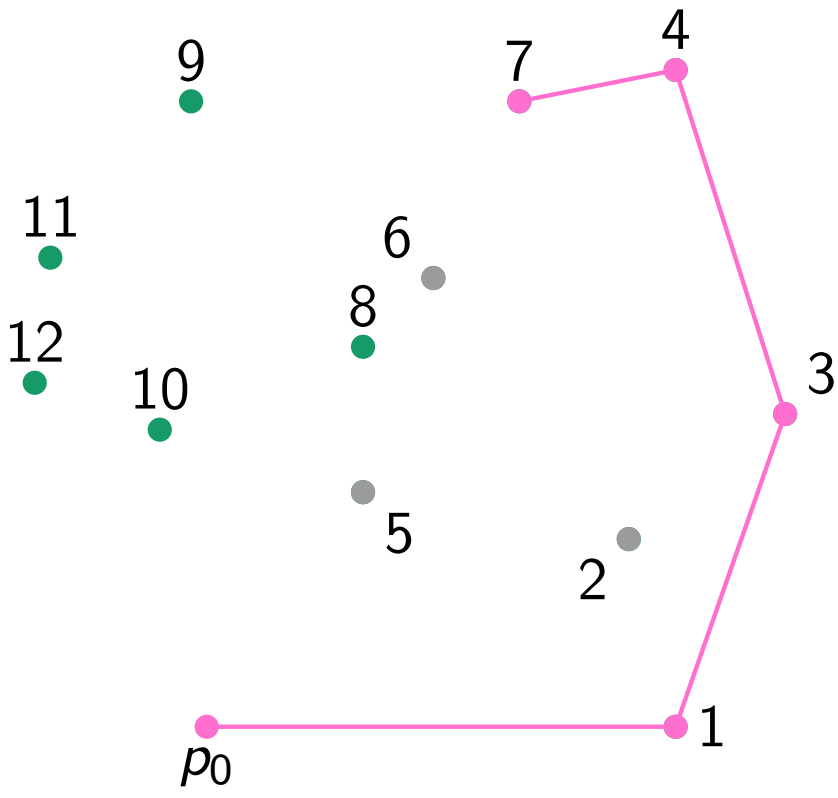

9
7
4
11
6
8
12
10
3
5
2
1
p_0

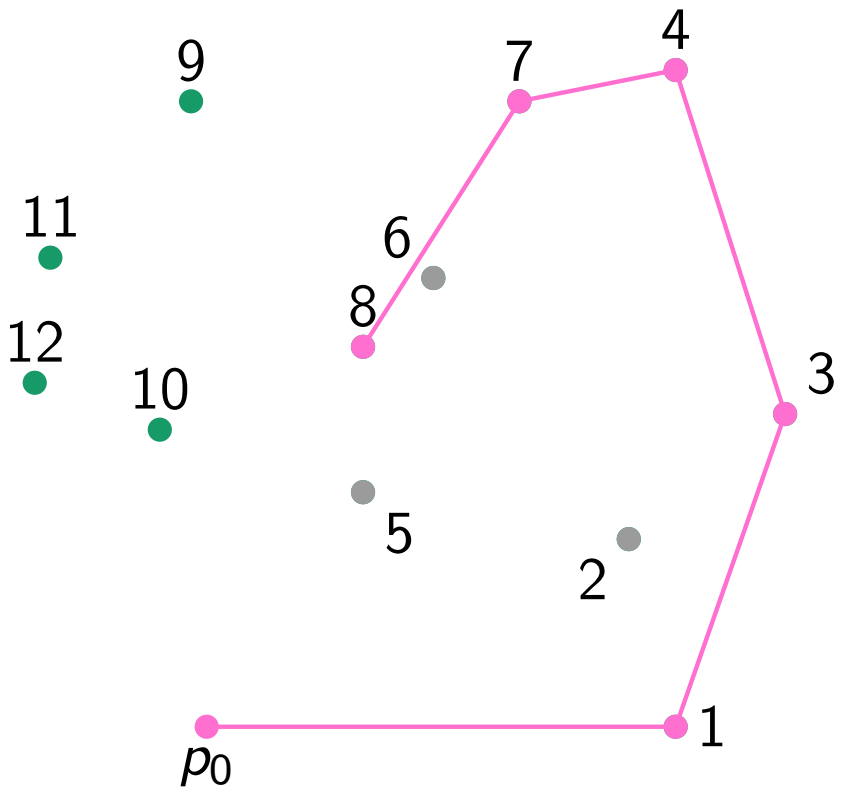

9
7
4
11
6
8
12
10
3
5
2
1
p_0

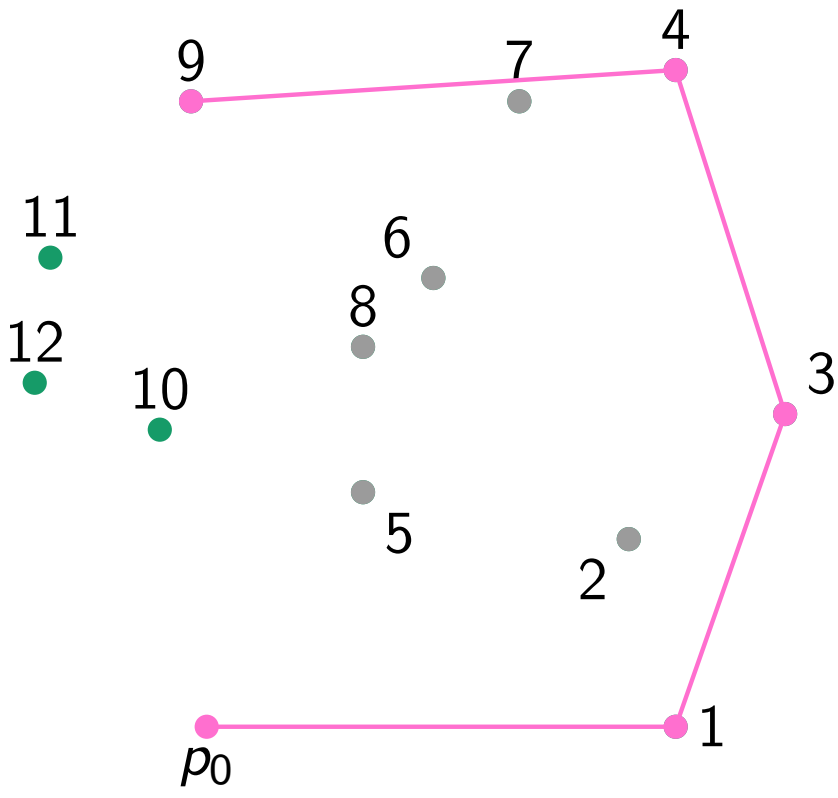

9
7
4
11
6
8
12
10
3
5
2
1
p_0

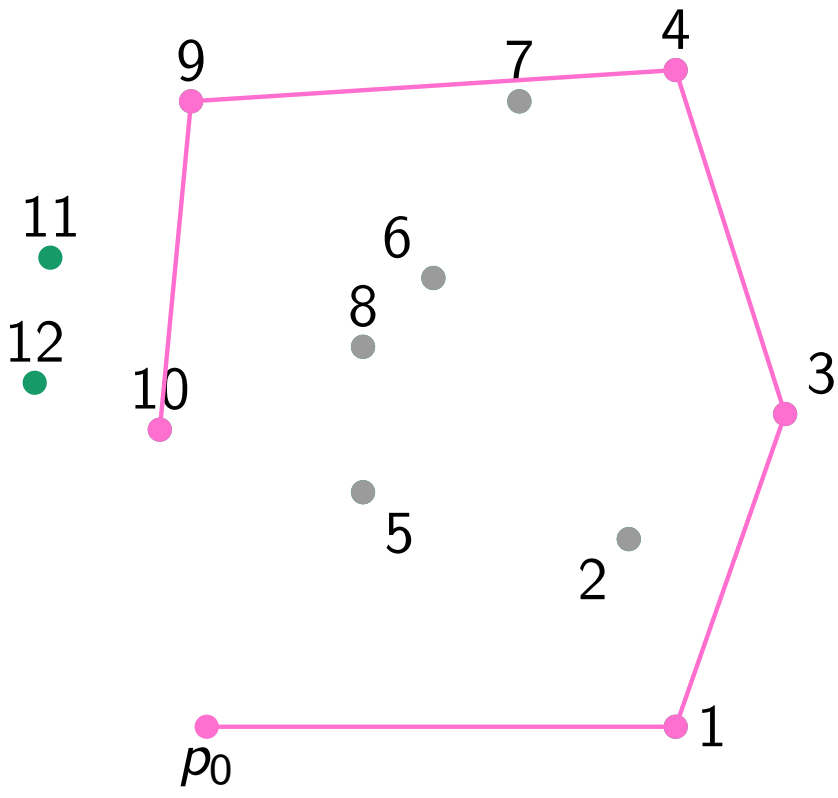

9
7
4
11
6
8
12
10
3
5
2
1
p_0

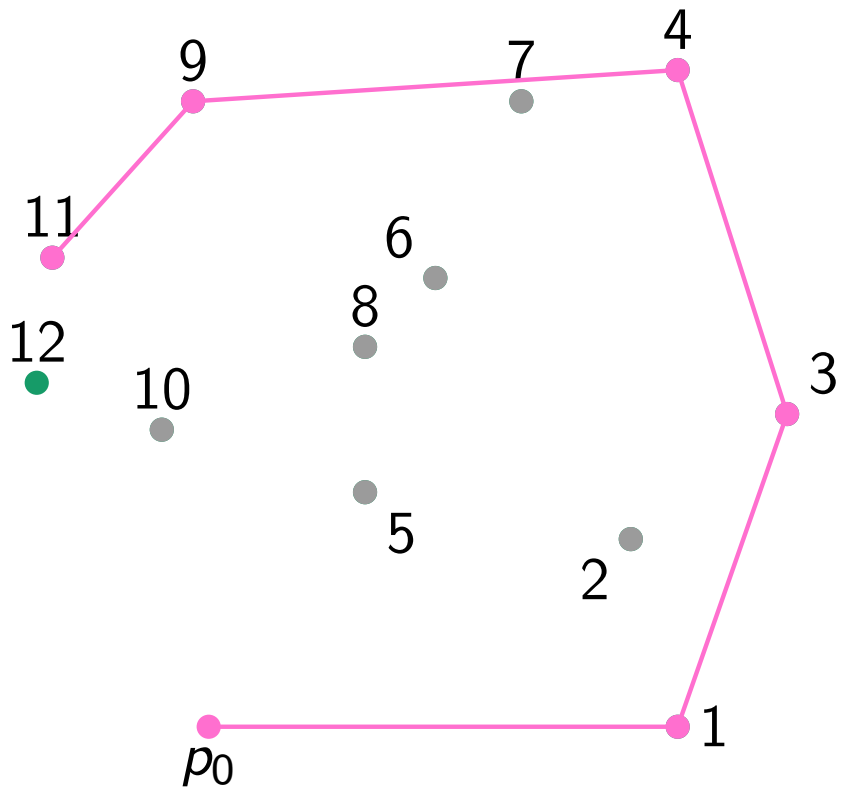

9
7
4
11
6
8
12
10
3
5
2
1
p_0

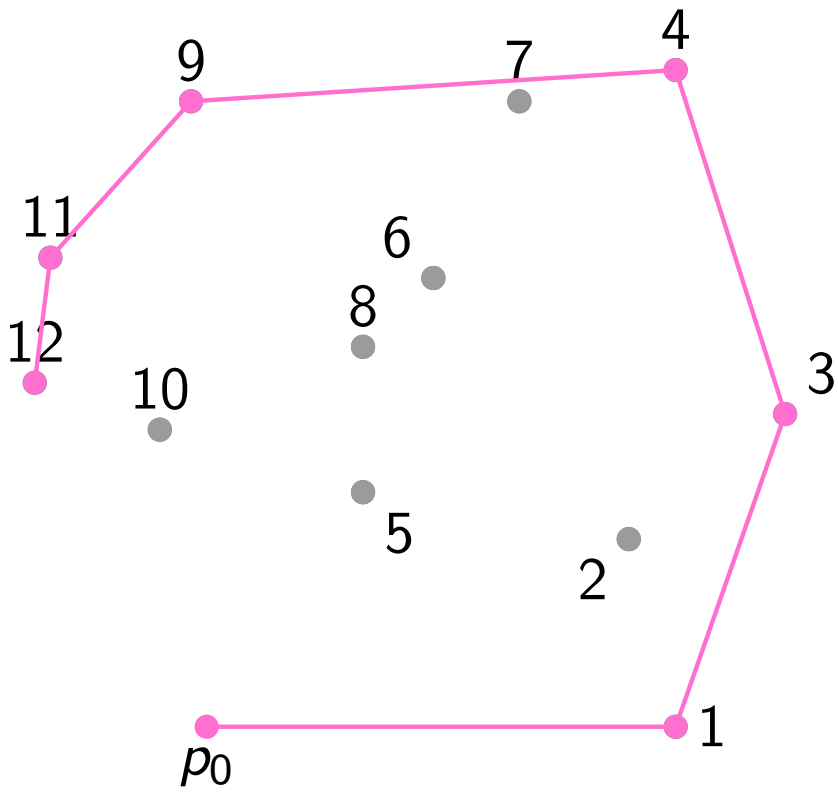

9
7
4
11
6
8
12
10
3
5
2
1
p_0

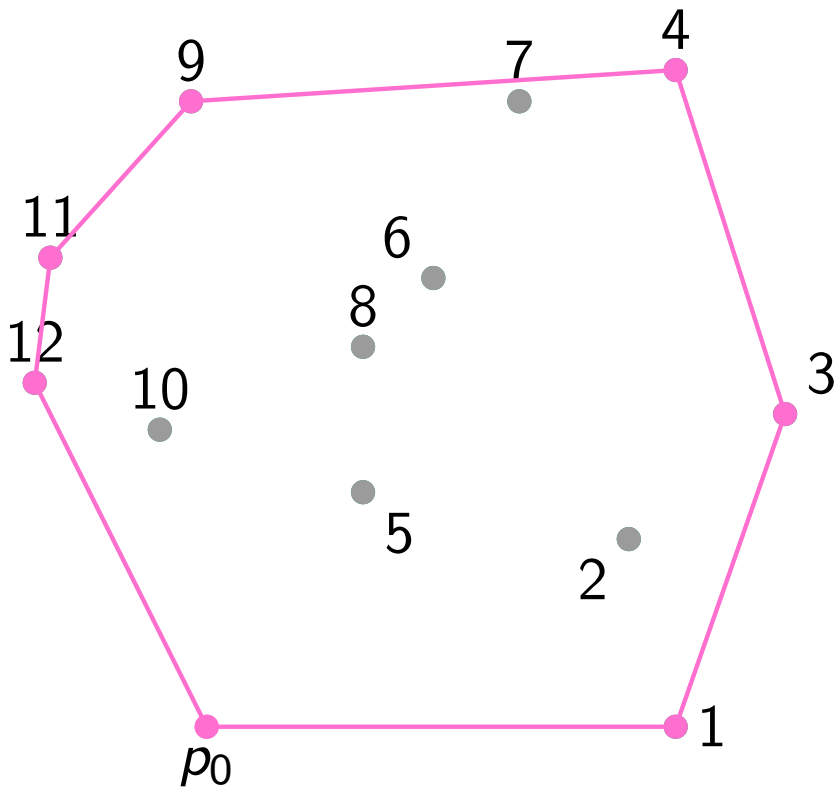

9
7
4
11
6
8
12
10
3
5
2
1
p_0

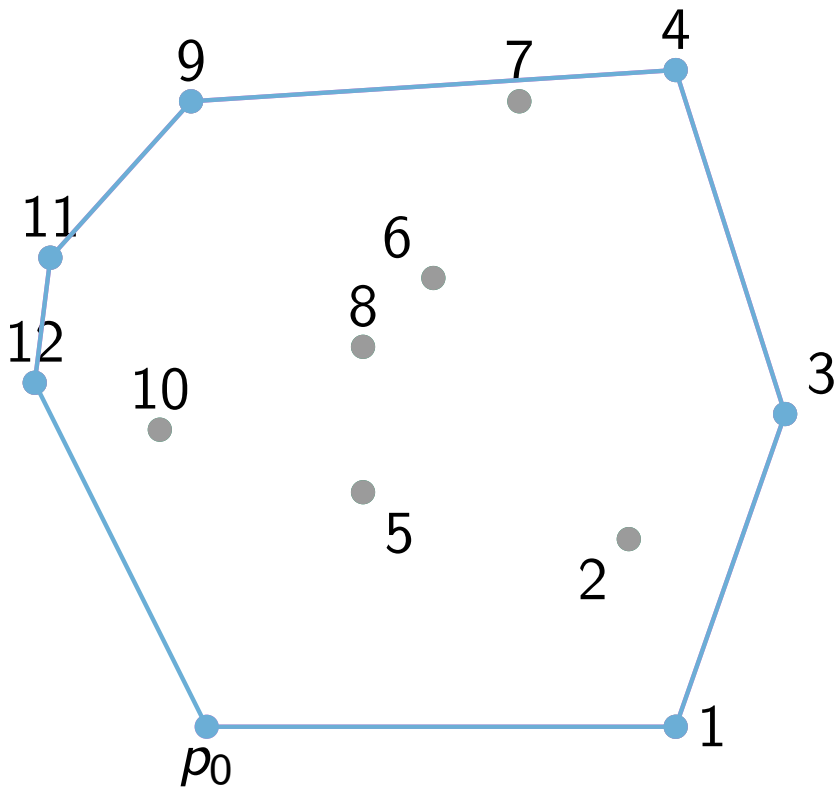

9
7
4
11
6
8
12
10
3
5
2
1
p_0


9
7
4
11
6
8
12
10
3
5
2
1
p_0


9
7
4
11
6
8
12
10
3
5
2
1
p_0


9
7
4
11
6
8
12
10
3
5
2
1
p_0


9
7
4
11
6
8
12
10
3
5
2
1
p_0


9
7
4
11
6
8
12
10
3
5
2
1
p_0

Code tìm bao lồi

- Sắp xếp vec-tơ điểm `P` theo hệ số góc, với `P[0]` là điểm có tung độ nhỏ nhất.

```
int convex_hull(vector<point> P) {
    int n = size(P),
    sort(P.begin(), P.end(), cmp);
    vector<point> S;
    S.push_back(P[n-1]);
    S.push_back(P[0]);
    S.push_back(P[1]);
    i = 2;
    while (i < n) {
        j = (int)S.size()-1;
        if (ccw(S[j-1], S[j], P[i]) || j==2 )
            S.push_back(P[i++]);
        // (j==2) de kiem tra cac diem thang hang
        else S.pop_back();
