計算幾何

sam571128

November 2, 2021

用電腦算幾何?

計算幾何，顧名思義就是用電腦來計算幾何問題，但是，看到幾何，大家應該會很害怕，因為數學上的幾何真的很難。不過事實上，在高中的競賽題目中，需要用的的幾何並不多。

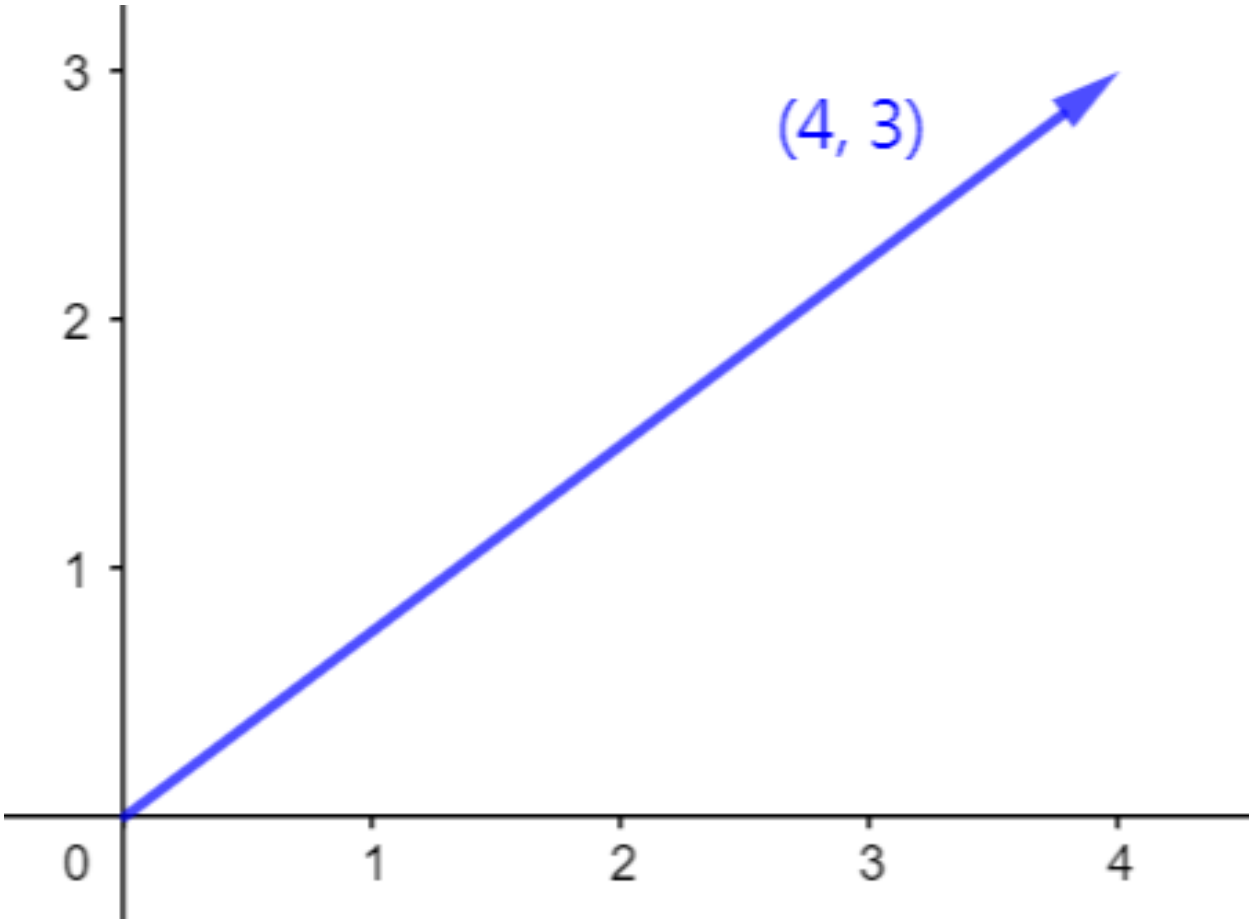
用電腦算幾何?

高中計算幾何涵蓋的範圍大概是以下幾種:

1. 向量
2. 凸包
3. 旋轉卡尺

向量

向量，簡單來說，他有一個方向，也有一個量，在數學上會長這樣



向量

向量的表示法:

- 多元組: $(v_1, v_2, \cdots)$
- 矩陣: $\begin{bmatrix} v_1 \\ v_2 \\ \vdots \end{bmatrix}$ 或 $\begin{bmatrix} v_1 & v_2 & \cdots \end{bmatrix}$

向量

而一個向量的長度，或者說是他的量，我們可以用畢氏定理得到

$$\sqrt{v_1^2 + v_2^2 + \cdots}$$

向量

而一個向量的長度，或者說是他的量，我們可以用畢氏定理得到

$$\sqrt{v_1^2 + v_2^2 + \cdots}$$

向量

符號表示:

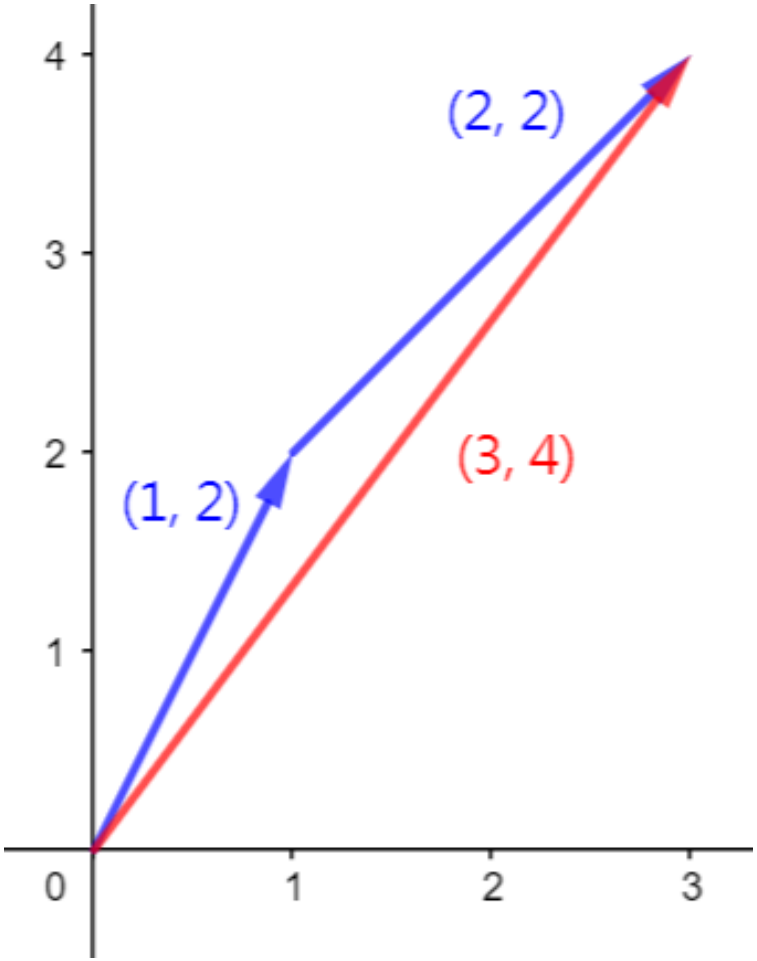
- 由 A 點指到 B 點的向量，$\overrightarrow{AB}$ 或 $B - A$
- 由原點 O 指到 A 點的向量，$\overrightarrow{A}$
- A 向量的長度，$|\overrightarrow{A}|$

向量的運算

- 加法 $+$
- 減法 $-$
- 純量乘法 $c\overrightarrow{A}$
- 內積 $\cdot$ (Dot Product)
- 外積 $\times$ (Cross Product)

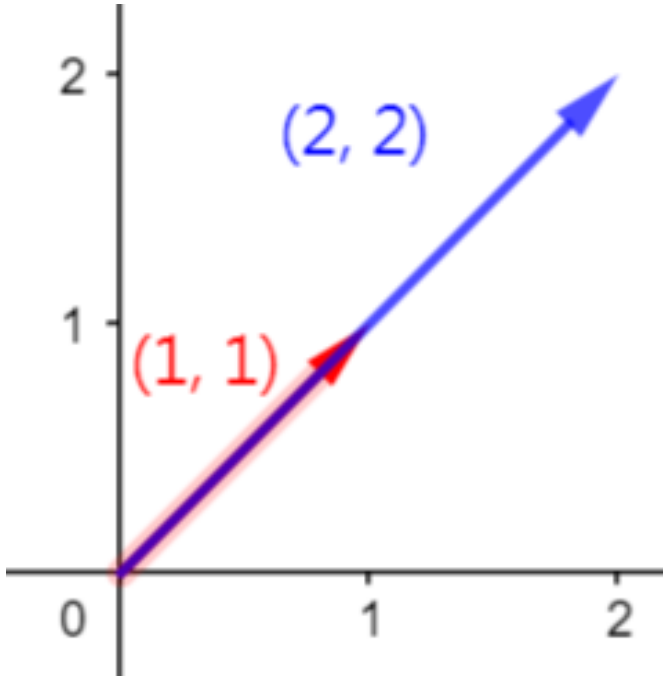
向量的運算

加法，$\overrightarrow{A} + \overrightarrow{B} = (u_1, u_2, \cdots) + (v_1, v_2, \cdots) = (u_1 + v_1, u_2 + v_2, \cdots)$



