

# 기하 알고리즘 1

최백준 [choi@startlink.io](mailto:choi@startlink.io)

---

CCW

# CCW


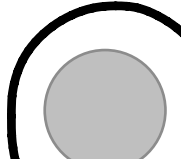
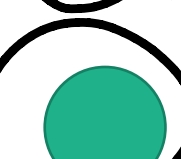
Counter Clockwise

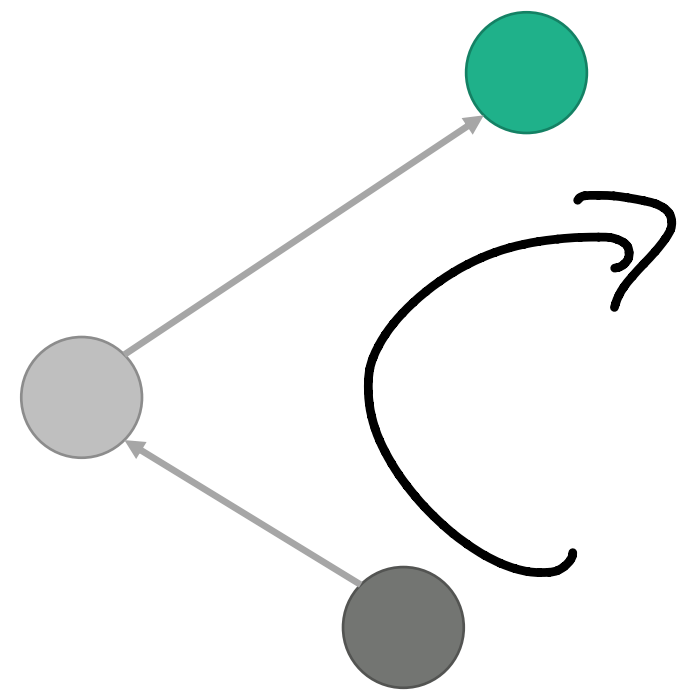
3

- P1 ( $x_1, y_1$ ), P2 ( $x_2, y_2$ ), P3 ( $x_3, y_3$ )가 있을 때
- P1  $\rightarrow$  P2  $\rightarrow$  P3가 어떤 방향인지 알아낼 수 있다.
- 1: 반시계방향
- 0: 일직선
- -1: 시계방향

# CCW

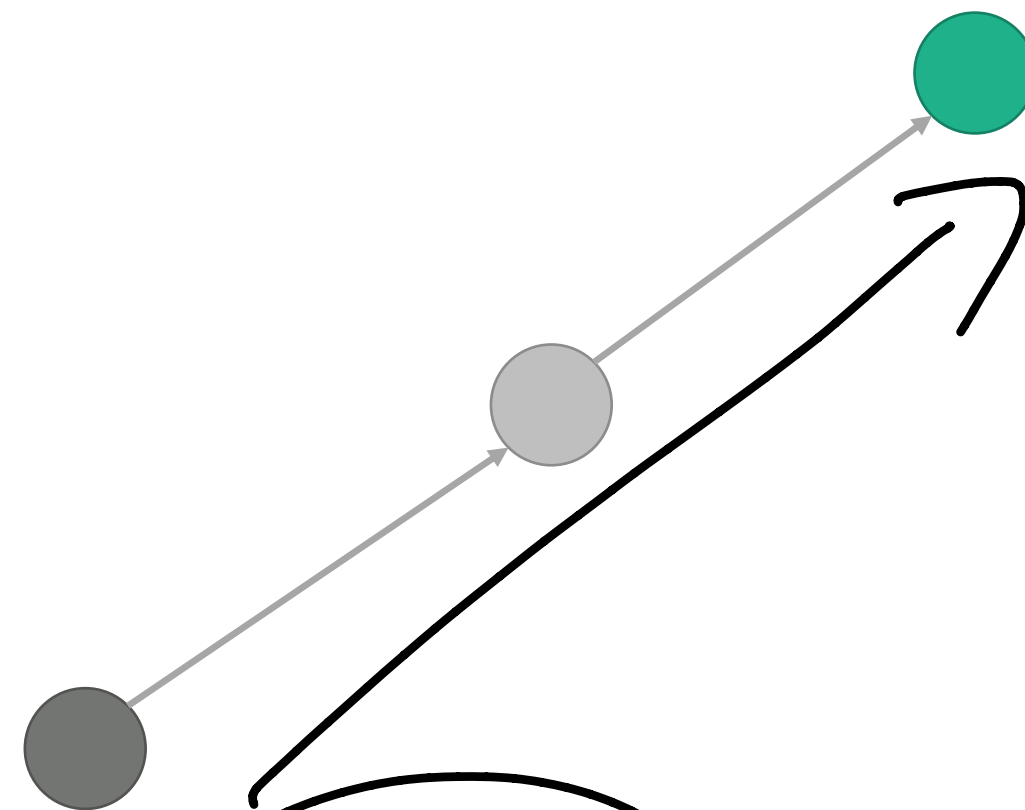
Counter Clockwise

- P1 
- P2 
- P3 



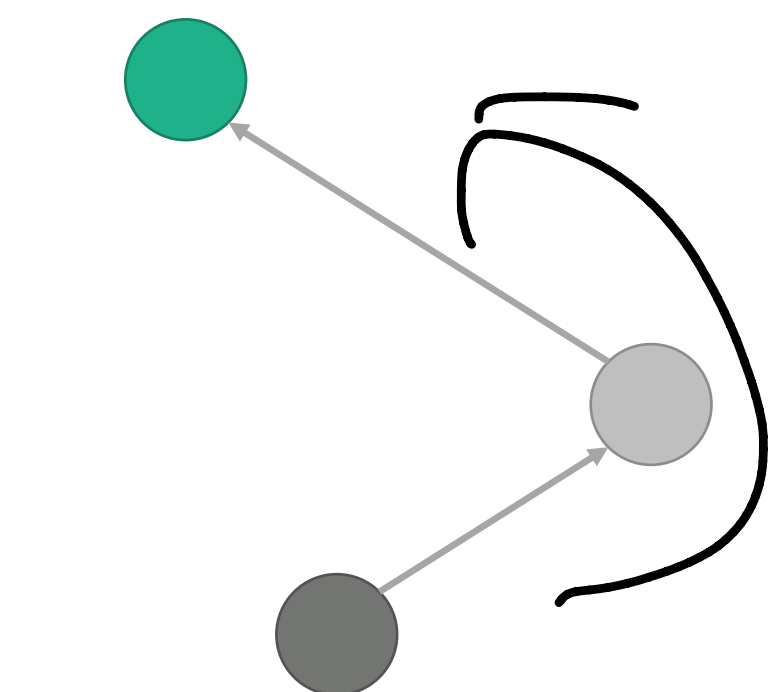
시계 방향 -1

                      
시계



일직선 0



반시계 방향 1

                      
반시계

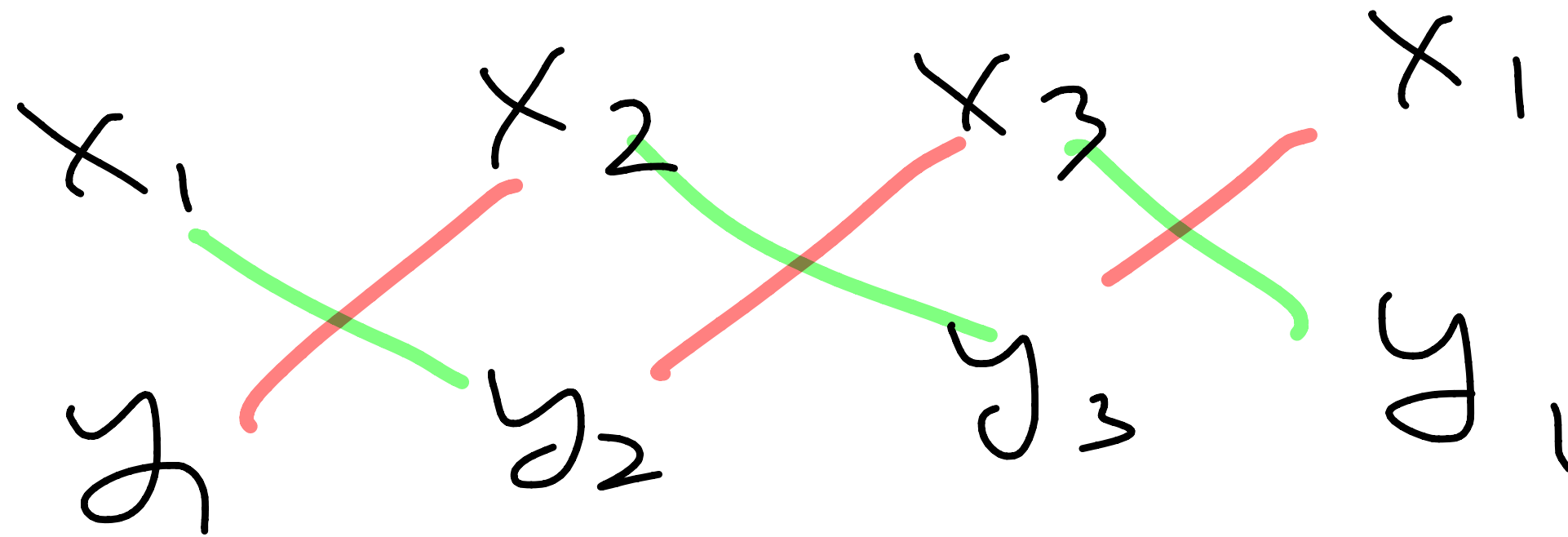
# CCW

Counter Clockwise

§173

5

- 두 벡터  $\overrightarrow{P_1P_2}$ ,  $\overrightarrow{P_1P_3}$ 의 벡터곱의 부호와 같은 의미를 갖는다.
- $P_1 (x_1, y_1)$ ,  $P_2 (x_2, y_2)$ ,  $P_3 (x_3, y_3)$
- $x_1 \times y_2 + x_2 \times y_3 + x_3 \times y_1 - y_1 \times x_2 - y_2 \times x_3 - y_3 \times x_1$



# CCW

<https://www.acmicpc.net/problem/11758>

32<sub>2</sub>    6584

- 소스: <http://codeplus.codes/df2ac2eafafc4b60aa2cb5ba875c64f8>

# 다각형의 넓이

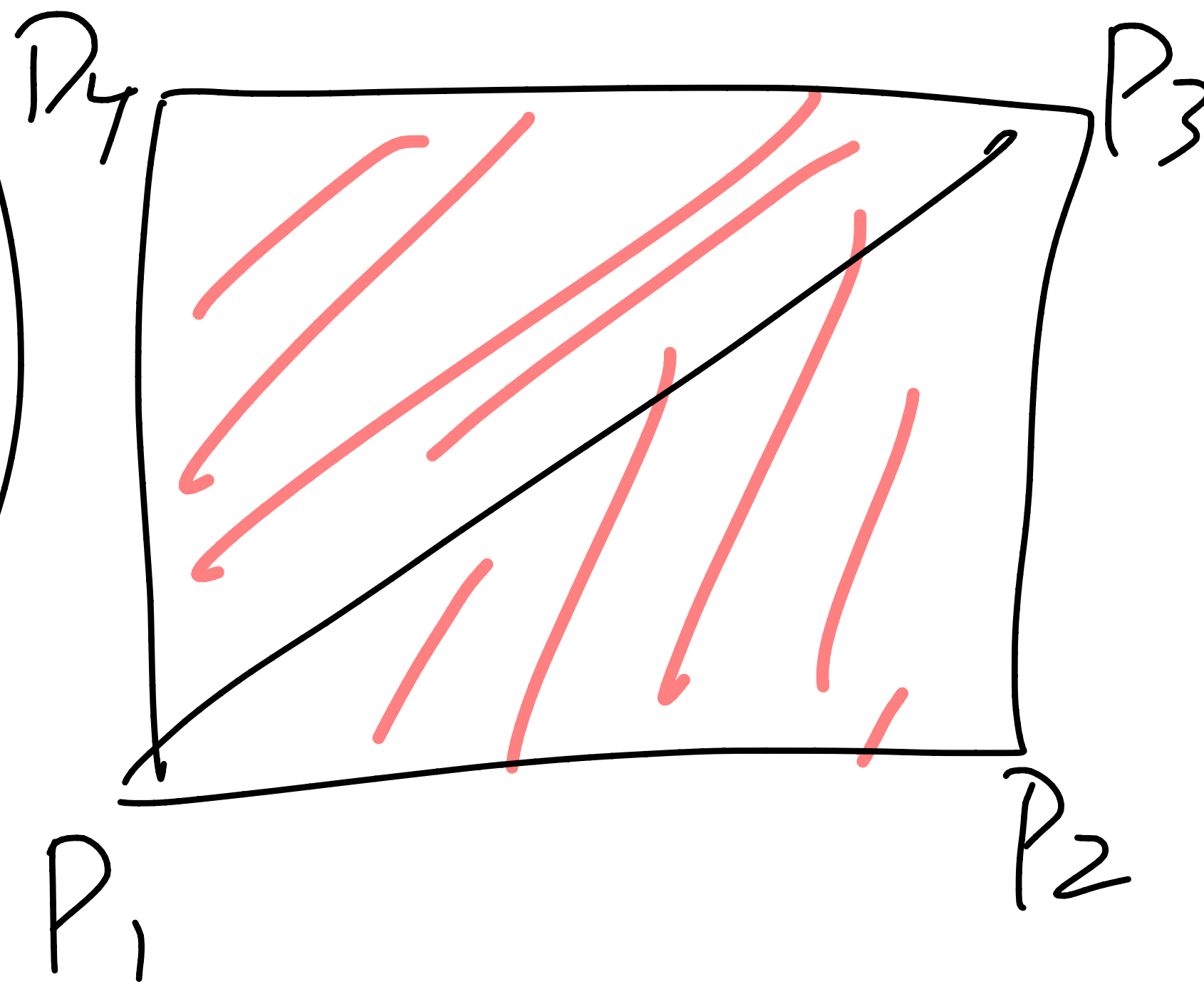
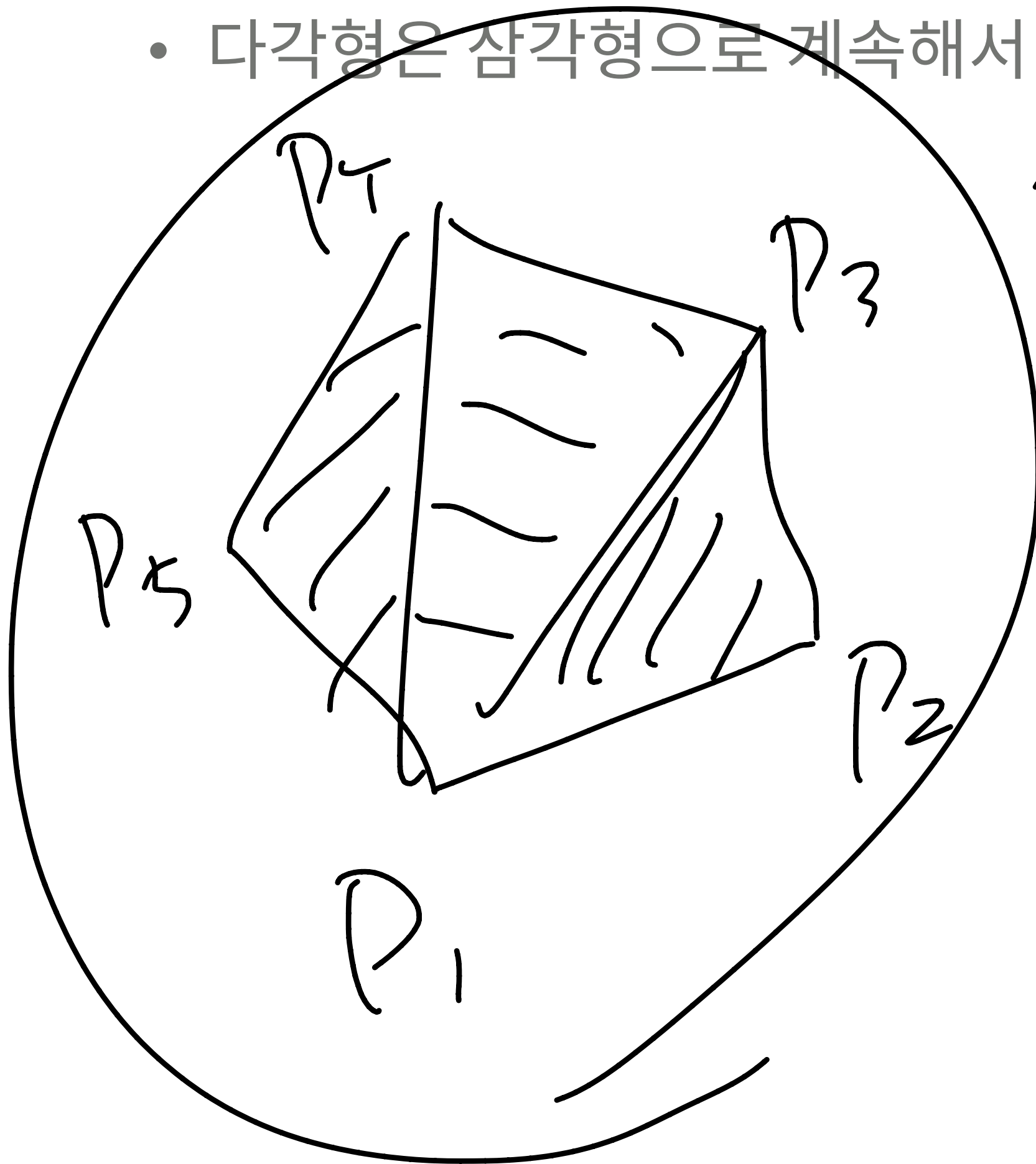
---