    }
    return S;
}
```

Code khác tham khảo

```
point hull[MAXN];
int convex_hull(vector<point> p) {
    int n = size(p), l = 0;
    sort(p.begin(), p.end(), cmp);
    for (int i = 0; i < n; i++) {
        if (i > 0 && p[i] == p[i - 1])
            continue;
        while (l >= 2 &&
                ccw(hull[l - 2], hull[l - 1], p[i]) >= 0)
            l--;
        hull[l++] = p[i];
    }
    int r = l;
    for (int i = n - 2; i >= 0; i--) {
        if (p[i] == p[i + 1])
            continue;
        while (r - l >= 1 &&
                ccw(hull[r - 2], hull[r - 1], p[i]) >= 0)
            r--;
        hull[r++] = p[i];
    }
    return l == 1 ? 1 : r - 1;
}
```

Ngoài ra còn nhiều thuật toán tìm bao lồi khác:

Ngoài ra còn nhiều thuật toán tìm bao lồi khác:

- Thuật toán “Gói quà" hay còn gọi là thuật toán Javis March với độ phức tạp $\mathcal{O}(n)$.

Ngoài ra còn nhiều thuật toán tìm bao lồi khác:

- Thuật toán “Gói quà" hay còn gọi là thuật toán Javis March với độ phức tạp $\mathcal{O}(n)$.
- Thuật toán “Quick-hull", ý tưởng tương tự thuật toán Quicksort, độ phức tạp trung bình $\mathcal{O}(N \log N)$, và độ phức tạp trong tình huống tồi nhất là $\mathcal{O}(n^2)$.

Ngoài ra còn nhiều thuật toán tìm bao lồi khác:

