

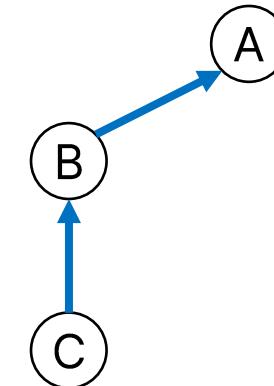
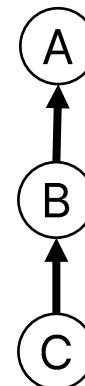
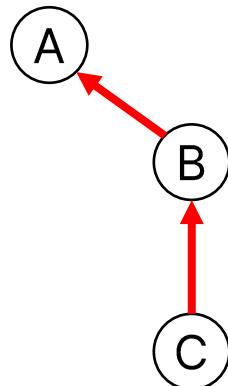
# **Geometry**

# 벡터의 외적

- $\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$  ( $2 \times 2$  행렬식)
- $\vec{a} \times \vec{b} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{vmatrix}$  와 같다. ( $\vec{i}, \vec{j}, \vec{k}$ 는 각각  $x, y, z$  축의 단위벡터이다.)
- $= \begin{vmatrix} a_2 & a_3 \\ b_2 & b_3 \end{vmatrix} \vec{i} - \begin{vmatrix} a_1 & a_3 \\ b_1 & b_3 \end{vmatrix} \vec{j} + \begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix} \vec{k}$
- $= \begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix} \vec{k}$  ( $a_3, b_3 = 0$ 이므로)  $= (0, 0, a_1b_2 - a_2b_1)$
- 두 벡터  $\vec{a}, \vec{b}$ 의 외적은  $\vec{a} \cdot \vec{b}$ 이며  **$\vec{a}, \vec{b}$ 가 이루는 평면에 수직인 방향의 벡터** 값이다.

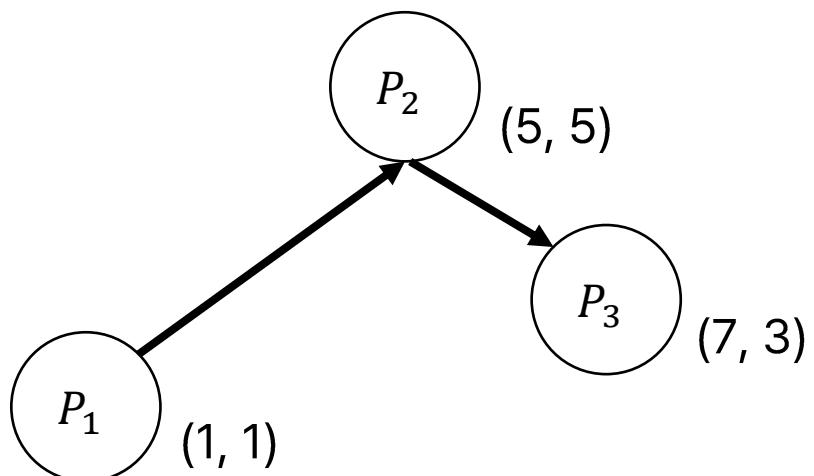
# CCW(Counterclockwise)

- 3개의 점  $A, B, C$ 를 이은 **직선의 방향성**을 구하는 알고리즘
- 시계 반대 방향이면 **양수**, 평행하면 **0**, 시계 방향이면 **음수**를 반환



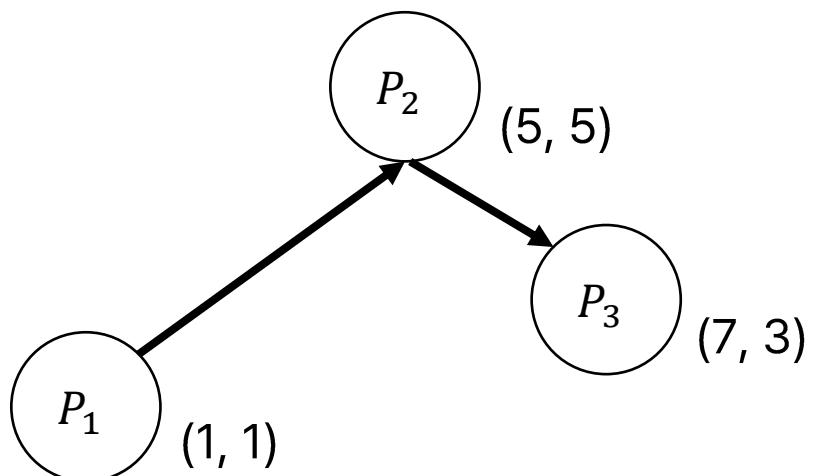
# CCW BOJ 11758

- 2차원 좌표 평면 위에 있는 점 3개  $P_1, P_2, P_3$ 가 주어진다.
- $P_1, P_2, P_3$ 를 순서대로 이은 선분이 어떤 방향을 이루고 있는지 구해보자.



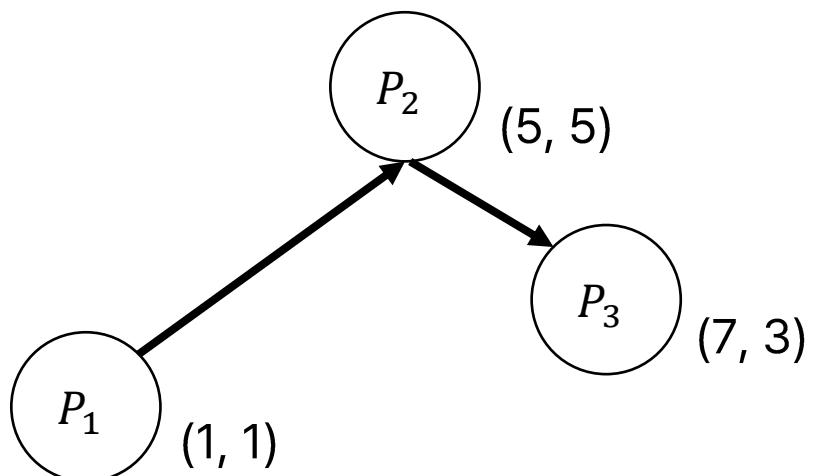
# CCW BOJ 11758

- 2차원 좌표 평면 위에 있는 점 3개  $P_1, P_2, P_3$ 가 주어진다.
- $P_1, P_2, P_3$ 를 순서대로 이은 선분이 어떤 방향을 이루고 있는지 구해보자.
- 두 벡터를 먼저 구성해보자.



# CCW BOJ 11758

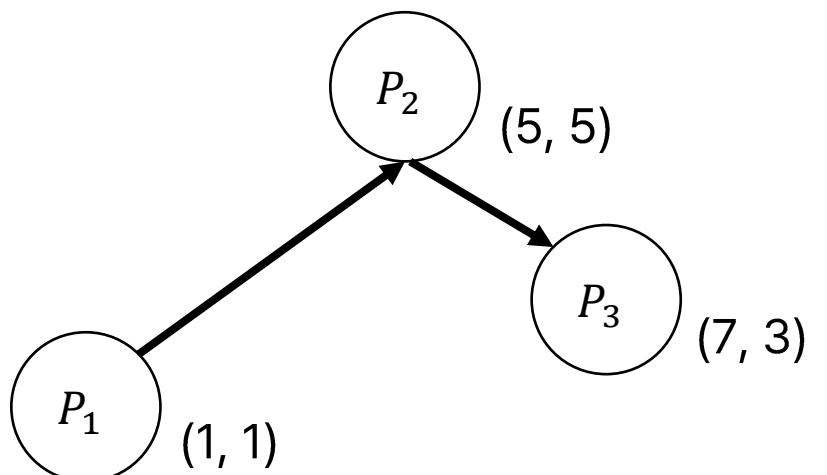
- 2차원 좌표 평면 위에 있는 점 3개  $P_1, P_2, P_3$ 가 주어진다.
- $P_1, P_2, P_3$ 를 순서대로 이은 선분이 어떤 방향을 이루고 있는지 구해보자.
- 두 벡터를 먼저 구성해보자.



- $\overrightarrow{P_1P_2} = (5 - 1, 5 - 1, 0 - 0) = (4, 4, 0)$
- $\overrightarrow{P_2P_3} = (7 - 5, 3 - 5, 0 - 0) = (2, -2, 0)$

# CCW BOJ 11758

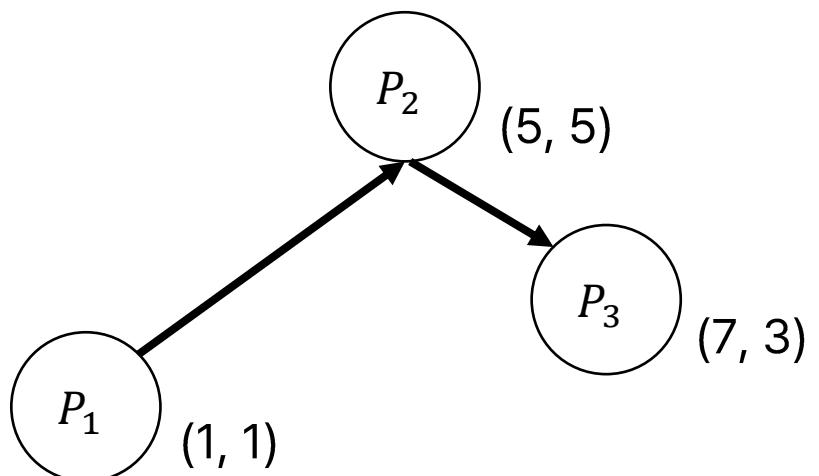
- 2차원 좌표 평면 위에 있는 점 3개  $P_1, P_2, P_3$ 가 주어진다.
- $P_1, P_2, P_3$ 를 순서대로 이은 선분이 어떤 방향을 이루고 있는지 구해보자.
- 외적을 구해주면 아래와 같이 나온다.



- $\overrightarrow{P_1P_2} = (5 - 1, 5 - 1, 0 - 0) = (4, 4, 0)$
- $\overrightarrow{P_2P_3} = (7 - 5, 3 - 5, 0 - 0) = (2, -2, 0)$
- $\overrightarrow{P_1P_2} \times \overrightarrow{P_2P_3} = (4 \times (-2) - (4 \times 2)) = -16$

# CCW BOJ 11758

- 2차원 좌표 평면 위에 있는 점 3개  $P_1, P_2, P_3$ 가 주어진다.
- $P_1, P_2, P_3$ 를 순서대로 이은 선분이 어떤 방향을 이루고 있는지 구해보자.
- 값이 음수이므로 시계방향이다.



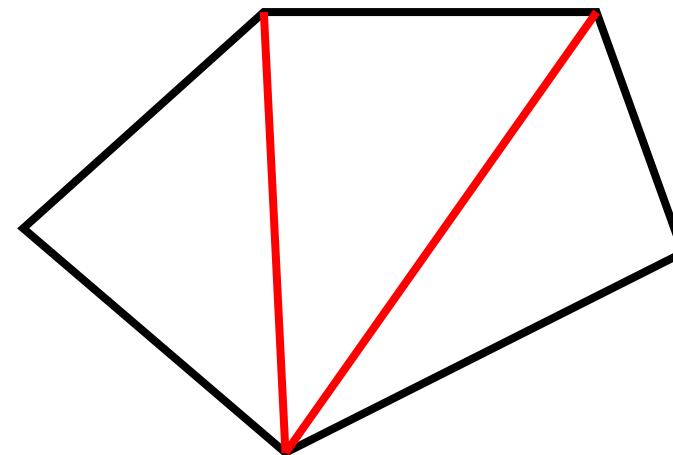
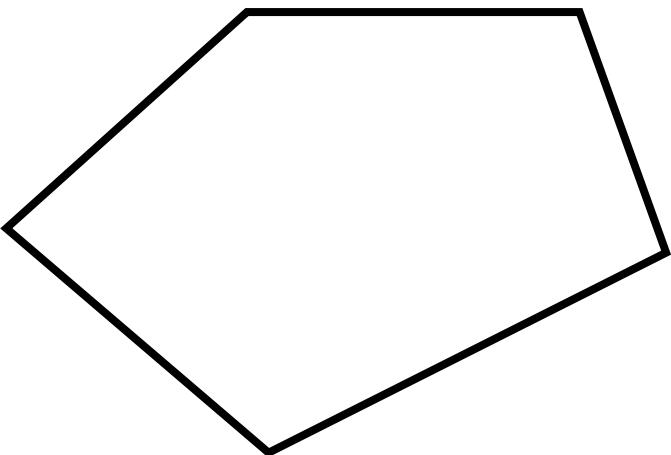
- $\overrightarrow{P_1P_2} = (5 - 1, 5 - 1, 0 - 0) = (4, 4, 0)$
- $\overrightarrow{P_2P_3} = (7 - 5, 3 - 5, 0 - 0) = (2, -2, 0)$
- $\overrightarrow{P_1P_2} \times \overrightarrow{P_2P_3} = (4 \times (-2) - (4 \times 2)) = -16$

# 벡터의 외적

- 두 벡터  $\vec{a}, \vec{b}$ 가 이루는 각의 크기를  $\theta$ 라고 하면  $|\vec{a} \times \vec{b}| = |\vec{a}| |\vec{b}| \sin \theta$  이다.
- 즉, 외적의 절댓값은 두 벡터가 만드는 평행사변형의 넓이이다.

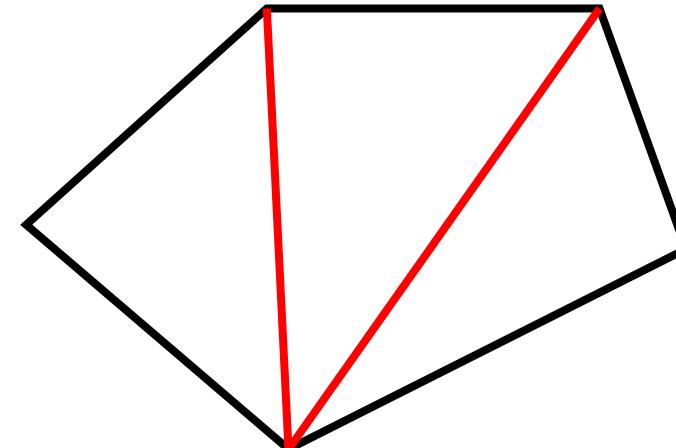
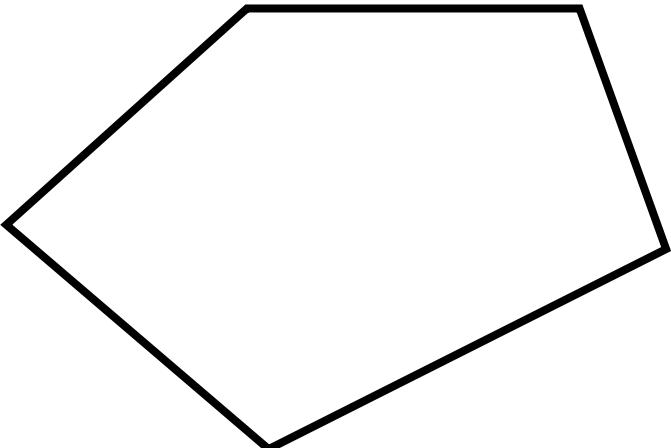
# 다각형의 넓이

- 외적의 절댓값은 변  $a, b$ 의 끼인각을  $\theta$ 로 가지는 **삼각형 넓이의 2배**이다.
- 볼록 다각형(모든 내각이  $180^\circ$ 미만인 다각형)은 아래와 같이 쉽게 나누어 구할 수 있다.



# 다각형의 넓이

- 외적의 절댓값은 변  $a, b$ 의 끼인각을  $\theta$ 로 가지는 **삼각형 넓이의 2배**이다.
- 볼록 다각형(모든 내각이  $180^\circ$ 미만인 다각형)은 아래와 같이 쉽게 나누어 구할 수 있다.
- 한 꼭짓점을 고정하고 한 방향으로 돌면서 외적 값을 구해준다.

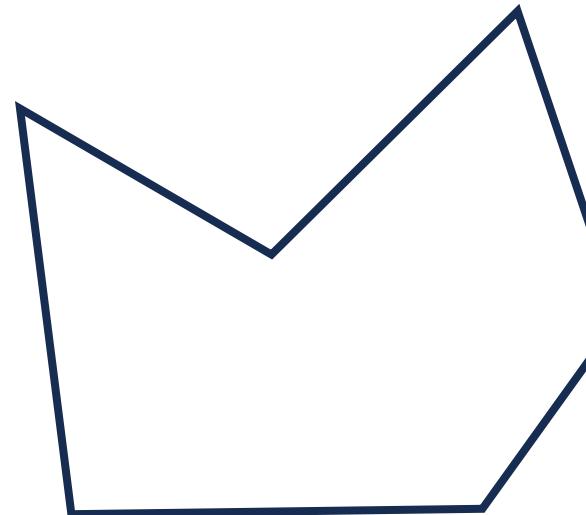


# 다각형의 넓이

- 외적의 절댓값은 변  $a, b$ 의 끼인각을  $\theta$ 로 가지는 **삼각형 넓이의 2배**이다.
- 볼록 다각형(모든 내각이  $180^\circ$ 미만인 다각형)은 아래와 같이 쉽게 나누어 구할 수 있다.
- 한 꼭짓점을 고정하고 한 방향으로 돌면서 외적 값을 구해준다.
- 만약 오목 다각형이면?

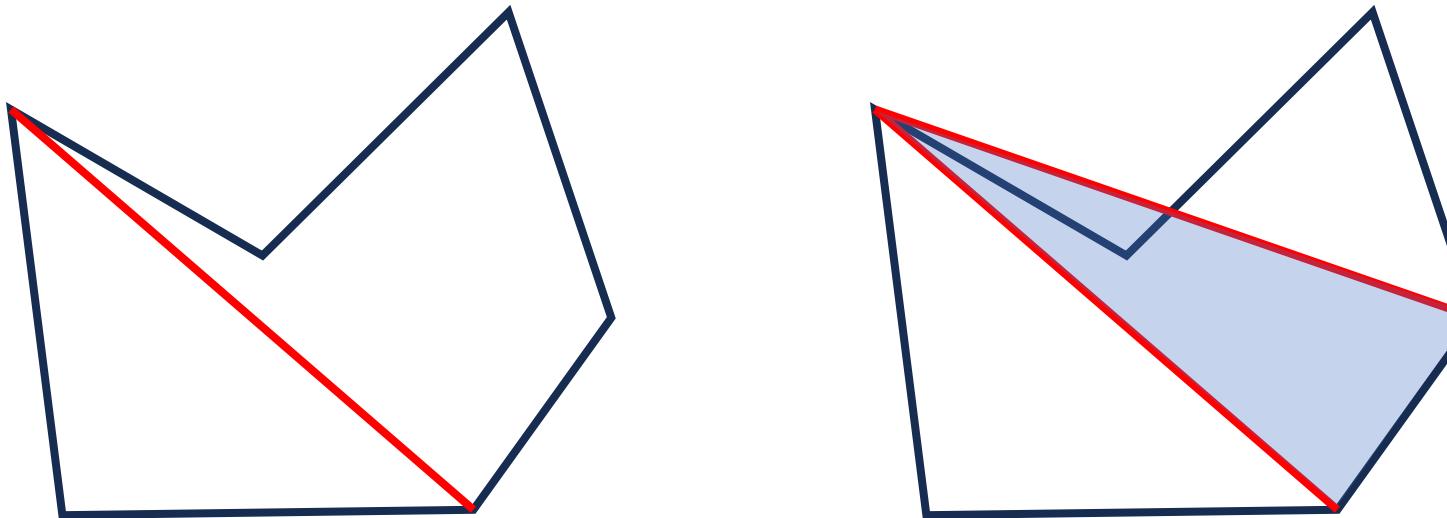
# 다각형의 넓이

- 외적의 절댓값은 변  $a, b$ 의 끼인각을  $\theta$ 로 가지는 **삼각형 넓이의 2배**이다.
- 오목 다각형도 볼록 다각형과 같은 방식으로 나눌 수 있을까?



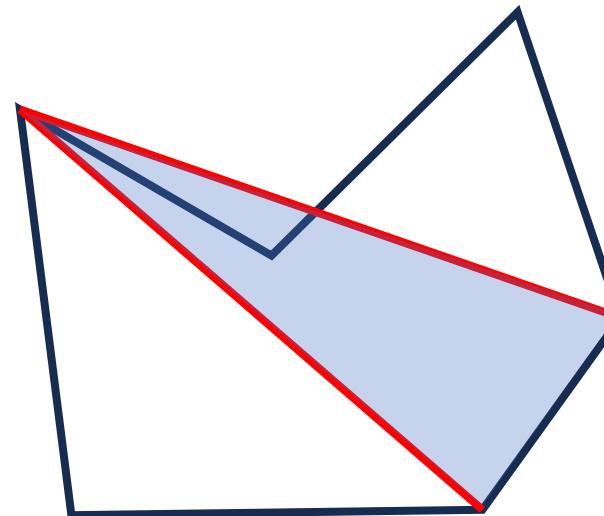
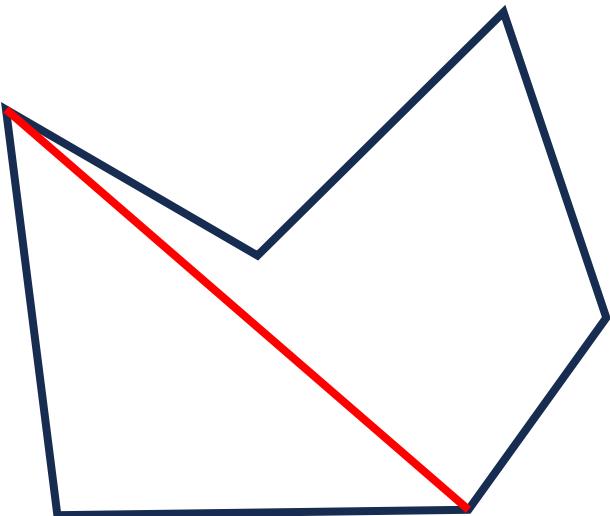
# 다각형의 넓이

- 외적의 절댓값은 변  $a, b$ 의 끼인각을  $\theta$ 로 가지는 **삼각형 넓이의 2배**이다.
- 오목 다각형도 볼록 다각형과 같은 방식으로 나눌 수 있을까?



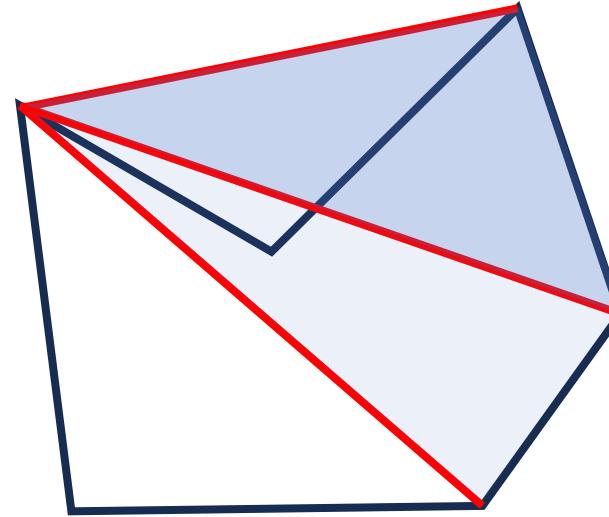
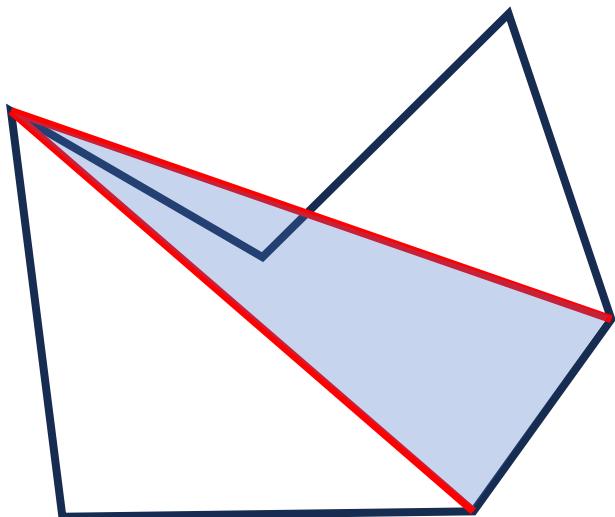
# 다각형의 넓이

- 외적의 절댓값은 변  $a, b$ 의 끼인각을  $\theta$ 로 가지는 **삼각형 넓이의 2배**이다.
- 다각형 외부를 포함하도록 나누어지는데, 괜찮을까?