# 다각형의 넓이

Area of Polygon

8

- 점 3개로 이루어진 삼각형의 경우 CCW 공식의 결과에  $\frac{1}{2}$ 를 곱하면 넓이를 구할 수 있다.
- 다각형은 삼각형으로 계속해서 나누어서 넓이를 구할 수 있다.

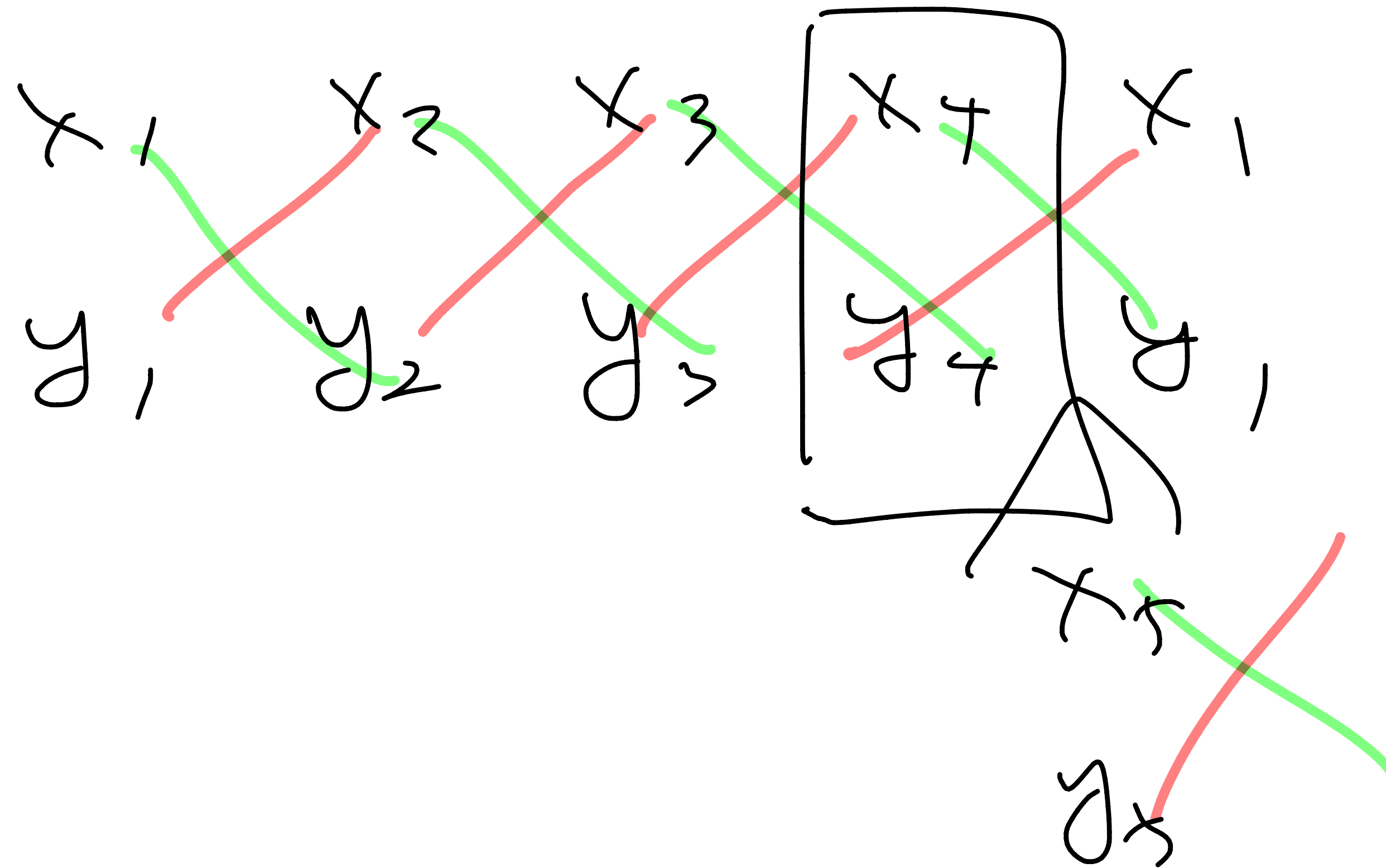




# 다각형의 면적

<https://www.acmicpc.net/problem/2166>

- 다각형의 면적을 구하는 문제



# 다각형의 면적

<https://www.acmicpc.net/problem/2166>

- 소스: <http://codeplus.codes/8837a32338f0461b90b219b0159faa5c>

# 선분의 교차

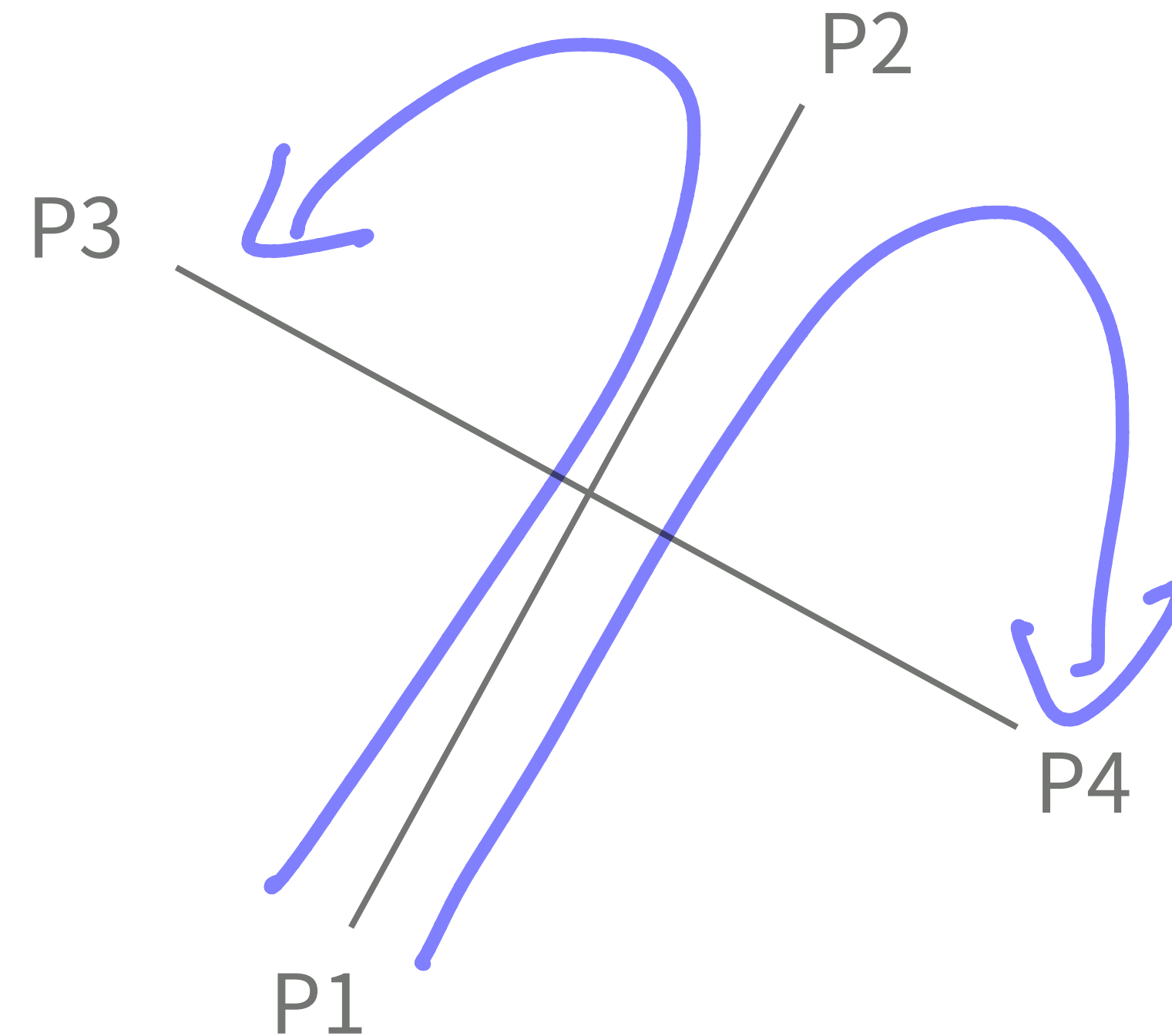
# 선분의 교차

## Intersection of Two Line Segments

- 두 선분의 교차 확인
- P1-P2-P3의 방향과 P1-P2-P4의 방향이 반대

$$L_1 = \overrightarrow{P_1 P_2}$$

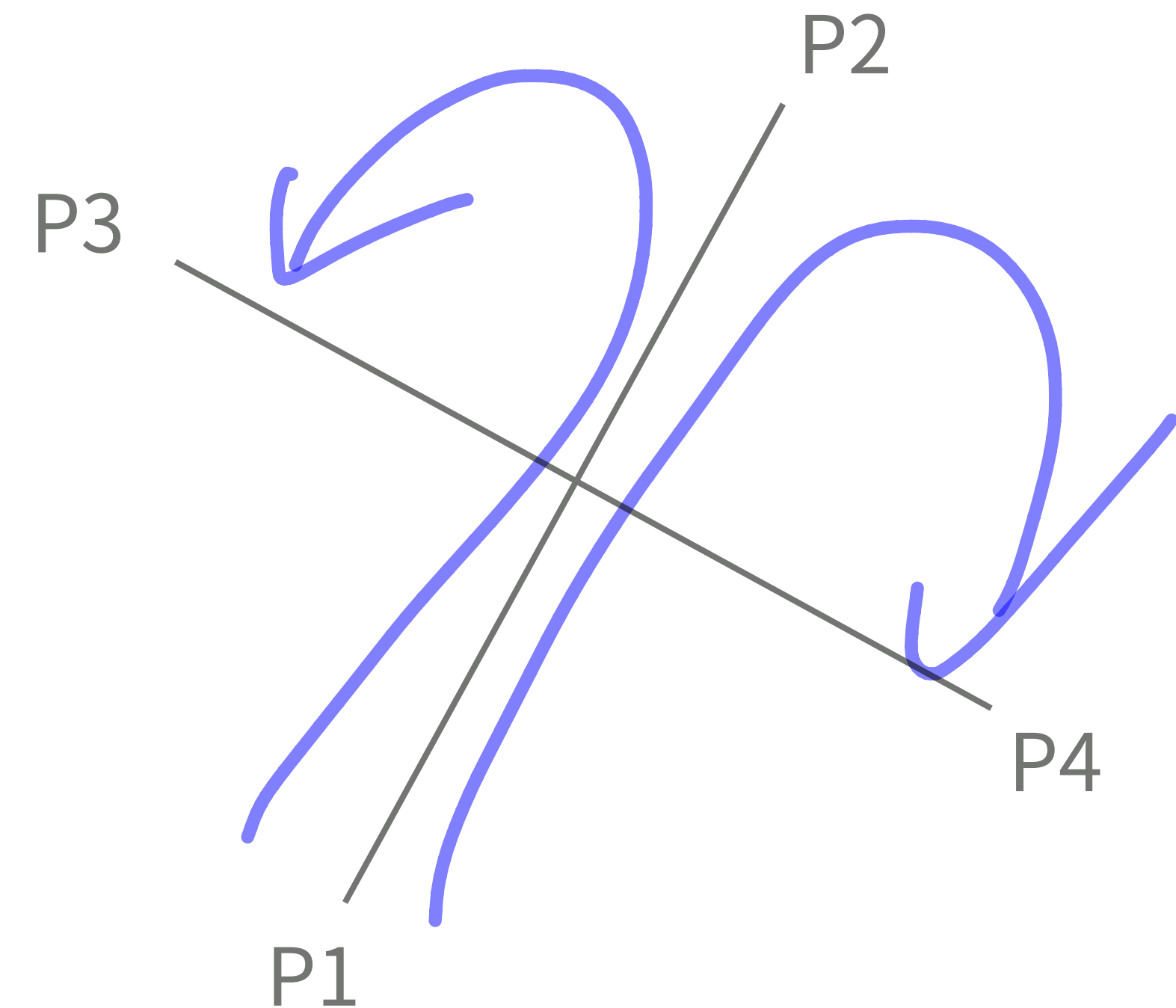
$$L_2 = \overrightarrow{P_3 P_4}$$



# 선분의 교차

## Intersection of Two Line Segments

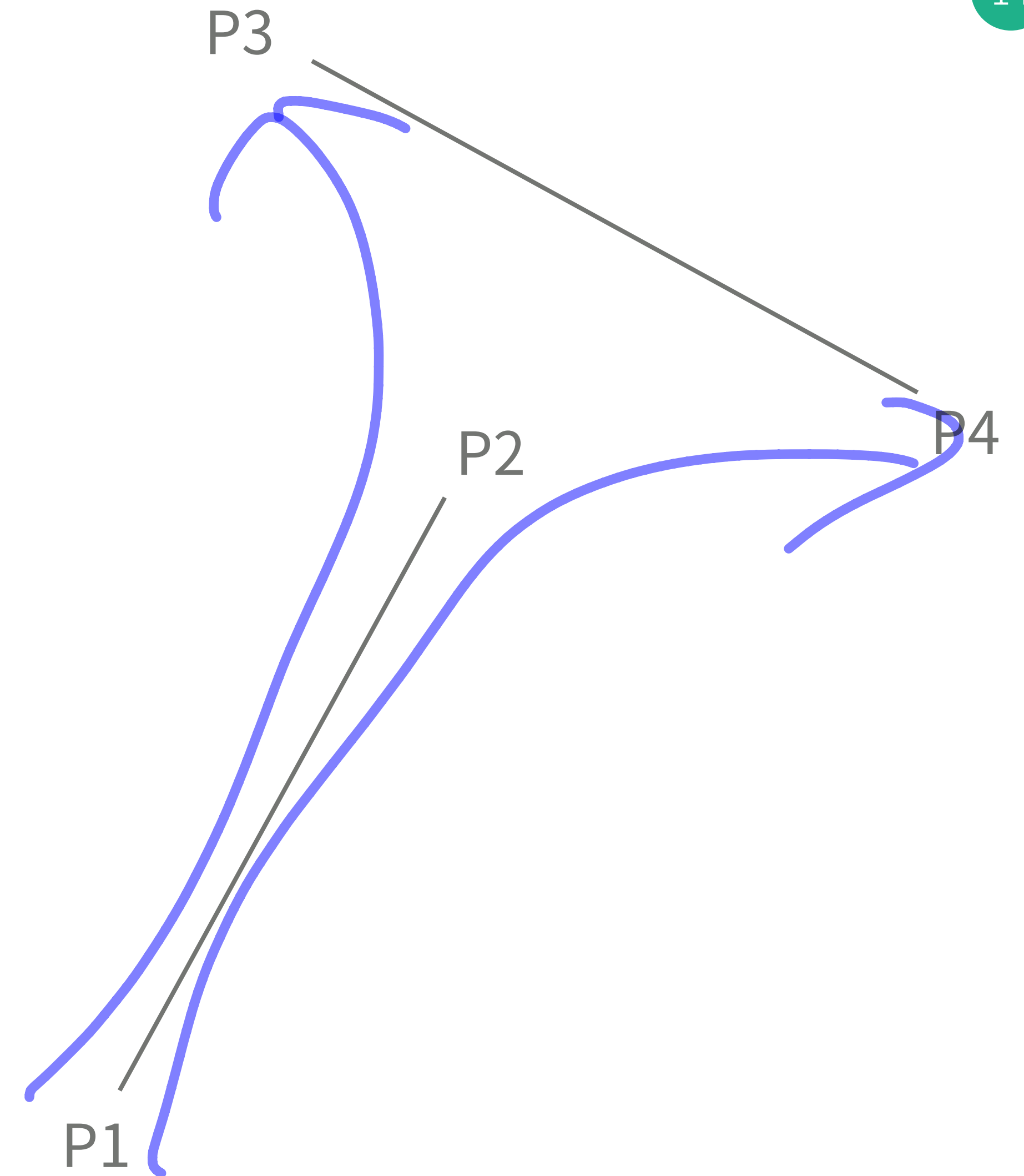
- 두 선분의 교차 확인
- P1-P2-P3의 방향과 P1-P2-P4의 방향이 반대
- 반례가 있다.



# 선분의 교차

Intersection of Two Line Segments

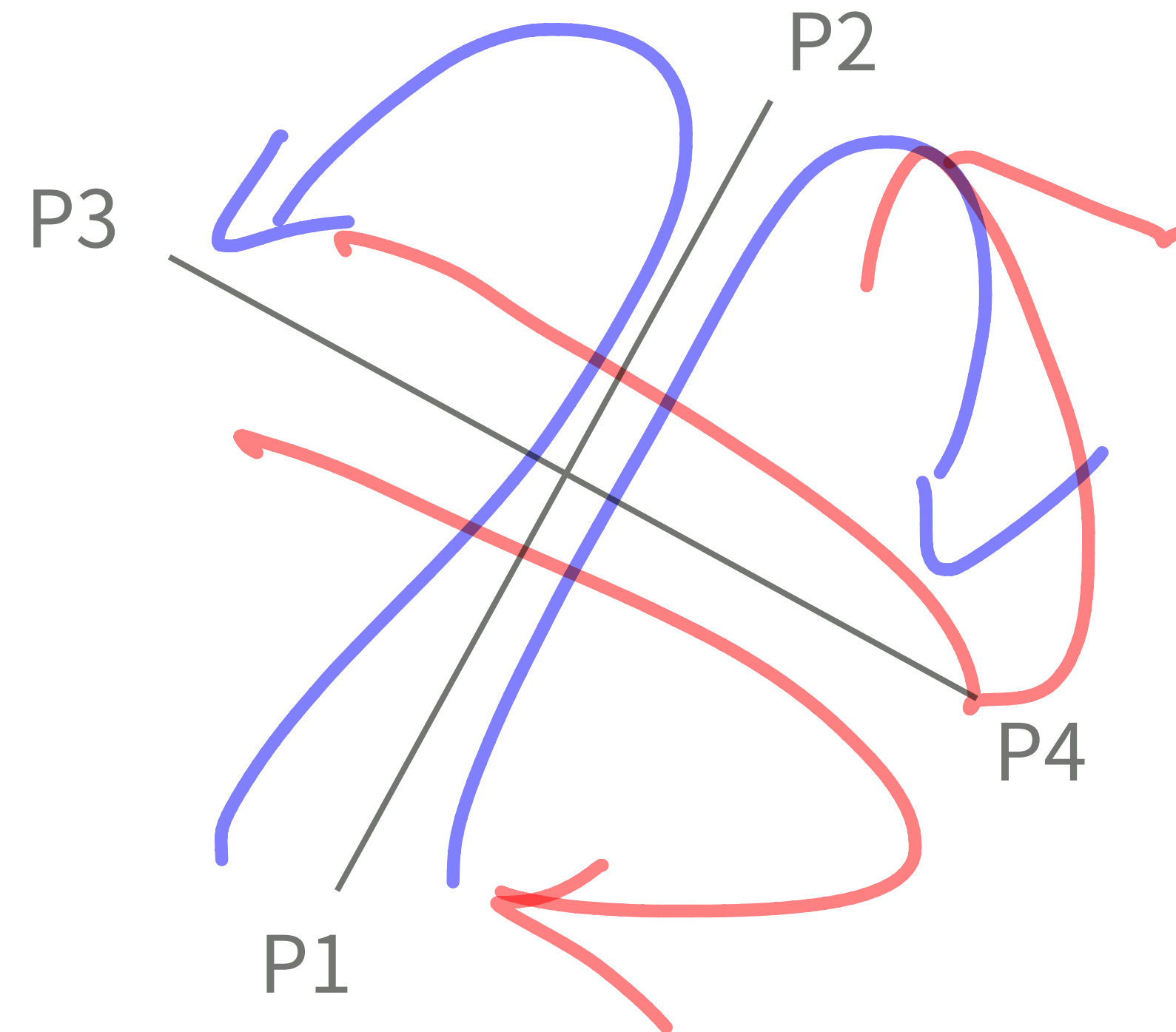
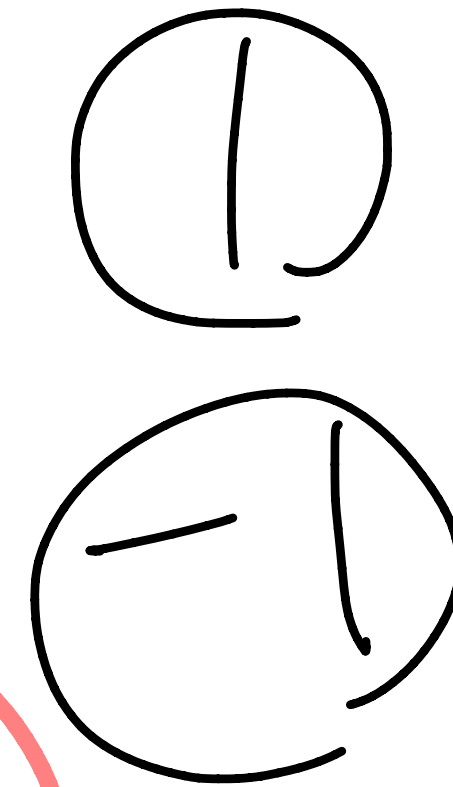
- 두 선분의 교차 확인
- P1-P2-P3의 방향과 P1-P2-P4의 방향이 달라야 함



# 선분의 교차

Intersection of Two Line Segments

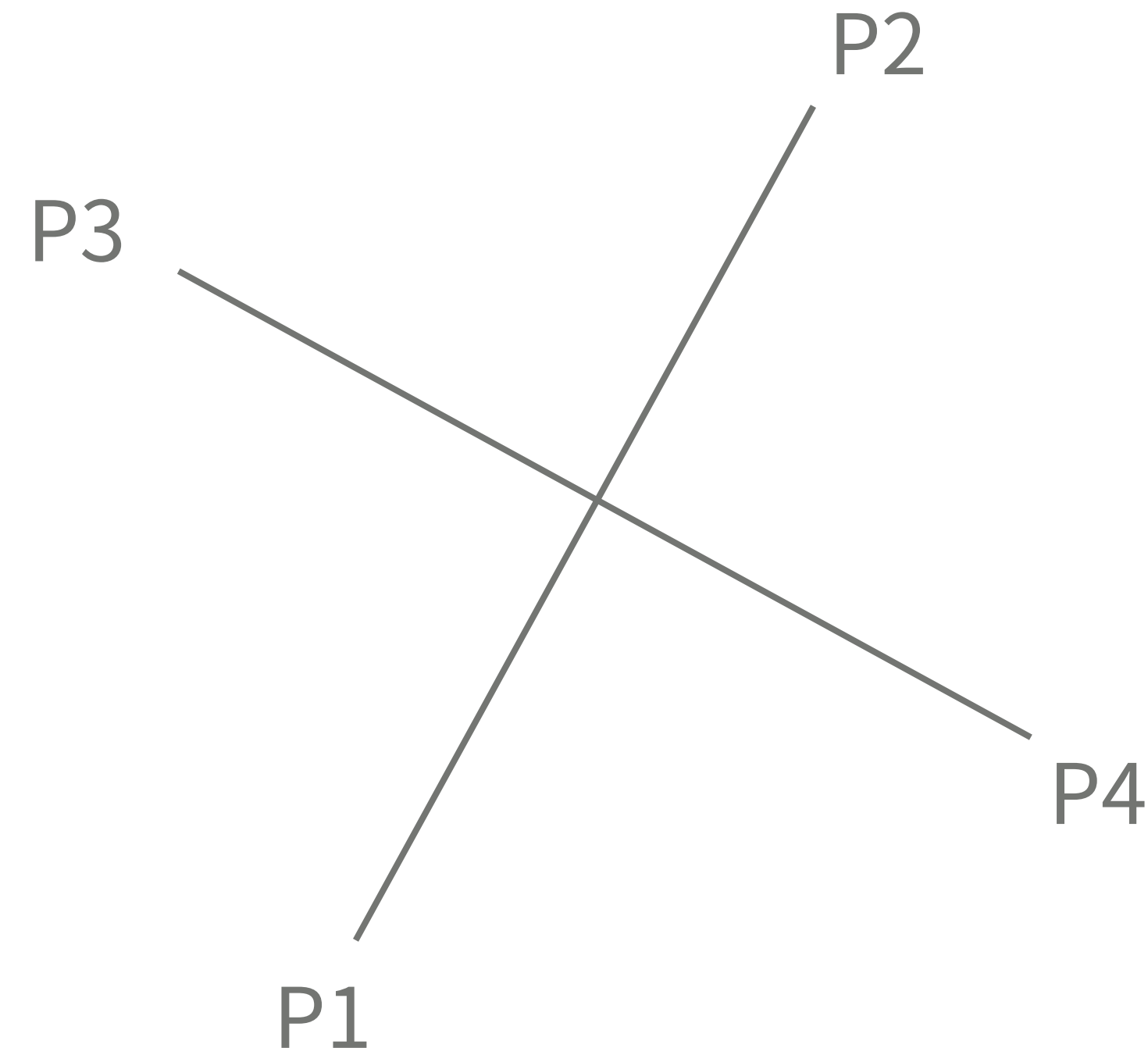
- 두 선분의 교차 확인
- P1-P2-P3의 방향과 P1-P2-P4의 방향이 반대
- P3-P4-P1의 방향과 P3-P4-P2의 방향도 반대



# 선분의 교차

## Intersection of Two Line Segments

- 두 선분의 교차 확인