- Thuật toán “Gói quà" hay còn gọi là thuật toán Javis March với độ phức tạp $\mathcal{O}(n)$.
- Thuật toán “Quick-hull", ý tưởng tương tự thuật toán Quicksort, độ phức tạp trung bình $\mathcal{O}(N \log N)$, và độ phức tạp trong tình huống tồi nhất là $\mathcal{O}(n^2)$.
- Chia để trị với độ phức tạp $\mathcal{O}(n^3)$.
- Tham khảo chi tiết các thuật toán tại
  `https://www.geeksforgeeks.org/convex-hull-simple-divide-conquer-algorithm/`

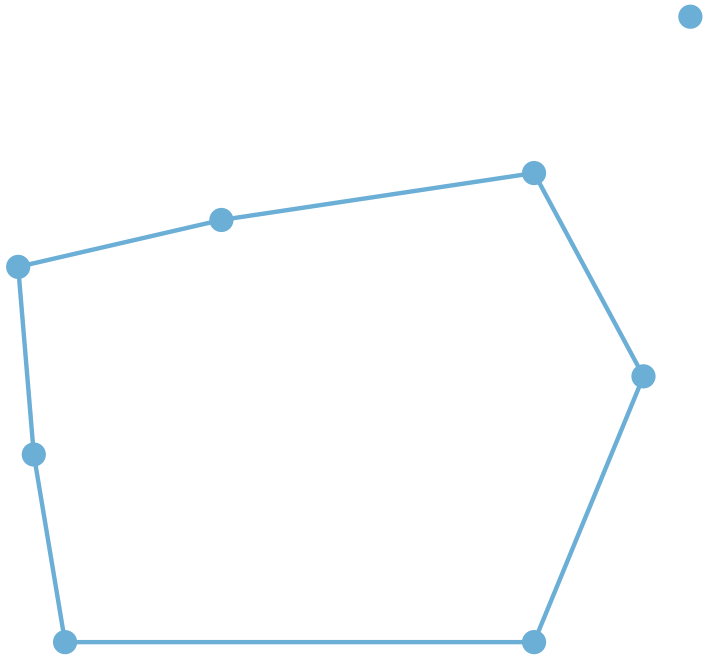
Ngoài ra còn nhiều thuật toán tìm bao lồi khác:

- Thuật toán “Gói quà" hay còn gọi là thuật toán Javis March với độ phức tạp $\mathcal{O}(n)$.
- Thuật toán “Quick-hull", ý tưởng tương tự thuật toán Quicksort, độ phức tạp trung bình $\mathcal{O}(N \log N)$, và độ phức tạp trong tình huống tồi nhất là $\mathcal{O}(n^2)$.
- Chia để trị với độ phức tạp $\mathcal{O}(n^3)$.
- Tham khảo chi tiết các thuật toán tại `https://www.geeksforgeeks.org/convex-hull-simple-divide-conquer-algorithm/`

Một số thuật toán có thể mở rộng cho 3 chiều hoặc đa chiều.

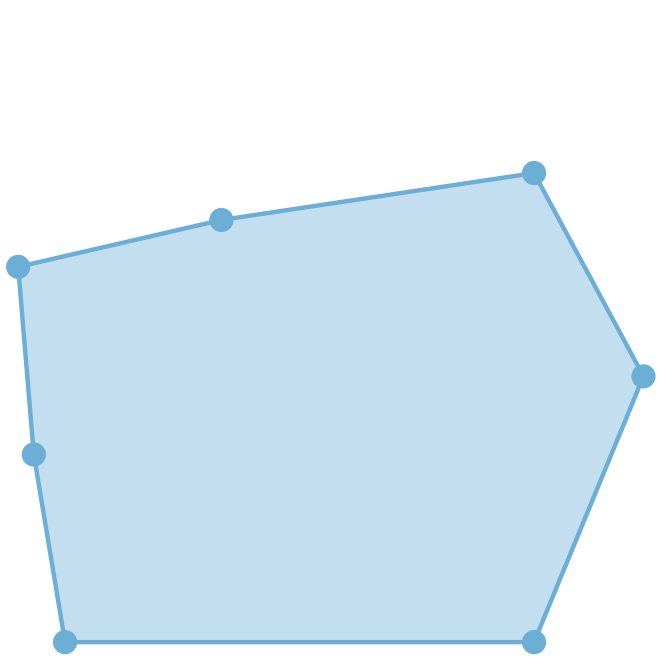
Kiểm tra một điểm nằm trong đa giác

Đa giác lồi:



Kiểm tra một điểm nằm trong đa giác

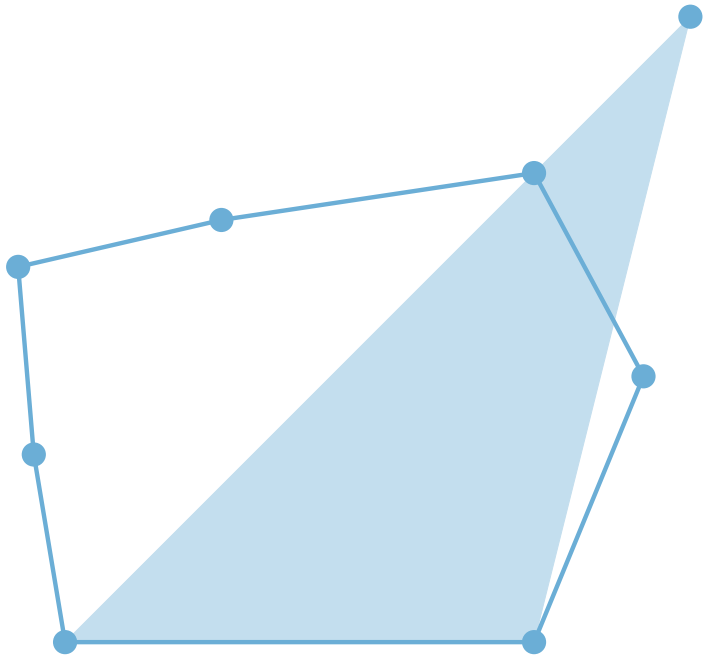
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.