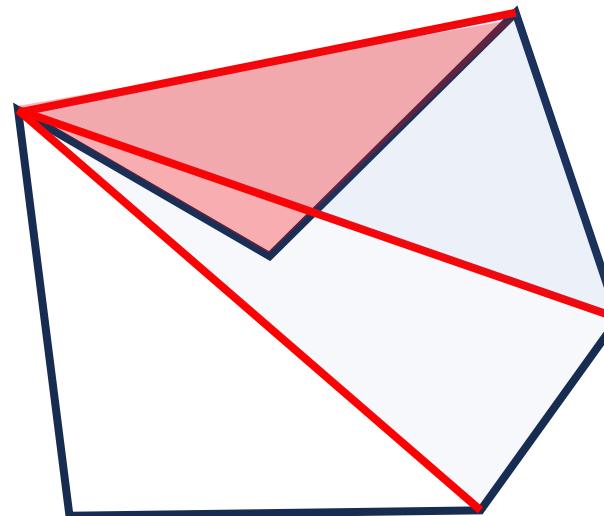
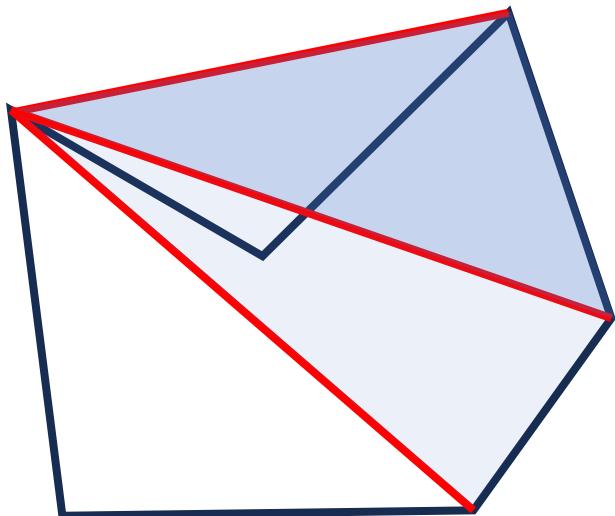
# 다각형의 넓이

- 외적의 절댓값은 변  $a, b$ 의 끼인각을  $\theta$ 로 가지는 **삼각형 넓이의 2배**이다.
- 다각형 외부를 포함하도록 나누어지는데, 괜찮을까?



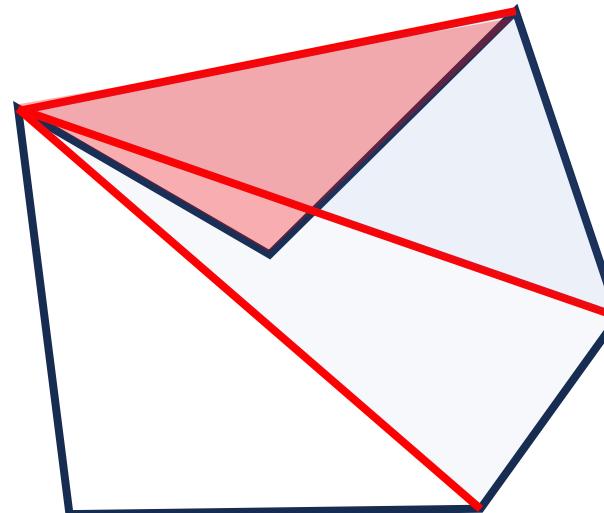
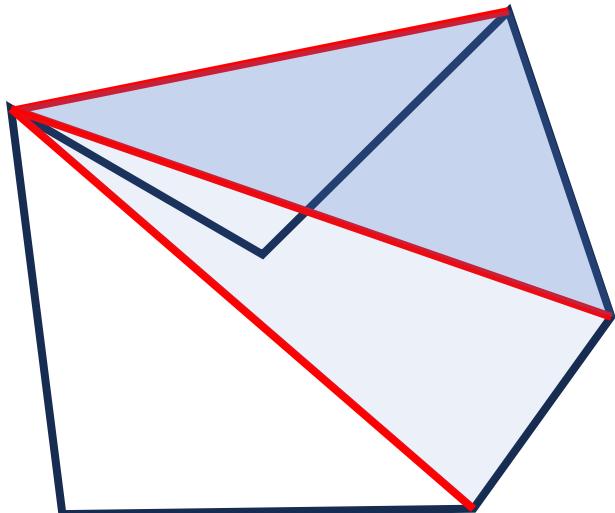
# 다각형의 넓이

- 외적의 절댓값은 변  $a, b$ 의 끼인각을  $\theta$ 로 가지는 **삼각형 넓이의 2배**이다.
- 다각형 외부를 포함하도록 나누어지는데, 괜찮을까?



# 다각형의 넓이

- 외적의 절댓값은 변  $a, b$ 의 끼인각을  $\theta$ 로 가지는 **삼각형 넓이의 2배**이다.
- 결국 기존의 방향과 반대 방향으로 외적을 하기에 부호가 반대로 되어 외부의 삼각형의 넓이를 빼는 역할을 한다.



# 다각형의 면적 BOJ 2166

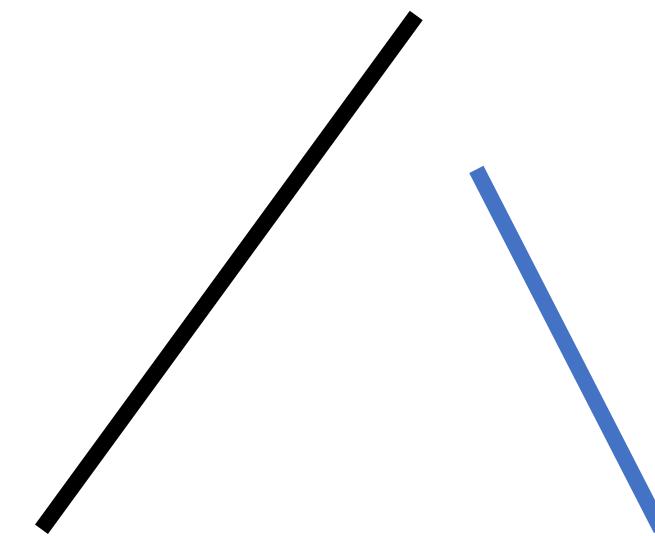
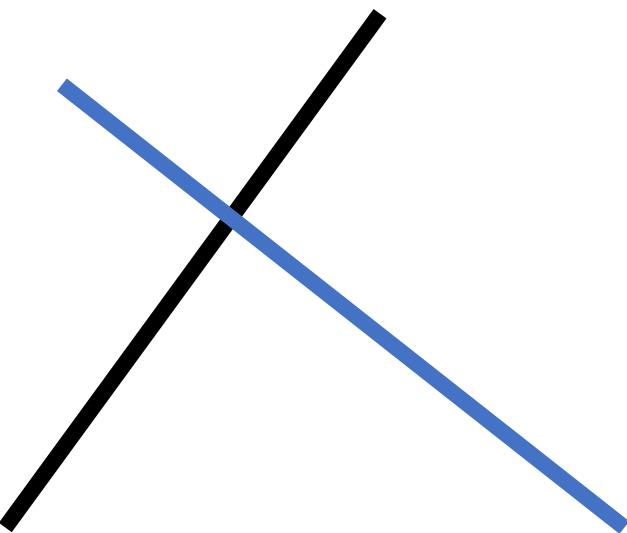
- 2차원 평면상에  $N(3 \leq N \leq 10,000)$ 개의 점으로 이루어진 다각형이 있다.
- 점은 다각형을 이루는 순서대로  $x, y$ 좌표가 주어진다.
- 이 다각형의 면적을 구해보자.

# 다각형의 면적 BOJ 2166

- 2차원 평면상에  $N(3 \leq N \leq 10,000)$ 개의 점으로 이루어진 다각형이 있다.
  - 점은 다각형을 이루는 순서대로  $x, y$ 좌표가 주어진다.
  - 이 다각형의 면적을 구해보자.
- 
- 앞에서와 같이, 한 꼭짓점을 고정하고 한 방향으로 돌면서 외적 값을 계속해서 더해준다.
  - 외적 값은 삼각형의 넓이의 2배라는 것을 주의하자.

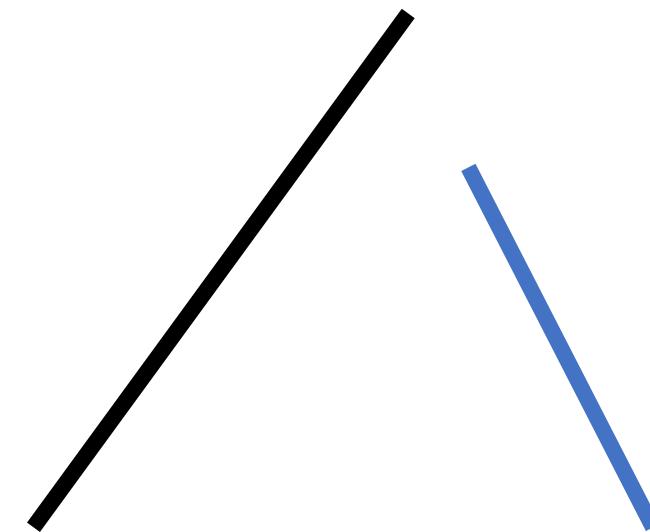
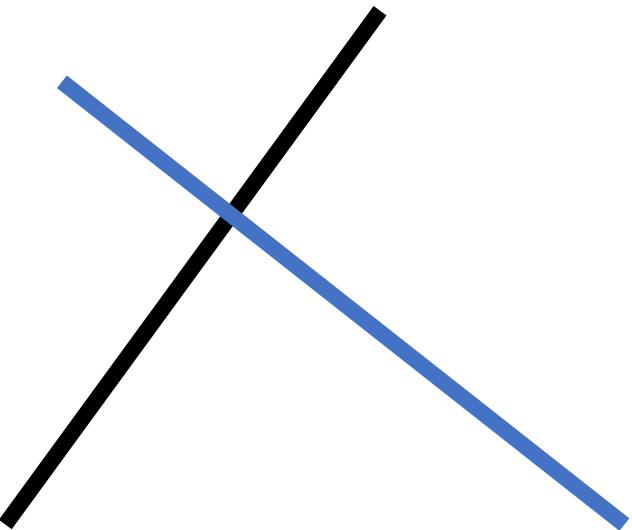
# 선분 교차

- 두 선분의 교차 여부를 판단해보자.



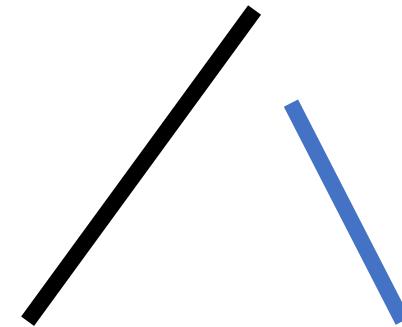
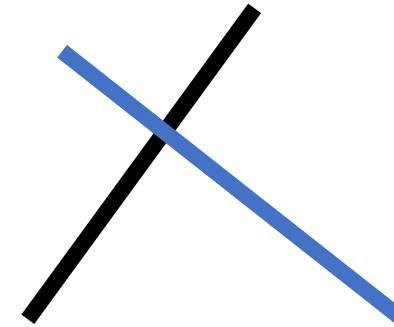
# 선분 교차

- 두 선분의 교차 여부를 판단해보자.
- 교차한 상태일 때, 점의 위치는 어떠한가?



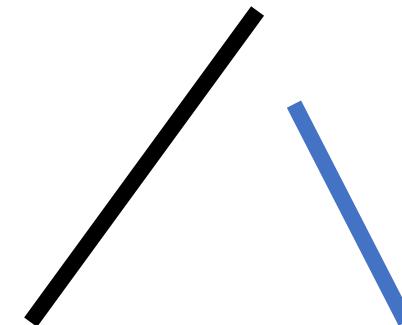
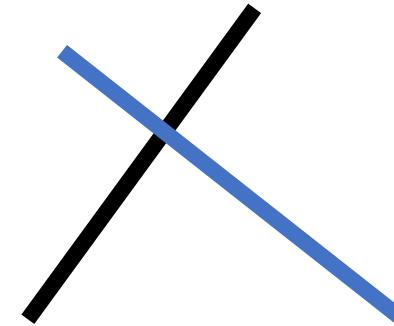
# 선분 교차

- 두 선분의 교차 여부를 판단해보자.
- 교차한 상태일 때, 점의 위치는 어떠한가?
- 한 선분이 점을 좌우로 갈라놓은 것처럼 보인다.



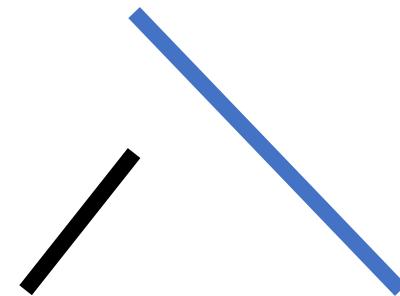
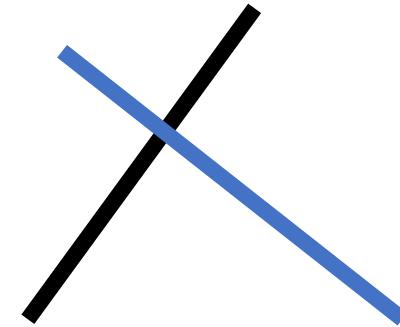
# 선분 교차

- 두 선분의 교차 여부를 판단해보자.
- 교차한 상태일 때, 점의 위치는 어떠한가?
- 한 선분을 기준으로 다른 선분의 점이 양쪽 방향에 있다.
- 반례는 없을까?



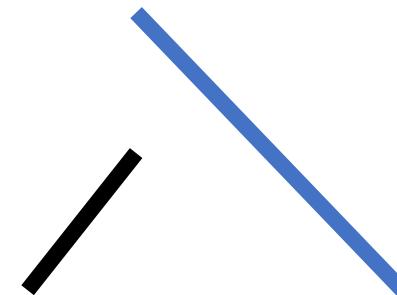
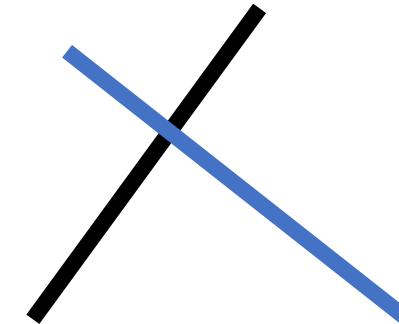
# 선분 교차

- 두 선분의 교차 여부를 판단해보자.
- 교차한 상태일 때, 점의 위치는 어떠한가?
- 한 선분을 기준으로 다른 선분의 점이 양쪽 방향에 있다.
- 반례는 없을까?



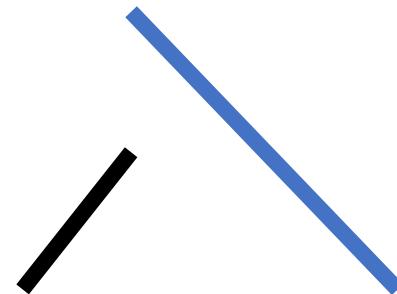
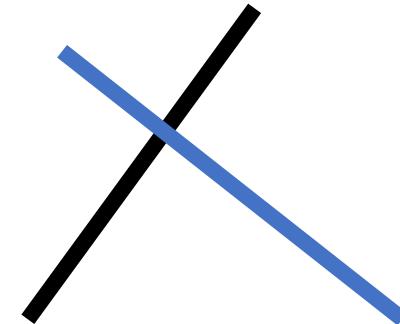
# 선분 교차

- 두 선분의 교차 여부를 판단해보자.
- 교차한 상태일 때, 점의 위치는 어떠한가?
- 해당 반례에서 한 가지 조건을 더 보충할 수 있다.



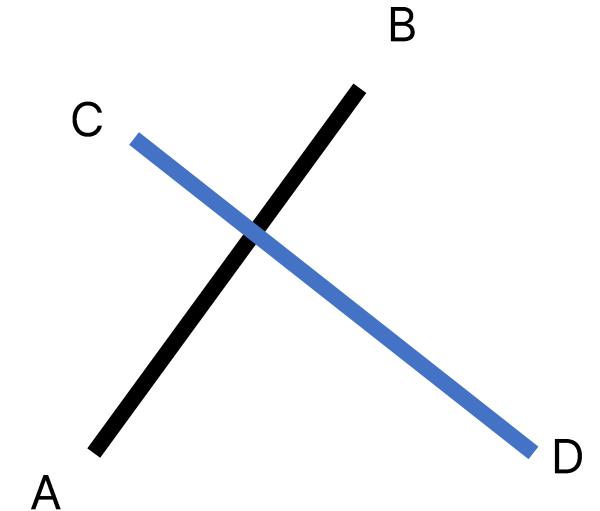
# 선분 교차

- 두 선분의 교차 여부를 판단해보자.
- 교차한 상태일 때, 점의 위치는 어떠한가?
- 해당 반례에서 한 가지 조건을 더 보충할 수 있다.
- **두 선분 모두** 한 선분을 기준으로 다른 선분의 점이 양쪽 방향에 있다.



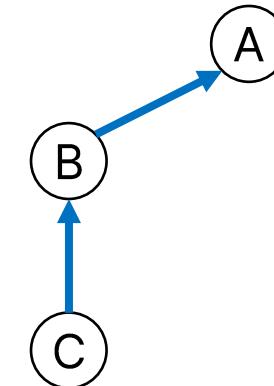
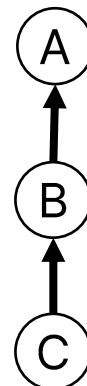
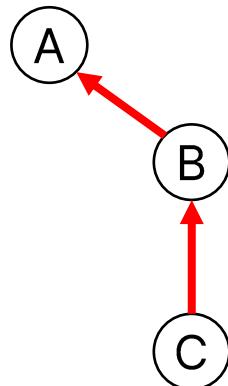
# 선분 교차

- 두 선분의 교차 여부를 판단해보자.
- **두 선분 모두** 한 선분을 기준으로 다른 선분의 점이 양쪽 방향에 있다.
- 그렇다면 이를 어떻게 구할까?



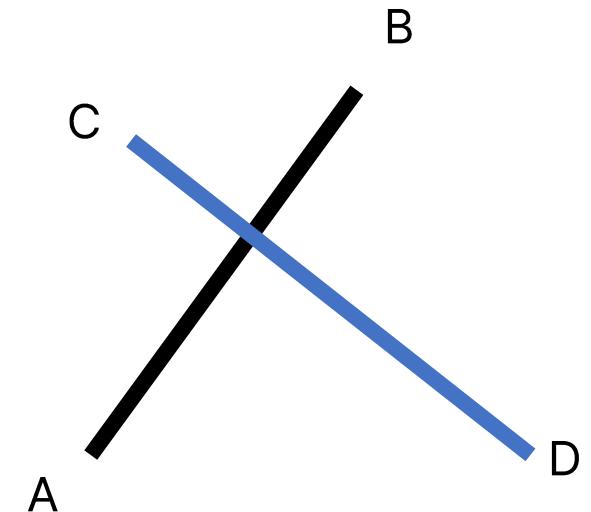
# CCW(Counterclockwise)

- 3개의 점  $A, B, C$ 를 이은 **직선의 방향성**을 구하는 알고리즘
- 시계 반대 방향이면 **양수**, 평행하면 **0**, 시계 방향이면 **음수**를 반환



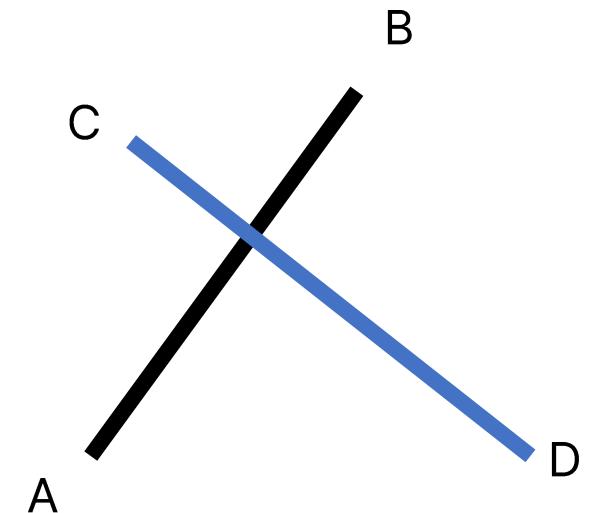
# 선분 교차

- 두 선분의 교차 여부를 판단해보자.
- **두 선분 모두** 한 선분을 기준으로 다른 선분의 점이 양쪽 방향에 있다.
- 방향이 반대인 것은 CCW값의 부호가 서로 반대라는 것이다.



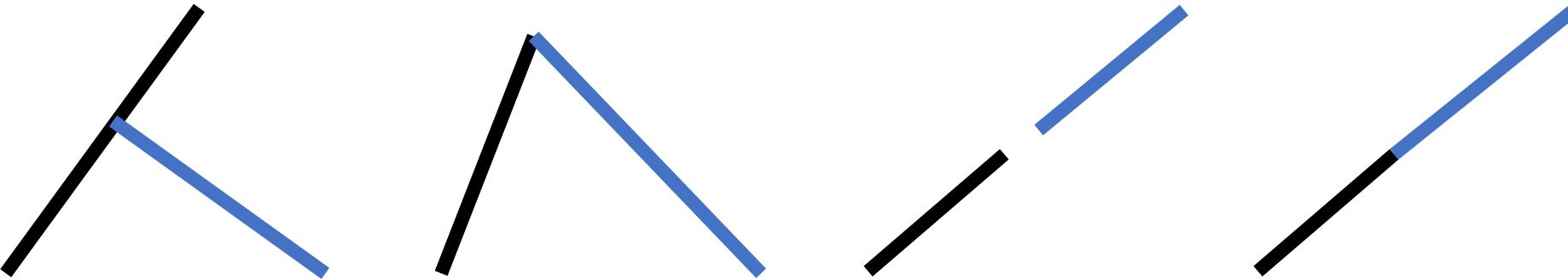
# 선분 교차

- 두 선분의 교차 여부를 판단해보자.
- **두 선분 모두** 한 선분을 기준으로 다른 선분의 점이 양쪽 방향에 있다.
- 방향이 반대인 것은 CCW값의 부호가 서로 반대라는 것이다.
  - $CCW(A, B, C) \times CCW(A, B, D) < 0$
  - $CCW(D, C, A) \times CCW(D, C, B) < 0$
- 위 두 조건이 만족하면 된다.



# 선분 교차

- 두 선분의 교차 여부를 판단해보자.
- 나머지 케이스들은 따로 체크를 해주도록 하자.