- P1-P2-P3의 방향과 P1-P2-P4의 방향이 반대
- P3-P4-P1의 방향과 P3-P4-P2의 방향도 반대
- $CCW(P1, P2, P3) \times CCW(P1, P2, P4) < 0$
- $CCW(P3, P4, P1) \times CCW(P3, P4, P2) < 0$





# 선분 교차 1

<https://www.acmicpc.net/problem/17386>

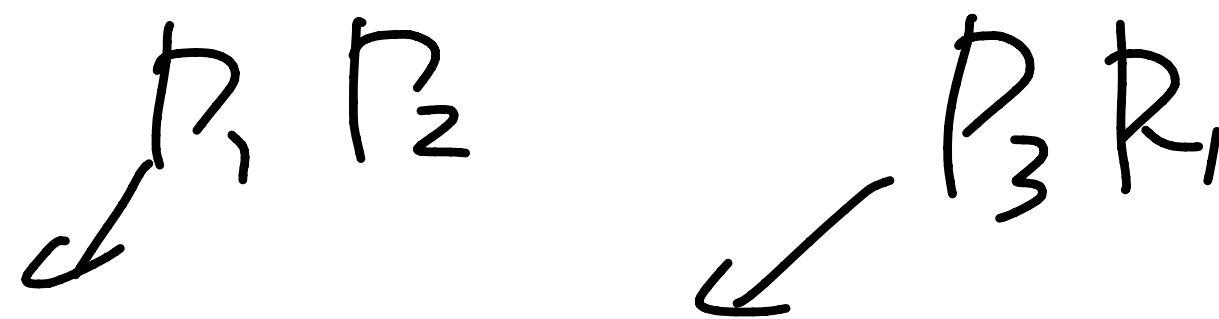
- 두 선분이 주어졌을 때, 교차하는지 아닌지 구하는 문제

• 세 점이 일직선 위에 있는 경우는 없다.

# 선분 교차 1

<https://www.acmicpc.net/problem/17386>

```
int ccw(Point p1, Point p2, Point p3) {  
    long long temp = p1.x * p2.y + p2.x * p3.y + p3.x * p1.y;  
    temp -= p1.y * p2.x + p2.y * p3.x + p3.y * p1.x;  
    if (temp > 0) return 1;  
    if (temp < 0) return -1;  
    return 0;  
}
```



```
bool cross(Line l1, Line l2) {
```

```
    int l1l2 = ccw(l1.first, l1.second, l2.first) * ccw(l1.first,  
l1.second, l2.second);
```

```
    int l2l1 = ccw(l2.first, l2.second, l1.first) * ccw(l2.first,  
l2.second, l1.second);
```

```
    return l1l2 < 0 && l2l1 < 0;
```

```
}
```

# 선분 교차 1

<https://www.acmicpc.net/problem/17386>

- 소스: <http://codeplus.codes/8f0c6386105b48e880b83a7762f56ba8>

# 선분 교차 2

<https://www.acmicpc.net/problem/17387>

- 두 선분이 주어졌을 때, 교차하는지 아닌지 구하는 문제
- 세 점이 일직선 위에 있을 수도 있다.