Kiểm tra một điểm nằm trong đa giác

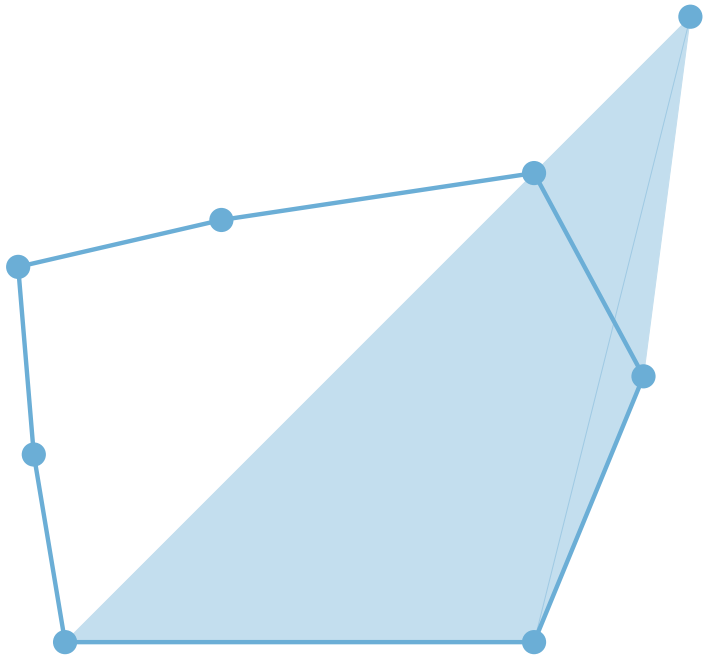
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.

Kiểm tra một điểm nằm trong đa giác

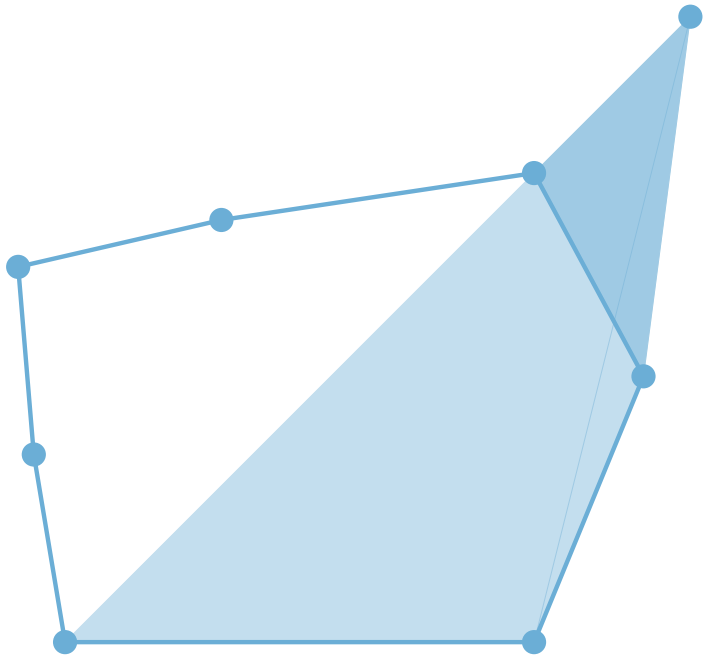
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.

Kiểm tra một điểm nằm trong đa giác

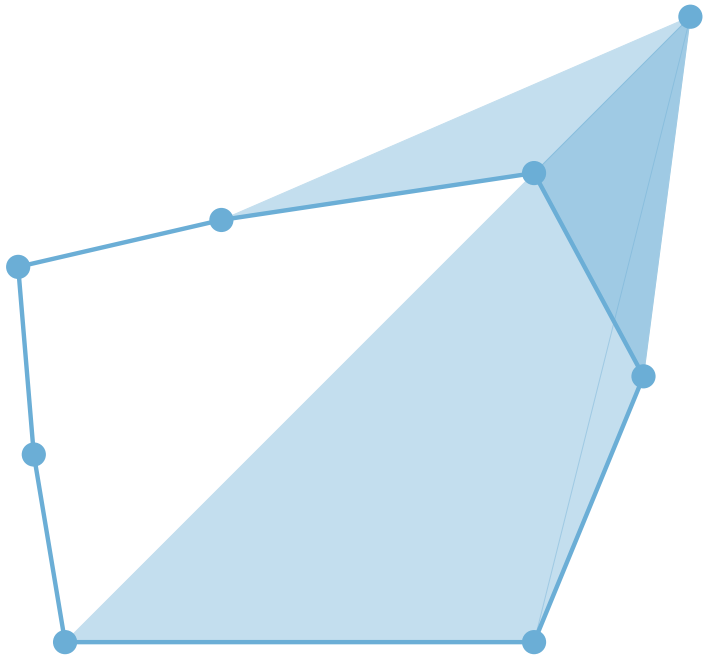
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.

Kiểm tra một điểm nằm trong đa giác

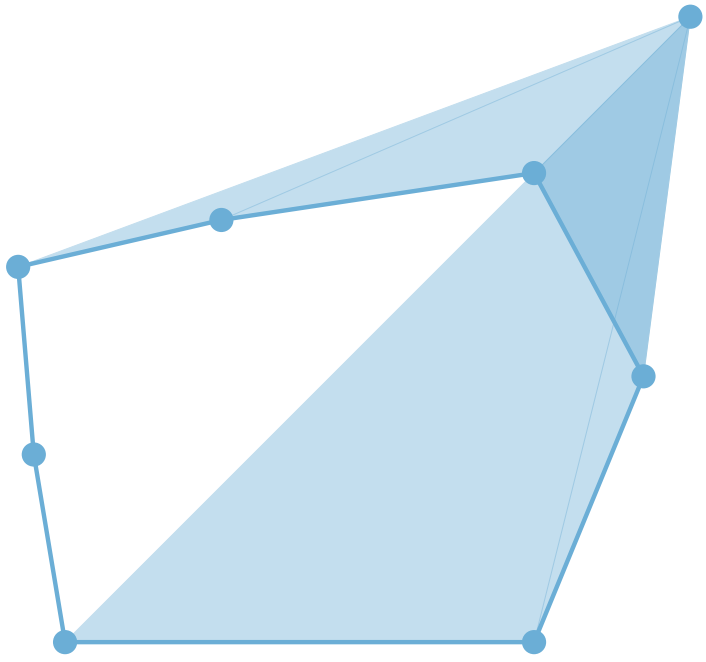
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.

Kiểm tra một điểm nằm trong đa giác

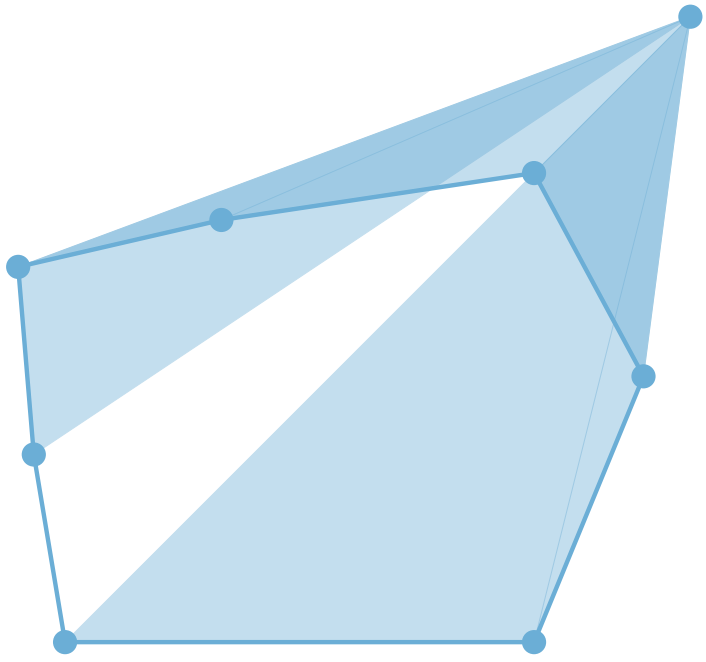
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.