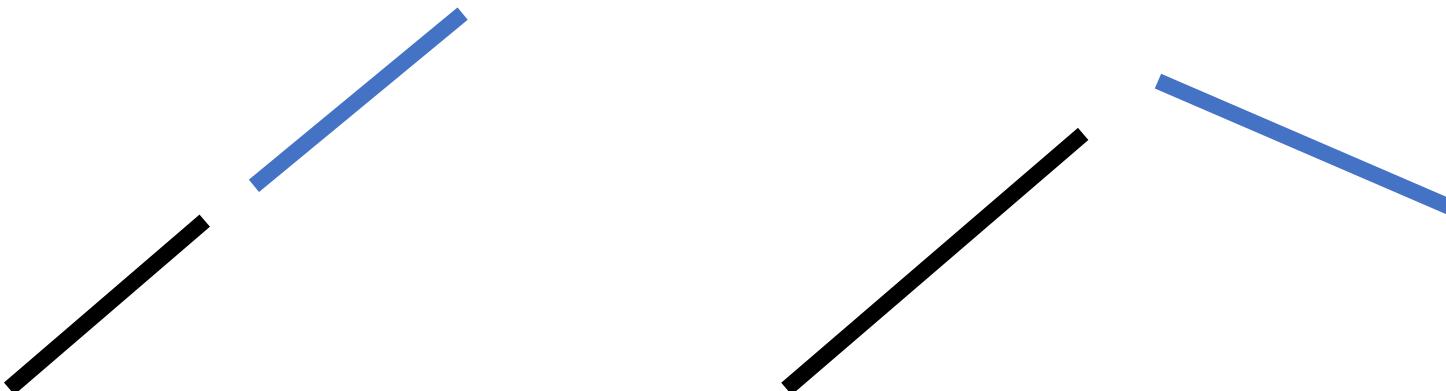


# 선분 교차 1 BOJ 17386

- 2차원 좌표 평면 위의 두 선분  $L_1, L_2$ 가 주어졌을 때, 두 선분이 교차하는지 구해보자.
- 세 점이 일직선 위에 있는 경우는 없다.

# 선분 교차 1 BOJ 17386

- 2차원 좌표 평면 위의 두 선분  $L_1, L_2$ 가 주어졌을 때, 두 선분이 교차하는지 구해보자.
- 세 점이 일직선 위에 있는 경우는 없다.
- 아래의 경우가 없는 것이다.



# 선분 교차 1 BOJ 17386

- 2차원 좌표 평면 위의 두 선분  $L_1, L_2$ 가 주어졌을 때, 두 선분이 교차하는지 구해보자.
- 세 점이 일직선 위에 있는 경우는 없다.
- 결국 아래의 식만 그대로 활용해주어도 된다.
  - $CCW(A, B, C) \times CCW(A, B, D) < 0$
  - $CCW(D, C, A) \times CCW(D, C, B) < 0$

# 선분 교차 2 BOJ 17387

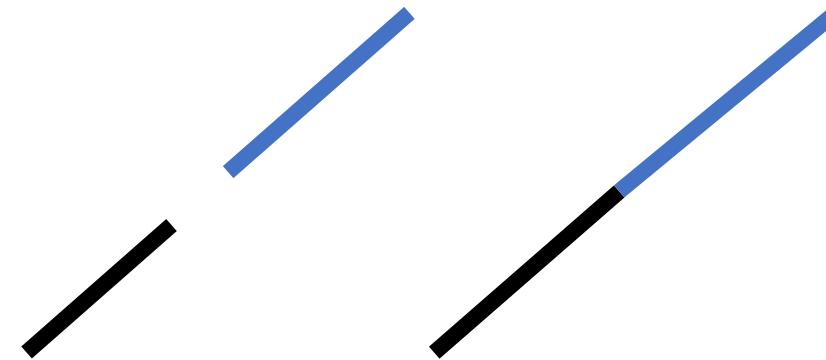
- 2차원 좌표 평면 위의 두 선분  $L_1, L_2$ 가 주어졌을 때, 두 선분이 교차하는지 구해보자.
- 한 선분의 끝 점이 다른 선분이나 끝 점 위에 있는 것도 교차하는 것이다.
- 이번에는 일직선 상 위에 존재할 수 있다.

# 선분 교차 2 BOJ 17387

- 2차원 좌표 평면 위의 두 선분  $L_1, L_2$ 가 주어졌을 때, 두 선분이 교차하는지 구해보자.
- 한 선분의 끝 점이 다른 선분이나 끝 점 위에 있는 것도 교차하는 것이다.
- 이번에는 일직선 상 위에 존재할 수 있다.
- 아래의 식으로 해결할 수 있을 듯 하다.
  - $CCW(A, B, C) \times CCW(A, B, D) \leq 0$
  - $CCW(D, C, A) \times CCW(D, C, B) \leq 0$

# 선분 교차 2 BOJ 17387

- 2차원 좌표 평면 위의 두 선분  $L_1, L_2$ 가 주어졌을 때, 두 선분이 교차하는지 구해보자.
- 한 선분의 끝 점이 다른 선분이나 끝 점 위에 있는 것도 교차하는 것이다.
- 이번에는 일직선 상 위에 존재할 수 있다.
- 아래의 식으로 해결할 수 있을 듯 하다.
  - $CCW(A, B, C) \times CCW(A, B, D) \leq 0$
  - $CCW(D, C, A) \times CCW(D, C, B) \leq 0$



그렇다면 이런 경우는 어떻게 해결할까?

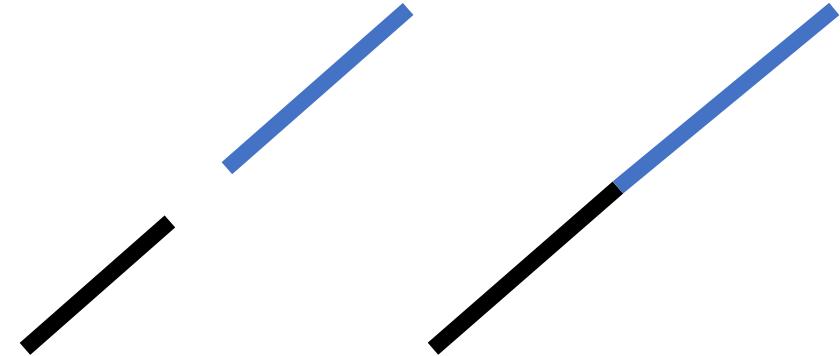
# 선분 교차 2 BOJ 17387

- 아래의 식으로 해결할 수 있을 듯 하다.

$$\cdot CCW(A, B, C) \times CCW(A, B, D) \leq 0$$

$$\cdot CCW(D, C, A) \times CCW(D, C, B) \leq 0$$

- 위의 두 식이 각각 0인 경우에서 추가적인 조건이 필요할 것 같다.



그렇다면 이런 경우는 어떻게 해결할까?

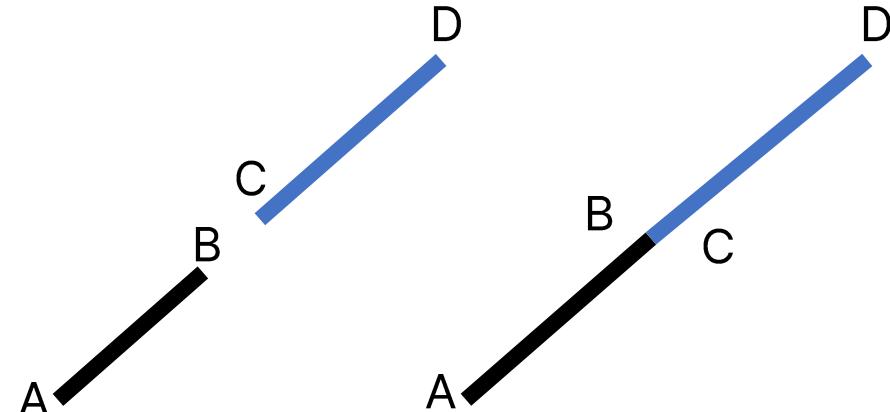
# 선분 교차 2 BOJ 17387

- 아래의 식으로 해결할 수 있을 듯 하다.

- $CCW(A, B, C) \times CCW(A, B, D) \leq 0$
- $CCW(D, C, A) \times CCW(D, C, B) \leq 0$

- 위의 두 식이 각각 0인 경우에서 추가적인 조건이 필요할 것 같다.

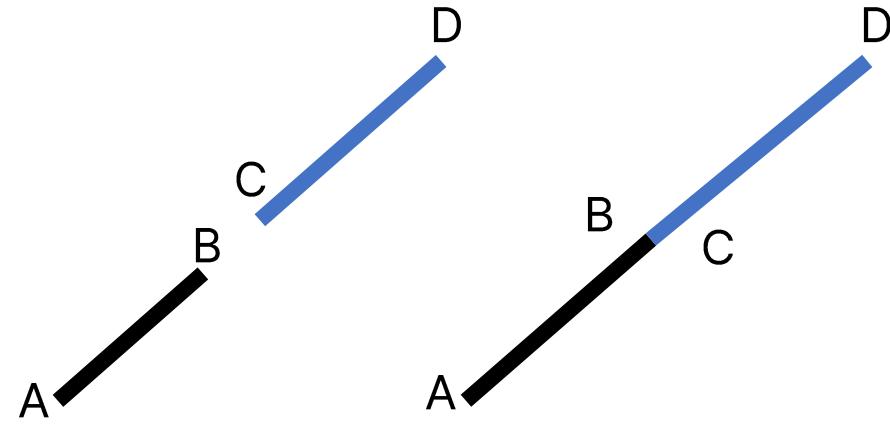
- 각 선분에서  $x$ 좌표가 작은 것이 각각 점  $A, C$ 라고 하자.
- 아래와 같이 위치한다면 일직선 상에서 겹치는 상태이다.
  - 점  $D$ 가 점  $A$ 의 오른쪽(혹은 동일한 위치)
  - 점  $C$ 가 점  $B$ 의 왼쪽(혹은 동일한 위치)



그렇다면 이런 경우는 어떻게 해결할까?

# 선분 교차 2 BOJ 17387

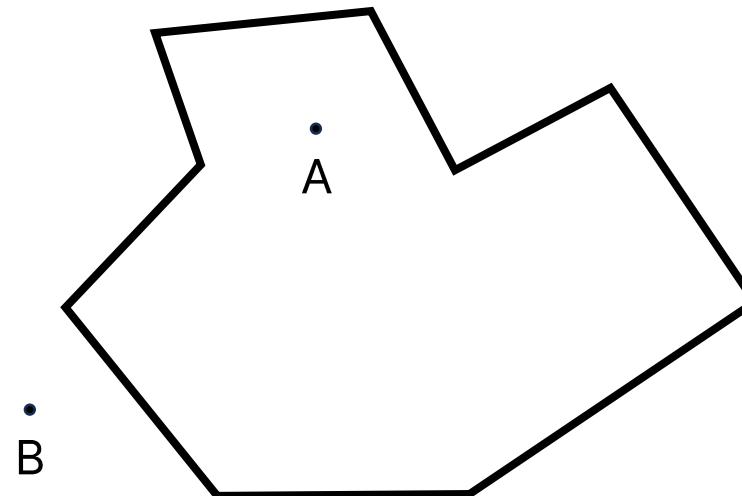
- 아래의 식으로 해결할 수 있을 듯 하다.
  - $CCW(A, B, C) \times CCW(A, B, D) \leq 0$
  - $CCW(D, C, A) \times CCW(D, C, B) \leq 0$
- 위의 두 식이 각각 0인 경우에서 추가적인 조건이 필요할 것 같다.
  - 각 선분에서  $x$ 좌표가 작은 것이 각각 점  $A, C$ 라고 하자.
  - 아래와 같이 위치한다면 일직선 상에서 겹치는 상태이다.
    - 점  $D$ 가 점  $A$ 의 오른쪽(혹은 동일한 위치)
    - 점  $C$ 가 점  $B$ 의 왼쪽(혹은 동일한 위치)
- 이 조건을 추가해준다면 해당 문제를 해결할 수 있다.



그렇다면 이런 경우는 어떻게 해결할까?

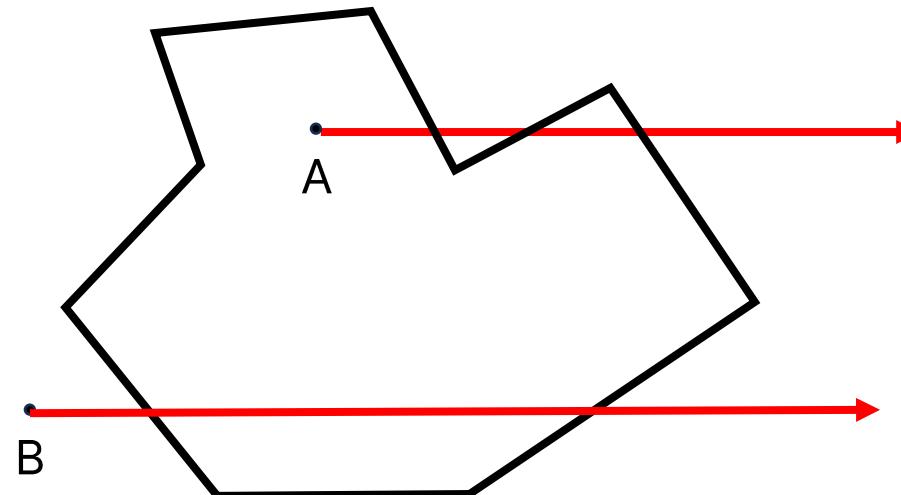
# 다각형 내부의 점 판정

- 점이 다각형 내부에 있는지 외부에 있는지를 구해보자.



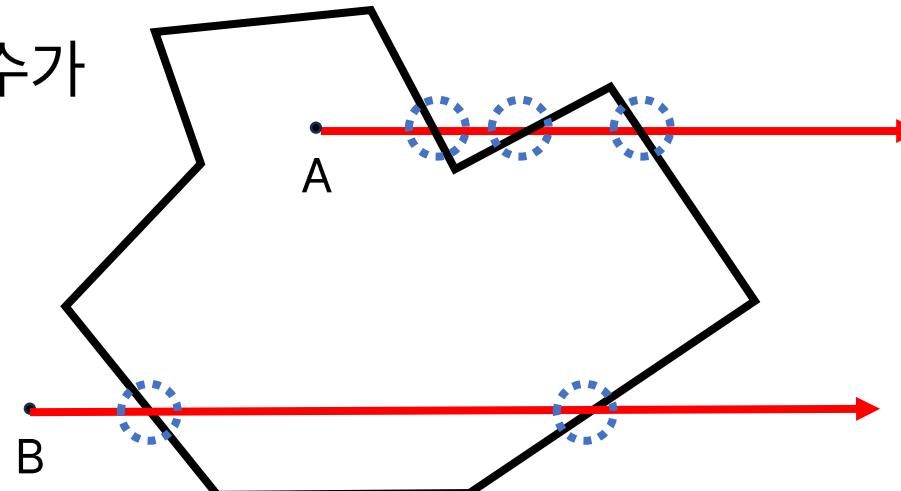
# 다각형 내부의 점 판정

- 점이 다각형 내부에 있는지 외부에 있는지를 구해보자.
- 점에서 반직선을 그어보자.



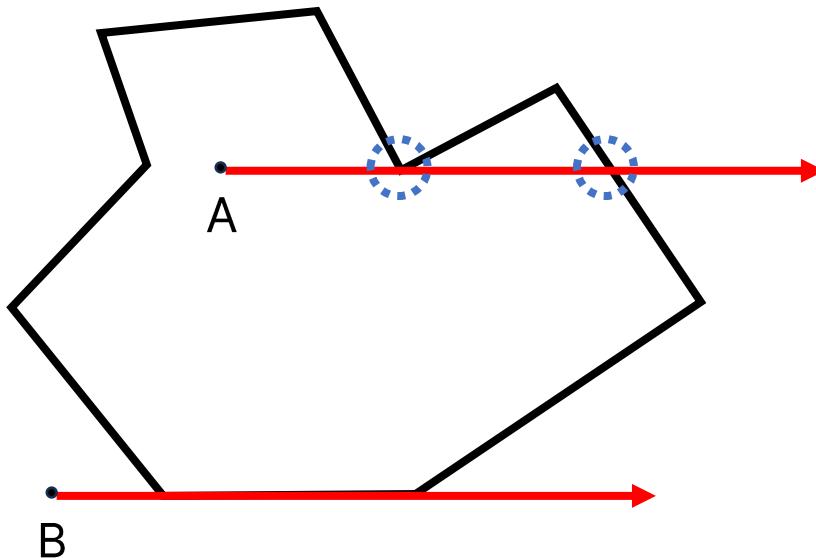
# 다각형 내부의 점 판정

- 점이 다각형 내부에 있는지 외부에 있는지를 구해보자.
- 점에서 반직선을 그어보자.
- 다각형과 만나는 점의 개수가
  - 홀수라면 내부이다.
  - 짝수라면 외부이다.



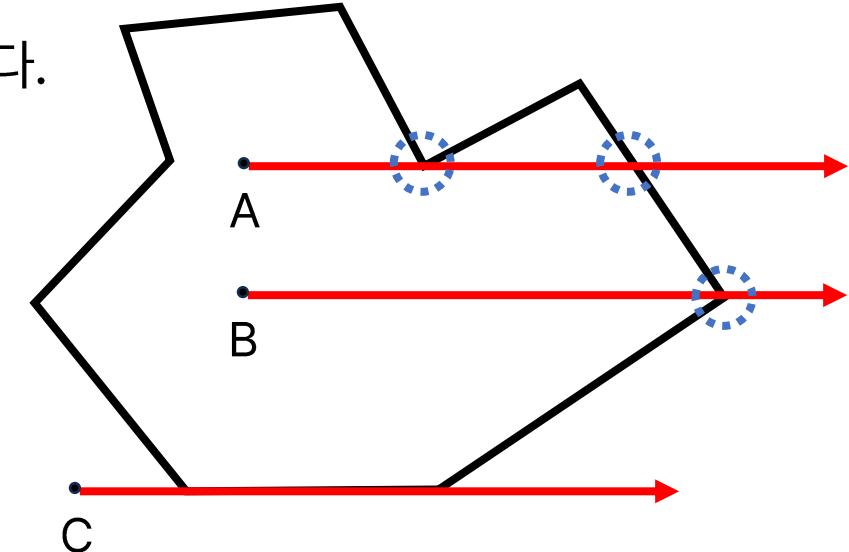
# 다각형 내부의 점 판정

- 점이 다각형 내부에 있는지 외부에 있는지를 구해보자.
- 아래의 경우에는 어떻게 할까?



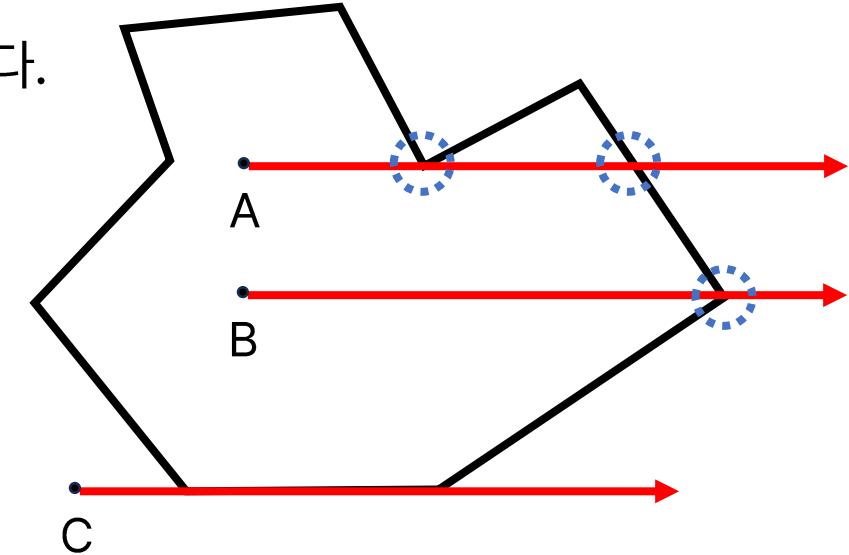
# 다각형 내부의 점 판정

- 점이 다각형 내부에 있는지 외부에 있는지를 구해보자.
- 아래의 경우에는 어떻게 할까?
  - 점  $A, B$ 에서처럼 꼭짓점에 있는 경우 반직선 기준 위 / 아래 선분 중 하나만 체크한다.
  - 점  $C$ 처럼 반직선이 선분을 포함하는 경우는 체크하지 않는다.



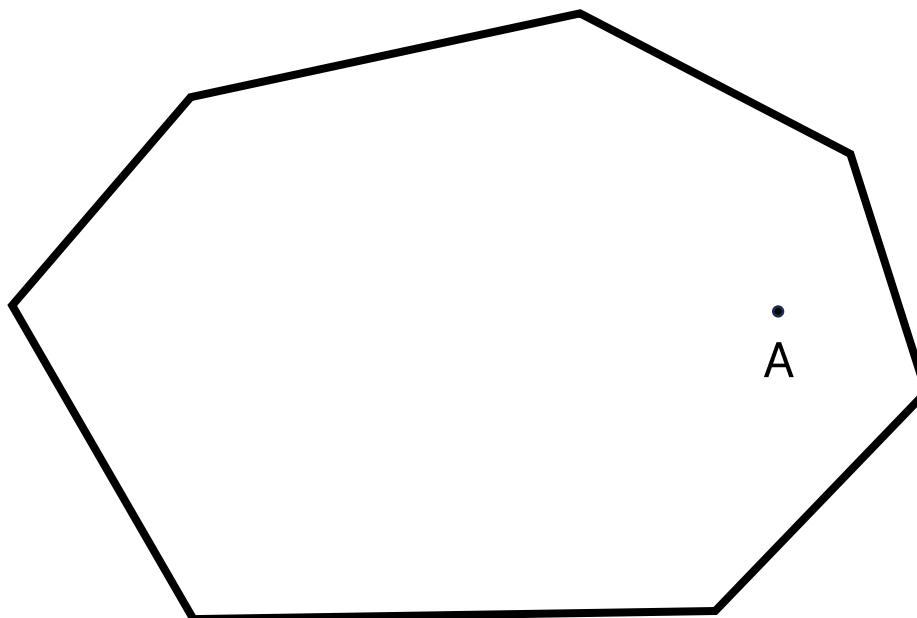
# 다각형 내부의 점 판정

- 점이 다각형 내부에 있는지 외부에 있는지를 구해보자.
- 아래의 경우에는 어떻게 할까?
  - 점  $A, B$ 에서처럼 꼭짓점에 있는 경우 반직선 기준 위 / 아래 선분 중 하나만 체크한다.
  - 점  $C$ 처럼 반직선이 선분을 포함하는 경우는 체크하지 않는다.
- 시간 복잡도는  $O(N)$ 이다.