# 선분 교차 2

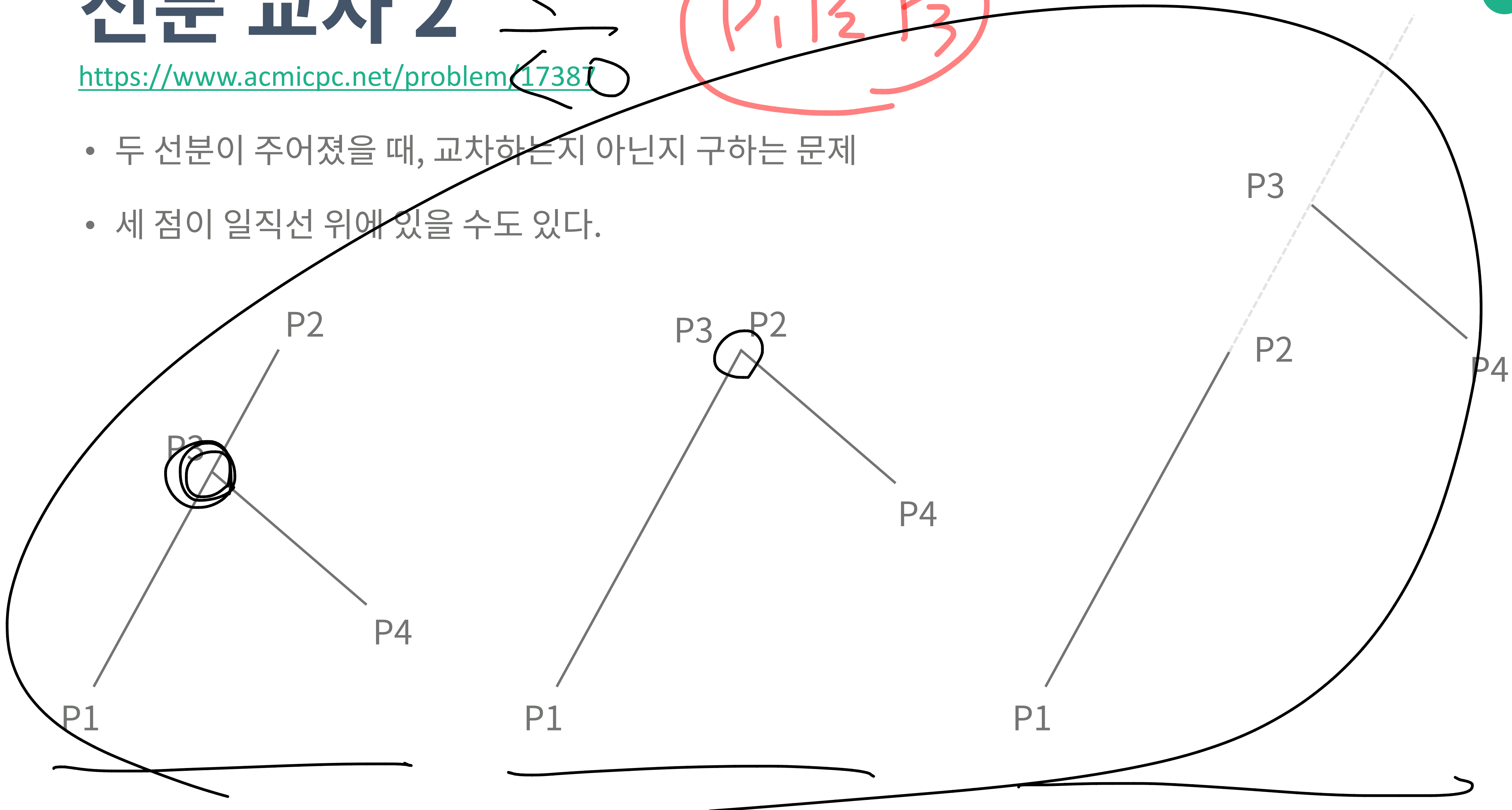
$\leq 0$   
 $\leq 0$

$P_1, P_2, P_3$

21

<https://www.acmicpc.net/problem/17387>

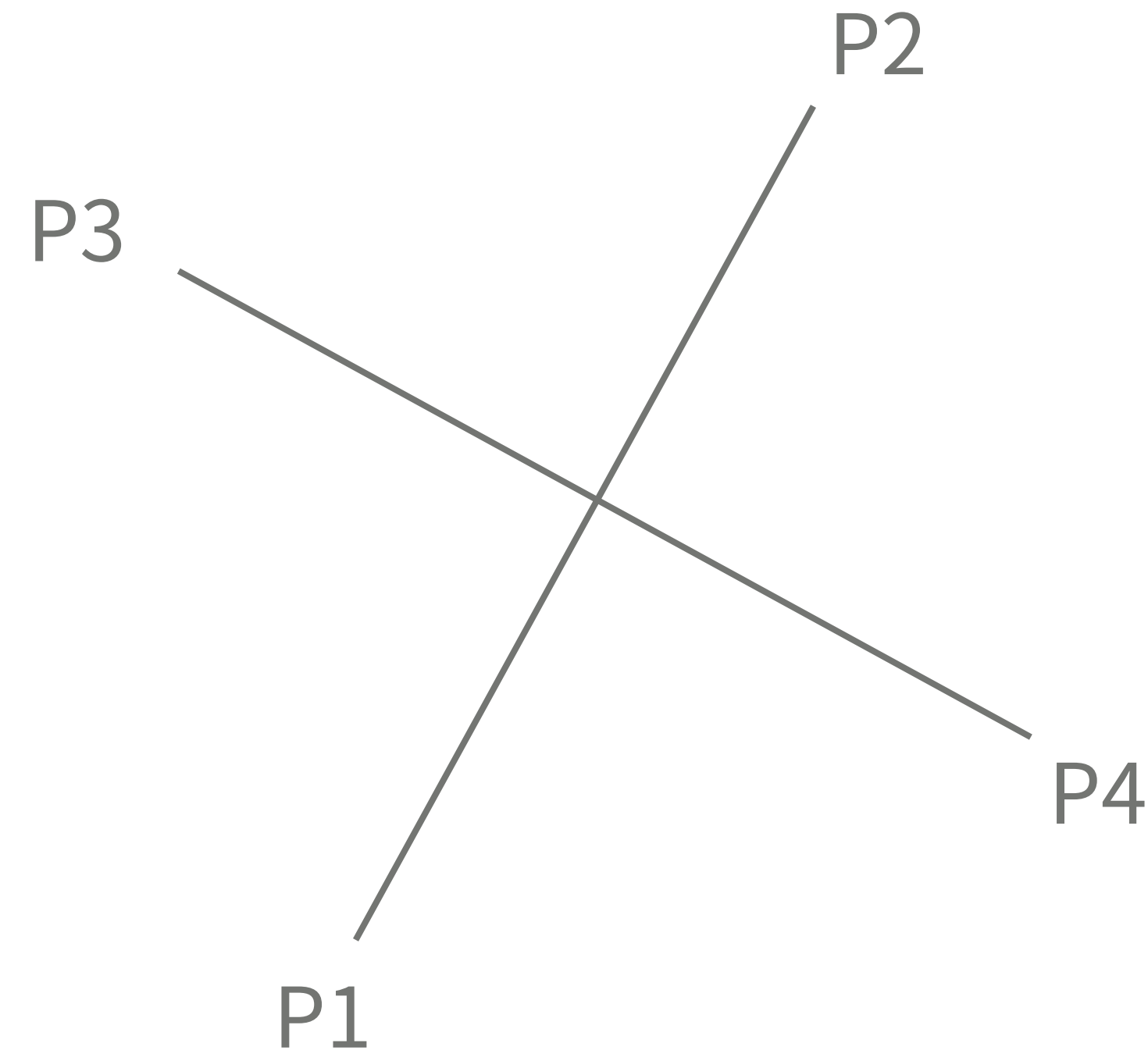
- 두 선분이 주어졌을 때, 교차하는지 아닌지 구하는 문제
- 세 점이 일직선 위에 있을 수도 있다.



# 선분의 교차

## Intersection of Two Line Segments

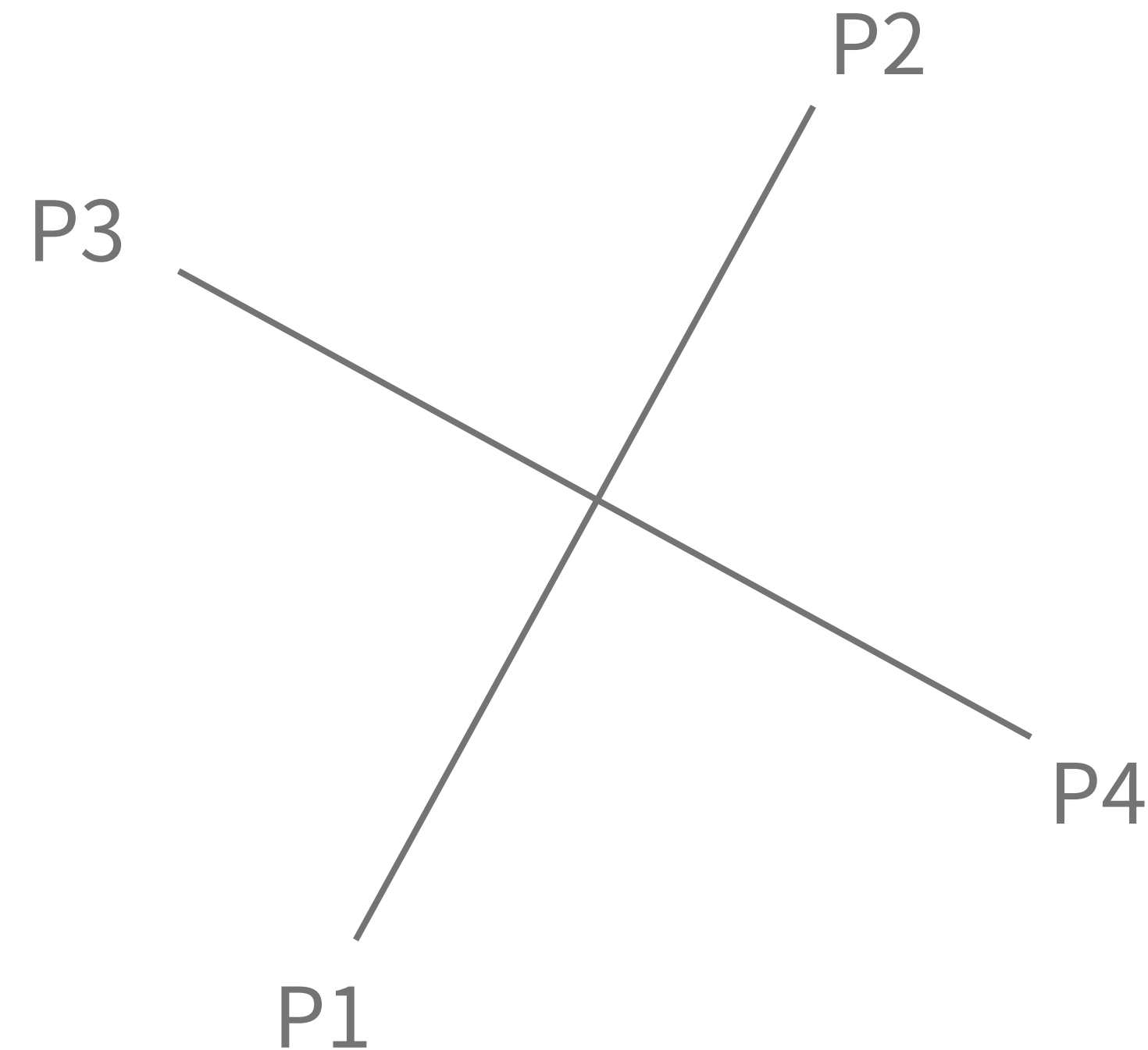
- 두 선분의 교차 확인
- P1-P2-P3의 방향과 P1-P2-P4의 방향이 반대
- P3-P4-P1의 방향과 P3-P4-P2의 방향도 반대
- $CCW(P1, P2, P3) \times CCW(P1, P2, P4) \leq 0$
- $CCW(P3, P4, P1) \times CCW(P3, P4, P2) \leq 0$



# 선분의 교차

## Intersection of Two Line Segments

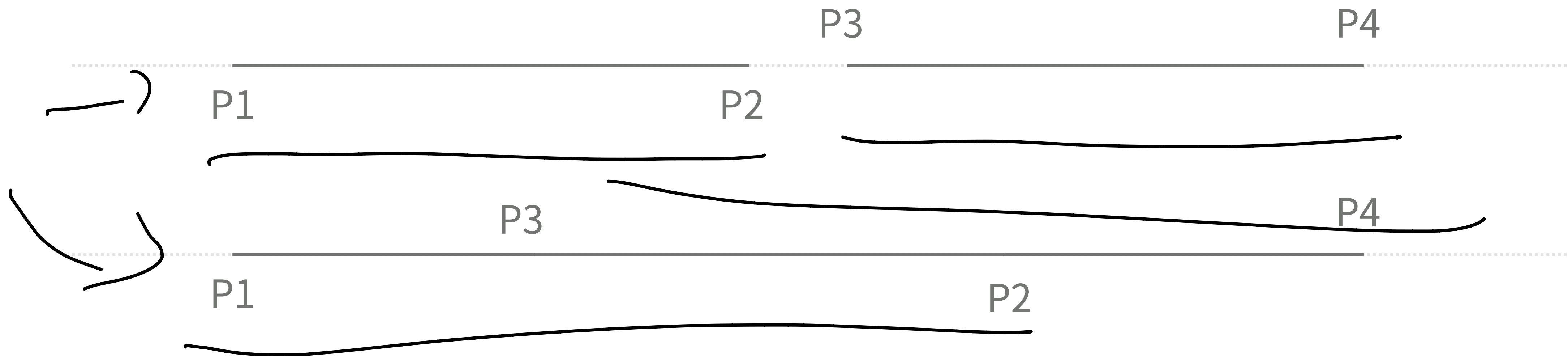
- 두 선분의 교차 확인
- $P1-P2-P3$ 의 방향과  $P1-P2-P4$ 의 방향이 반대
- $P3-P4-P1$ 의 방향과  $P3-P4-P2$ 의 방향도 반대
- $CCW(P1, P2, P3) \times CCW(P1, P2, P4) = 0$
- $CCW(P3, P4, P1) \times CCW(P3, P4, P2) = 0$
- 인 경우 반례가 있다.



# 선분의 교차

Intersection of Two Line Segments

- $CCW(P1, P2, P3) \times CCW(P1, P2, P4) = 0$
- $CCW(P3, P4, P1) \times CCW(P3, P4, P2) = 0$
- 인 경우 반례가 있다.