Kiểm tra một điểm nằm trong đa giác

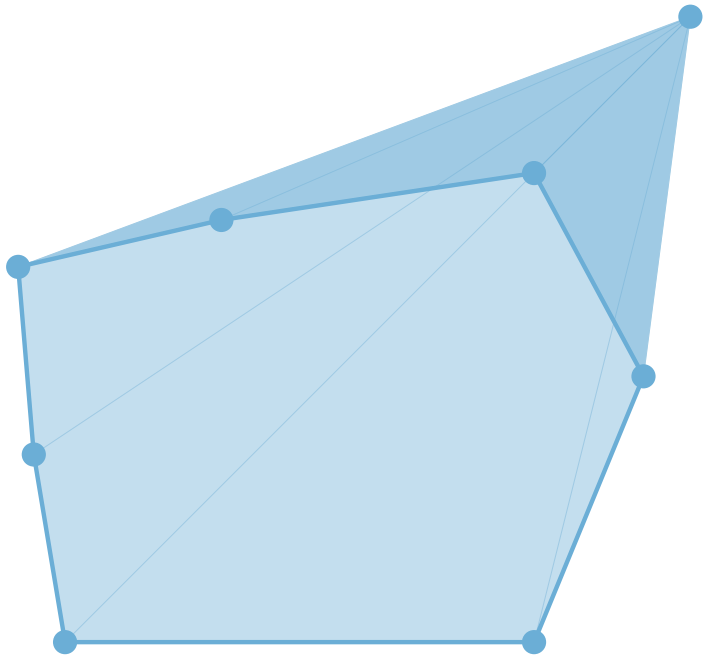
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.

Kiểm tra một điểm nằm trong đa giác

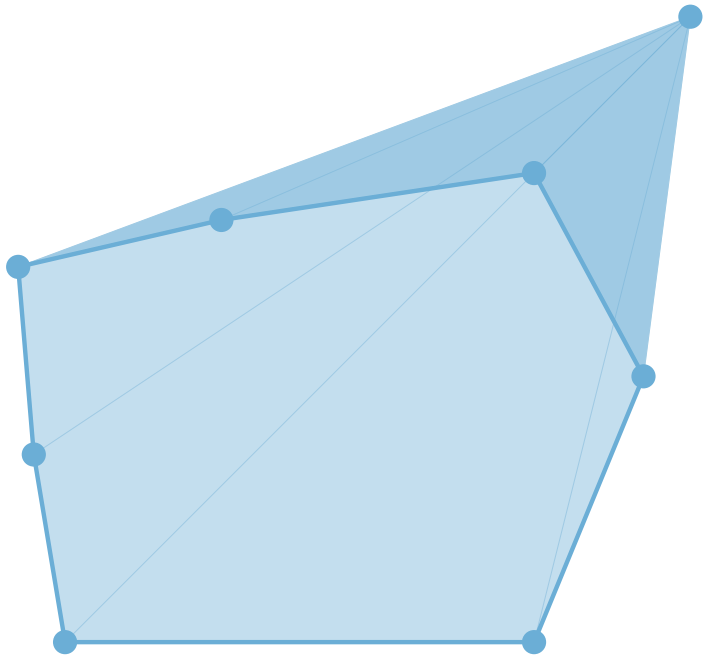
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.

Kiểm tra một điểm nằm trong đa giác

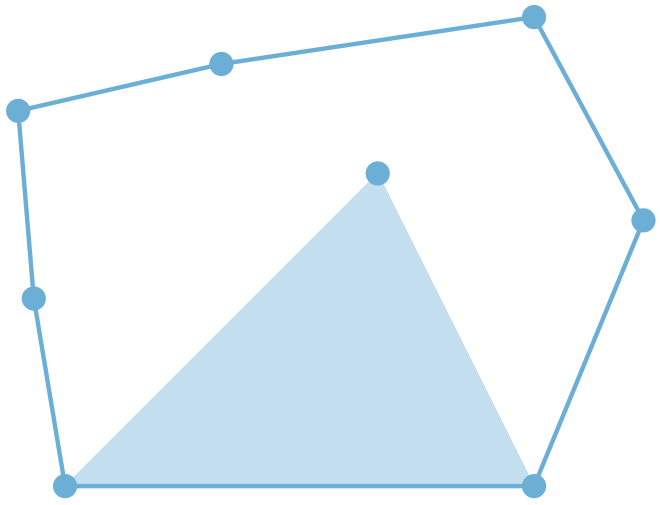
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.
- Điểm nằm trong đa giác. khi và chỉ khi tổng diện tích của tam giác bằng diện tích của đa giác.

Kiểm tra một điểm nằm trong đa giác

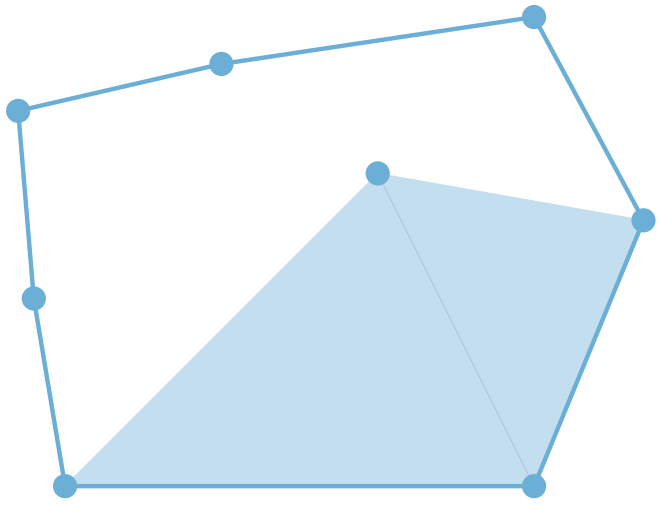
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.
- Điểm nằm trong đa giác. khi và chỉ khi tổng diện tích của tam giác bằng diện tích của đa giác.

Kiểm tra một điểm nằm trong đa giác

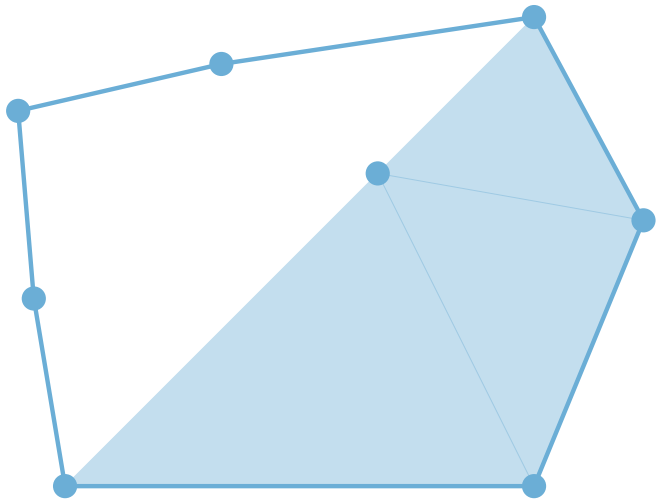
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.
- Điểm nằm trong đa giác. khi và chỉ khi tổng diện tích của tam giác bằng diện tích của đa giác.

Kiểm tra một điểm nằm trong đa giác

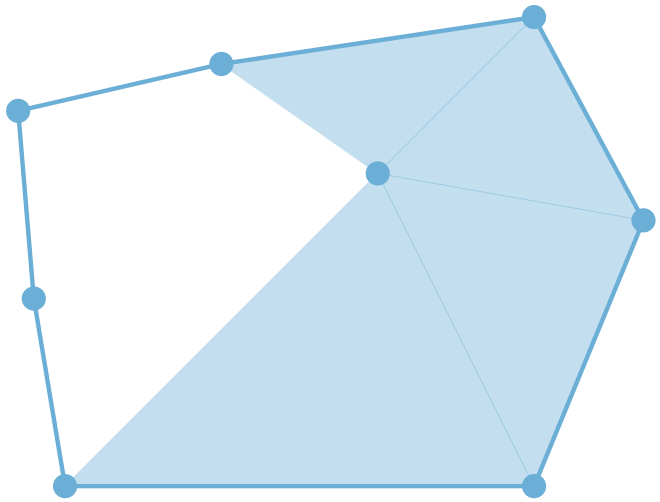
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.
- Điểm nằm trong đa giác. khi và chỉ khi tổng diện tích của tam giác bằng diện tích của đa giác.

Kiểm tra một điểm nằm trong đa giác

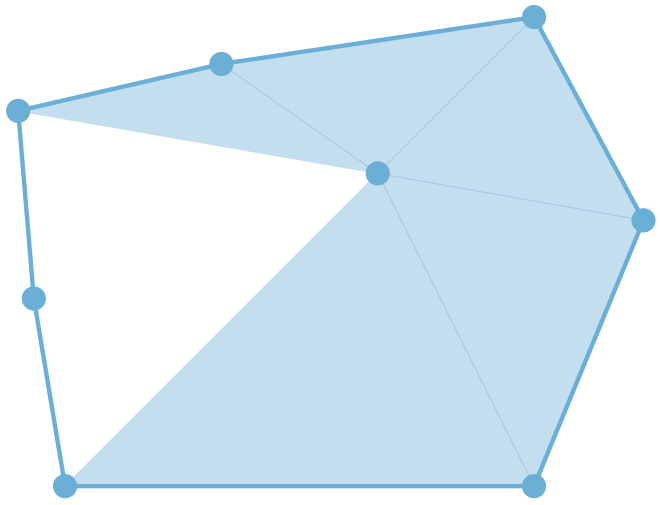
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.
- Điểm nằm trong đa giác. khi và chỉ khi tổng diện tích của tam giác bằng diện tích của đa giác.