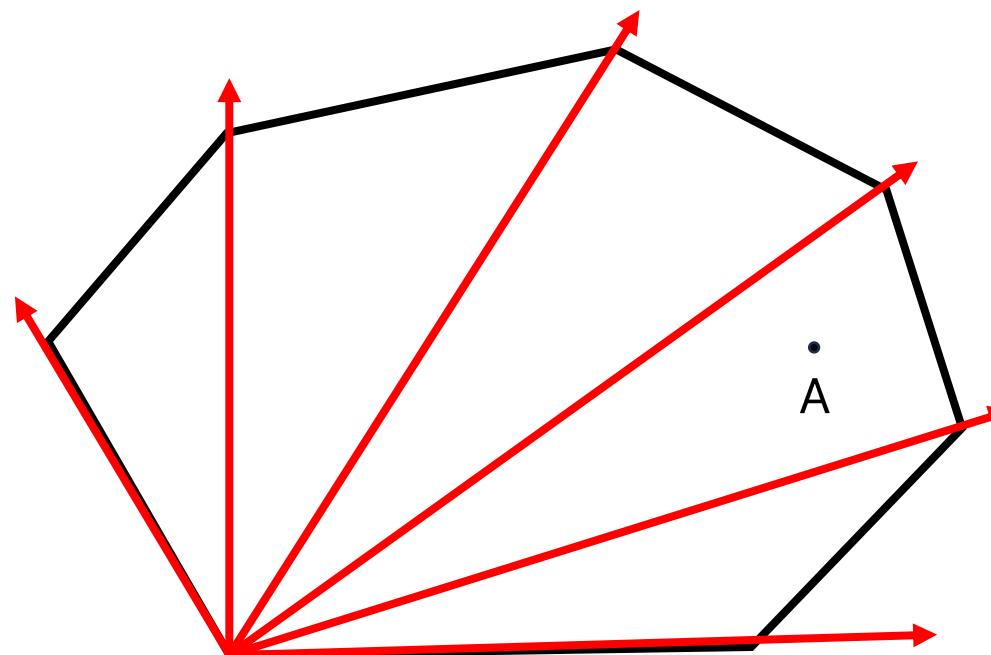
# 볼록 다각형 내부의 점 판정

- 점이 **볼록** 다각형 내부에 있는지 외부에 있는지를 구해보자.



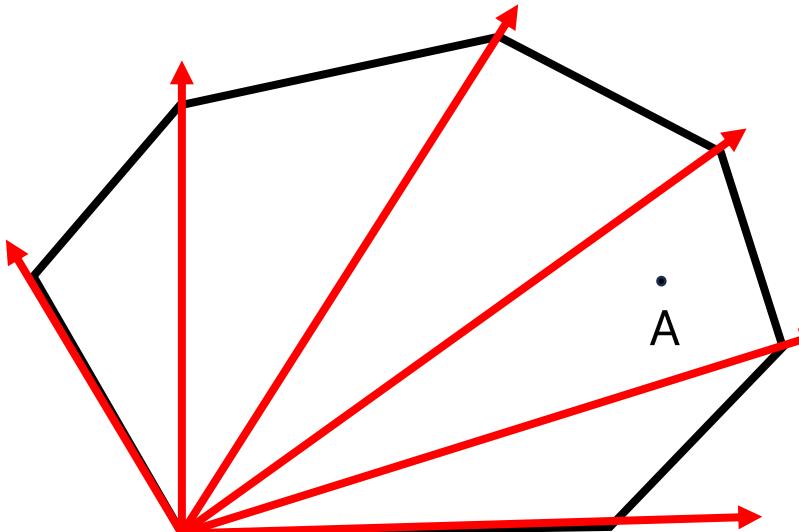
# 볼록 다각형 내부의 점 판정

- 점이 **볼록 다각형** 내부에 있는지 외부에 있는지를 구해보자.
- 볼록 다각형의 한 점에서 시작해, 다른 점들로 이어지는 반직선을 그어보자.



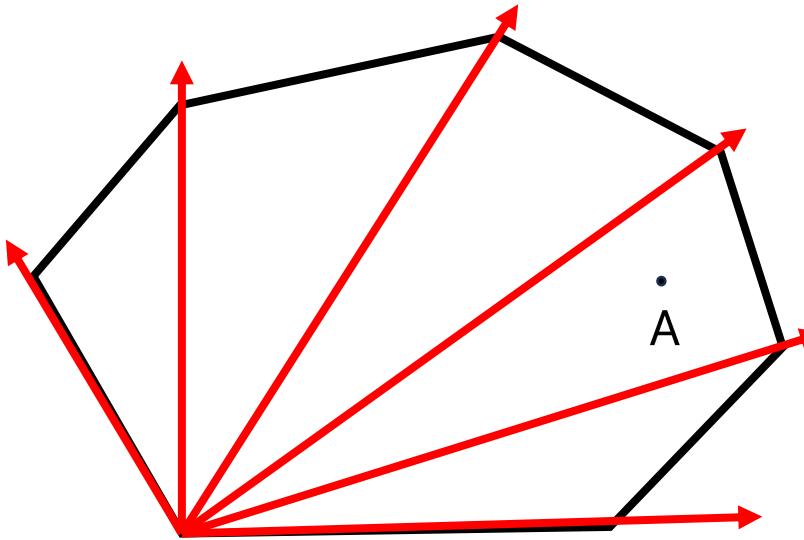
# 볼록 다각형 내부의 점 판정

- 점이 **볼록 다각형** 내부에 있는지 외부에 있는지를 구해보자.
- 볼록 다각형의 한 점에서 시작해, 다른 점들로 이어지는 반직선을 그어보자.
- $n$ 각형에서 점이 다각형 내부에 위치하려면 양 끝 반직선 사이에 점이 존재해야 한다.



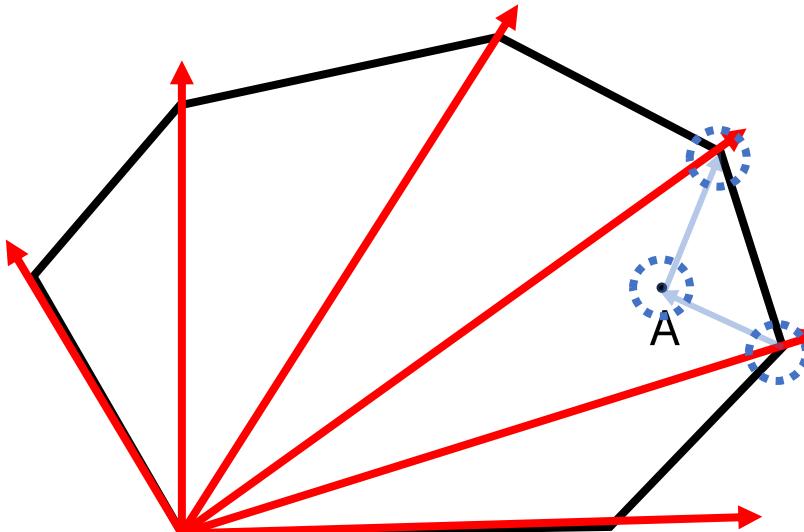
# 볼록 다각형 내부의 점 판정

- 점이 **볼록 다각형** 내부에 있는지 외부에 있는지를 구해보자.
- 볼록 다각형의 한 점에서 시작해, 다른 점들로 이어지는 반직선을 그어보자.
- 그 이후에는 두 반직선 사이의  $n - 2$ 개의 구간 중 어디에 위치하는지를 파악한다.



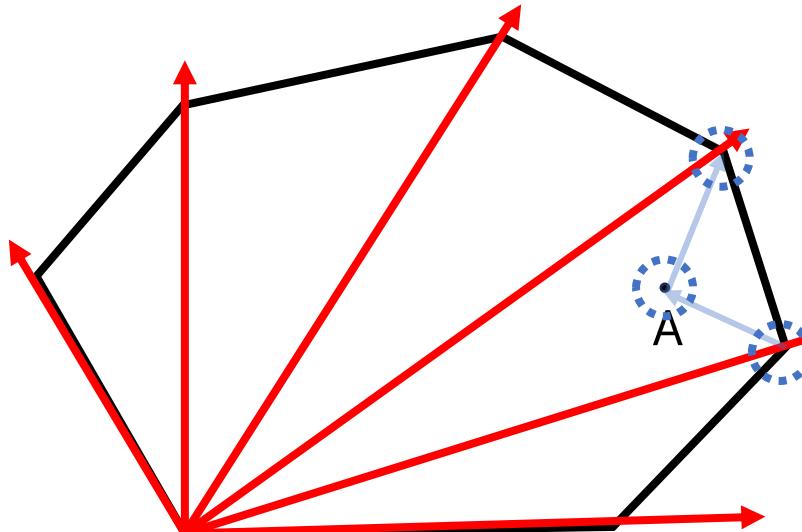
# 볼록 다각형 내부의 점 판정

- 점이 **볼록 다각형** 내부에 있는지 외부에 있는지를 구해보자.
- 볼록 다각형의 한 점에서 시작해, 다른 점들로 이어지는 반직선을 그어보자.
- 어떤 구간에 있는지 파악을 했다면 세 점의 방향성으로 판단을 하면 된다.



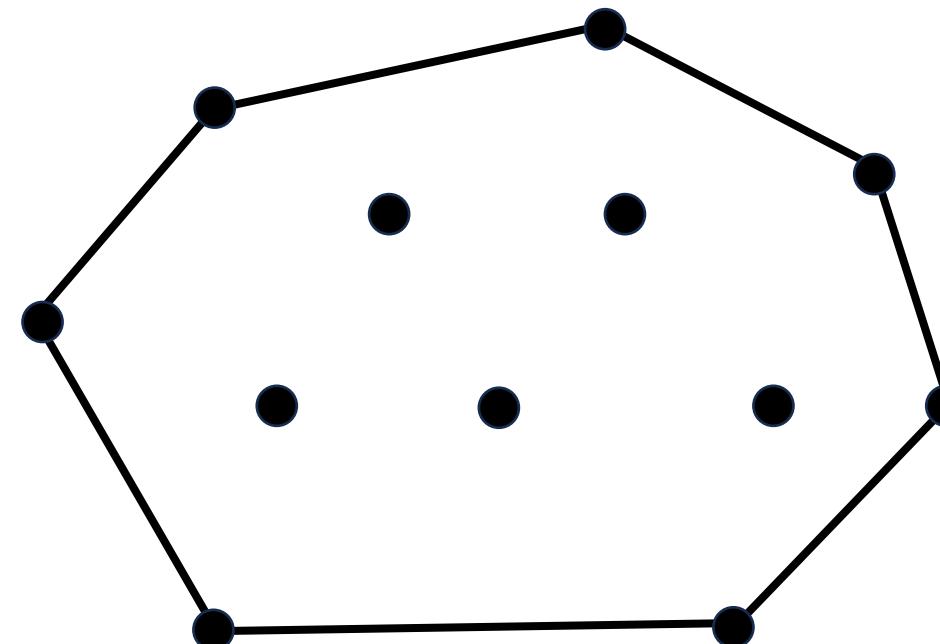
# 볼록 다각형 내부의 점 판정

- 점이 **볼록 다각형** 내부에 있는지 외부에 있는지를 구해보자.
- 볼록 다각형의 한 점에서 시작해, 다른 점들로 이어지는 반직선을 그어보자.
- 시간 복잡도는 볼록 다각형을 구했다는 가정하에  $O(\log N)$ 이다.



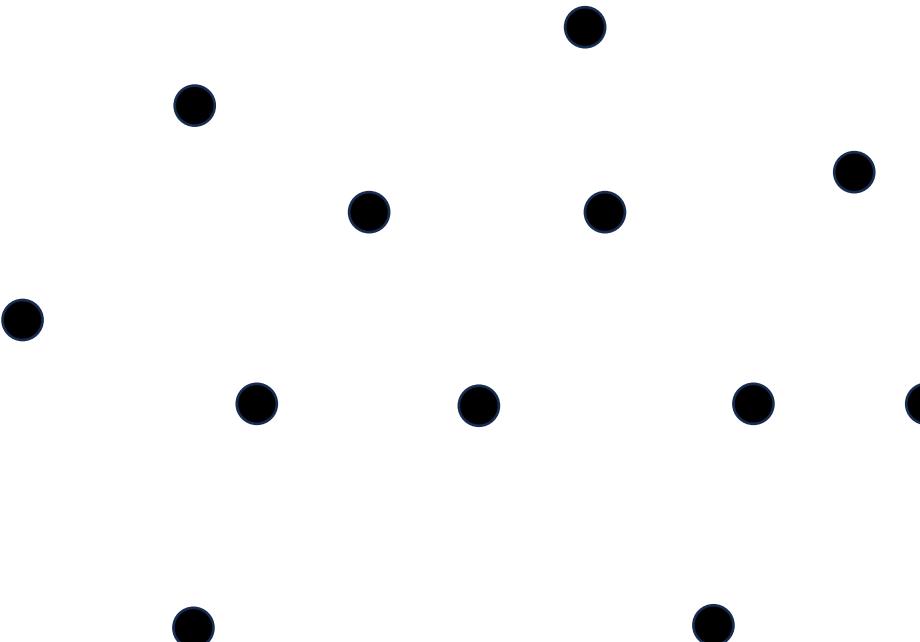
# Convex Hull

- 2차원 평면상에 있는 점 집합에서 볼록 다각형을 이루는 점을 찾는 알고리즘이다.



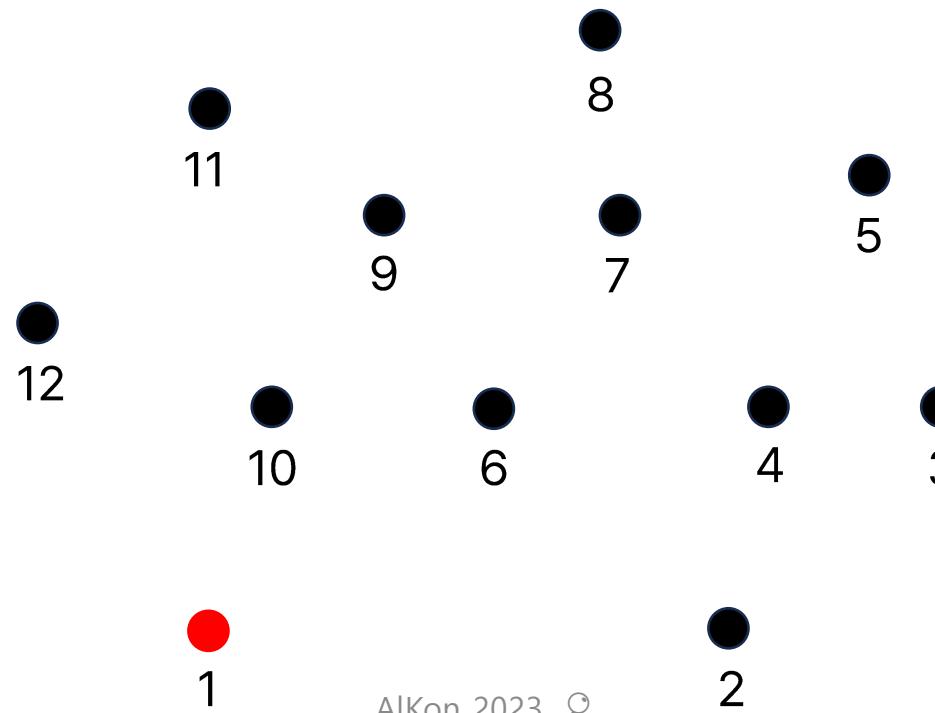
# Graham's Scan

- Convex hull 알고리즘 중 가장 대표적인 방식이다.



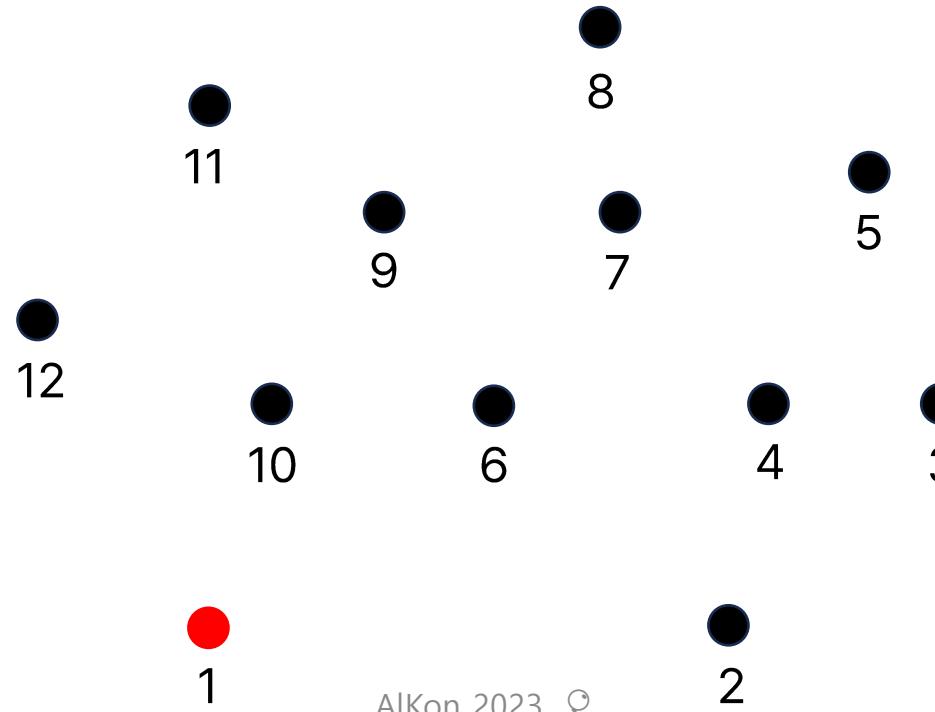
# Graham's Scan

- Convex hull 알고리즘 중 가장 대표적인 방식이다.
- 좌하단에 있는 점을 기준으로 다른 점들을 각도에 따라 정렬하는 전처리가 필요하다.



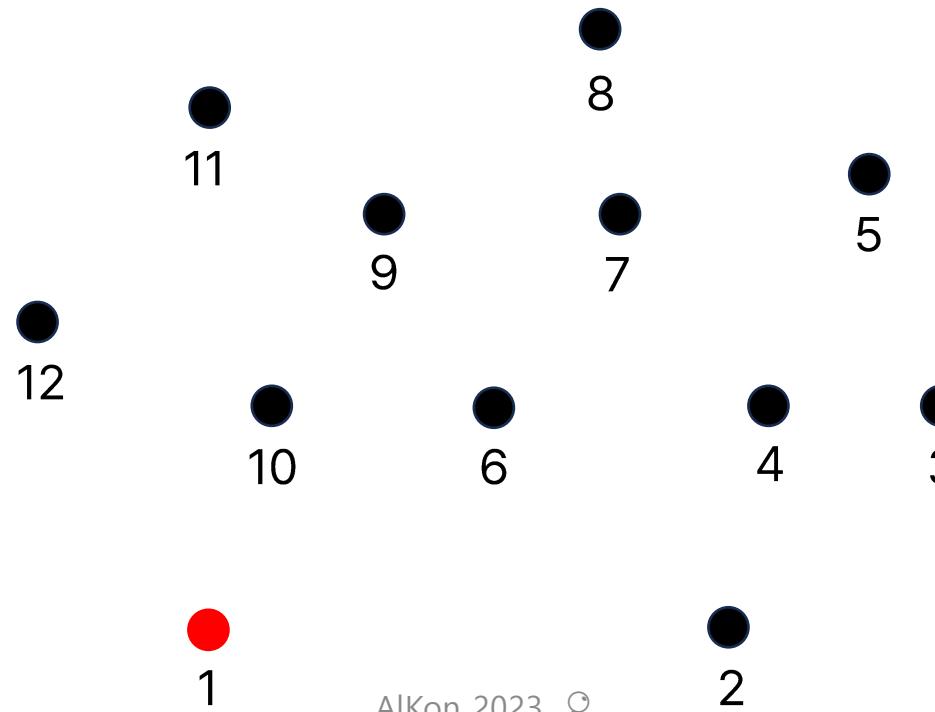
# Graham's Scan

- Convex hull 알고리즘 중 가장 대표적인 방식이다.
- 기준점은  $y$ 좌표,  $x$ 좌표가 작은 순으로 정렬하면 된다.



# Graham's Scan

- Convex hull 알고리즘 중 가장 대표적인 방식이다.
- 각도 정렬은 기준점과 비교하려는 두 점의 진행 방향이 반시계인지 판단하면 된다.



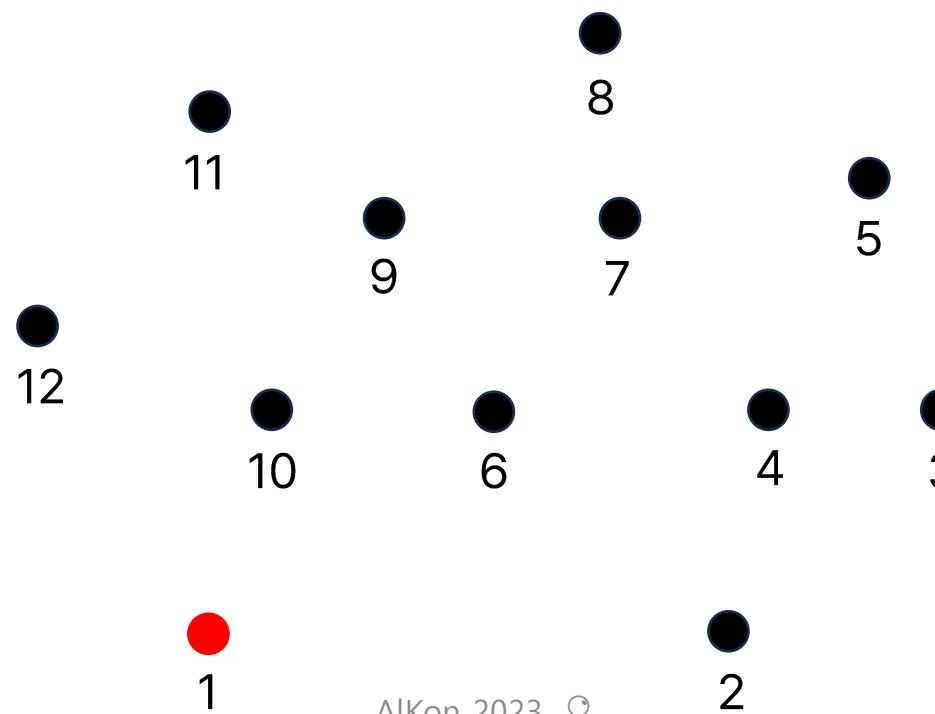
# Graham's Scan

```
bool cmp_ccw(const pair<int, int>& a, const pair<int, int>& b) {
    int c = ccw(v[0], a, b);
    if (c == 0) return dist(v[0], a) < dist(v[0], b);
    return c > 0;
}

sort(v.begin() + 1, v.end(), cmp_ccw);
```

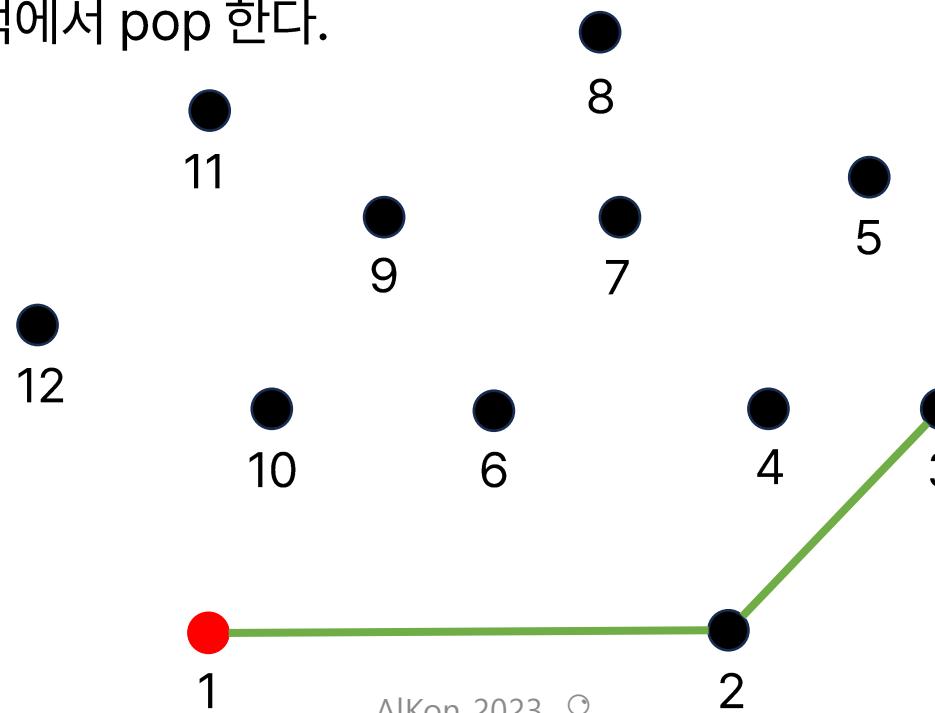
# Graham's Scan

- 스택에 첫번째 정점과 두번째 정점을 넣고 시작한다.



