# 基础

## 点(Point)

```
struct Point {
    db x, y;
};

inline db dis(const Point &a, const Point &b) {
    db dx = a.x - b.x;
    db dy = a.y - b.y;
    return sqrt(dx * dx + dy * dy);
}
```

## 向量(Vector)

```
using Vector = Point;

inline db dot(const Vector &a, const Vector &b) {
    return a.x * b.x + a.y * b.y;
}

inline db cross(const Vector &a, const Vector &b) {
    return a.x * b.y - a.y * b.x;
}

inline Vector operator-(const Point &a, const Point &b) {
    return Vector{a.x - b.x, a.y - b.y};
}
```

### 基础用法

- 将一个向量$\vec{a}$逆时针旋转$\theta$度

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \times \begin{bmatrix} a_x \\ b_x \end{bmatrix} = \begin{bmatrix} \cos\theta\, a_x - \sin\theta\, a_y \\ \sin\theta\, a_x + \cos\theta\, a_y \end{bmatrix}$$

## 点积(Dot)

$$\vec{a} \cdot \vec{b} = a_x b_x + a_y b_y$$

## 叉积(Cross)

$$\vec{a} \times \vec{b} = a_x b_y - a_y b_x$$

- 平行四边形面积: $\|\vec{a}\|\|\vec{b}\|\,|\sin\theta| = \|\vec{a}\times\vec{b}\|$
- 向量平行: $\vec{a}\times\vec{b} = 0$
- **TO_LEFT测试**
    - 判断点$P$在向量$AB$的左侧还是右侧
    - $\vec{a}\times\vec{b} > 0$ $P$在向量$AB$左侧
    - $\vec{a}\times\vec{b} < 0$ $P$在向量$AB$右侧
    - $\vec{a}\times\vec{b} = 0$ $P$在向量$AB$上

# 线段(Segment)

```
struct Segment {
    Point a, b;
};
```

# 基础用法

- 判断点$P$是否在线段$AB$上(含端点)
    - $\vec{a}\times\vec{b} = \vec{0}$
    - $\vec{a}\cdot\vec{b} \le 0$
- 判断线段$AB$，$CD$是否相交
    - 特判三点共线和四点共线
    - 通过叉积判断
    - 点C和点D在线段AB的不同侧
    - 点A和点B在线段CD的不同侧

# 直线

点向式 `struct Line { Point p, v; };`

## 基础用法

- 输入直线$(P, \vec{v})$与点A，求A到直线距离

$$\|\vec{AB}\| = \|\vec{PA}\|\,|\sin\theta| = \frac{\|\vec{v}\times\vec{PA}\|}{\|\vec{v}\|}$$

- 输入直线$(P, \vec{v})$与点A，求A在直线上的投影点B 点乘算投影
- 两直线求交点 直线 $\{P_1, \vec{v_1}\}\{P_2, \vec{v_2}\}$

$$\begin{cases} \dfrac{\|P_1Q\|}{\sin\alpha} = \dfrac{\|P_1P_2\|}{\sin\beta} \\ \|\vec{v}_2\times\vec{P_2P_1}\| = \|\vec{v}_2\|\|\vec{P_2P_1}\|\sin\alpha \end{cases}$$

$$\|\vec{v}_1\times\vec{v}_2\| = \|\vec{v}_1\|\|\vec{v}_2\|\sin\beta$$

$$\|P_1Q\| = \frac{\|\vec{v}_2 \times \vec{P_2P_1}\|\|\vec{v}_1\|}{\|\vec{v}_1 \times \vec{v}_2\|}$$

$$\vec{OQ} = \vec{OP_1} + \vec{P_1Q} = \vec{OP_1} + \frac{\|P_1Q\|}{\|\vec{v}_1\|}\vec{v}_1 = \vec{OP_1} + \frac{\|\vec{v}_2 \times \vec{P_2P_1}\|}{\|\vec{v}_1 \times \vec{v}_2\|}\vec{v}_1$$

# 多边形

`struct Polygon { vector<Point> p; };` 一般默认按照逆时针排序

## 基础用法

- 计算多边形面积(三角剖分) $S = \frac{1}{2}\|\sum_{i=0}^{n-1} \vec{OP_I} \times \vec{OP_{(i+1)\,mod\,n}}\|$
- 判断点是否在多边形内部
    1. 从该点引出一条射线，如果与多边形有奇数个交点，则在内部，否则在多边形外部 (不能交到顶点)
    2. 遍历多边形的点，如果转动圈数为0，点在多边形外部，否则在内部 (计算角度有精度误差)
    3. 水平引出一条射线，逆时针依次遍历边，如果边从上向下穿过射线，val--，否则val++，如果val=0则点在多边形外 (优秀)

```

```

# 圆