Kiểm tra một điểm nằm trong đa giác

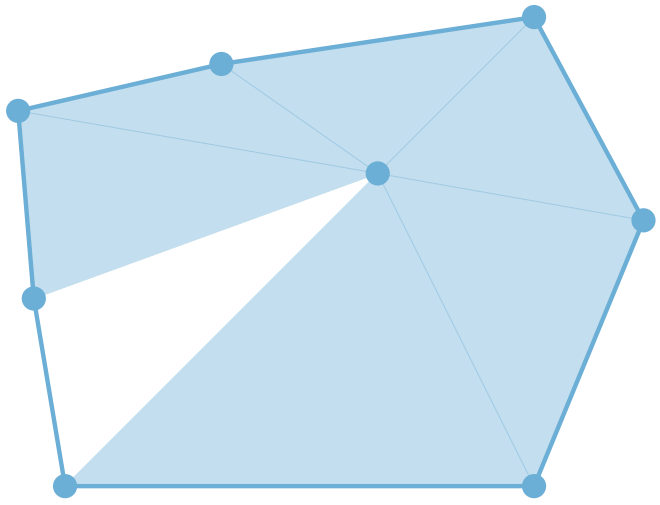
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.
- Điểm nằm trong đa giác. khi và chỉ khi tổng diện tích của tam giác bằng diện tích của đa giác.

Kiểm tra một điểm nằm trong đa giác

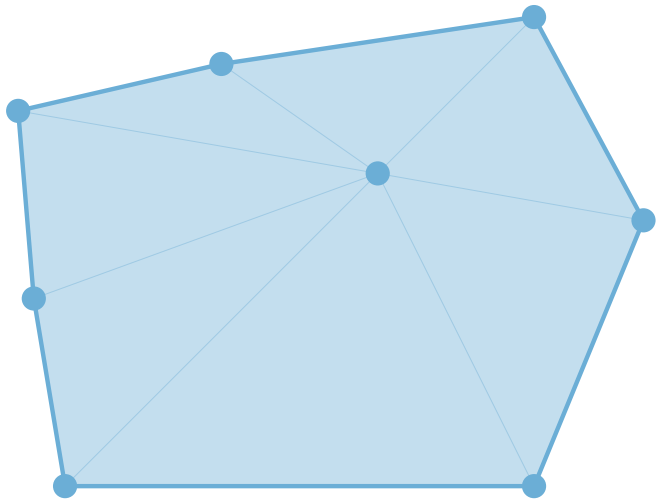
Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.
- Điểm nằm trong đa giác. khi và chỉ khi tổng diện tích của tam giác bằng diện tích của đa giác.

Kiểm tra một điểm nằm trong đa giác

Đa giác lồi:



- Bắt đầu bằng việc tính diện tích của đa giác.
- Để kiểm tra xem điểm cần xét có nằm trong đa giác hay không, hãy tổng hợp diện tích của các tam giác tạo bởi điểm đó và mọi cặp điểm liền kề.
- Điểm nằm trong đa giác. khi và chỉ khi tổng diện tích của tam giác bằng diện tích của đa giác.

Điểm trong đa giác lõm

Nếu đa giác không lồi thì sao?

Điểm trong đa giác lõm

Nếu đa giác không lồi thì sao?

- Sử dụng thuật toán *luật chẵn-lẻ*.

Điểm trong đa giác lõm

Nếu đa giác không lồi thì sao?

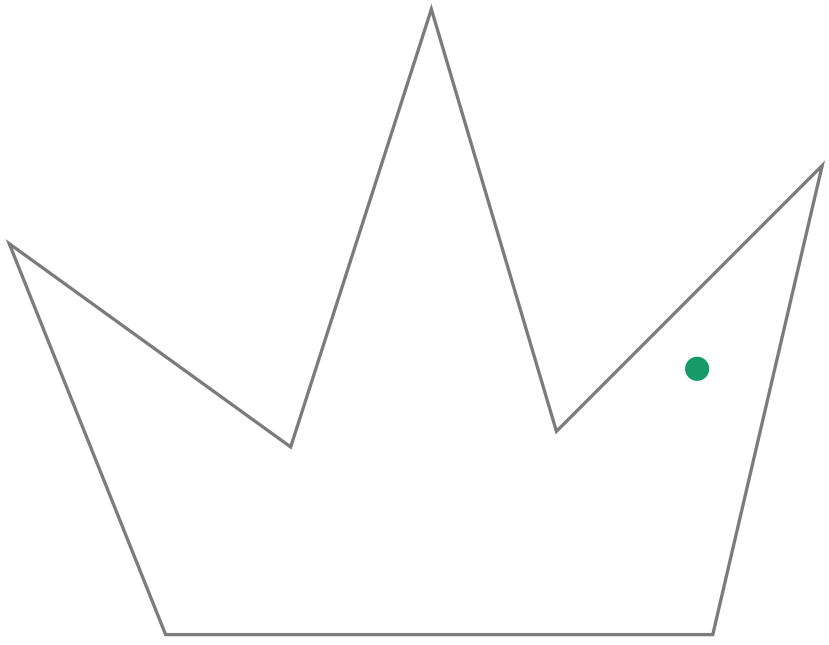
- Sử dụng thuật toán *luật chẵn-lẻ*.
- Kiểm tra một tia xuyên qua đa giác đến điểm cần xét.

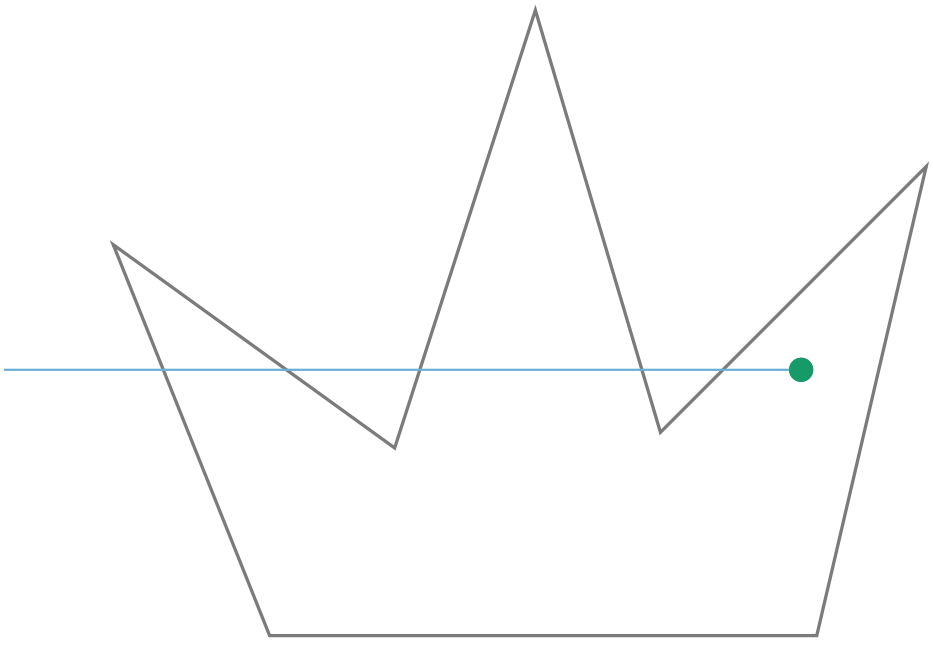
Điểm trong đa giác lõm

Nếu đa giác không lồi thì sao?

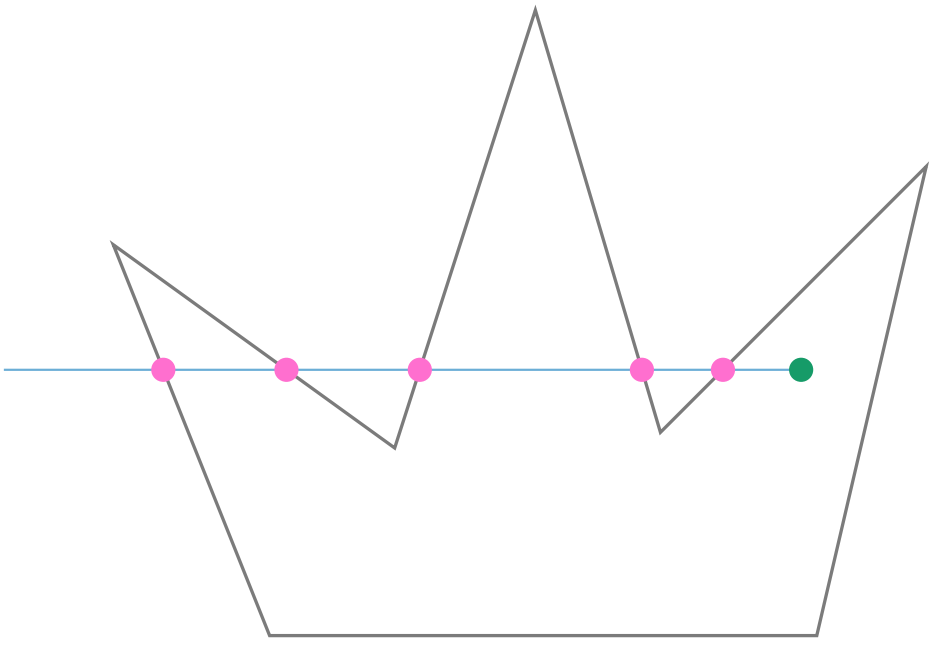
- Sử dụng thuật toán *luật chẵn-lẻ*.
- Kiểm tra một tia xuyên qua đa giác đến điểm cần xét.
- Nếu tia đi qua cạnh của đa giác, thế thì nó luân phiên đi từ bên ngoài vào bên trong, rồi bên ngoài vào bên trong.