# Graham's Scan

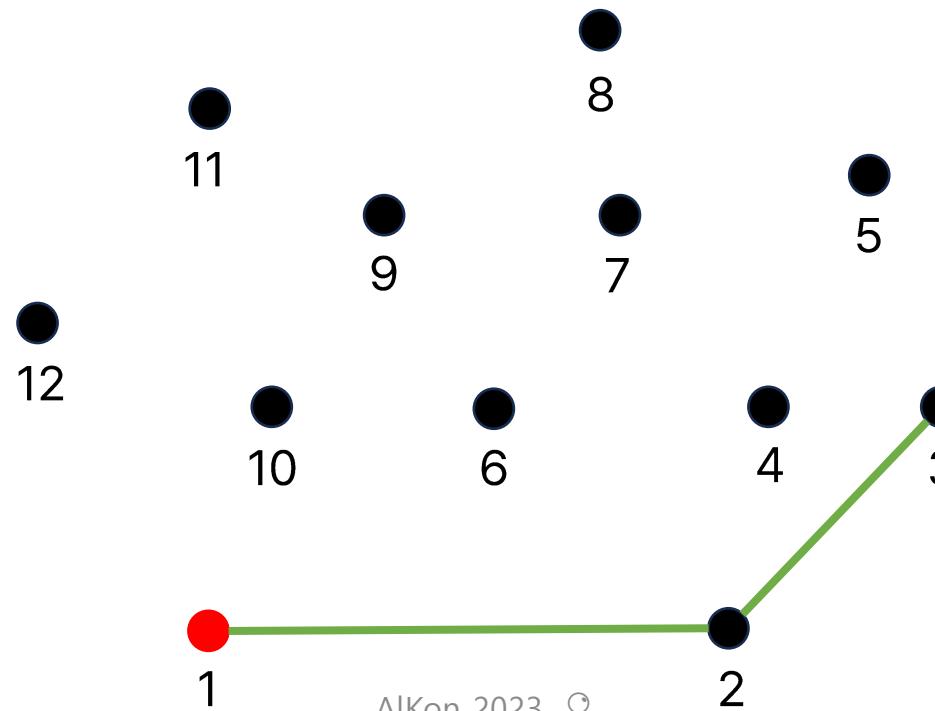
- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 반대 방향이라면 Convex Hull(볼록 껌질)이 될 수 있다.
  - 스택 맨 위의 점만 스택에서 pop 한다.



# Graham's Scan

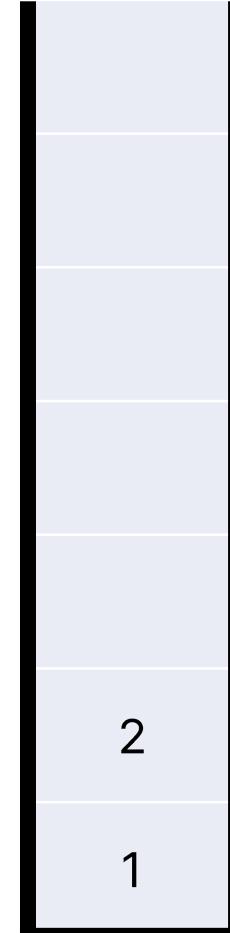
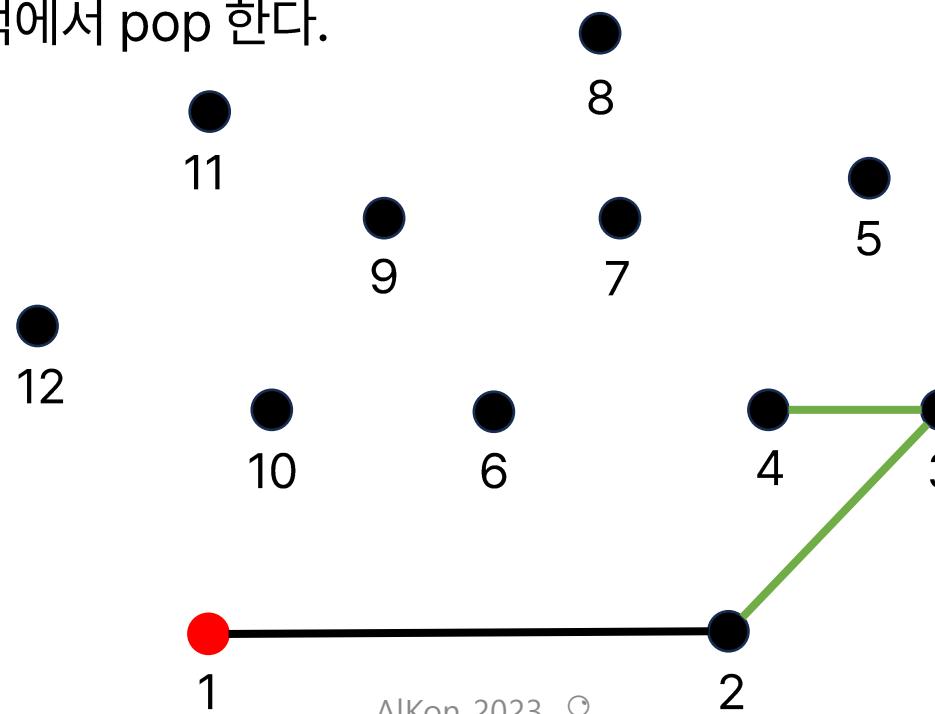
$$CCW(1, 2, 3) > 0$$

- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 반대 방향이라면 pop 했던 점을 다시 스택에 넣고 비교한 다음 점도 넣는다.



# Graham's Scan

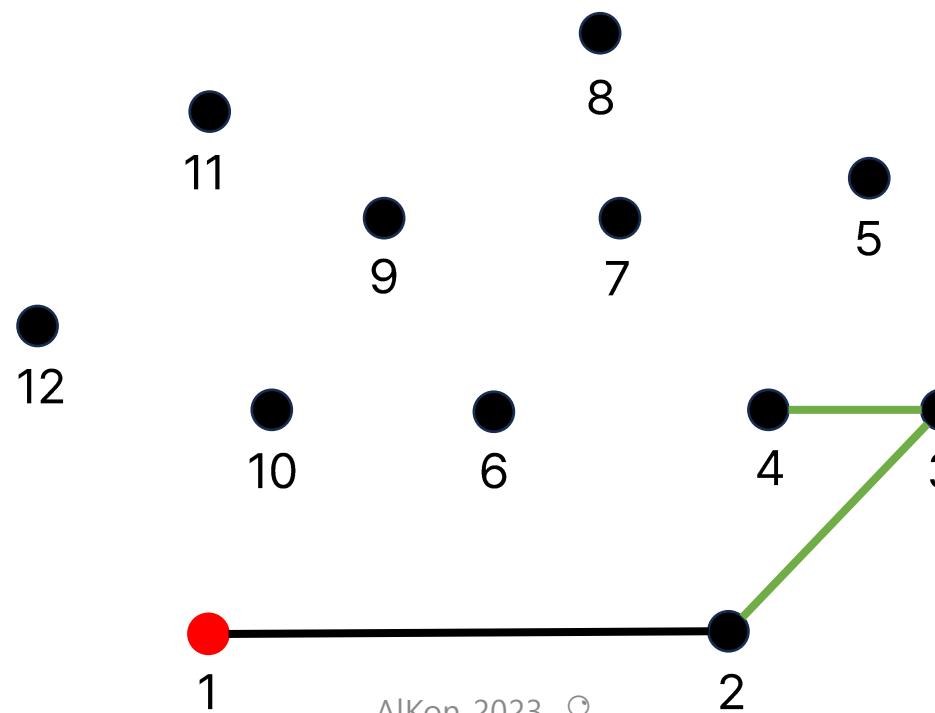
- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 반대 방향이라면 Convex Hull(볼록 껌질)이 될 수 있다.
  - 스택 맨 위의 점만 스택에서 pop 한다.



# Graham's Scan

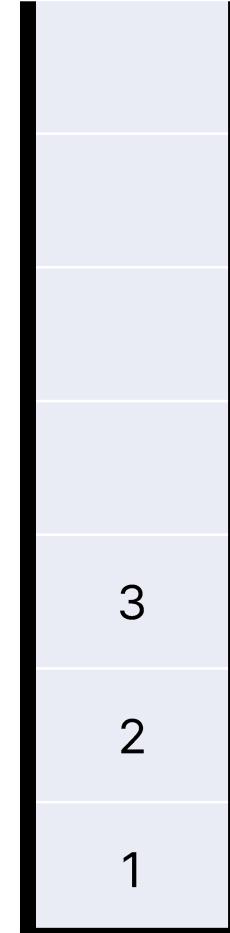
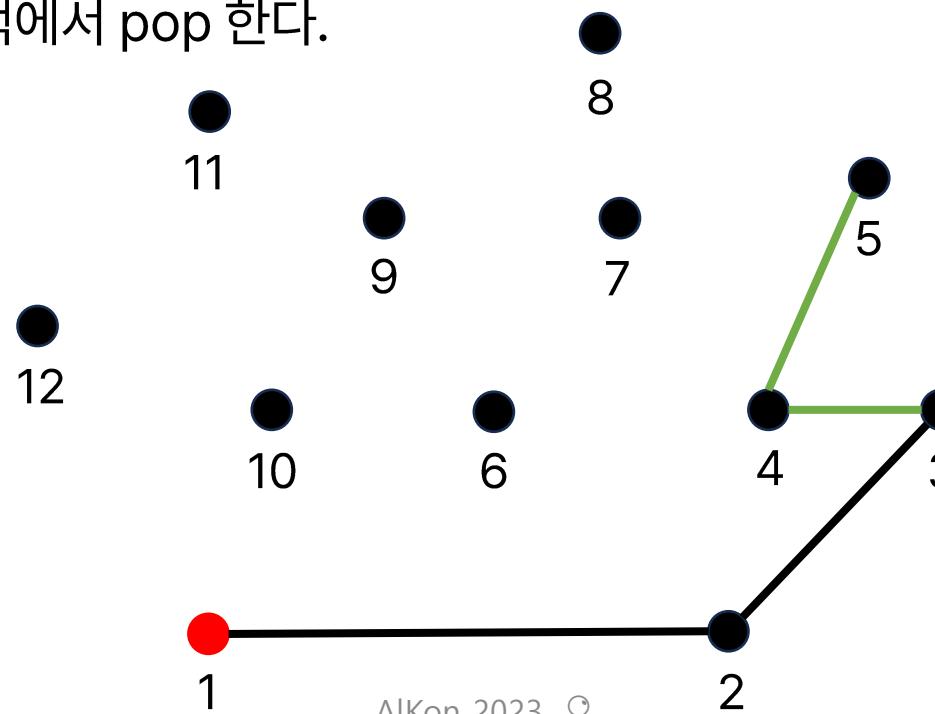
$$CCW(2, 3, 4) > 0$$

- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 반대 방향이라면 pop 했던 점을 다시 스택에 넣고 비교한 다음 점도 넣는다.



# Graham's Scan

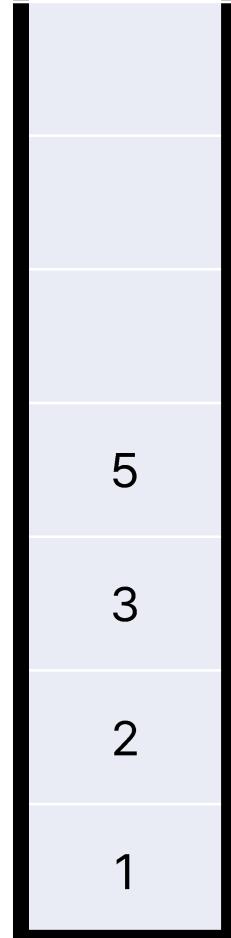
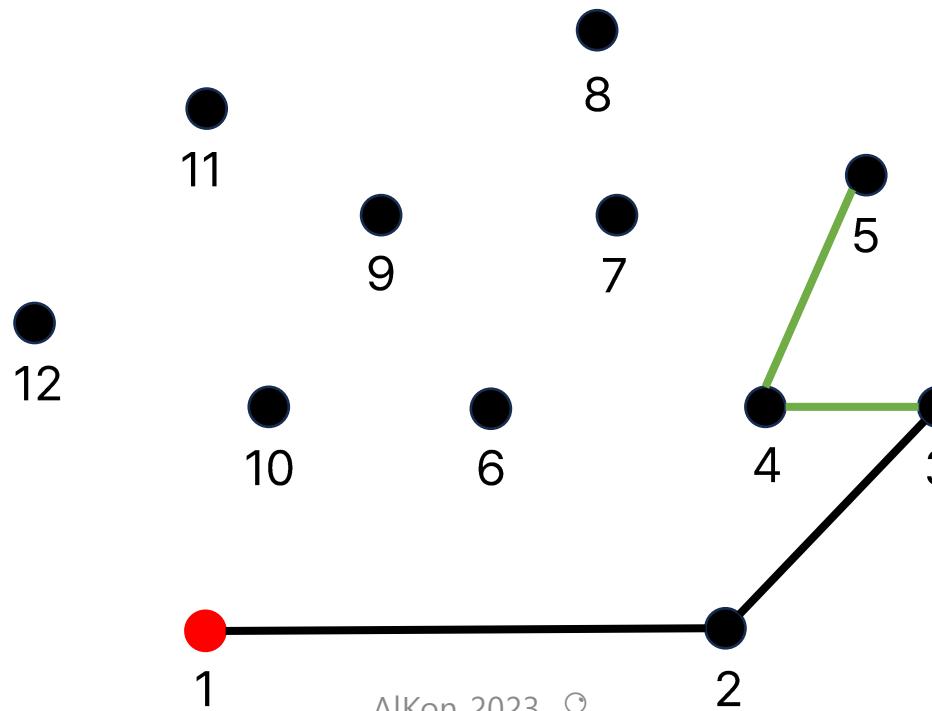
- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 방향이라면 Convex Hull(볼록 껍질)이 될 수 없다.
  - 스택 맨 위의 점만 스택에서 pop 한다.



# Graham's Scan

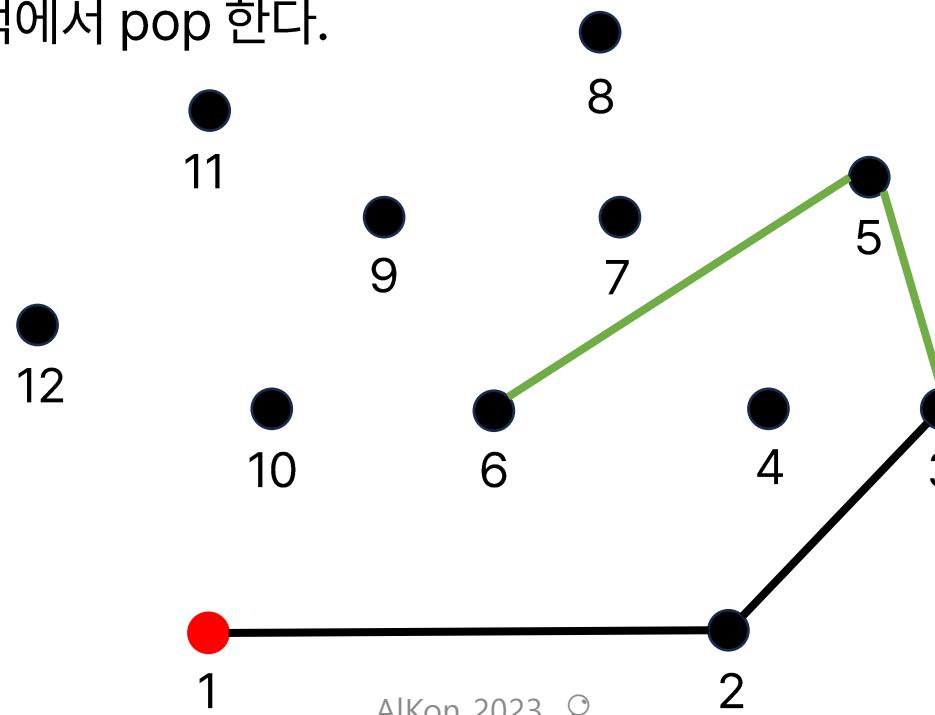
$$CCW(3, 4, 5) < 0$$

- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 방향이라면 pop 했던 점을 스택에 넣지 않고 비교한 다음 점만 넣는다.



# Graham's Scan

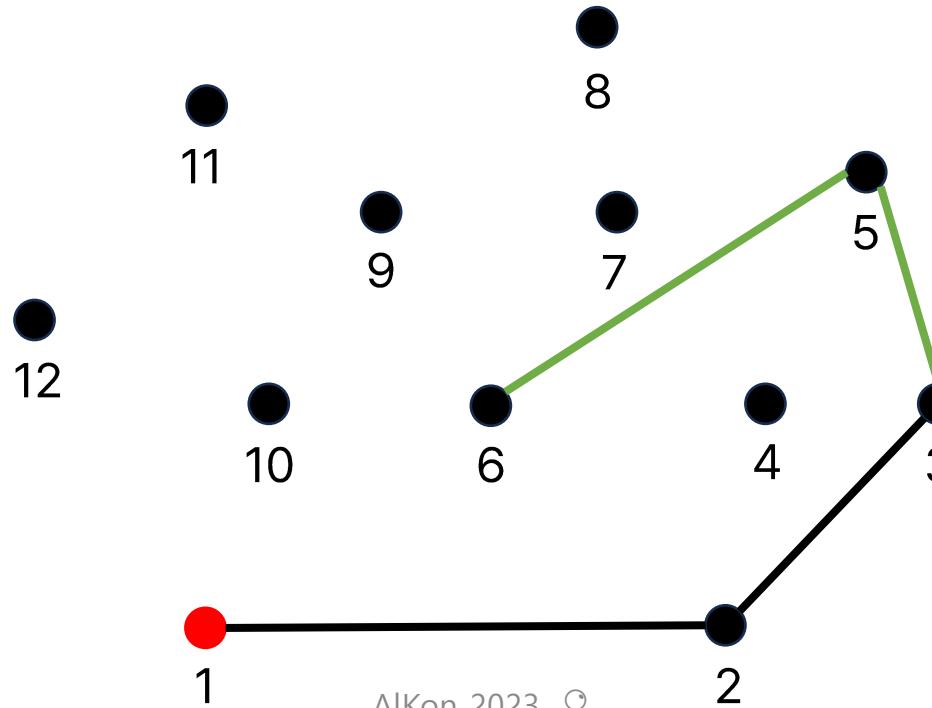
- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 반대 방향이라면 Convex Hull(볼록 껌질)이 될 수 있다.
  - 스택 맨 위의 점만 스택에서 pop 한다.



# Graham's Scan

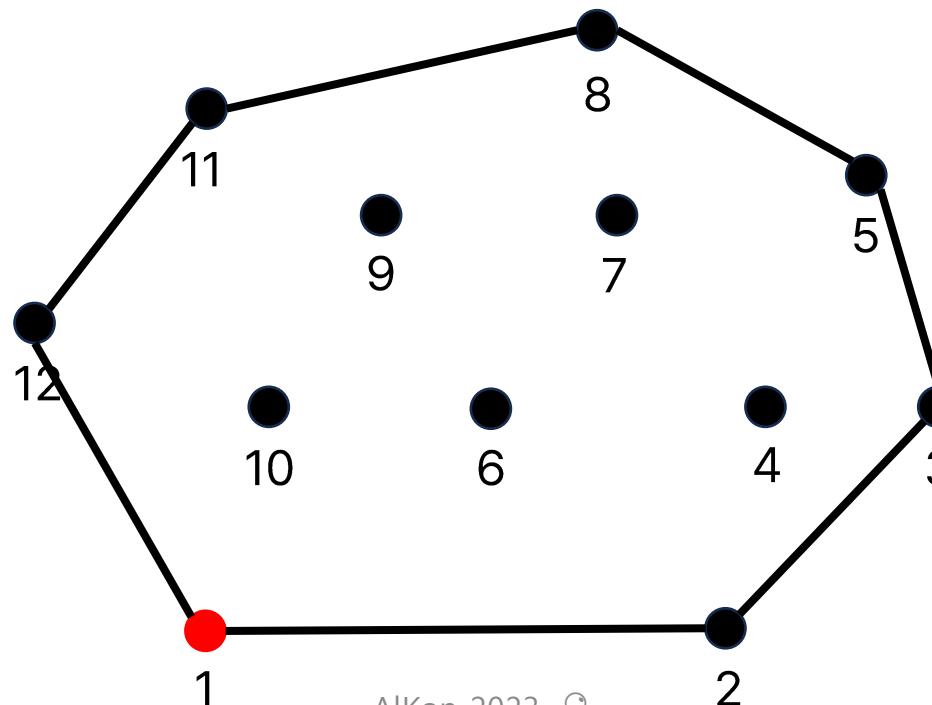
$$CCW(3, 5, 6) > 0$$

- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 반대 방향이라면 pop 했던 점을 다시 스택에 넣고 비교한 다음 점도 넣는다.



# Graham's Scan

- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 이를 계속해서 반복해주면 스택에 볼록 껹질을 이루는 점만 남게 된다.



12
11
8
5
3
2
1

# Graham's Scan

- Convex hull 알고리즘 중 가장 대표적인 방식이다.
- 시간 복잡도는 정렬에 종속되어  $O(N \log N)$ 이 된다.

# 볼록 껍질 BOJ 1708

- 점 집합이 주어졌을 때, 볼록 껍질을 이루는 점의 개수를 구해보자.
- 볼록 껍질의 변에 점이 여러 개 있는 경우 가장 양 끝 점만 개수에 포함한다.

# 볼록 껍질 BOJ 1708

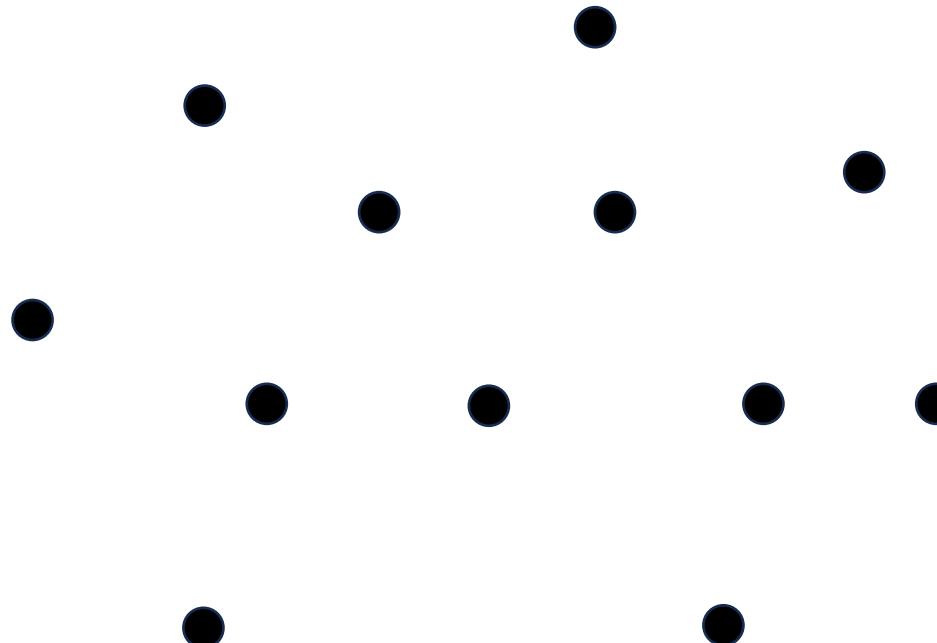
- 점 집합이 주어졌을 때, 볼록 껍질을 이루는 점의 개수를 구해보자.
- 볼록 껍질의 변에 점이 여러 개 있는 경우 가장 양 끝 점만 개수에 포함한다.
- 다른 처리 없이 Graham's Scan을 진행해주고, 스택에 들어있는 원소의 개수를 출력한다.