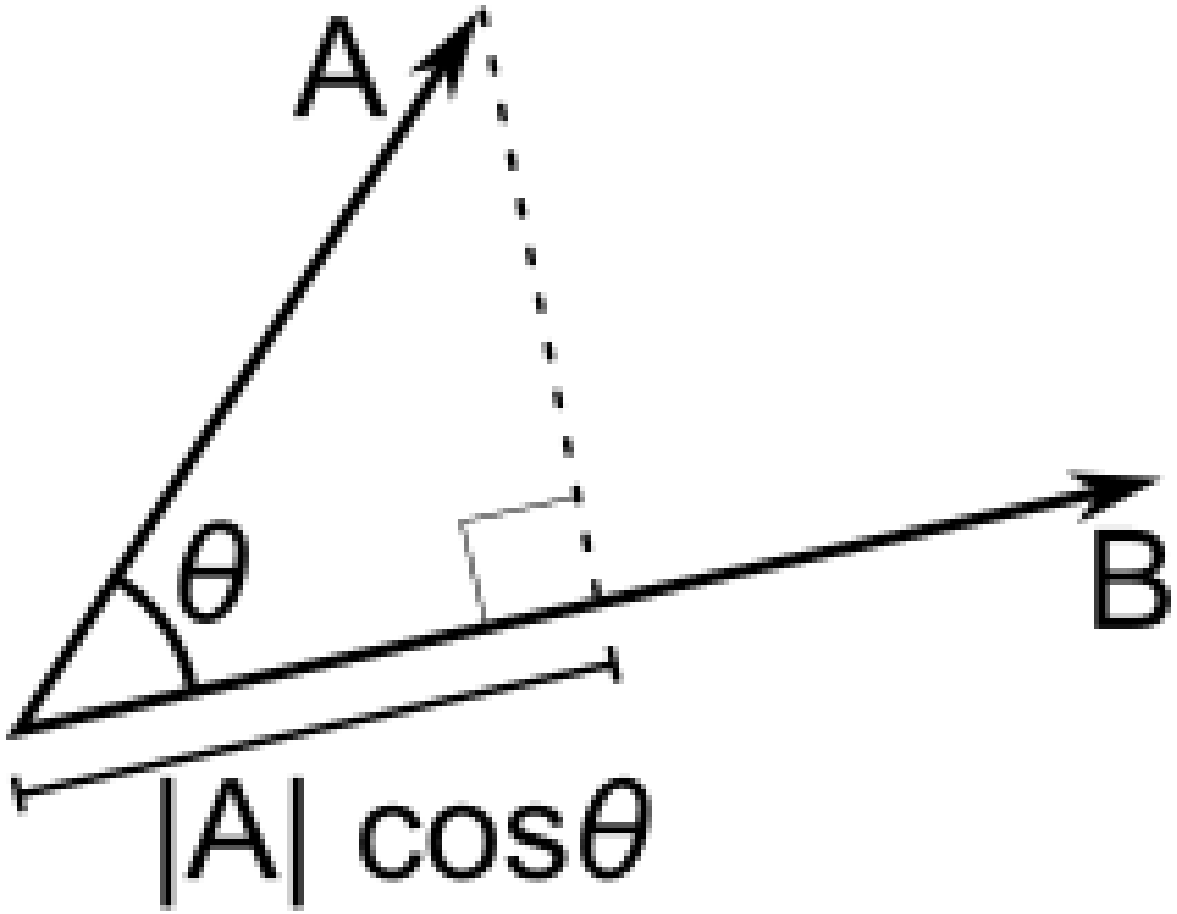
向量的運算

純量乘法，$c\overrightarrow{A} = c(u_1, u_2, \cdots) = (cu_1, cu_2, \cdots)$



向量的運算

內積，$\overrightarrow{A} \cdot \overrightarrow{B} = (u_1, u_2, \cdots) \cdot (v_1, v_2, \cdots) = u_1 v_1 + u_2 v_2 + \cdots$