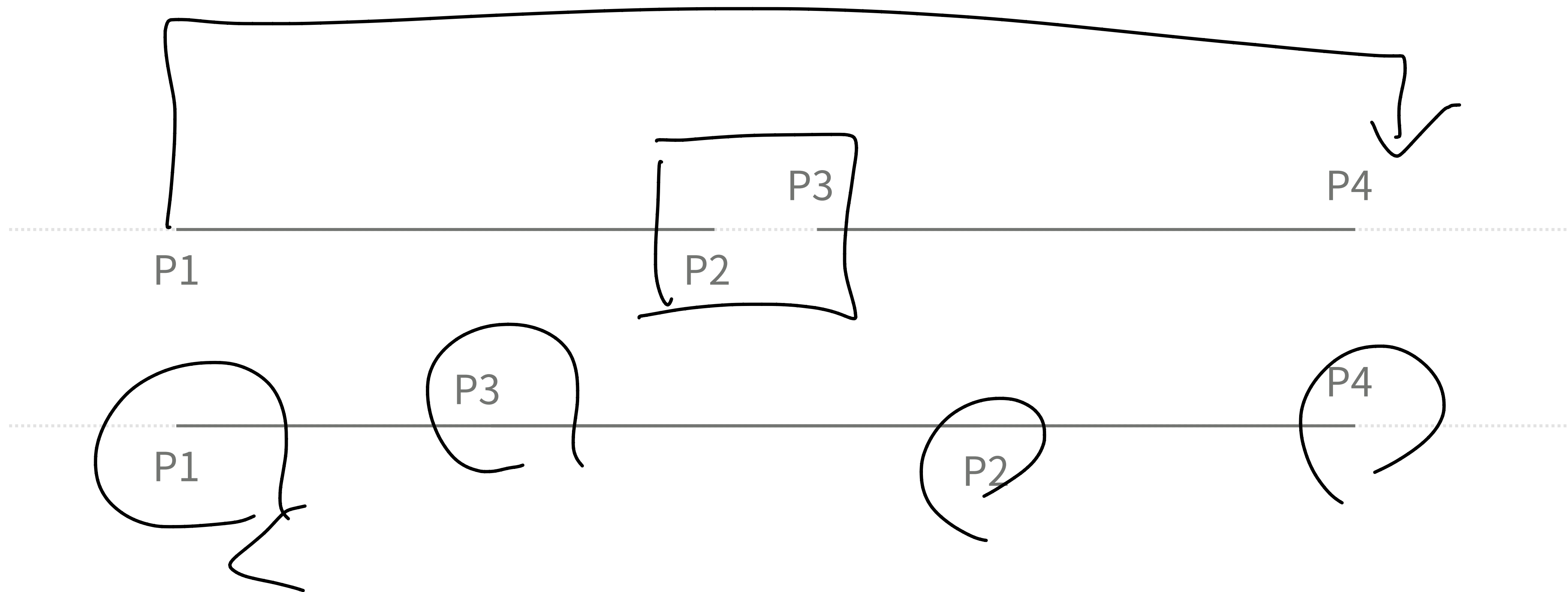




# 선분의 교차

## Intersection of Two Line Segments

- P2에 대한 P3의 위치와 P1에 대한 P4의 위치를 조사해야 한다.



# 선분 교차 2

<https://www.acmicpc.net/problem/17387>

```
bool cross(Line l1, Line l2) {  
    int l1l2 = ccw(l1.first, l1.second, l2.first) * ccw(l1.first,  
l1.second, l2.second);  
    int l2l1 = ccw(l2.first, l2.second, l1.first) * ccw(l2.first,  
l2.second, l1.second);  
    if (l1l2 == 0 && l2l1 == 0) {  
        if (l1.first > l1.second) swap(l1.first, l1.second);  
        if (l2.first > l2.second) swap(l2.first, l2.second);  
        return l2.first <= l1.second && l1.first <= l2.second;  
    }  
    return l1l2 <= 0 && l2l1 <= 0;  
}
```

$P_3$                    $P_2$                    $P_1$                    $P_4$

# 선분 교차 2

<https://www.acmicpc.net/problem/17387>

- 소스: <http://codeplus.codes/88e128a4137d47379abaf0369e7947fa>

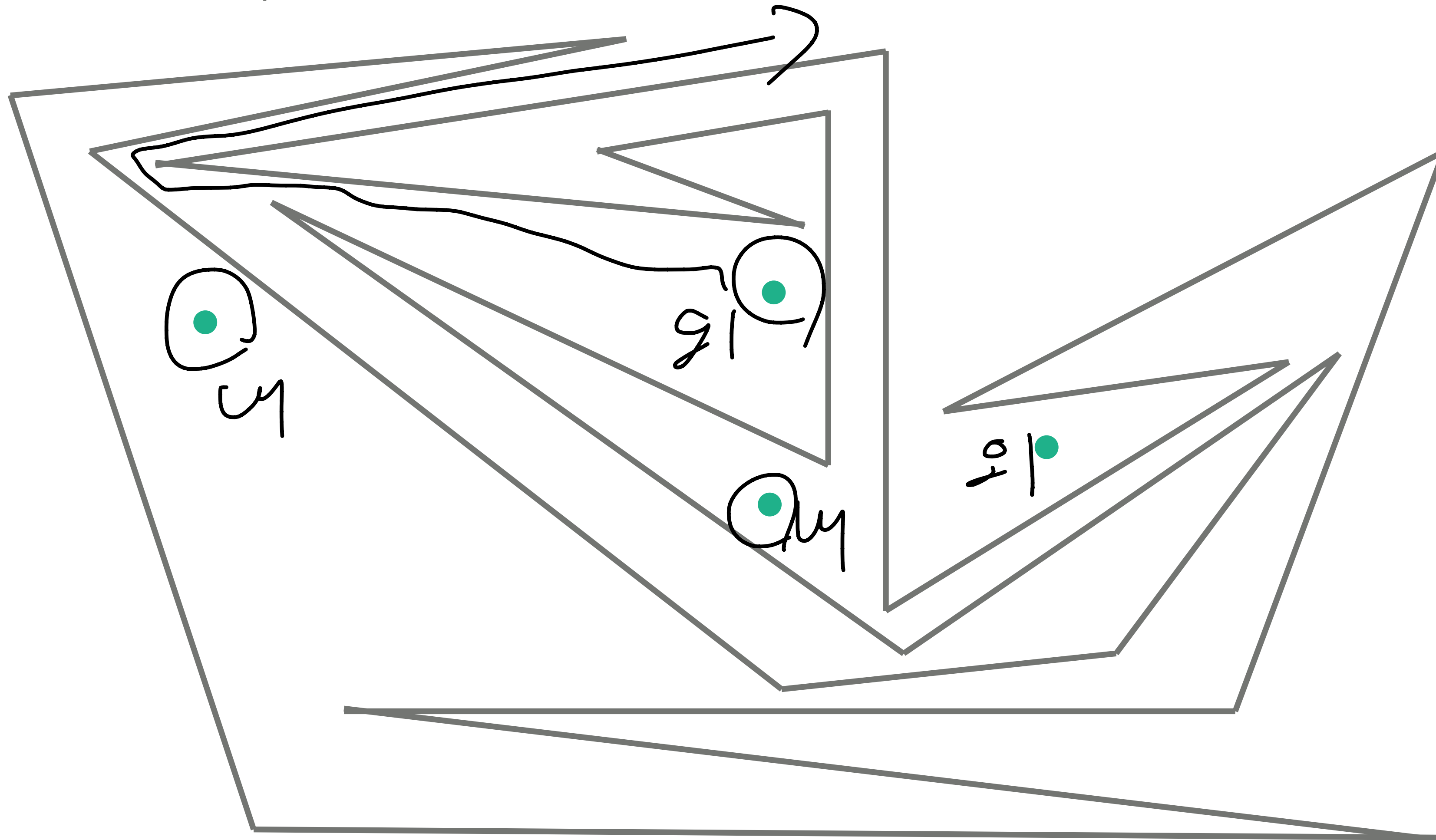
# 다각형의 내부/외부

---

# 다각형의 내부/외부

Polygon Inside, Outside

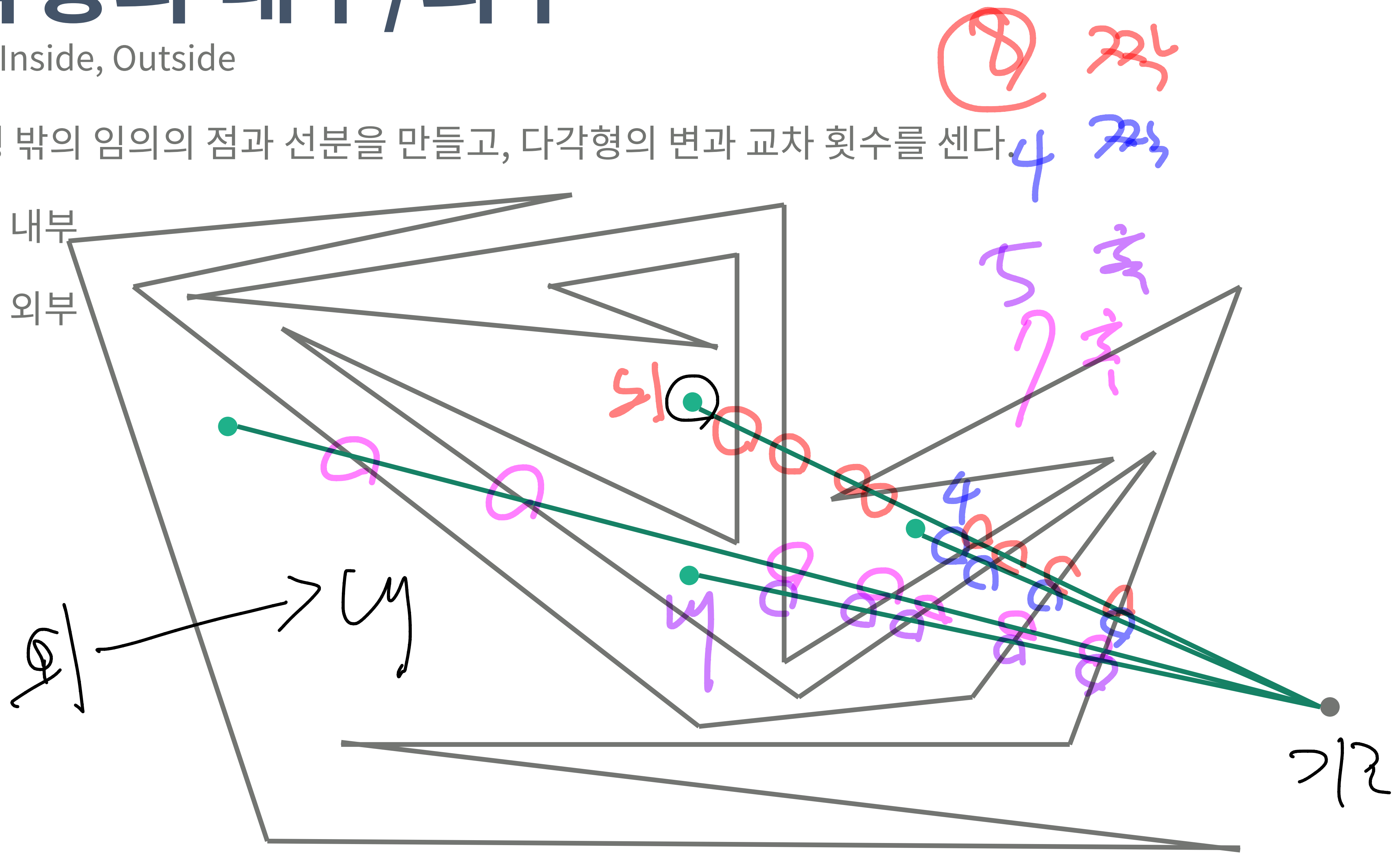
- 어떤 점이 다각형의 내부/외부에 있는지 알아보자



# 다각형의 내부/외부

Polygon Inside, Outside

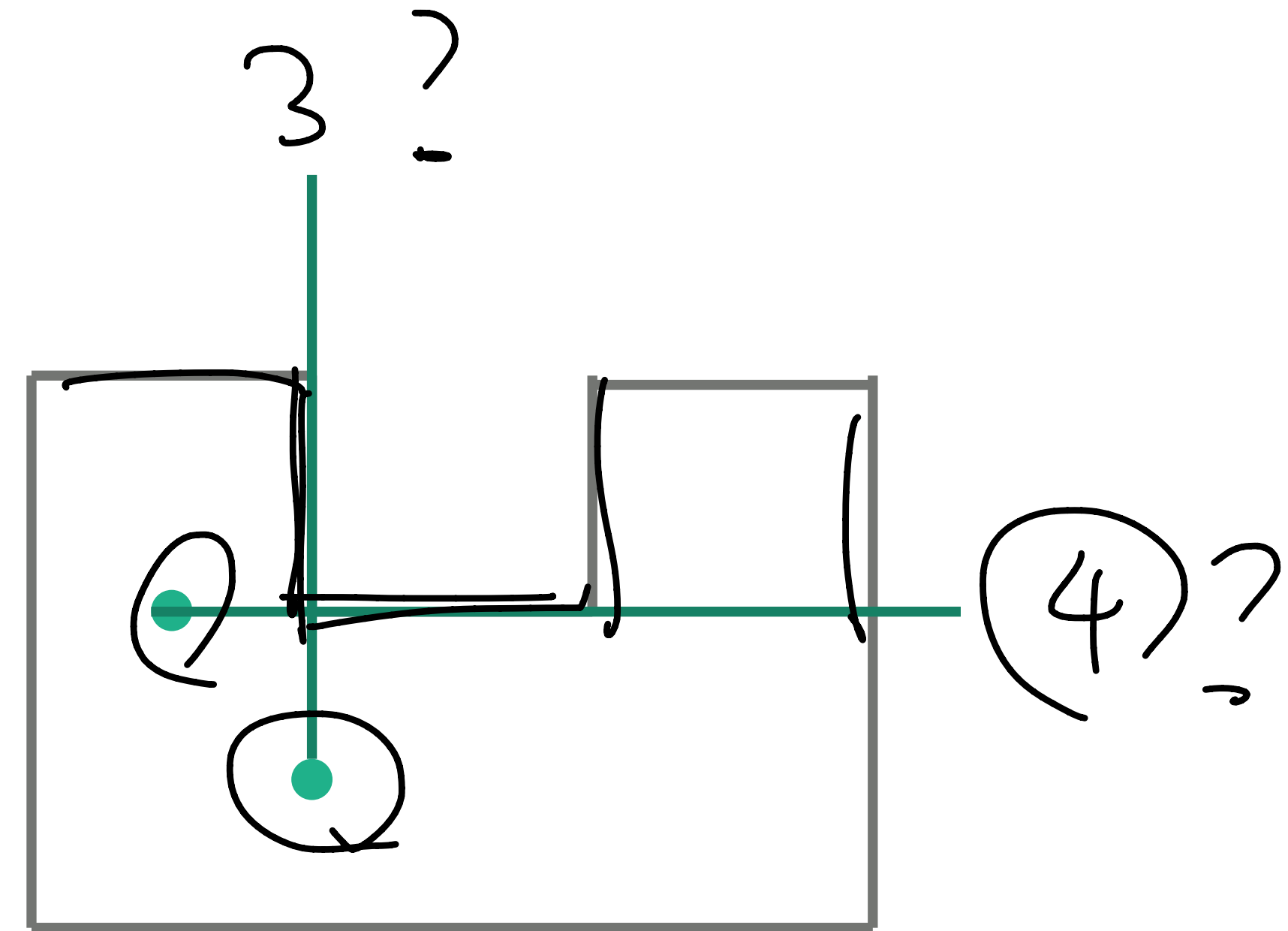
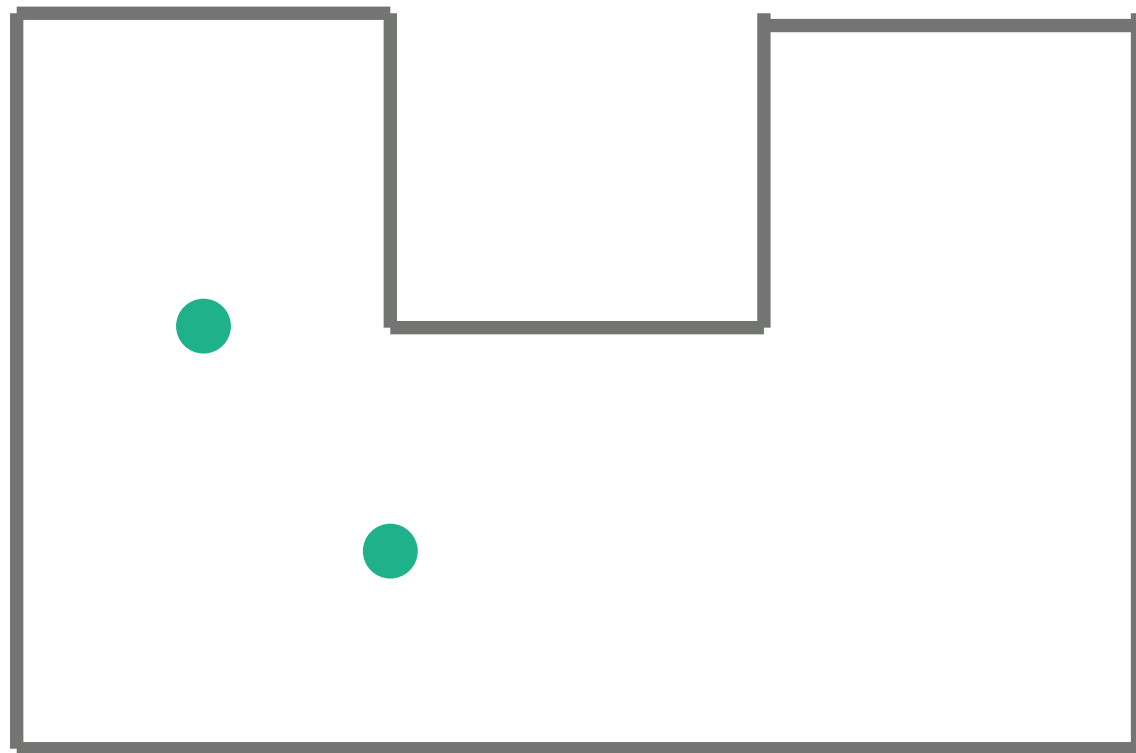
- 다각형 밖의 임의의 점과 선분을 만들고, 다각형의 변과 교차 횟수를 센다.
- 홀수 = 내부
- 짝수 = 외부



# 다각형의 내부/외부

Polygon Inside, Outside

- 홀수/짝수를 계산하기 어려운 경우도 있다.