# 격자점 컨벡스헐 BOJ 2699

- 점 집합이 주어졌을 때, 볼록 껌질을 이루는 점들을 구해보자.
- 볼록 껌질 문제와는 다르게 아래의 조건이 없다.
  - 볼록 껌질의 변에 점이 여러 개 있는 경우 가장 양 끝 점만 개수에 포함한다.
- 해당 점들을 볼록 껌질에 포함시키기 위해 반시계 방향과 일직선 방향 모두를 고려해주자.

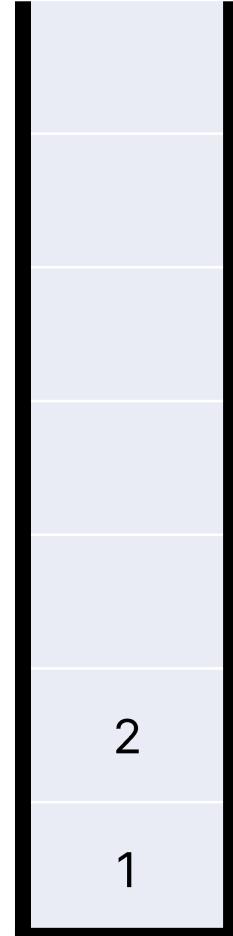
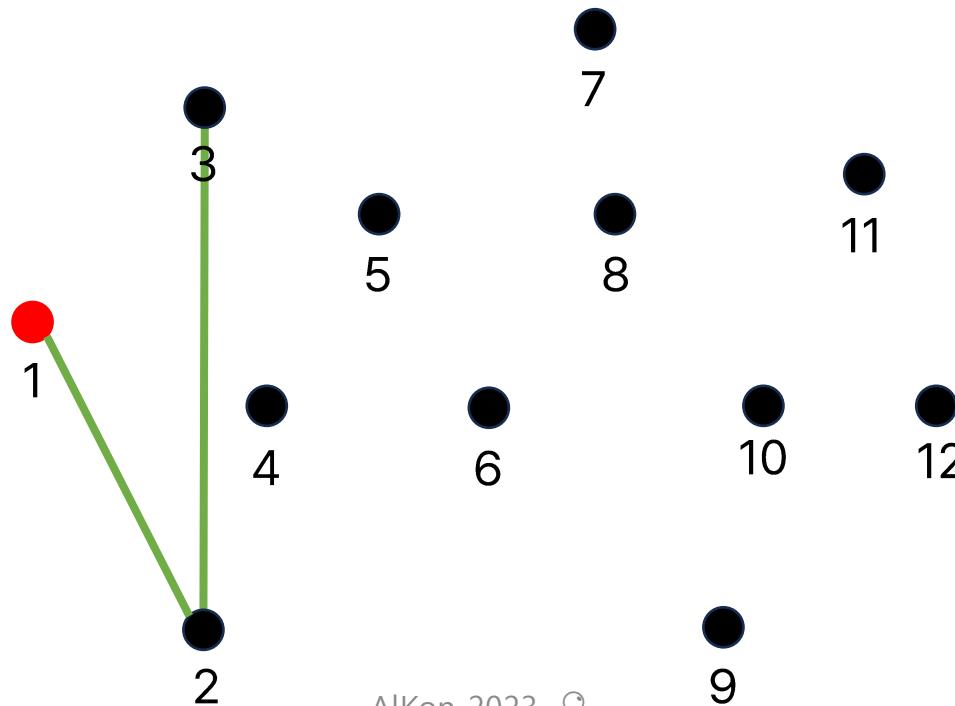
# Monotone Chain

- Graham's Scan은 각도에 따라 정렬하는 부분이 있어서 번거로울 수 있다.
- Monotone Chain은 upper hull과 lower hull로 나누어 구하는 알고리즘이다.



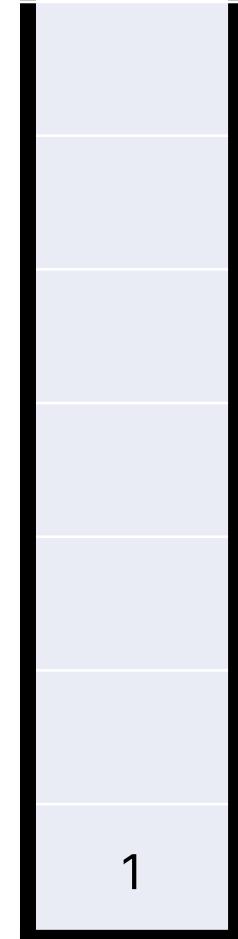
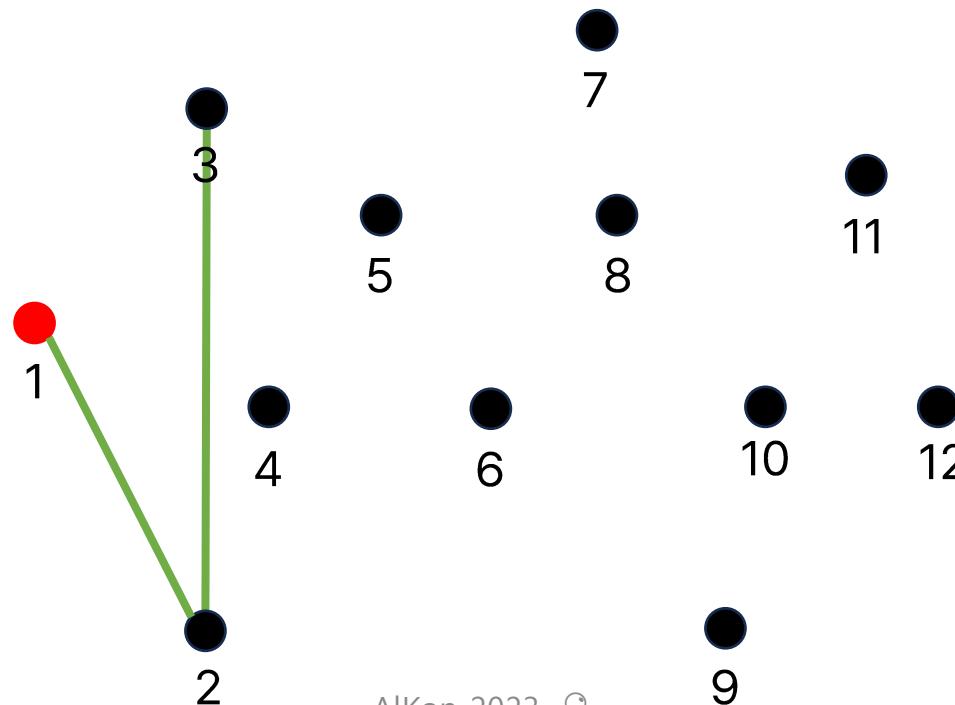
# Monotone Chain

- 우선  $x, y$ 좌표 순서대로 정렬을 해준다.
- 첫번째 정점과 두번째 정점을 스택에 넣고 시작한다.



# Monotone Chain

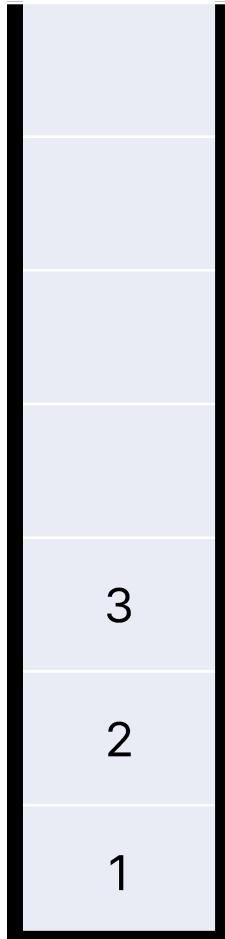
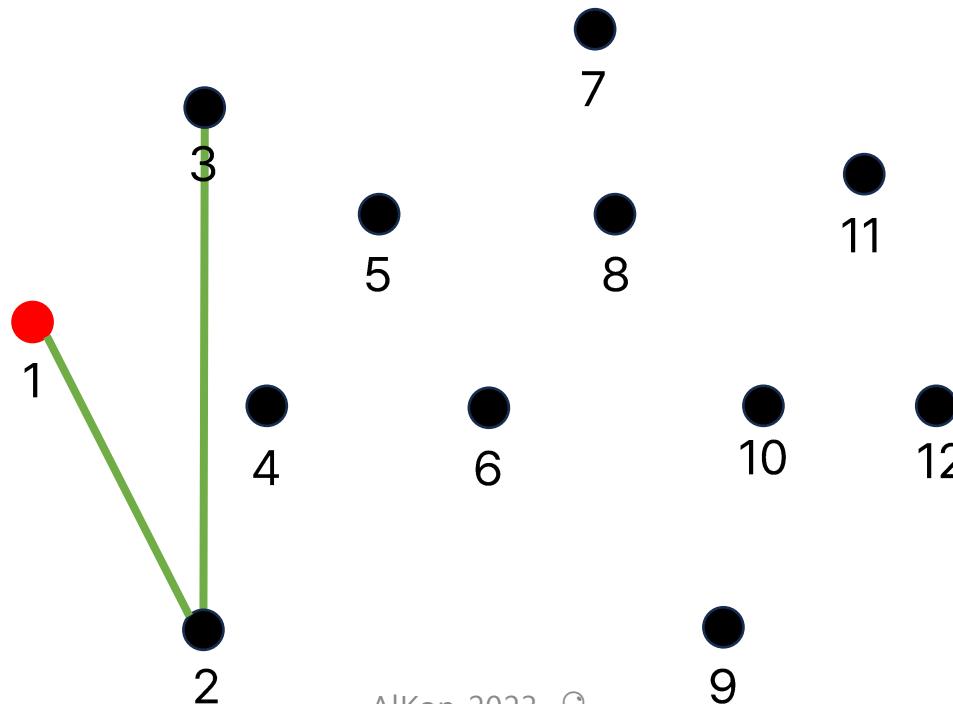
- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 스택 맨 위의 점만 스택에서 pop 한다.



# Monotone Chain

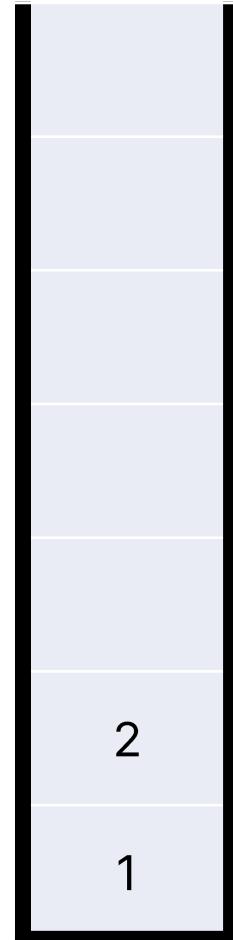
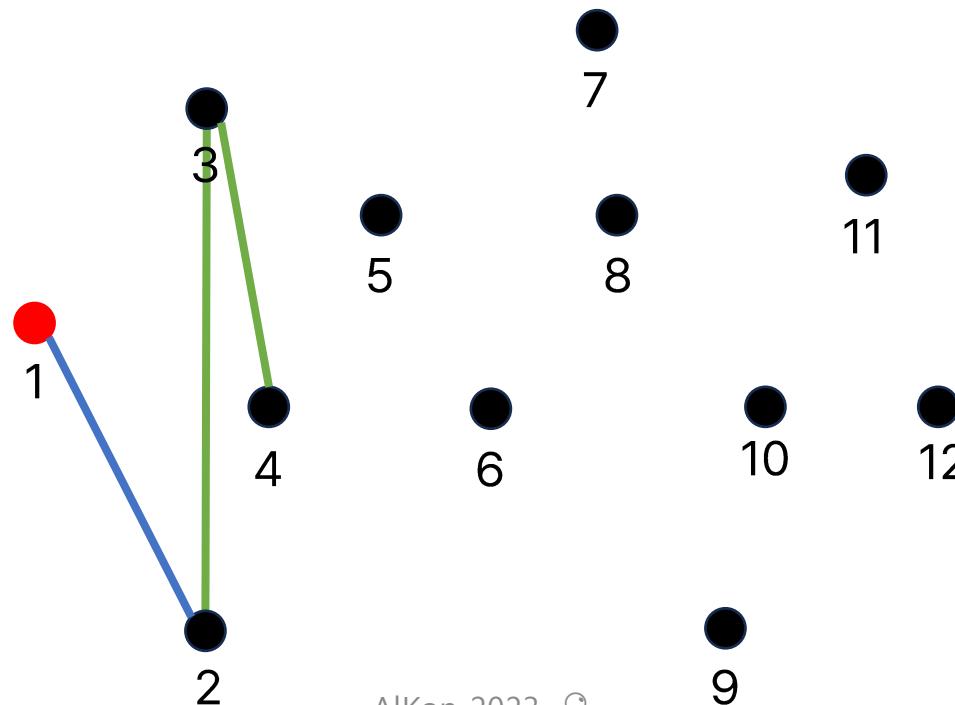
$$CCW(1, 2, 3) > 0$$

- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 반대 방향이라면 pop 했던 점을 다시 스택에 넣고 비교한 다음 점도 넣는다.



# Monotone Chain

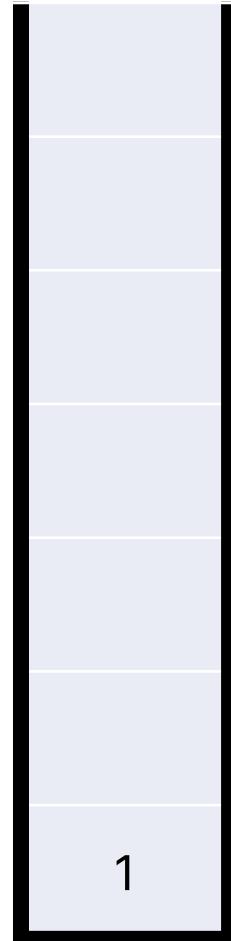
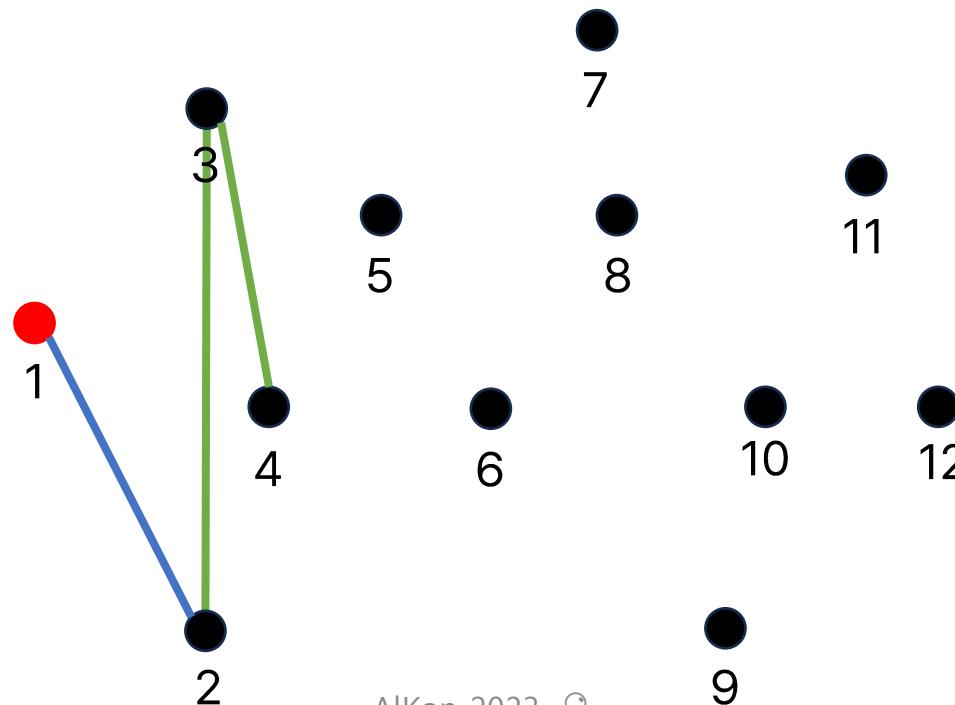
- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 스택 맨 위의 점만 스택에서 pop 한다.



# Monotone Chain

$$CCW(2, 3, 4) < 0$$

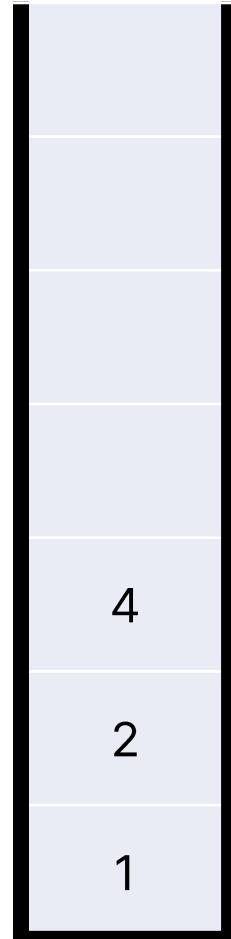
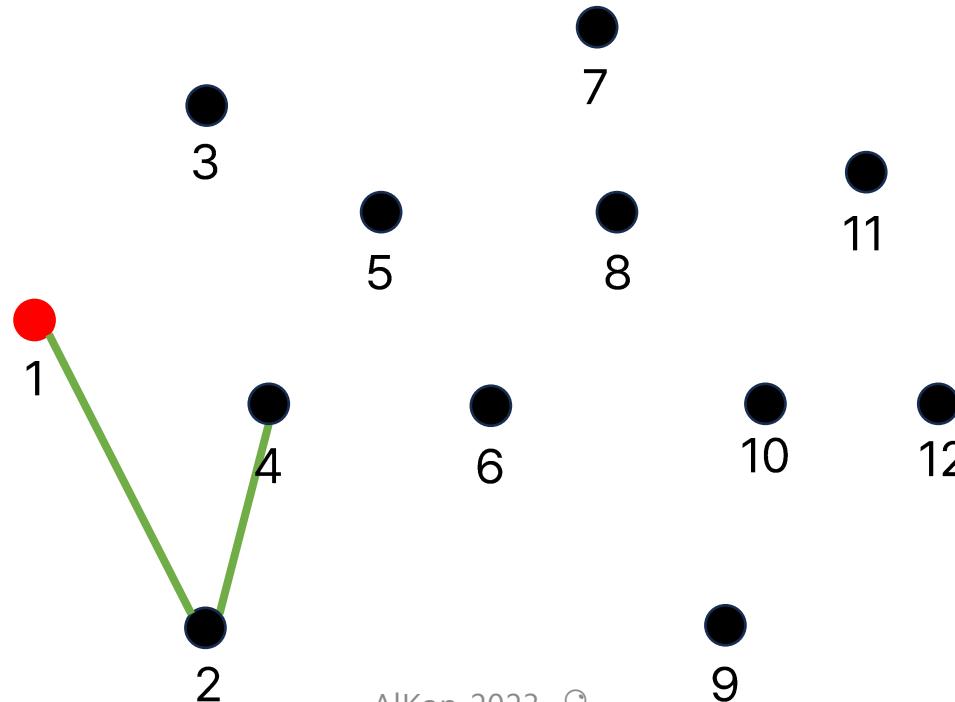
- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 방향이라면 pop을 이어간다.



# Monotone Chain

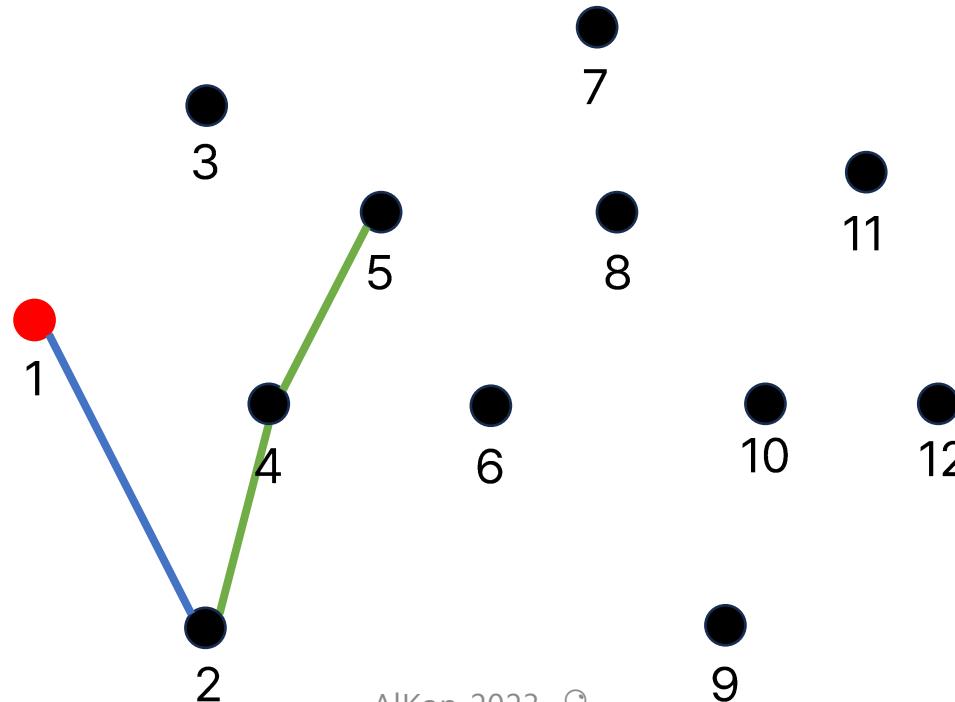
$$CCW(1, 2, 4) < 0$$

- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 반대 방향이라면 pop 했던 마지막 점과 비교한 다음 점을 스택에 넣는다.