向量的運算

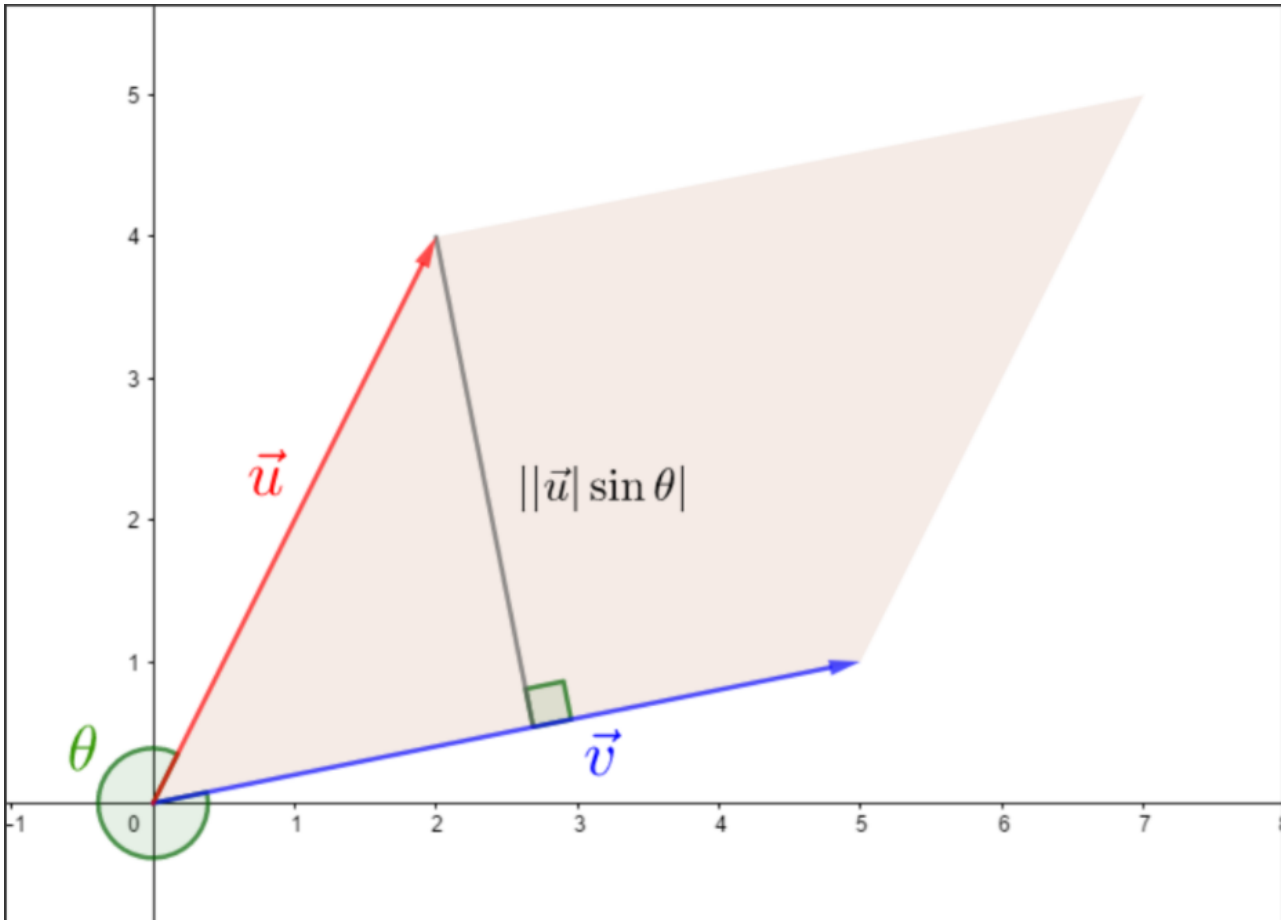
內積，又等於 $\overrightarrow{A} \cdot \overrightarrow{B} = |\overrightarrow{A}||\overrightarrow{B}| \cos(\theta)$

向量的運算

而根據這個性質，當 $\vec{u} \perp \vec{v}$ 時，$\vec{u} \cdot \vec{v} = 0$

向量的運算

外積，二維向量的外積 $\vec{u} \times \vec{v} = \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} = x_1 y_2 - x_2 y_1$