# 다각형의 내부/외부

Polygon Inside, Outside

- 다각형 외부의 점과 연결한 선분이 다각형의 변과 일치하는 경우가 없게 외부 점을 정한다.



# 지민이의 테러

<https://www.acmicpc.net/problem/1688>

- 어떤 점이 다각형의 내부/외부인지 알아내는 문제

# 지민이의 테러

<https://www.acmicpc.net/problem/1688>

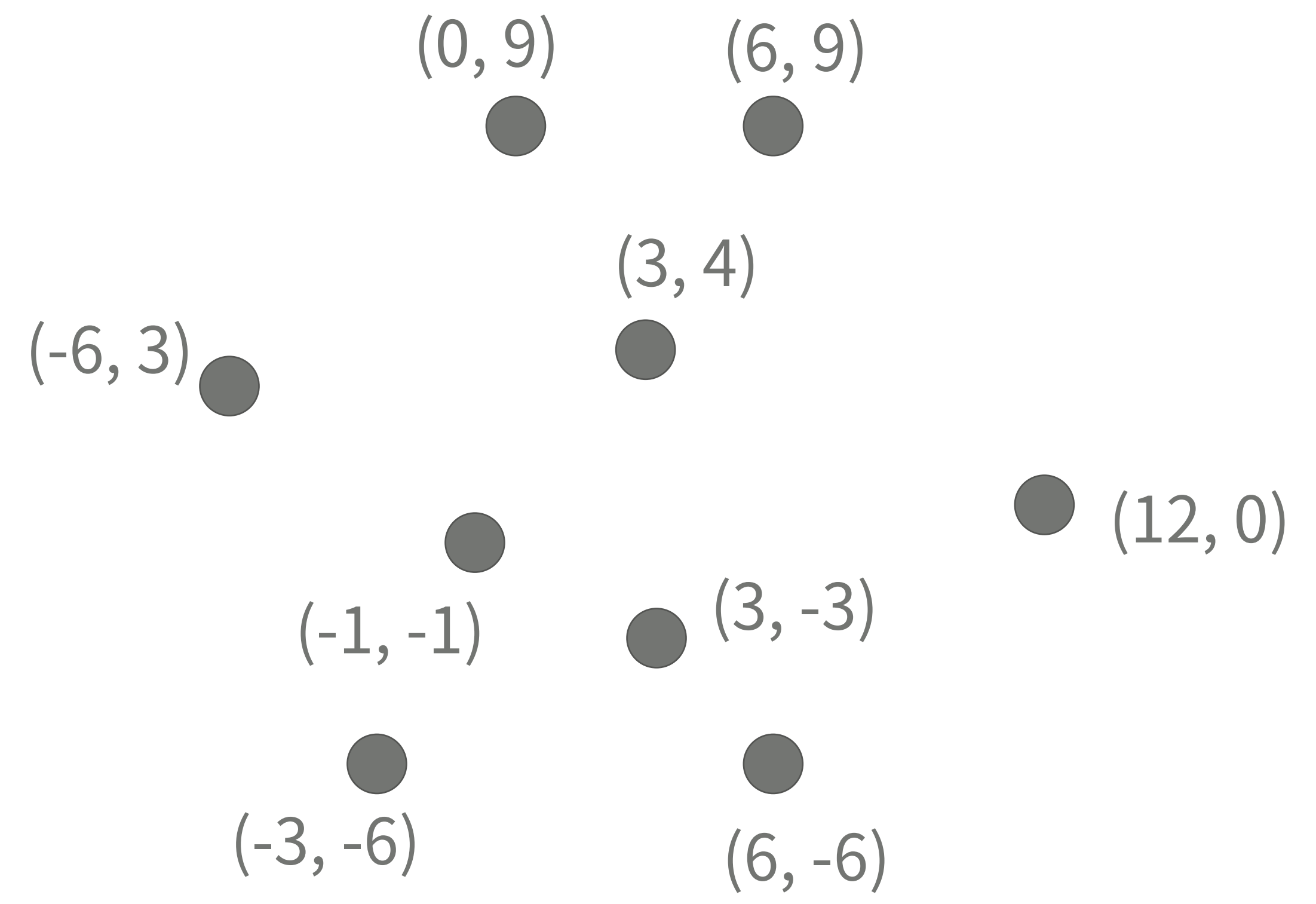
- 소스: <http://codeplus.codes/5212f2b599aa4d81a076650e1d039ab1>

# 볼록 껍질

# 볼록 껍질

Convex Hull

- 다각형을 감싸는 가장 작은 볼록 다각형



# 볼록 껍질

Convex Hull

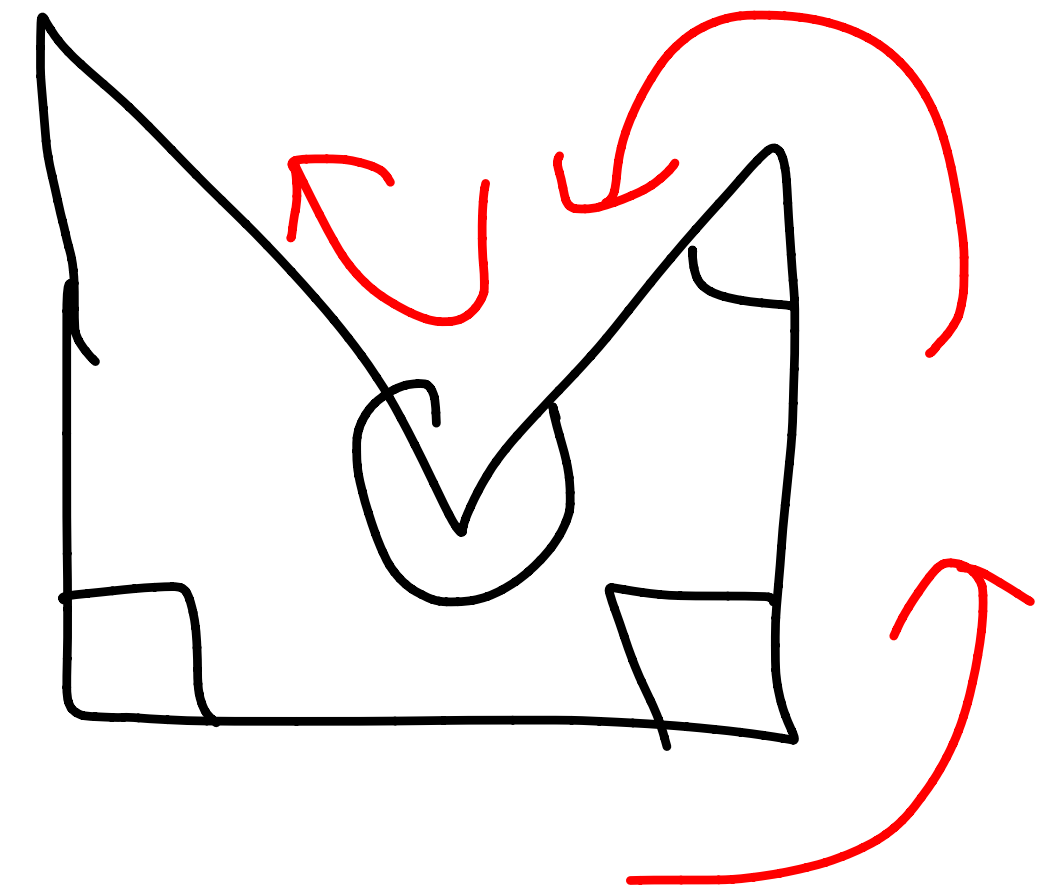
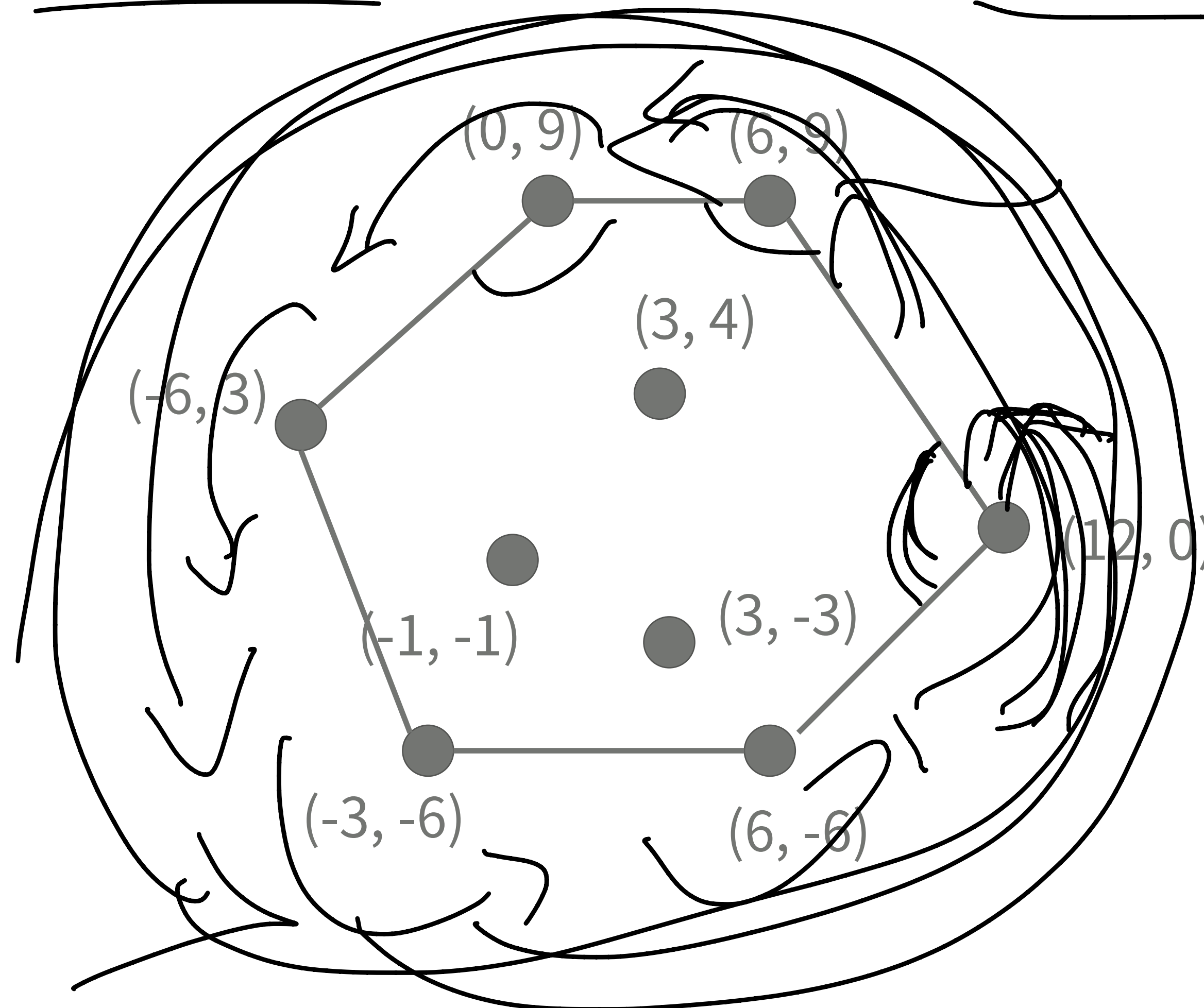
37