CẨN THẬN TRƯỜNG HỢP TIA TRÙNG VỚI MỘT CẠNH ĐA GIÁC!

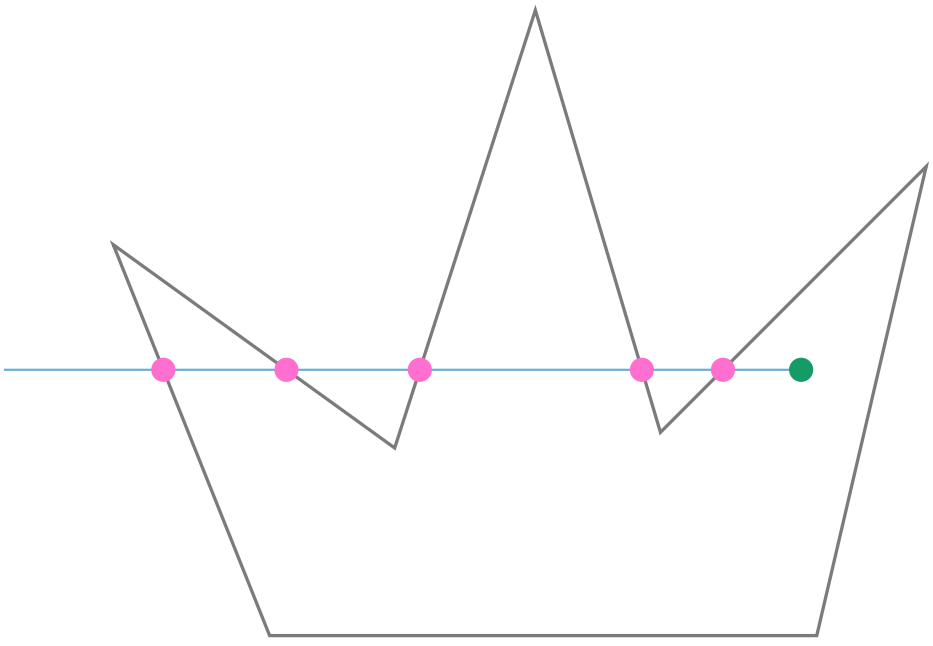




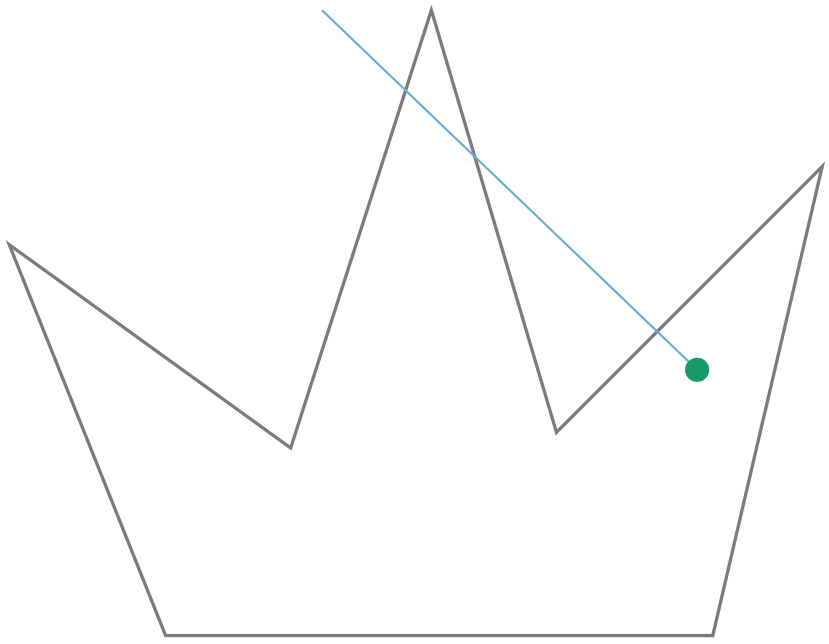
- Tia từ bên ngoài của đa giác đến điểm cần xét.



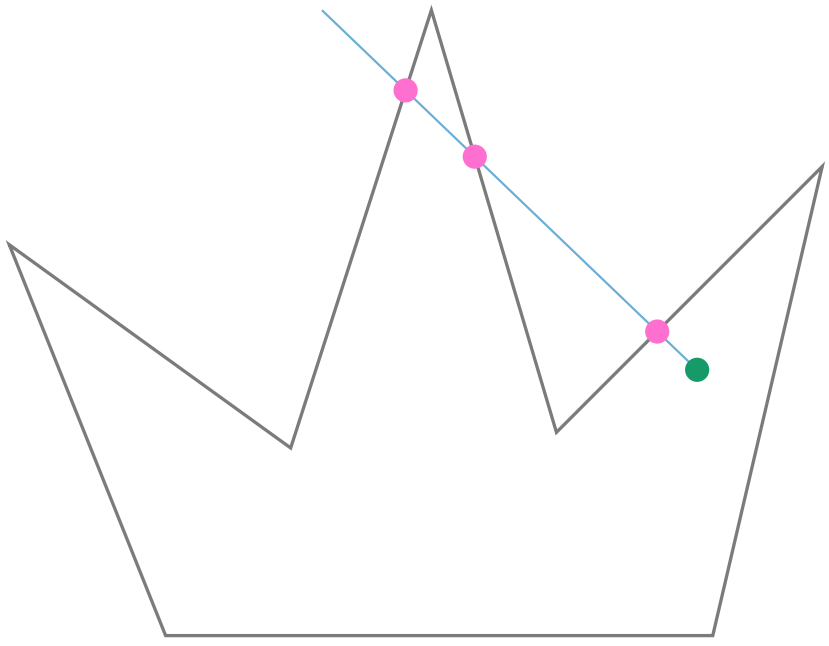
- Tia từ bên ngoài của đa giác đến điểm cần xét.
- Đếm số giao điểm.



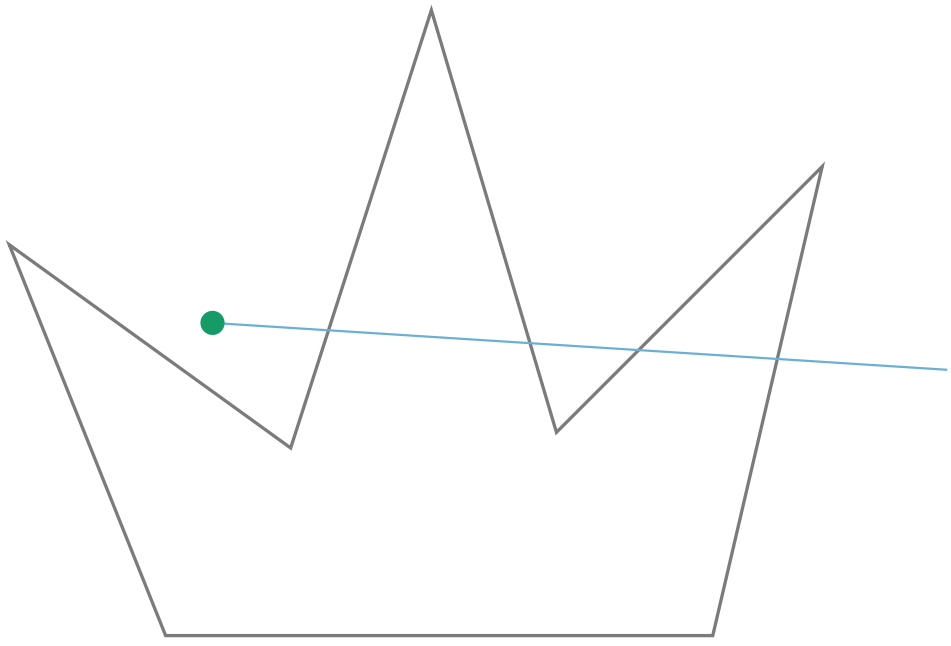
- Tia từ bên ngoài của đa giác đến điểm cần xét.
- Đếm số giao điểm.
- Nếu lẻ, thì điểm nằm trong đa giác.
- Nếu chẵn, thì điểm nằm ngoài đa giác.