# Monotone Chain

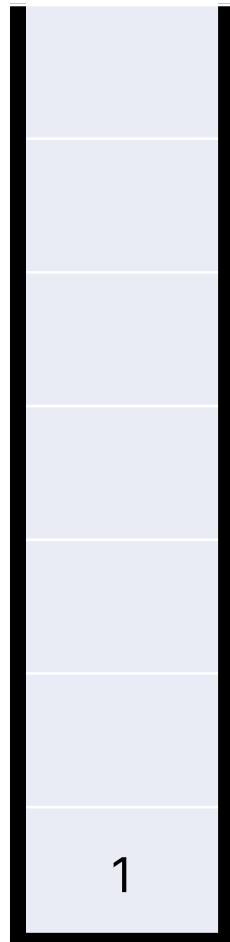
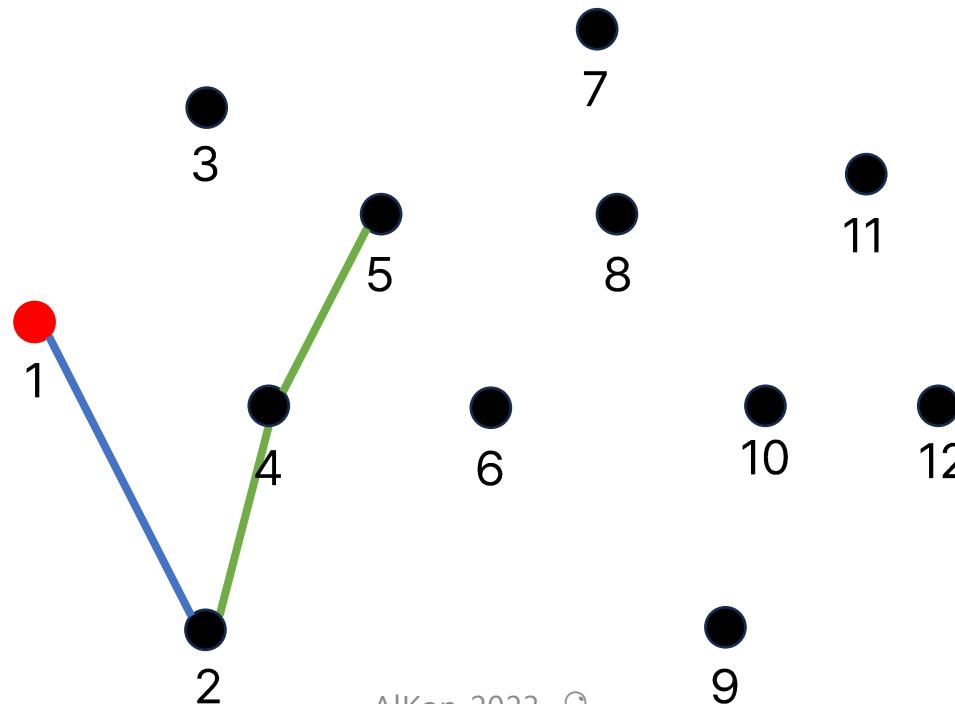
- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 스택 맨 위의 점만 스택에서 pop 한다.



# Monotone Chain

$$CCW(2, 4, 5) < 0$$

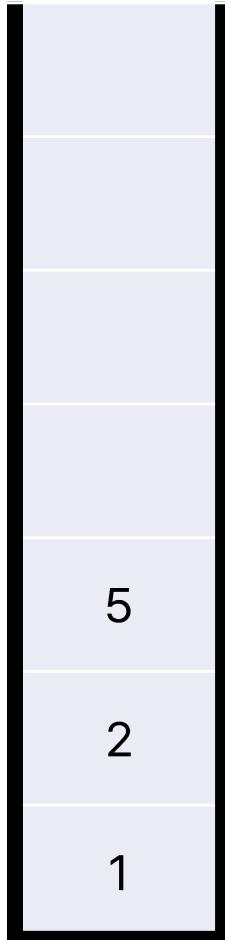
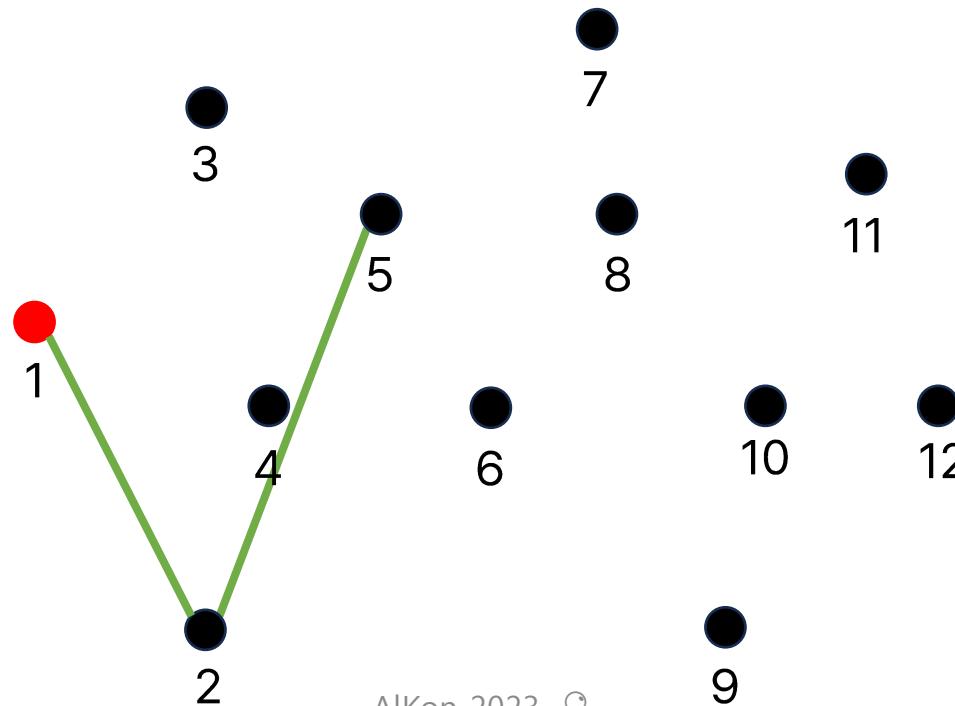
- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 방향이라면 pop을 이어간다.



# Monotone Chain

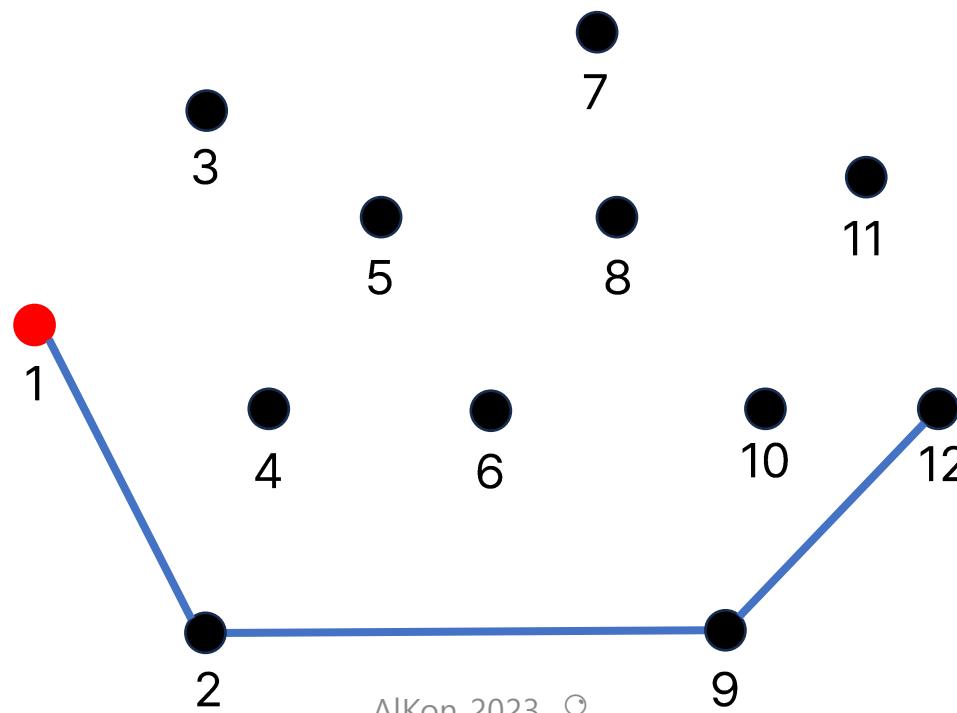
$$CCW(1, 2, 5) < 0$$

- 스택 상단의 두 점과 그 다음에 나올 점과의 진행 방향을 판단한다.
  - 시계 반대 방향이라면 pop 했던 마지막 점과 비교한 다음 점을 스택에 넣는다.



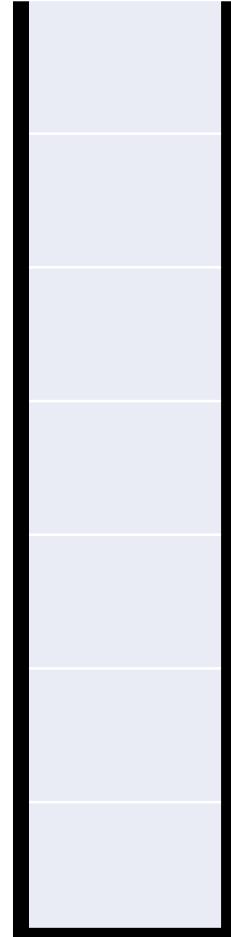
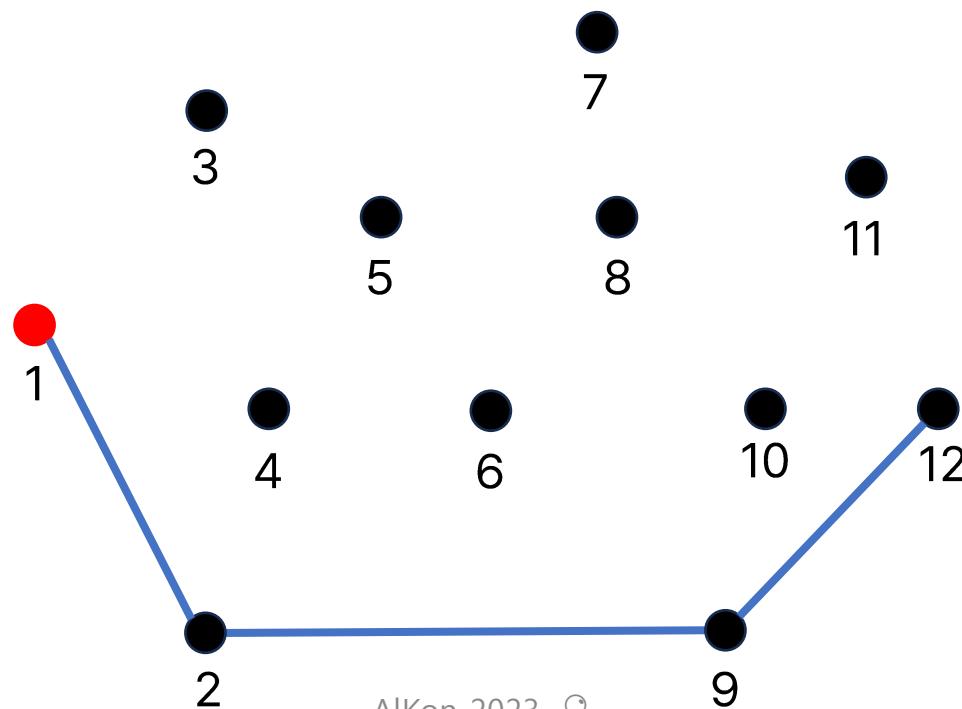
# Monotone Chain

- 진행이 완료되면 아래와 같은 lower hull을 구할 수 있다.



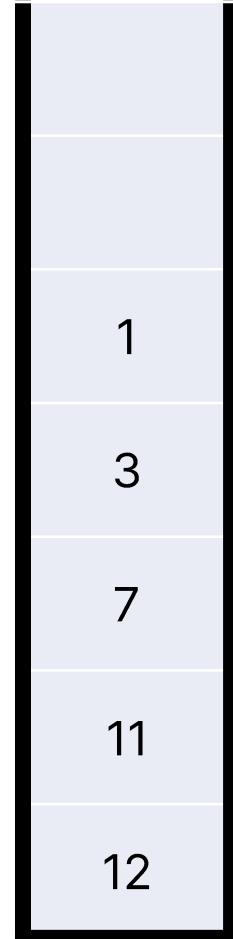
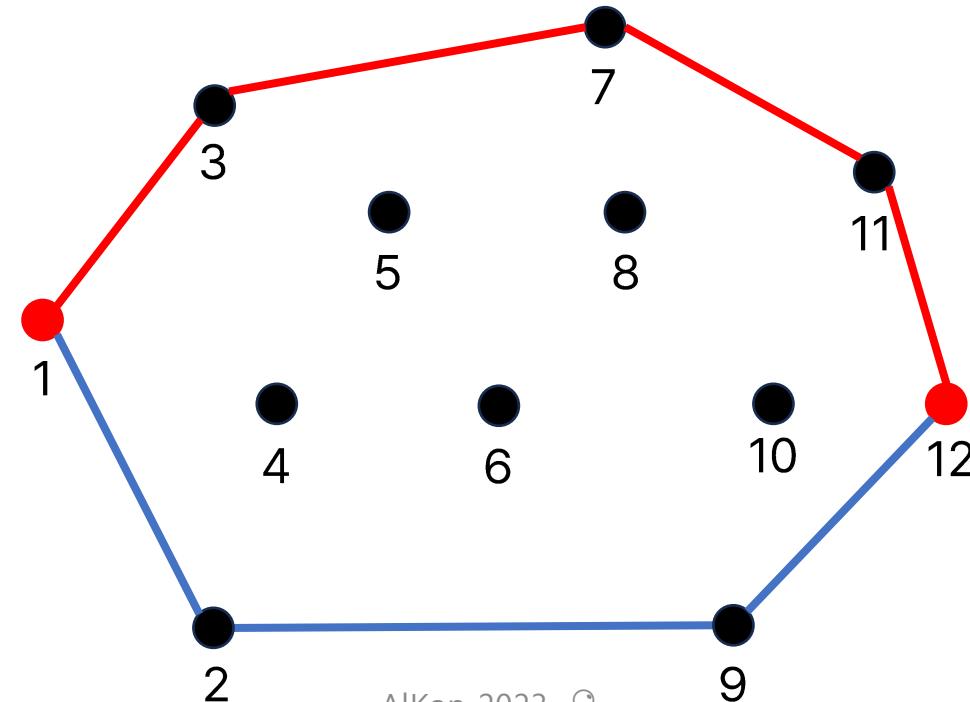
# Monotone Chain

- upper hull에서는 점을 거꾸로 읽으면 진행하면 된다.



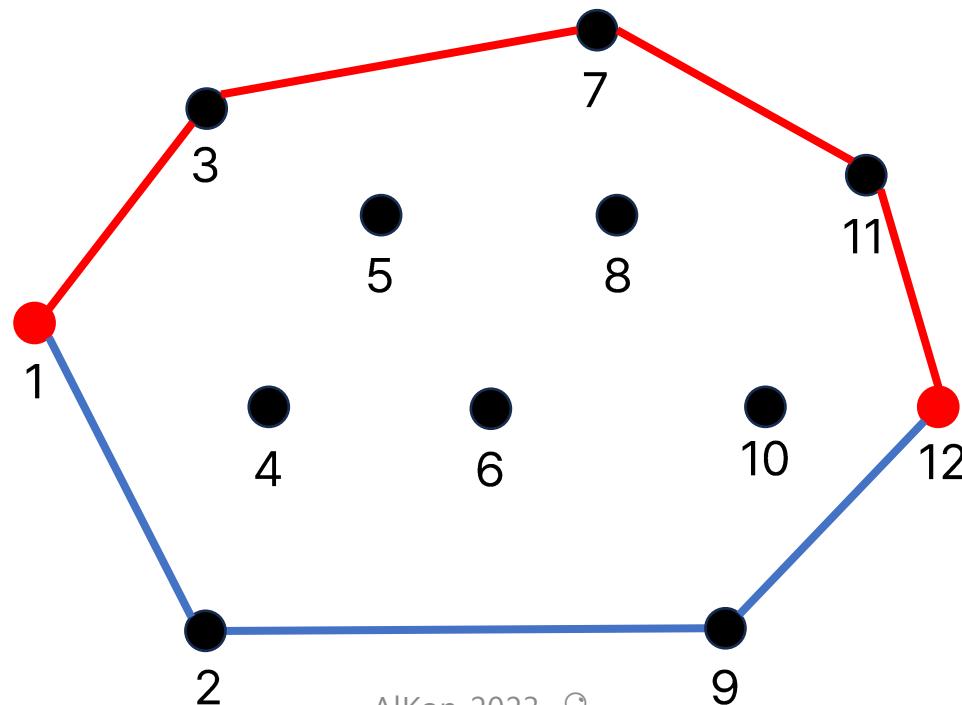
# Monotone Chain

- upper hull에서는 점을 거꾸로 읽으면 진행하면 된다.
- 그렇다면 그림과 같이 hull에 속한 집합을 얻을 수 있다.



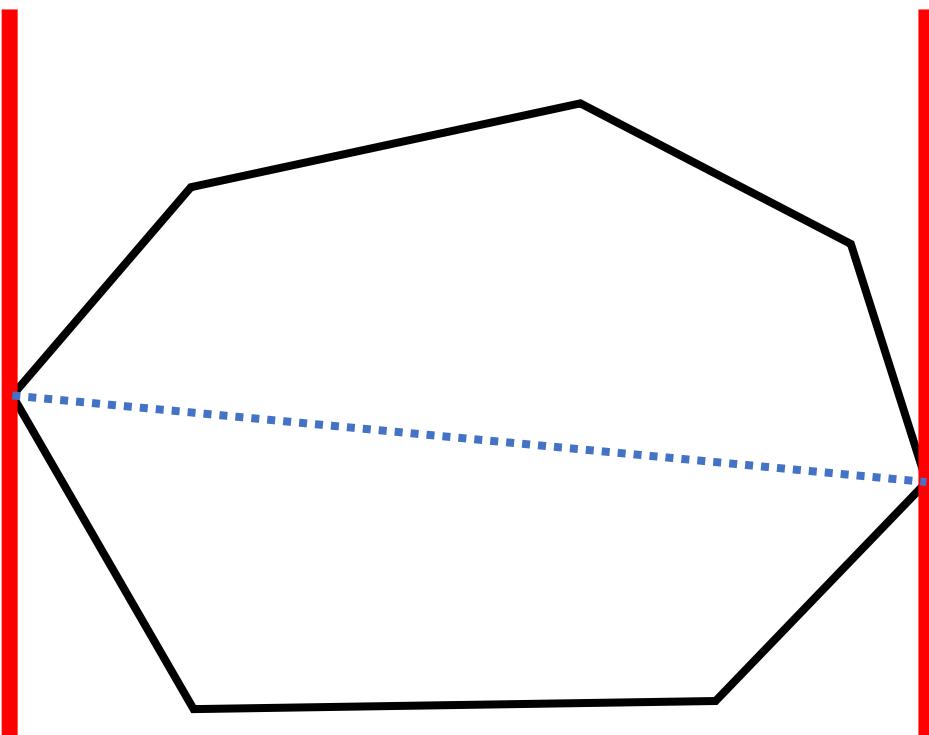
# Monotone Chain

- upper hull과 lower hull의 시작점은 서로 겹친다.
- 각 hull의 마지막 점을 제외하고 합쳐주면 된다.



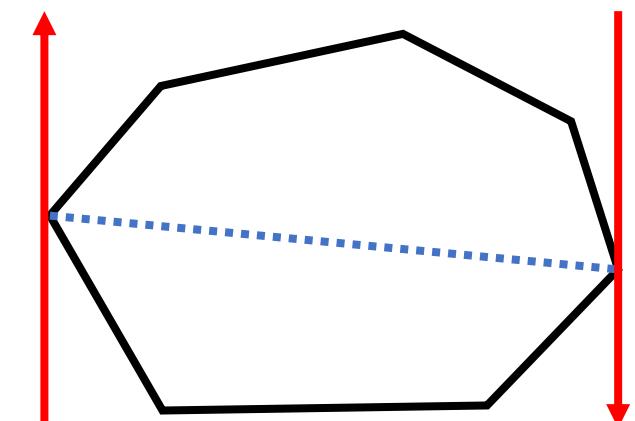
# Rotating Calipers

- 볼록 다각형의 지름을 구하는 알고리즘



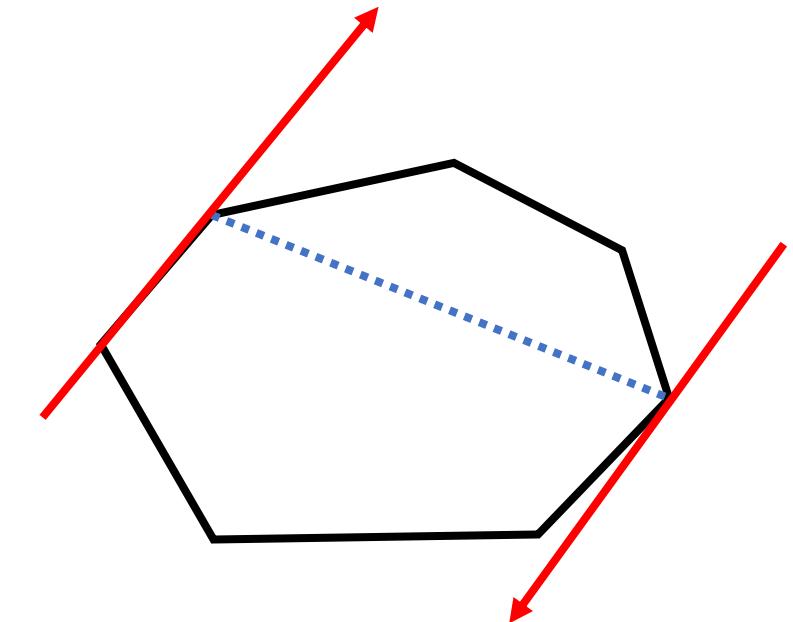
# Rotating Calipers

- 볼록 다각형의 지름을 구하는 알고리즘
- 가장 왼쪽과 오른쪽의 점에서 시작한다.



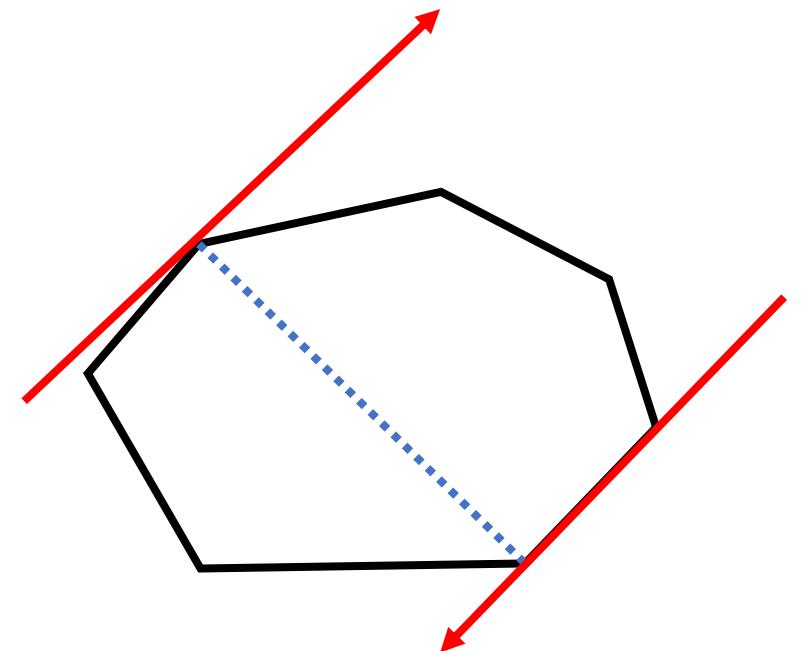
# Rotating Calipers

- 볼록 다각형의 지름을 구하는 알고리즘
- 다음 진행 방향에서 만나는 점과의 각도 중 더 작은 쪽 만큼 두 벡터를 회전시킨다.



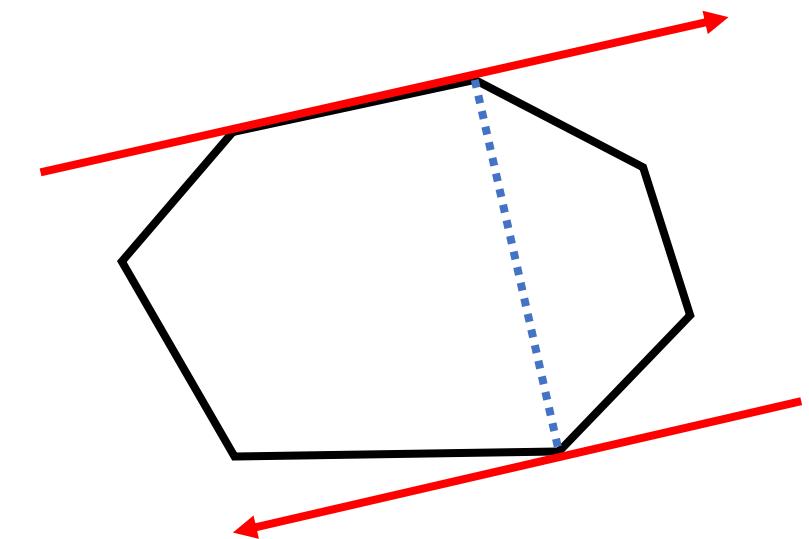
# Rotating Calipers

- 볼록 다각형의 지름을 구하는 알고리즘
- 다음 진행 방향에서 만나는 점과의 각도 중 더 작은 쪽 만큼 두 벡터를 회전시킨다.
- 이를 계속해서 반복한다.