- 다각형을 감싸는 가장 작은 볼록 다각형

볼록 다각형

내각  $< 180^\circ$

ㄴ

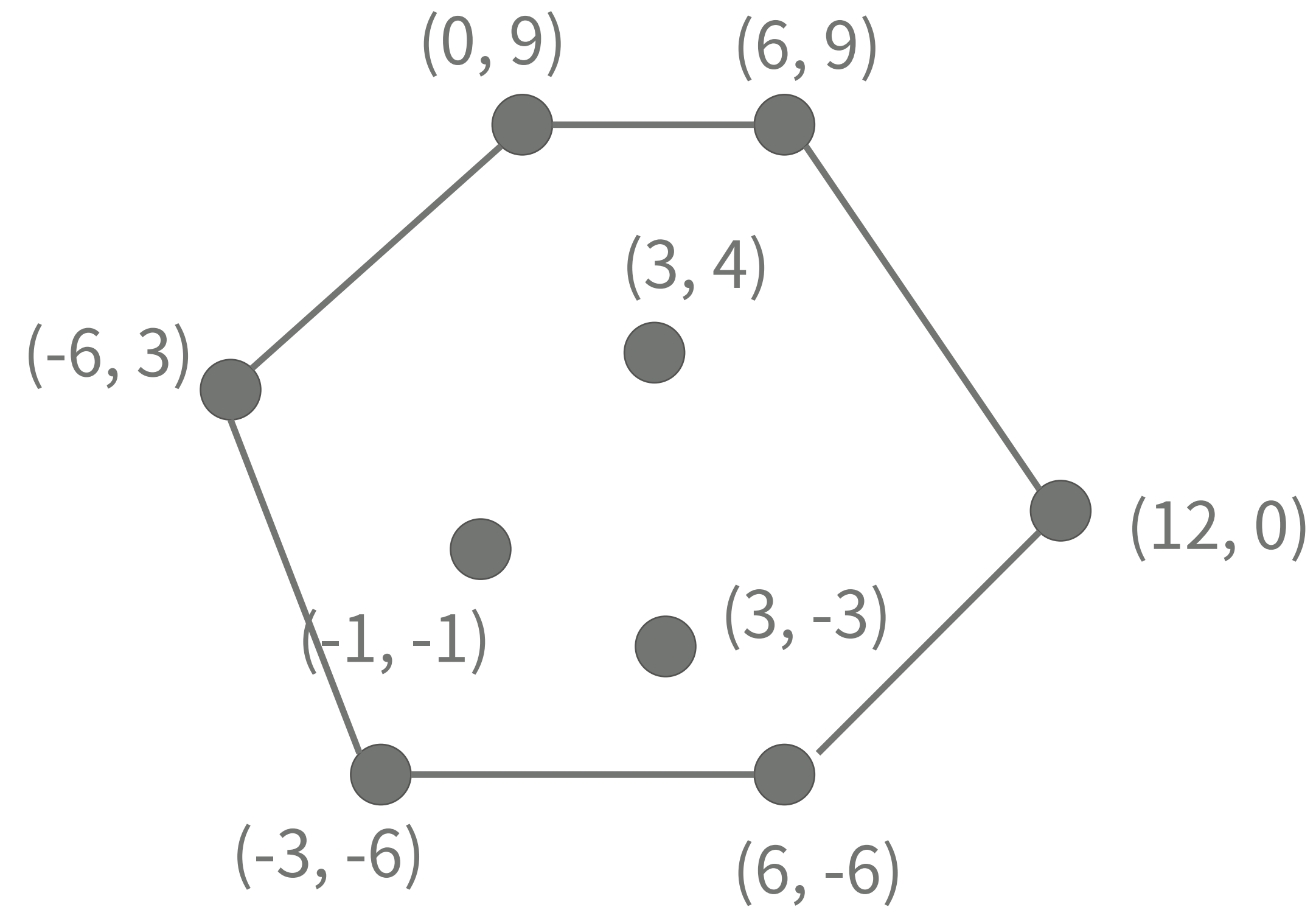


# 볼록 껍질

Convex Hull

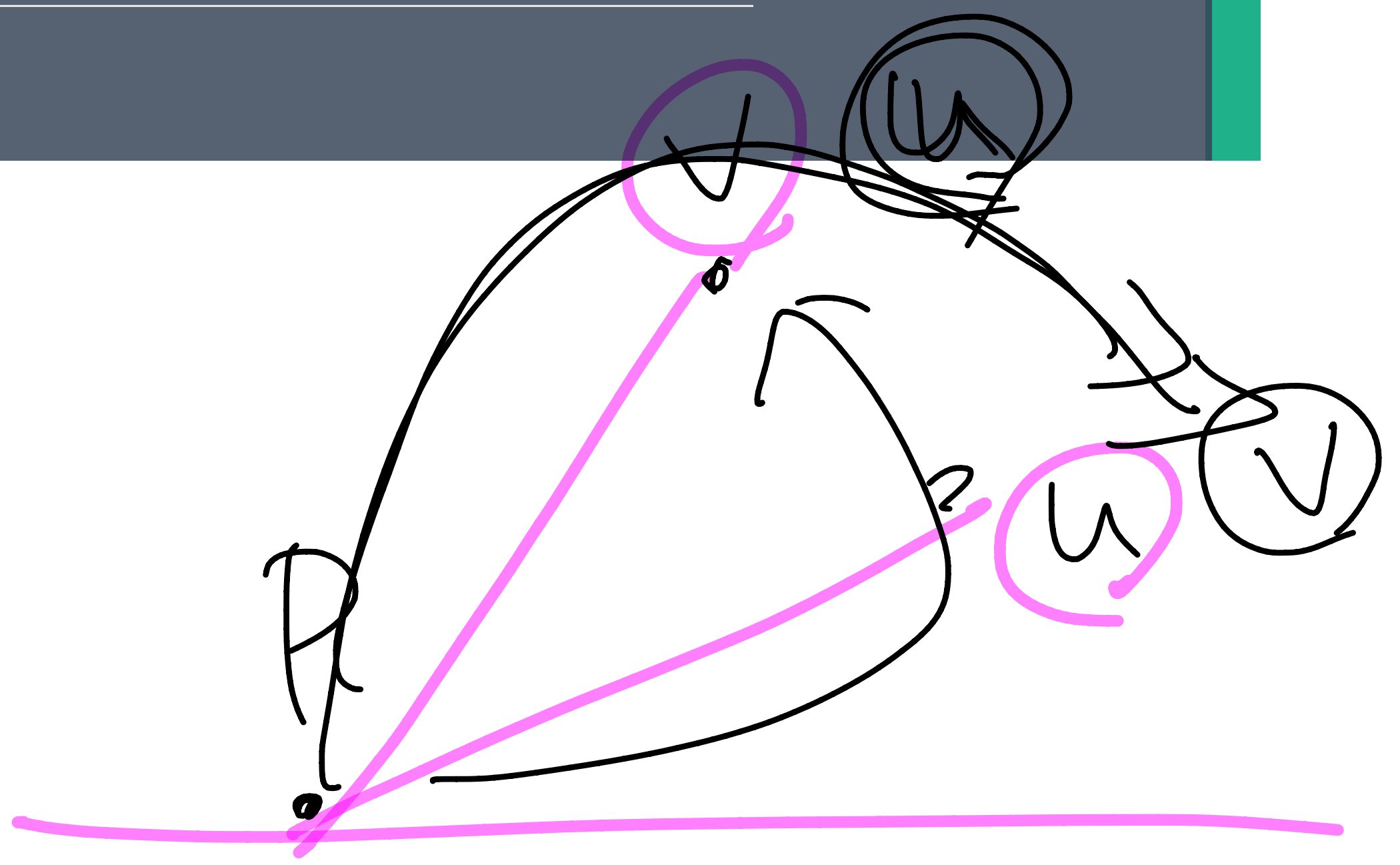
38

- 그라함 스캔을 이용해서 볼록 껍질을 구한다



$P$   $u$   $H(2)$   
 $P_{u,v}$   
 $u$ 가  $v$ 보다  $C$ 가  
크다/작다

# 그라함 스캔

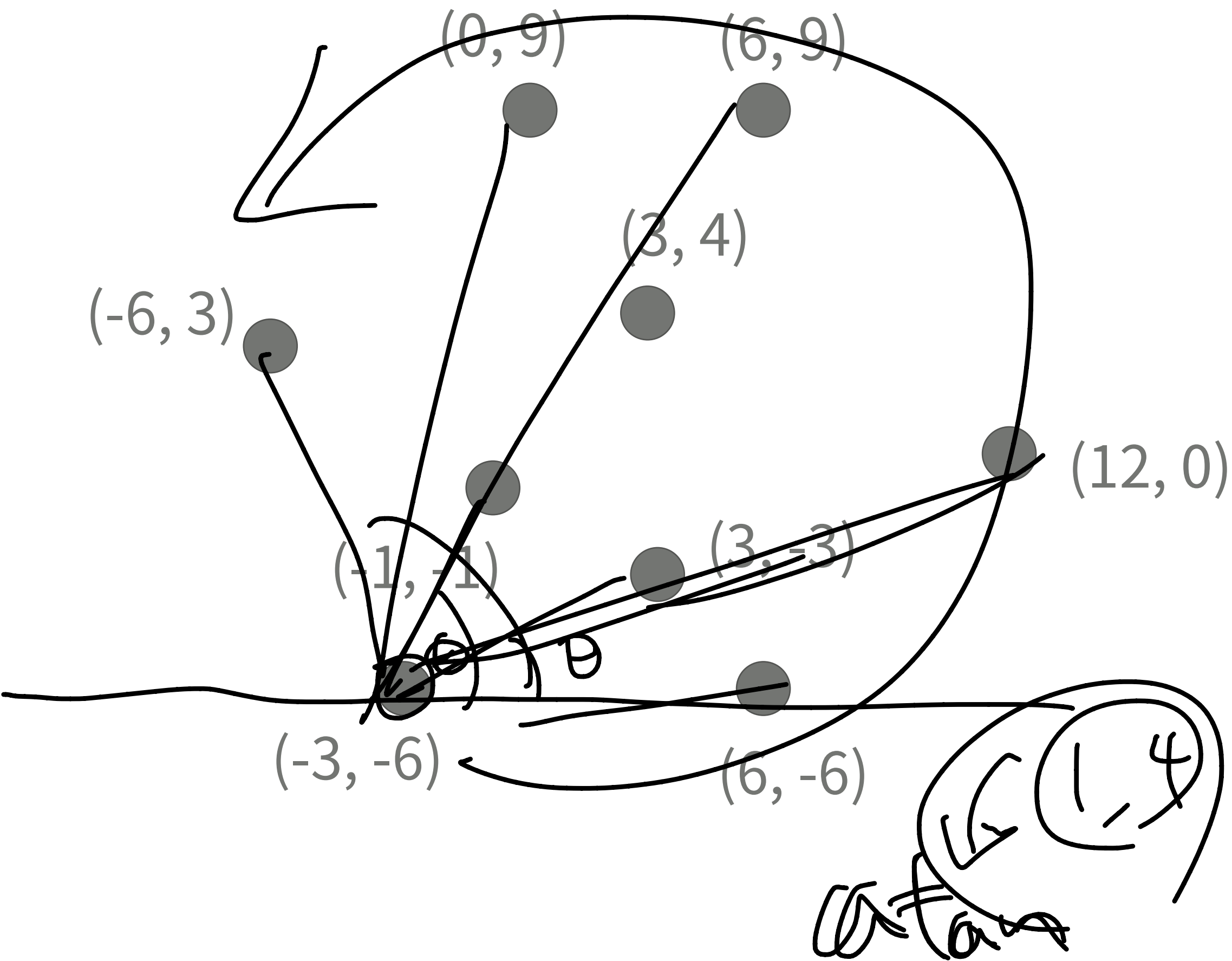


$Cmp(u, v)$   
 $u$ 가  $v$ 보다 앞  
true  
false  
-1  
0  
1

# 볼록 껍질

Convex Hull

- 가장 왼쪽 아래 점을 기준으로 각도순 정렬한다 (CCW 이용)



$(x_1, y_1)$

$(x_2, y_2)$

$$x = x_2 - x_1$$

$$y = y_2 - y_1$$



$$\tan \theta = \frac{y}{x}$$

$$\theta = \tan^{-1} \frac{y}{x}$$

1. -3 -6

2. 6 -6

3. 12 0

4. 3 -3

5. 3 4

6. 6 9

7. -1 -1

8. 0 9

9. -6 3

$\tan 2$

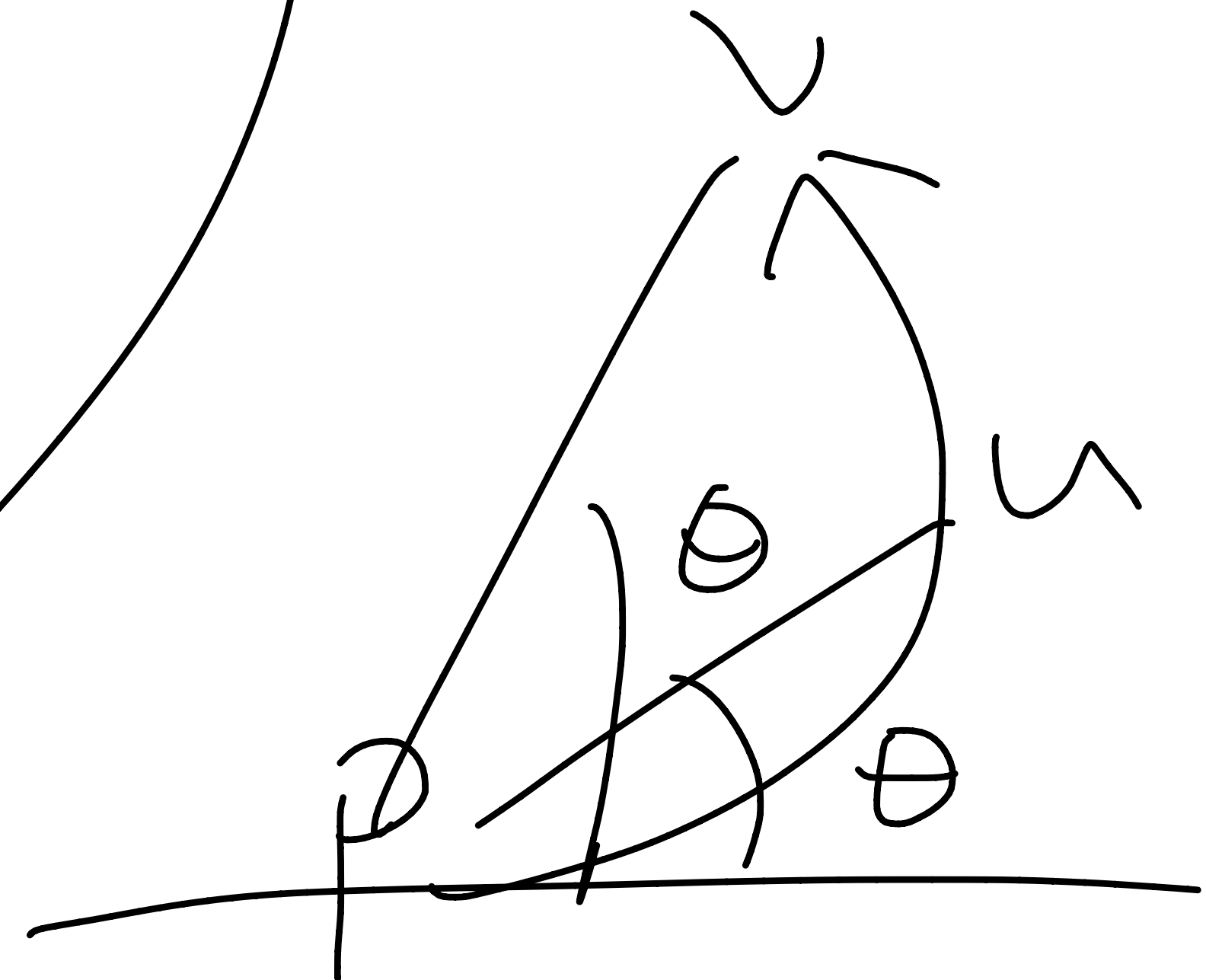


# 볼록 껍질

Convex Hull

```
bool cmp(const Point &u, const Point &v) {  
    int temp = ccw(p, u, v);  
    if (temp == 0) {  
        return dist(p, u) <= dist(p, v);  
    } else {  
        return temp == 1;  
    }  
}
```