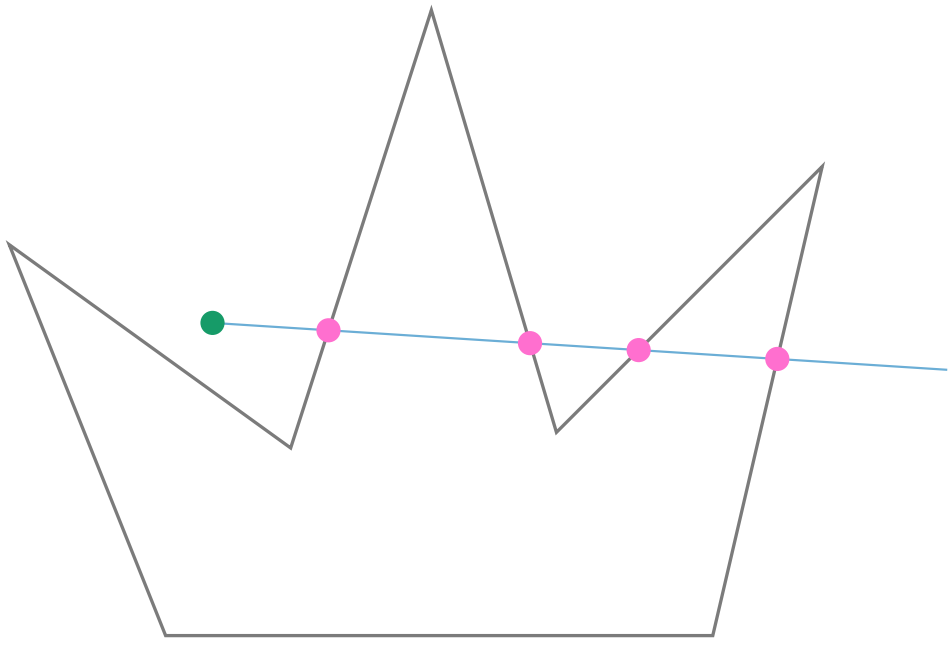
- Tia từ bên ngoài của đa giác đến điểm cần xét.
- Đếm số giao điểm.
- Nếu lẻ, thì điểm nằm trong đa giác.
- Nếu chẵn, thì điểm nằm ngoài đa giác.
- Không quan trọng là chọn tia nào.



- Tia từ bên ngoài của đa giác đến điểm cần xét.
- Đếm số giao điểm.
- Nếu lẻ, thì điểm nằm trong đa giác.
- Nếu chẵn, thì điểm nằm ngoài đa giác.
- Không quan trọng là chọn tia nào.



- Tia từ bên ngoài của đa giác đến điểm cần xét.
- Đếm số giao điểm.
- Nếu lẻ, thì điểm nằm trong đa giác.
- Nếu chẵn, thì điểm nằm ngoài đa giác.
- Không quan trọng là chọn tia nào.

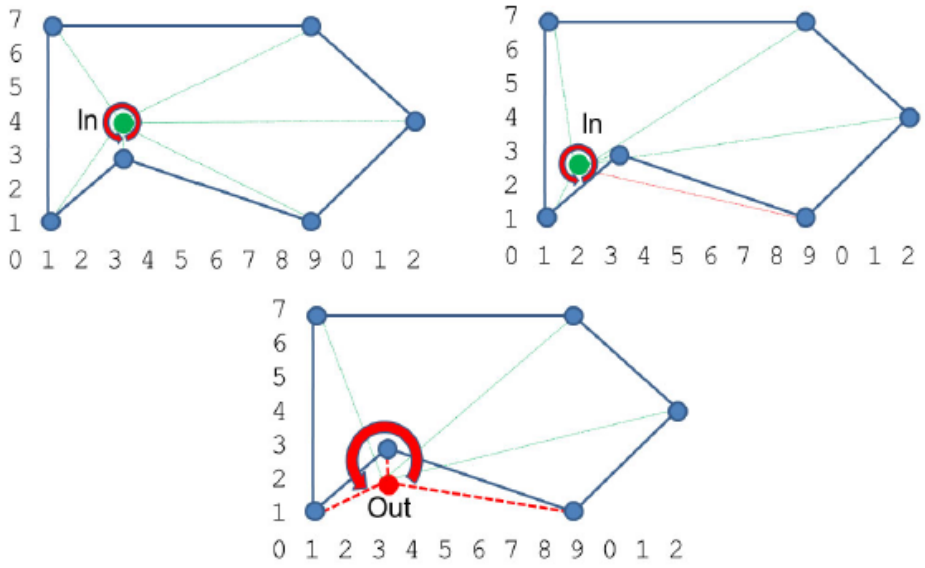


- Tia từ bên ngoài của đa giác đến điểm cần xét.
- Đếm số giao điểm.
- Nếu lẻ, thì điểm nằm trong đa giác.
- Nếu chẵn, thì điểm nằm ngoài đa giác.
- Không quan trọng là chọn tia nào.

Cách khác: Tính theo tổng các góc

Để kiểm tra một điểm pt thuộc miền trong của đa giác P hay không, ta có thể làm theo cách sau :

- Lấy tổng tất cả các góc giữa 3 điểm $P[i], pt, P[i+1]$với $(P[i] - P[i+1])$là các cạnh liên tiếp của P. Nếu tổng bằng 2πthì pt thuộc miền trong của P.



- Code:

```
bool inPolygon(point pt, const vector<point> &P) {
if ((int)P.size() == 0) return false;
double sum = 0; // gia thiet dinh dau trung dinh cuoi
for (int i = 0; i < (int)P.size()-1; i++) {
if (ccw(pt, P[i], P[i+1]))
sum += angle(P[i], pt, P[i+1]); // quay trai/ccw
else sum -= angle(P[i], pt, P[i+1]); } // quay phai/cw
return fabs(fabs(sum) - 2*PI) < EPS; }
```

Tìm cặp điểm gần nhất

Cho một tập điểm, hãy xác định cặp điểm có khoảng cách giữa chúng là nhỏ nhất.

Thuật toán chia để trị;

Tìm cặp điểm gần nhất

Cho một tập điểm, hãy xác định cặp điểm có khoảng cách giữa chúng là nhỏ nhất.

Thuật toán chia để trị;

- Sắp xếp các điểm theo toạ độ x.

Tìm cặp điểm gần nhất

Cho một tập điểm, hãy xác định cặp điểm có khoảng cách giữa chúng là nhỏ nhất.

Thuật toán chia để trị;

- Sắp xếp các điểm theo toạ độ x.
- Chia tập thành hai tập con có kích thước bằng nhau theo đường thẳng đứng có toạ độ x chính giữa.

Tìm cặp điểm gần nhất

Cho một tập điểm, hãy xác định cặp điểm có khoảng cách giữa chúng là nhỏ nhất.

Thuật toán chia để trị;

- Sắp xếp các điểm theo toạ độ x.
- Chia tập thành hai tập con có kích thước bằng nhau theo đường thẳng đứng có toạ độ x chính giữa.
- Giải bài toán theo cách đệ qui với tập con trái và tập con phải.

Tìm cặp điểm gần nhất

Cho một tập điểm, hãy xác định cặp điểm có khoảng cách giữa chúng là nhỏ nhất.

Thuật toán chia để trị;

- Sắp xếp các điểm theo toạ độ x.
- Chia tập thành hai tập con có kích thước bằng nhau theo đường thẳng đứng có toạ độ x chính giữa.
- Giải bài toán theo cách đệ qui với tập con trái và tập con phải.
- Sắp xếp hai tập con theo toạ độ y.

Tìm cặp điểm gần nhất

Cho một tập điểm, hãy xác định cặp điểm có khoảng cách giữa chúng là nhỏ nhất.

Thuật toán chia để trị;

- Sắp xếp các điểm theo toạ độ x.
- Chia tập thành hai tập con có kích thước bằng nhau theo đường thẳng đứng có toạ độ x chính giữa.
- Giải bài toán theo cách đệ qui với tập con trái và tập con phải.
- Sắp xếp hai tập con theo toạ độ y.
- Tìm khoảng cách nhỏ nhất giữa các cặp điểm mà mỗi điểm nằm trên một phía khác nhau của đường thẳng chia.

ĐỐI VỚI CÁC BÀI TOÁN HÌNH HỌC:

LUÔN PHẢI XEM XÉT KỸ LƯỠNG CÁC TRƯỜNG HỢP ĐẶC BIỆT!