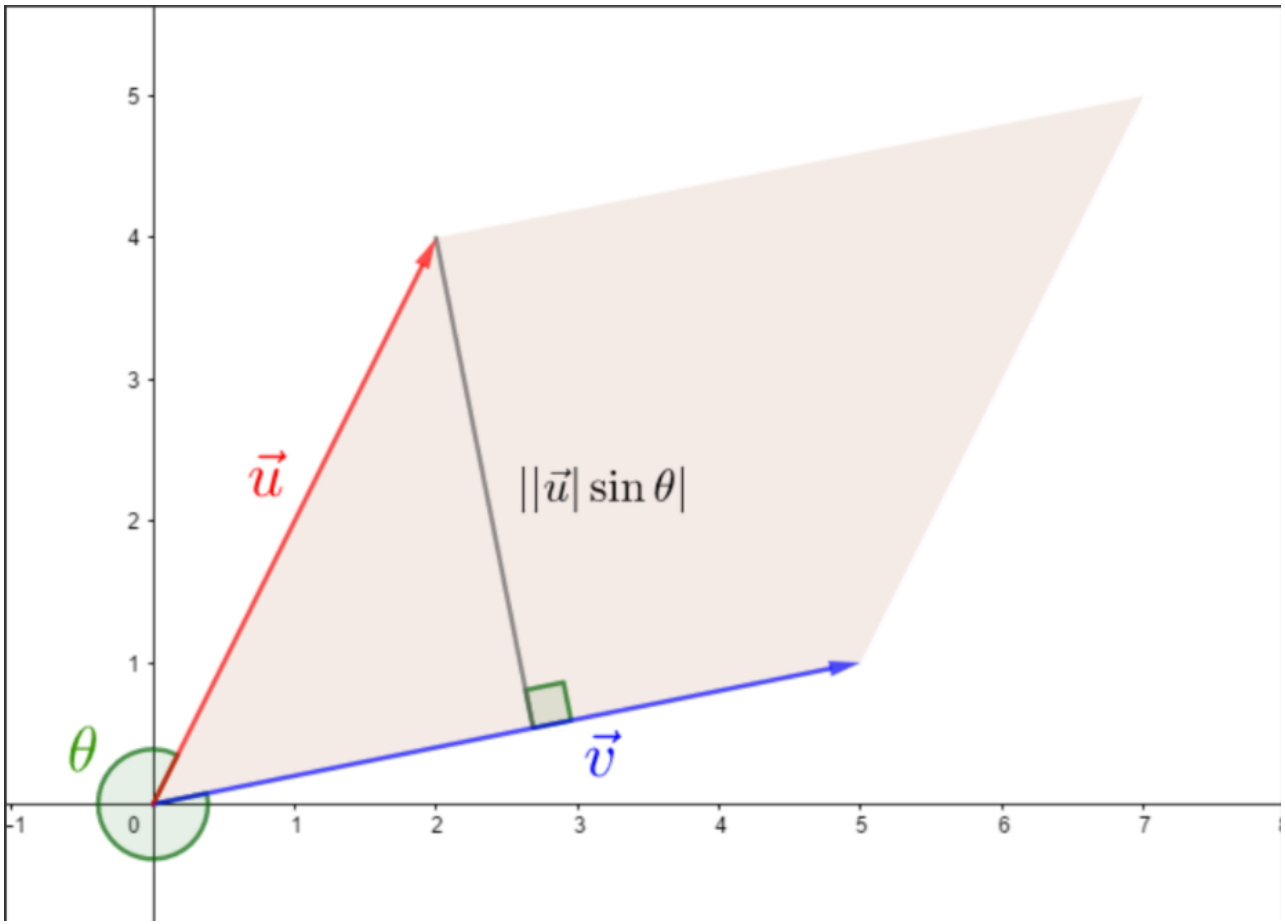


向量的運算

外積，又等於 $\overrightarrow{A} \times \overrightarrow{B} = |\overrightarrow{A}||\overrightarrow{B}| \sin(\theta)$，也等於兩個向量夾成的平行四邊形面積

向量的運算

而根據外積，我們可以得到兩個向量夾出的**有向面積**



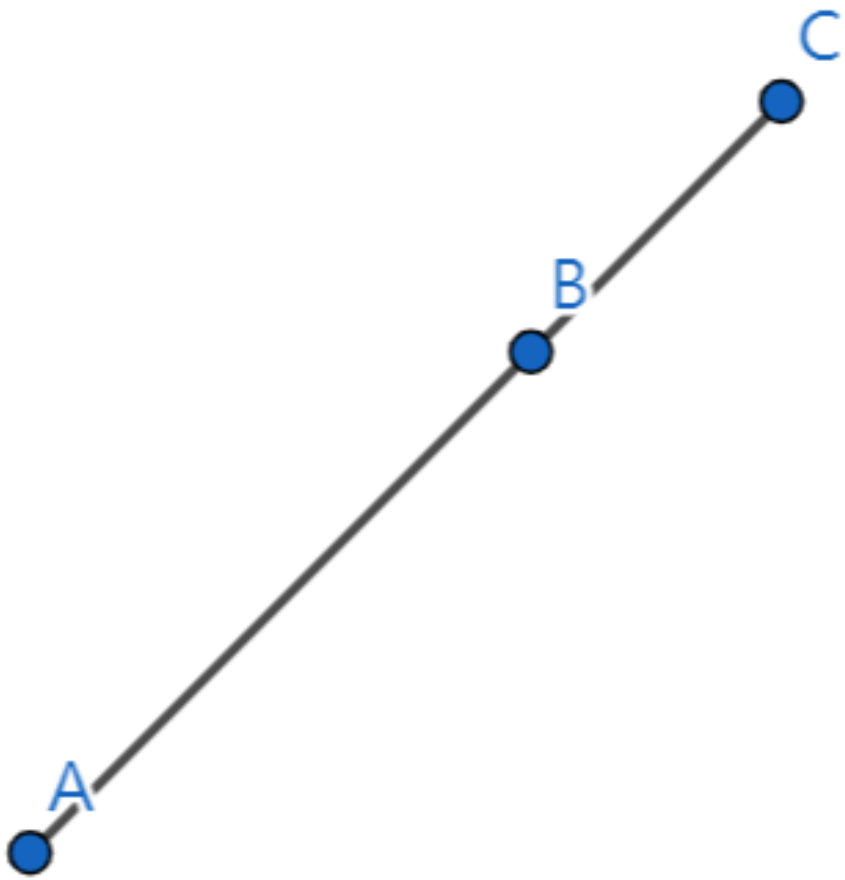
向量的運算

這些東西寫成程式會長這樣

```
typedef pair<int,int> point;
#define x first
#define y second

point operator +(point a, point b){
    return {a.x+b.x,a.y+b.y};
}

point operator -(point a, point b){
    return {a.x-b.x,a.y-b.y};
}

int operator ^(point a, point b){
    return a.x*b.y-a.y*b.x;
}

int operator *(point a, point b){
    return a.x*b.x+a.y*b.y;
}

int abs(point a){
    return a.x*a.x+a.y*a.y;
}
```