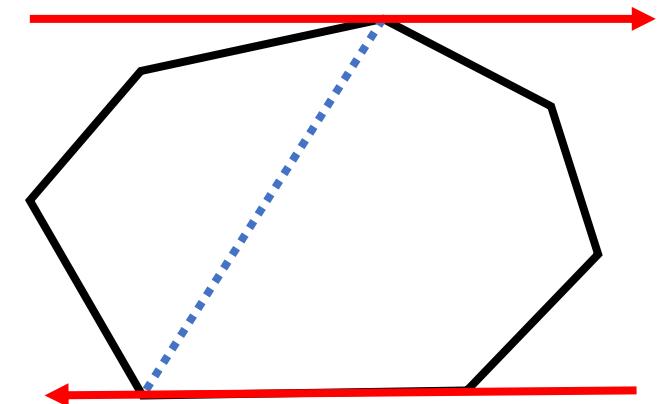
# Rotating Calipers

- 볼록 다각형의 지름을 구하는 알고리즘
- 다음 진행 방향에서 만나는 점과의 각도 중 더 작은 쪽 만큼 두 벡터를 회전시킨다.
- 이를 계속해서 반복한다.



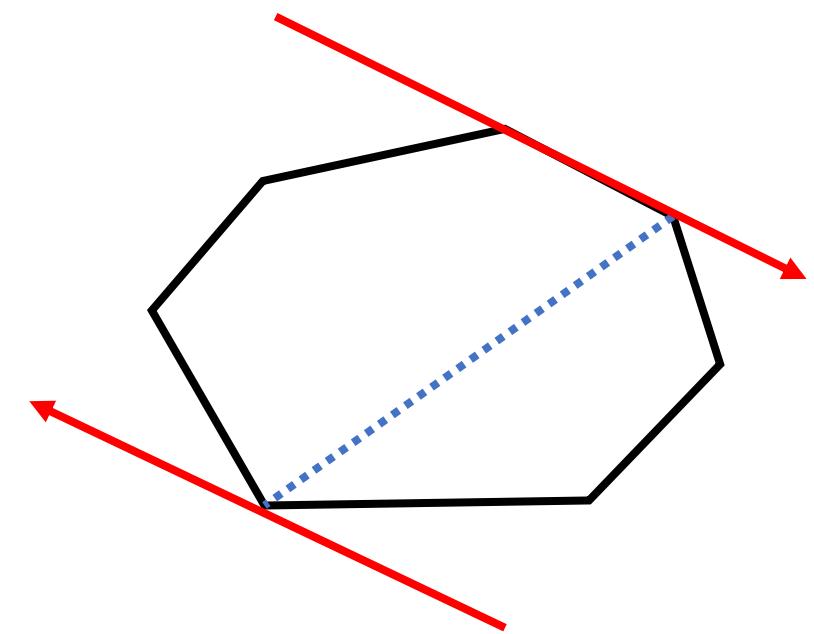
# Rotating Calipers

- 볼록 다각형의 지름을 구하는 알고리즘
- 다음 진행 방향에서 만나는 점과의 각도 중 더 작은 쪽 만큼 두 벡터를 회전시킨다.
- 이를 계속해서 반복한다.



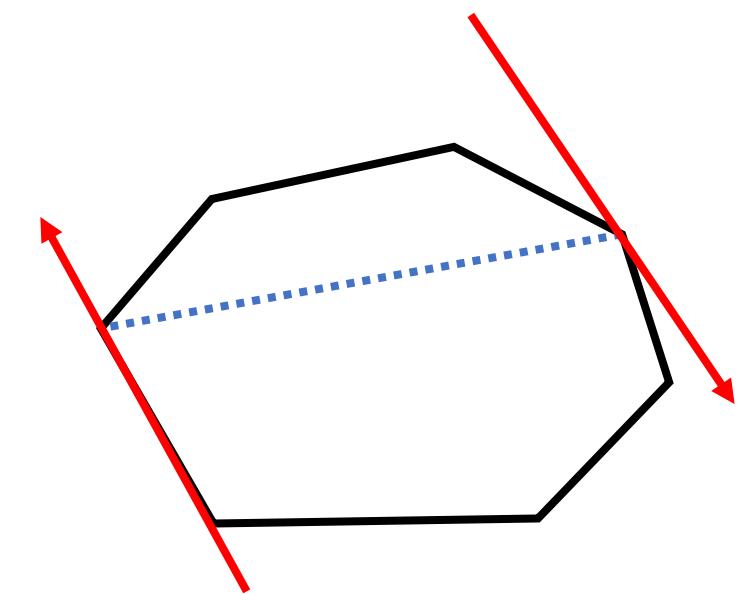
# Rotating Calipers

- 볼록 다각형의 지름을 구하는 알고리즘
- 다음 진행 방향에서 만나는 점과의 각도 중 더 작은 쪽 만큼 두 벡터를 회전시킨다.
- 이를 계속해서 반복한다.



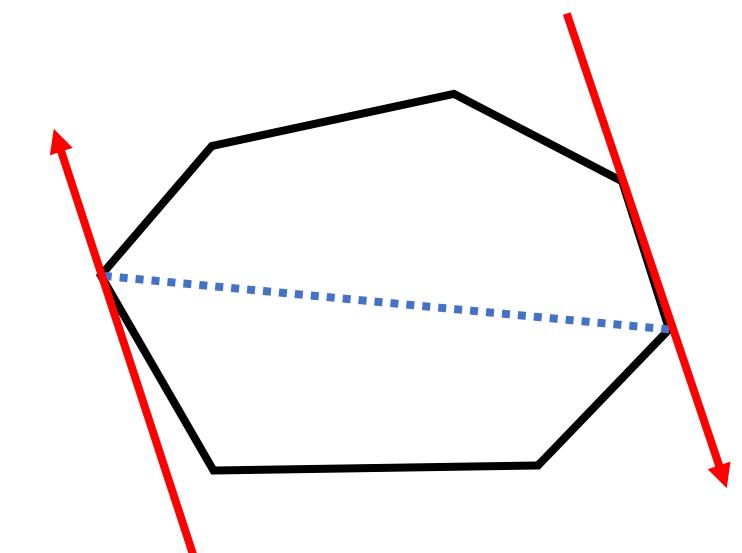
# Rotating Calipers

- 볼록 다각형의 지름을 구하는 알고리즘
- 다음 진행 방향에서 만나는 점과의 각도 중 더 작은 쪽 만큼 두 벡터를 회전시킨다.
- 이를 계속해서 반복한다.



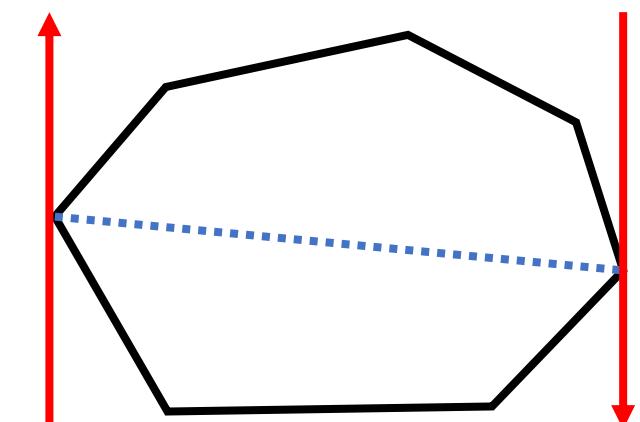
# Rotating Calipers

- 볼록 다각형의 지름을 구하는 알고리즘
- 다음 진행 방향에서 만나는 점과의 각도 중 더 작은 쪽 만큼 두 벡터를 회전시킨다.
- 이를 계속해서 반복한다.



# Rotating Calipers

- 볼록 다각형의 지름을 구하는 알고리즘
- 다음 진행 방향에서 만나는 점과의 각도 중 더 작은 쪽 만큼 두 벡터를 회전시킨다.
- 한 바퀴를 다시 돌아온 경우 종료한다.
- 시간 복잡도는 볼록 껍질을 구했다는 전제 하에  $O(N)$ 이다.



# 가장 먼 두 점 BOJ 2049

- 2차원 평면상에  $n$ 개의 점이 주어졌을 때, 이 점들 중 가장 먼 두 점을 구해보자.

# 가장 먼 두 점 BOJ 2049

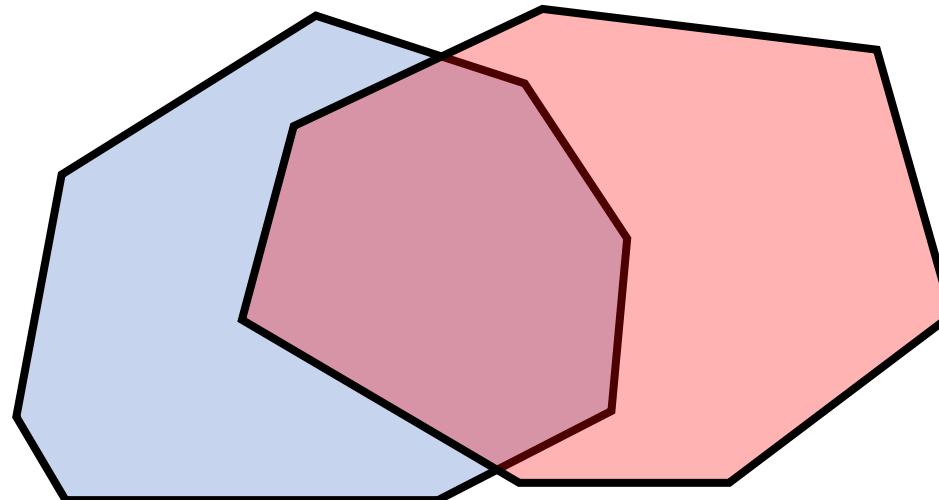
- 2차원 평면상에  $n$ 개의 점이 주어졌을 때, 이 점들 중 가장 먼 두 점을 구해보자.
- 볼록 껍질을 만든 다음, Rotating Calipers를 이용해 구해주면 된다.

# 넓이 BOJ 1077

- 두 개의 볼록 다각형이 주어진다.
- 두 볼록 다각형의 넓이는 0이 아니다.
- 볼록 다각형의 꼭짓점은 다른 볼록 다각형의 변 위에 있지 않다.
- 볼록 다각형을 이루는 꼭짓점의 개수는 100개 이하이다.
- 두 볼록 다각형의 겹쳐진 넓이를 구해보자.

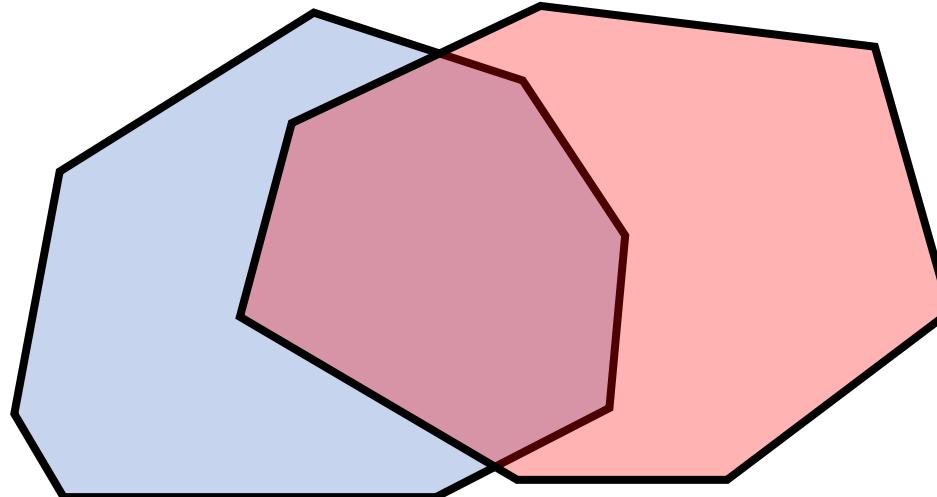
# 넓이 BOJ 1077

- 두 볼록 다각형의 겹쳐진 넓이를 구해보자.



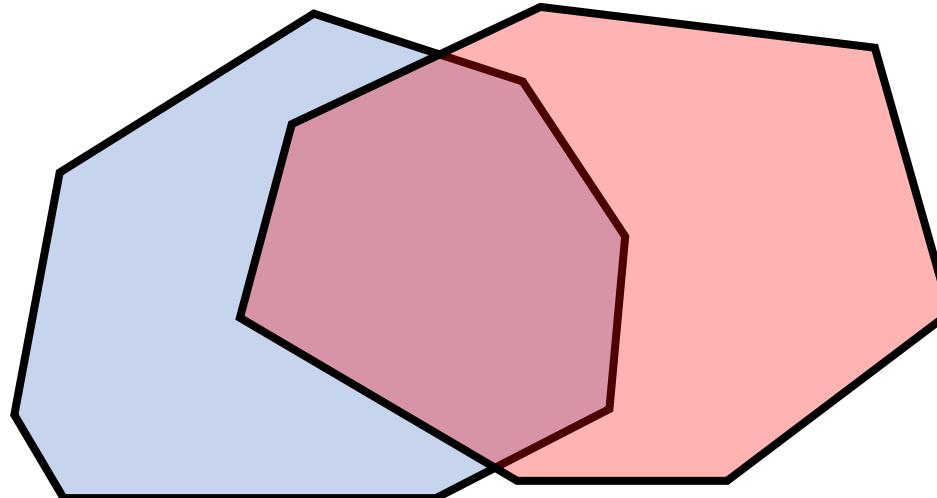
# 넓이 BOJ 1077

- 두 볼록 다각형의 겹쳐진 넓이를 구해보자.
- 겹쳐진 부분의 특징으로는 무엇이 있을까?



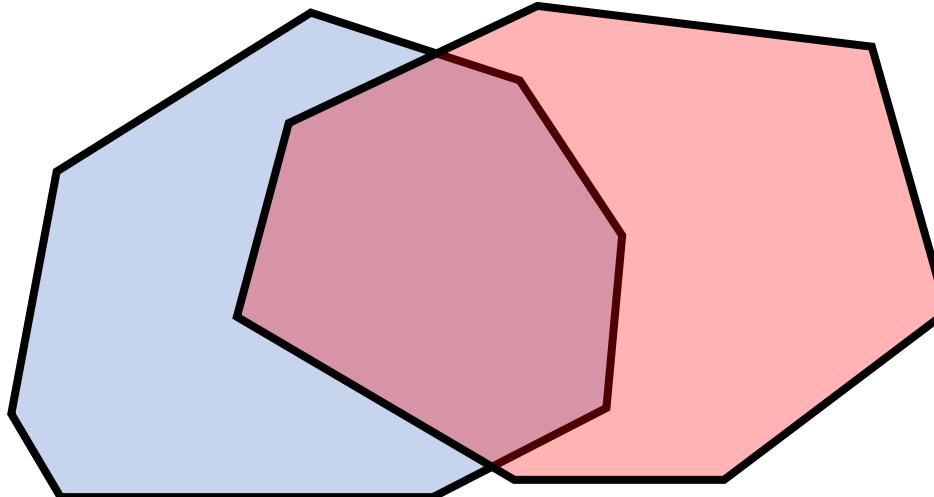
# 넓이 BOJ 1077

- 두 볼록 다각형의 겹쳐진 넓이를 구해보자.
- 겹쳐진 부분의 특징으로는 무엇이 있을까?
  - 겹쳐진 부분은 볼록 다각형이다.



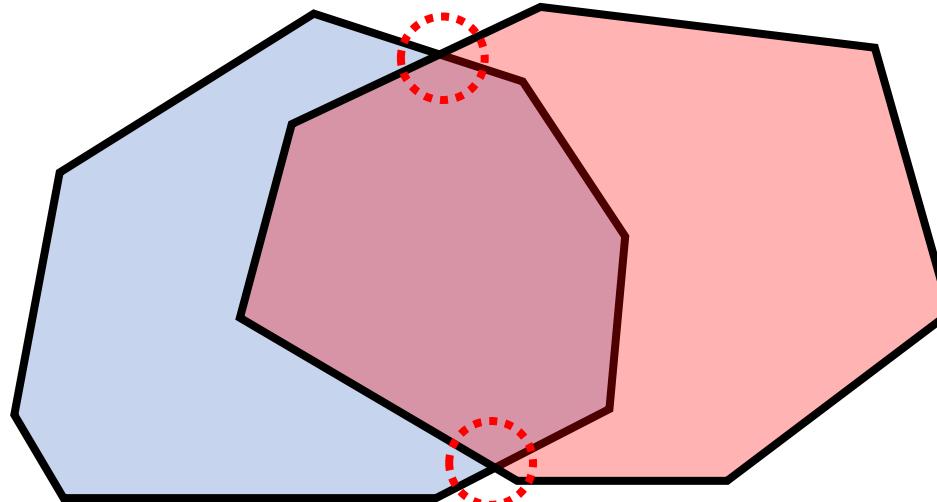
# 넓이 BOJ 1077

- 두 볼록 다각형의 겹쳐진 넓이를 구해보자.
- 겹쳐진 부분의 특징으로는 무엇이 있을까?
  - 해당 볼록 다각형의 꼭짓점은 다각형 내부의 점들과 2개의 교차점으로 이루어져 있다.



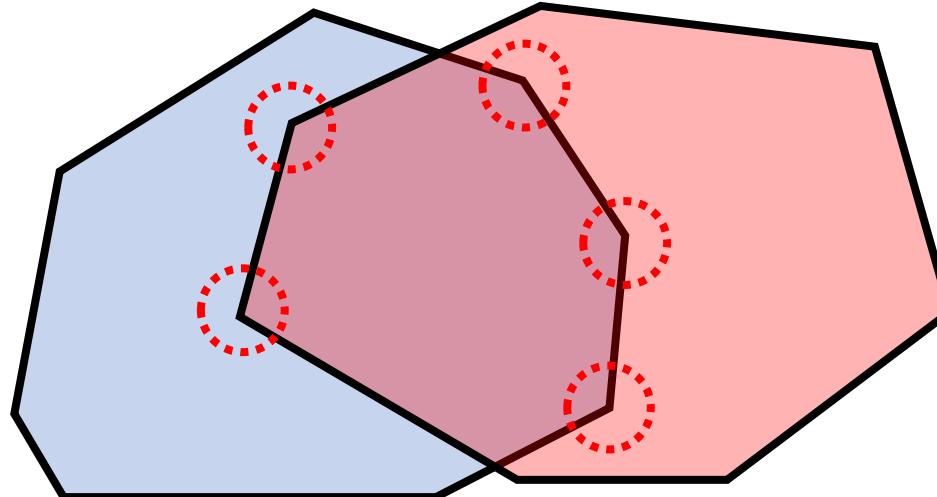
# 넓이 BOJ 1077

- 두 볼록 다각형의 겹쳐진 넓이를 구해보자.
- 볼록 다각형을 이루는 점의 개수가 100개 이하이므로, 교차점은 모든 선분을 탐색하여 찾 아주어도 무관하다.



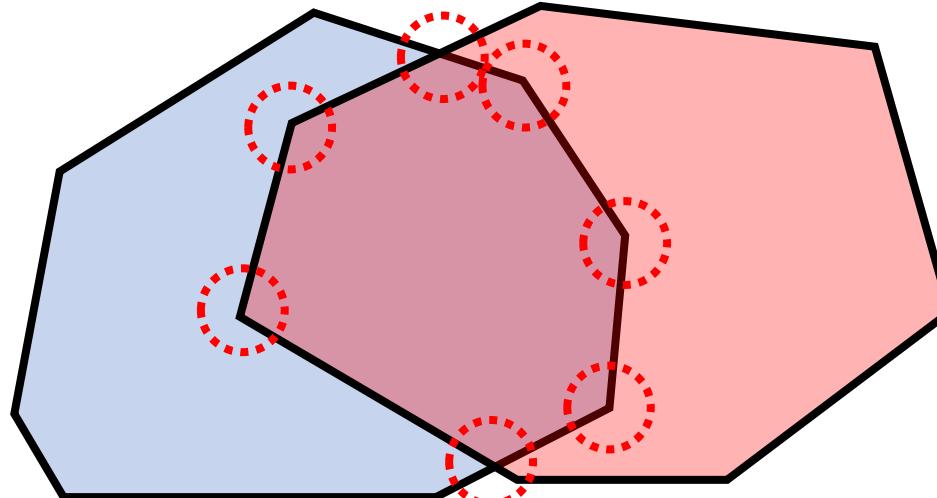
# 넓이 BOJ 1077

- 두 볼록 다각형의 겹쳐진 넓이를 구해보자.
- 다각형 내부의 점 판정을 통해 다각형 내부에 있는 점들을 모두 찾아준다.



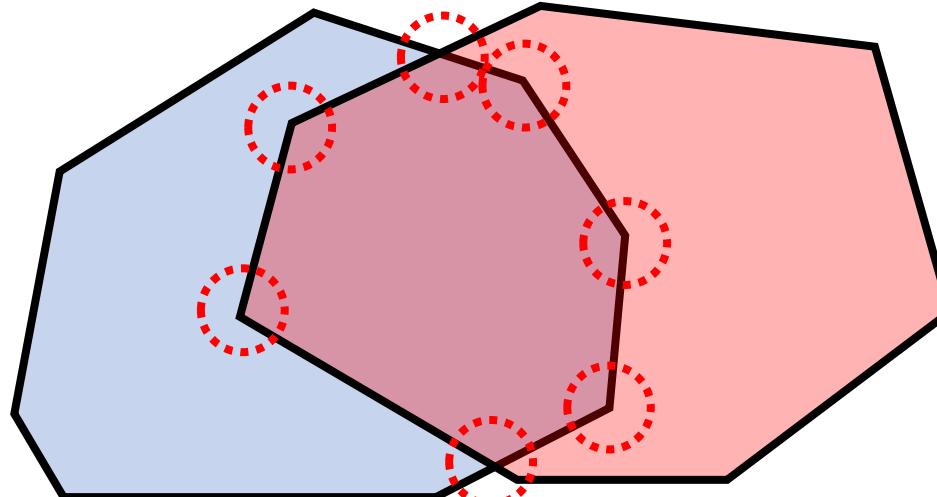
# 넓이 BOJ 1077

- 두 볼록 다각형의 겹쳐진 넓이를 구해보자.
- 찾은 점들로 볼록 다각형을 구성해준다.



# 넓이 BOJ 1077

- 두 볼록 다각형의 겹쳐진 넓이를 구해보자.
- 해당 볼록 다각형의 넓이를 구해준다.



# 연습 문제

- 11758 : CCW
- 2166 : 다각형의 면적
- 17386 : 선분 교차 1
- 17387 : 선분 교차 2
- 1708 : 볼록 껍질
- 2699 : 격자점 컨벡스헐
- 2049 : 가장 먼 두 점
- 1077 : 넓이

# Reference

- <https://anz1217.tistory.com/107>
- <https://en.wikipedia.org/wiki/Calipers>
- <https://hellogaon.tistory.com/40>