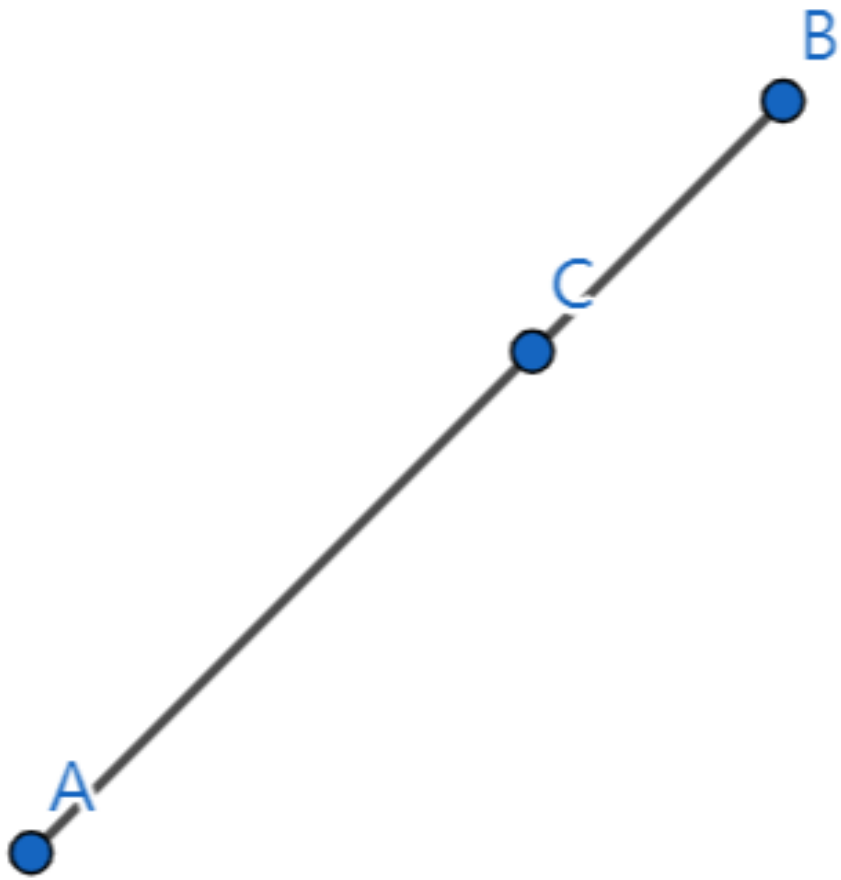
三點共線

$$(B - A) \times (C - A) = 0$$



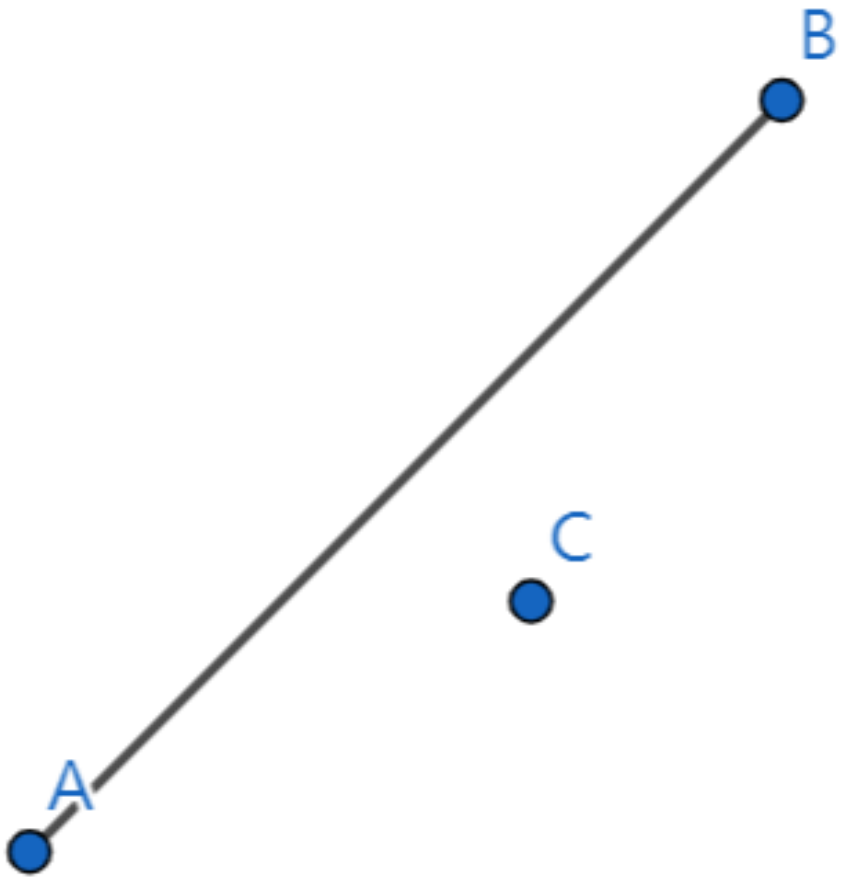
C 是否在 $\overline{AB}$ 上

三點共線，且 $\overrightarrow{CA} \cdot \overrightarrow{CB}$



C 點在 $\overrightarrow{AB}$ 的左邊還右邊

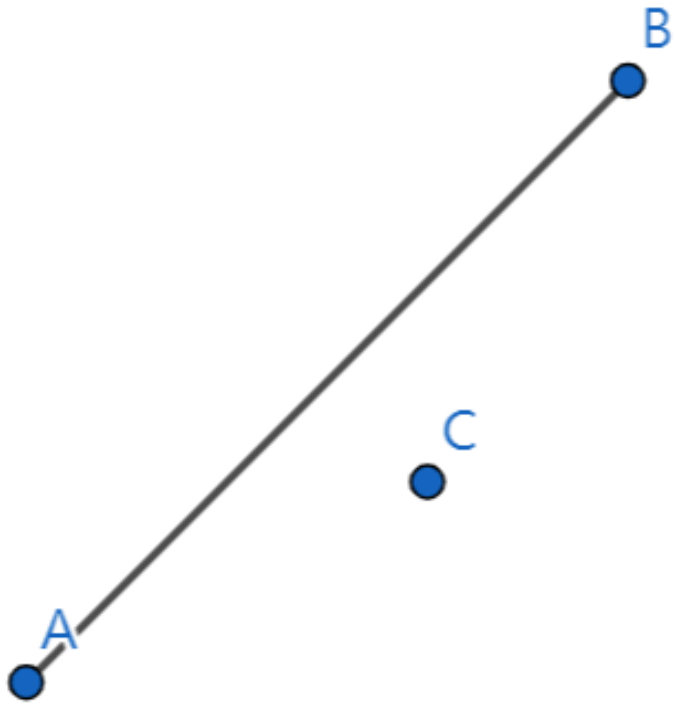
左邊 $\overrightarrow{AC} \times \overrightarrow{BC} > 0$，右邊 $\overrightarrow{AC} \times \overrightarrow{BC} < 0$



線段相交

$$\begin{cases} (\overrightarrow{AB} \times \overrightarrow{AC}) \cdot (\overrightarrow{AB} \times \overrightarrow{AD}) < 0 \\ (\overrightarrow{CD} \times \overrightarrow{CA}) \cdot (\overrightarrow{CD} \times \overrightarrow{CB}) < 0 \end{cases}$$

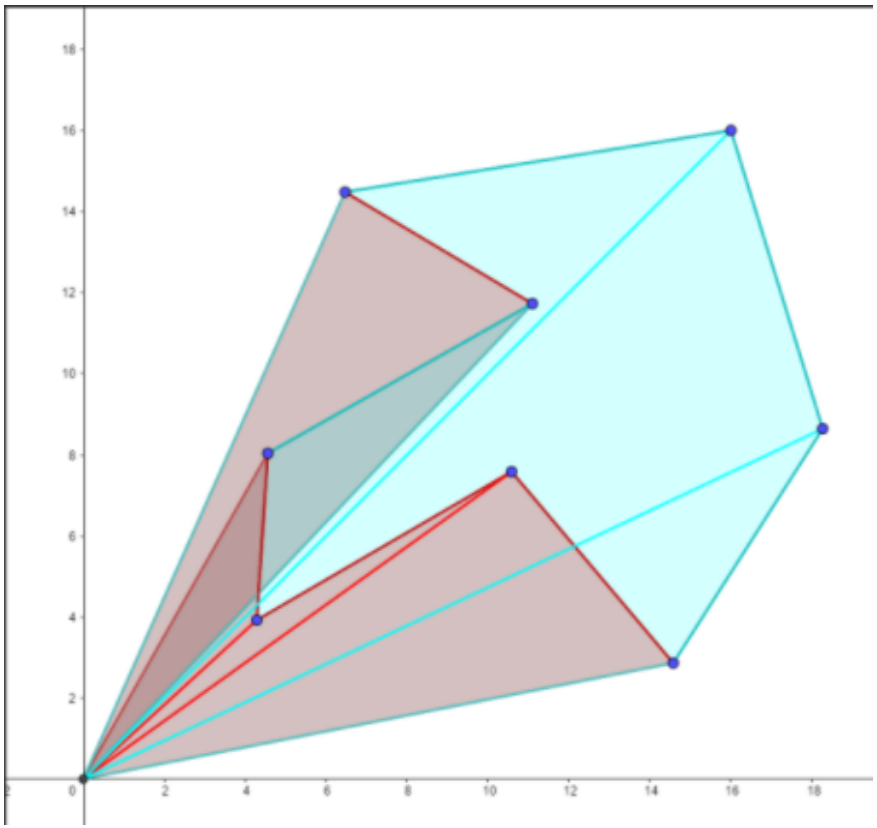
特判兩端點



多邊形面積

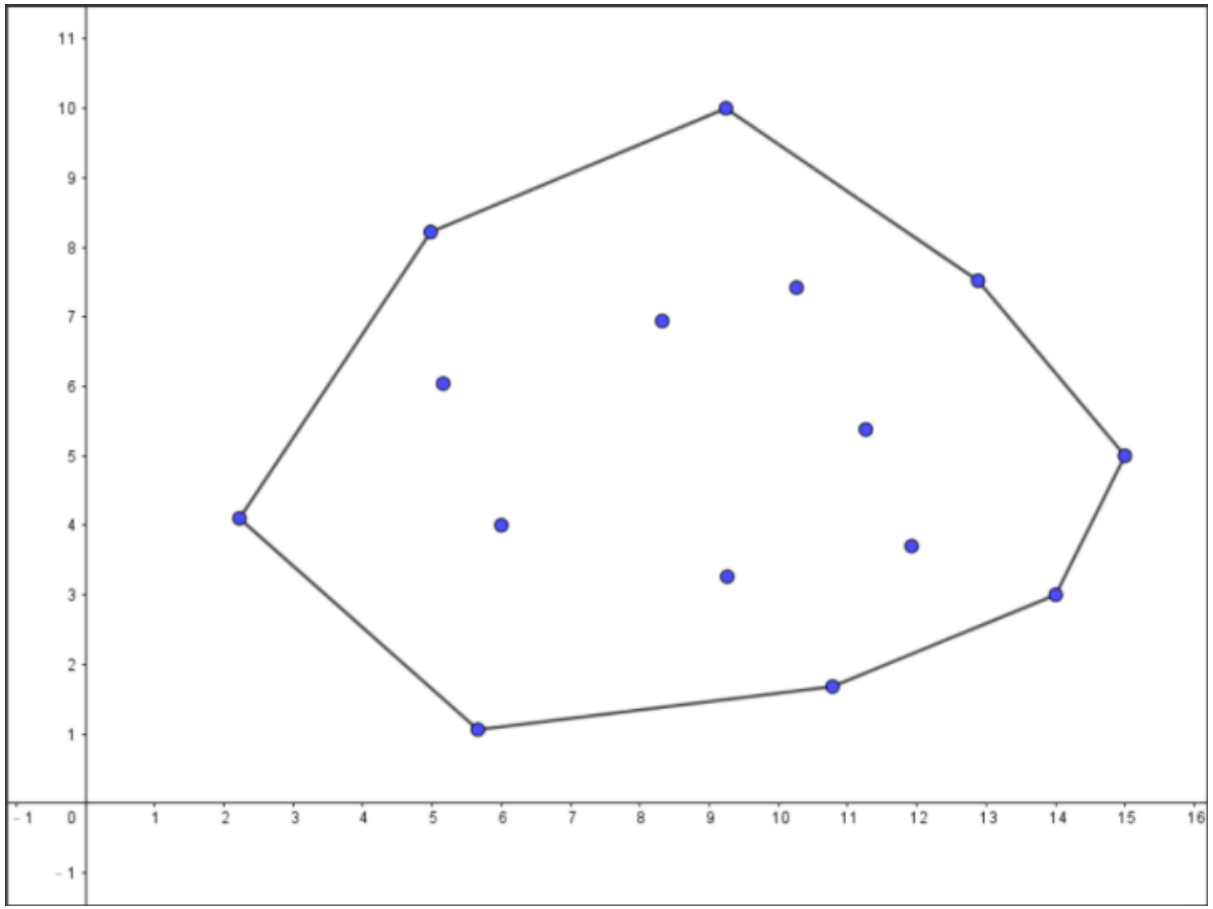
對於一個多邊形 $p_0, p_1, \cdots$ (順時針或逆時針)

我們可以用測量師公式，$\sum \begin{vmatrix} x_i & x_{i+1} \\ y_i & y_{i+1} \end{vmatrix}$



凸包

對於平面上的一些點，找到面積最小的多邊形可以包覆住所有點



凸包

而凸包主要有兩種演算法可以找

- Graham's Scan
- Andrew's Monotone Chain

而後者比較常用且好懂，因此我們這裡只講後者

凸包

步驟:

1. 對點依照 x 做排序
2. 依序去加入一個 stack
3. 若是加入這個點之後與前兩個點出現負的面積，就 pop
4. 最後會得到下凸包
5. 反過來再做一次，得到上凸包
6. 合併

凸包

程式碼:

```
vector<point> getCH(vector<point> v){
    int n = v.size();
    sort(v.begin(),v.end());
    vector<pair<T,T>> hull;
    for(int i = 0;i < 2;i++){
        int t = hull.size();
        for(auto x : v){
            while(hull.size()-t>=2&&((hull[hull.size()-1]-hull[hull.size()-2])^(x-hull[hull.size()-2]))<=0)
                hull.pop_back();
            hull.push_back(x);
        }
        hull.pop_back();
        reverse(v.begin(), v.end());
    }
    return hull;
}
```

極角排序

將點依照離中心的角度做排序

```
bool cmp(point a,point b){
    a=a-c, b=b-c; //c is center
    if(fabs(atan2(a.y,a.x)-atan2(b.y,b.x)) > eps)
    return atan2(a.y,a.x) < atan2(b.y,b.x);
    return abs(a) < abs(b);
}
```

總結

對於計算幾何，我會的也不是很多，但競賽中用的到的差不多就這些。

下次我們應該會來談談圖論。