(1, -1) 방향  
반시계 방향



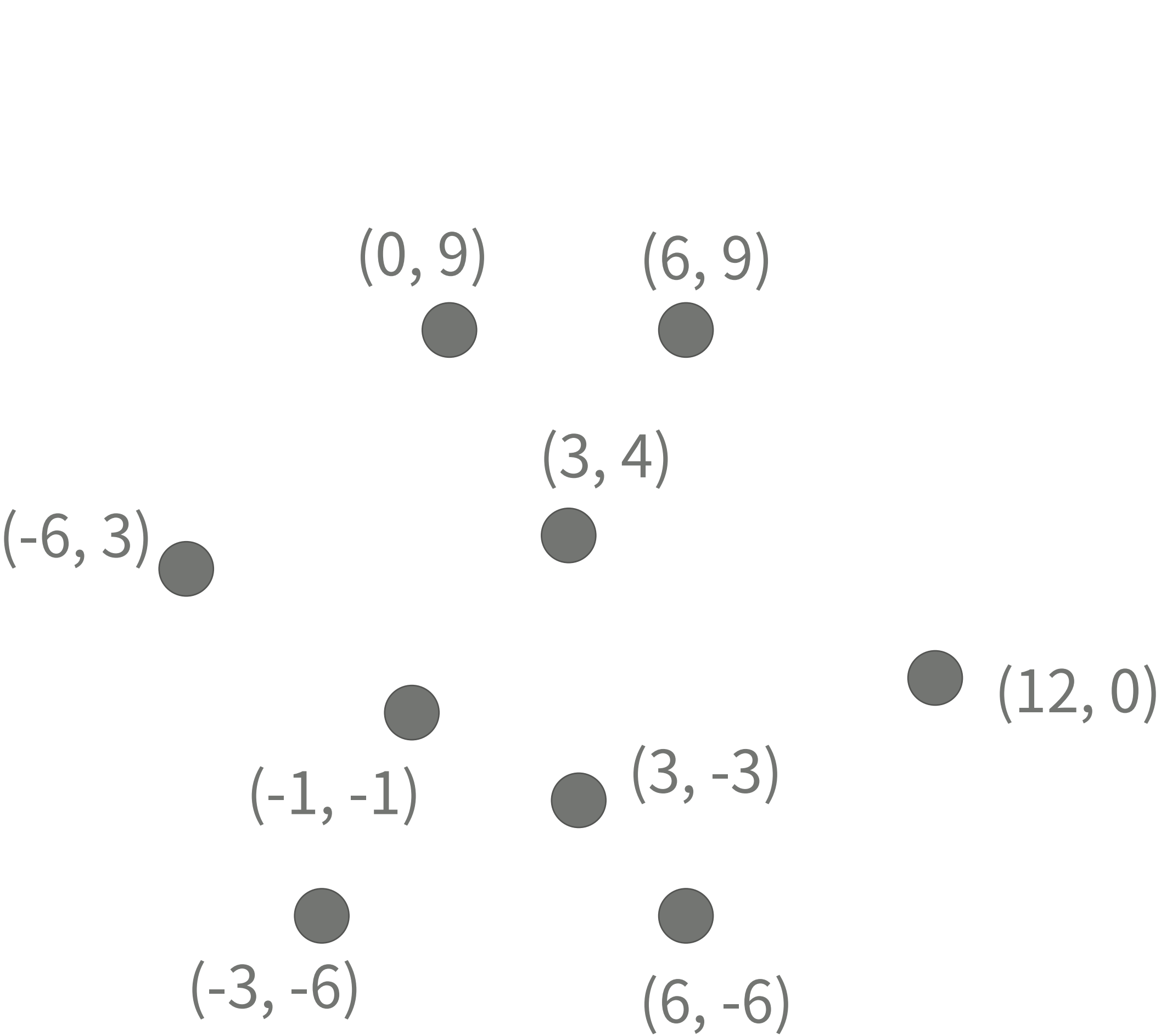
# 볼록 껍질

## Convex Hull

- 이제 새로운 점  $p$ 를 스택에 넣을 때, 다음을 검사해야 한다
- $(\text{stack}[s-2], \text{stack}[s-1], p)$  가 반시계 방향이 될 때 까지  $\text{stack}$ 에서  $\text{pop}$ 을 수행한다
- 그 다음  $p$ 를 스택에 넣는다

# 볼록 껍질

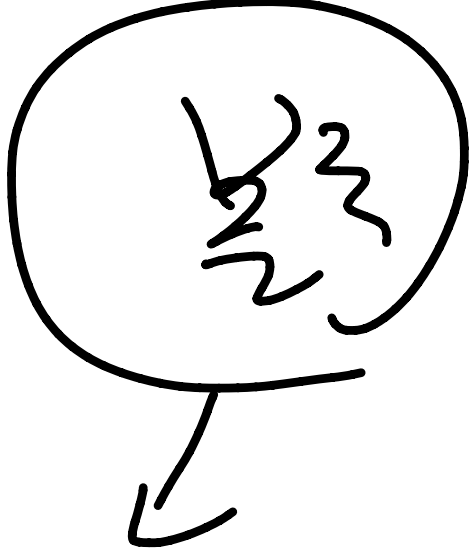
Convex Hull



스택	
1.	-3 -6
2.	6 -6
3.	12 0
4.	3 -3
5.	3 4
6.	6 9
7.	-1 -1
8.	0 9
9.	-6 3

# 볼록 껍질

Convex Hull



스택

1. -3 -6 → -3 -6

2. 6 -6 → 6 -6

3. 12 0

4. 3 -3

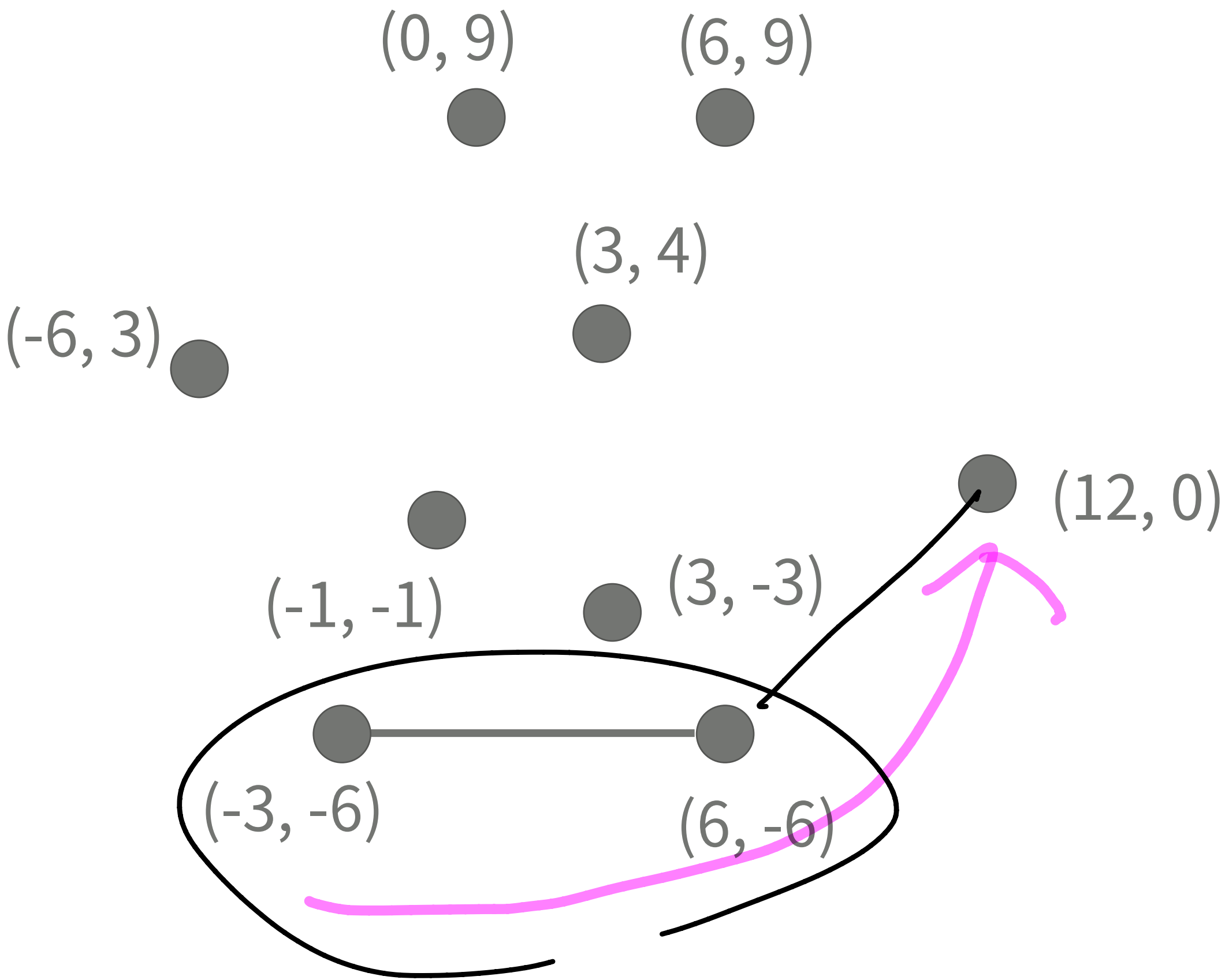
5. 3 4

6. 6 9

7. -1 -1

8. 0 9

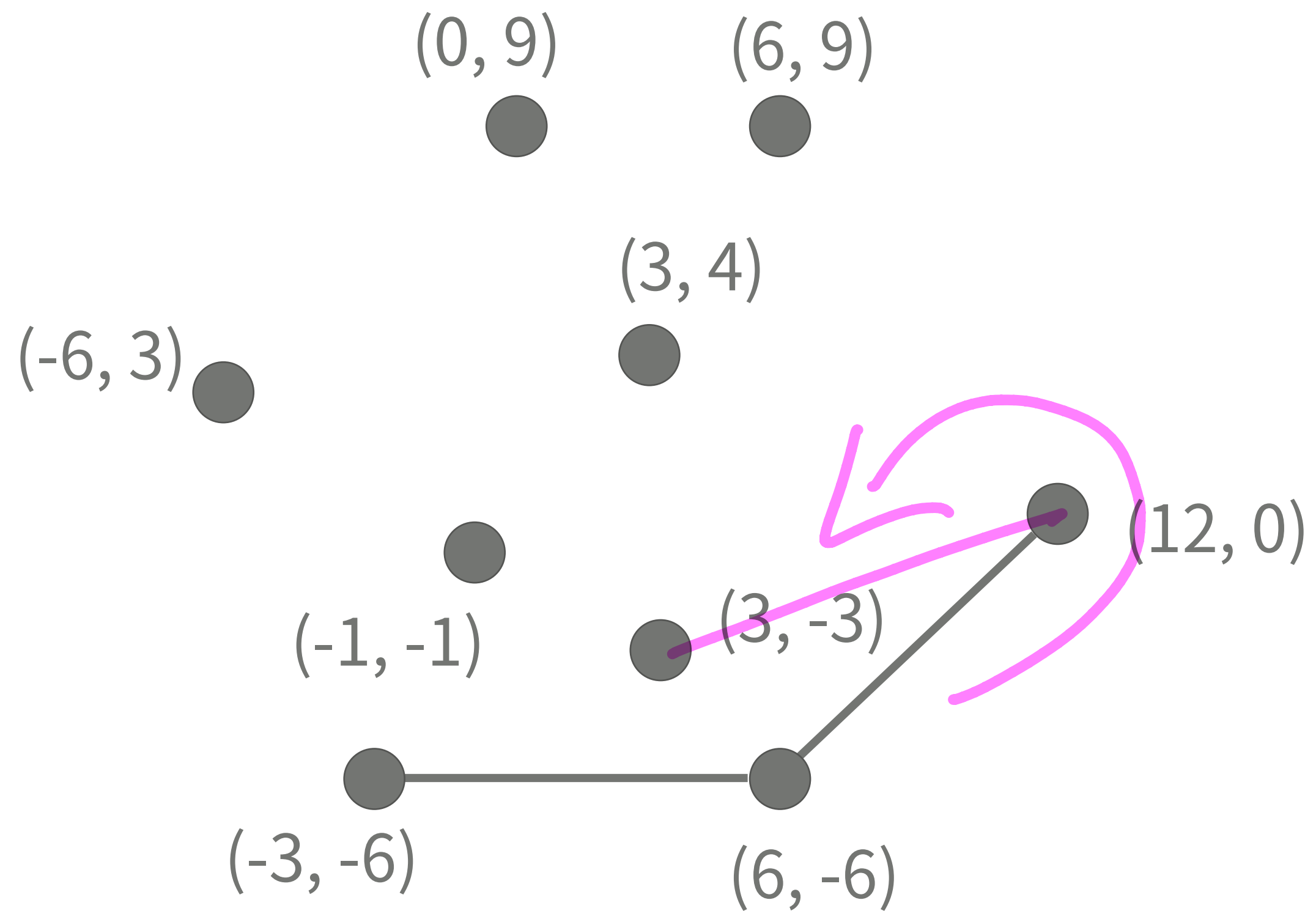
9. -6 3



# 볼록 껍질

Convex Hull

45



스택

1. -3 -6

-3 -6

2. 6 -6

6 -6

3. 12 0

12 0

4. 3 -3

5. 3 4

6. 6 9

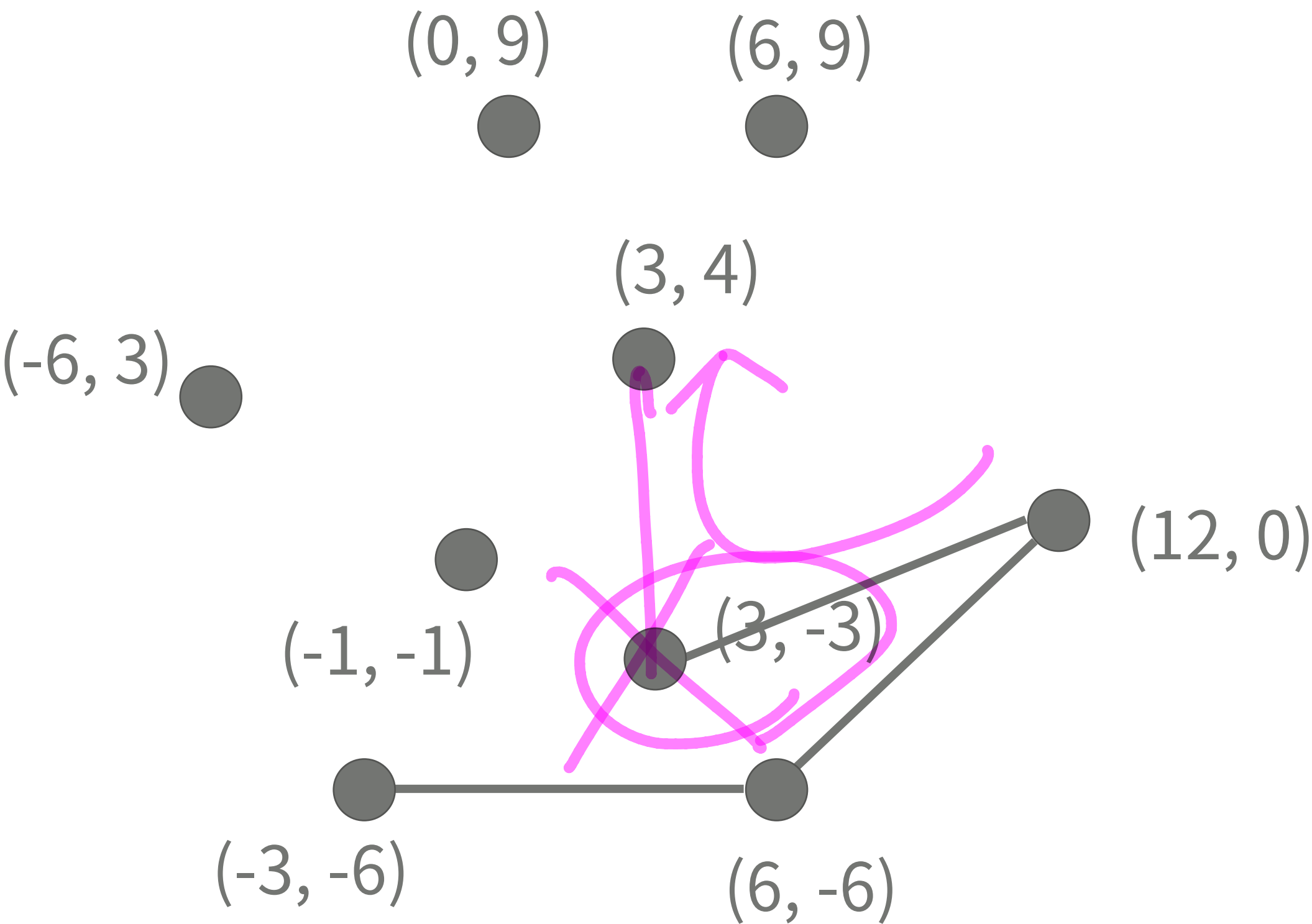
7. -1 -1

8. 0 9

9. -6 3

# 볼록 껍질

Convex Hull

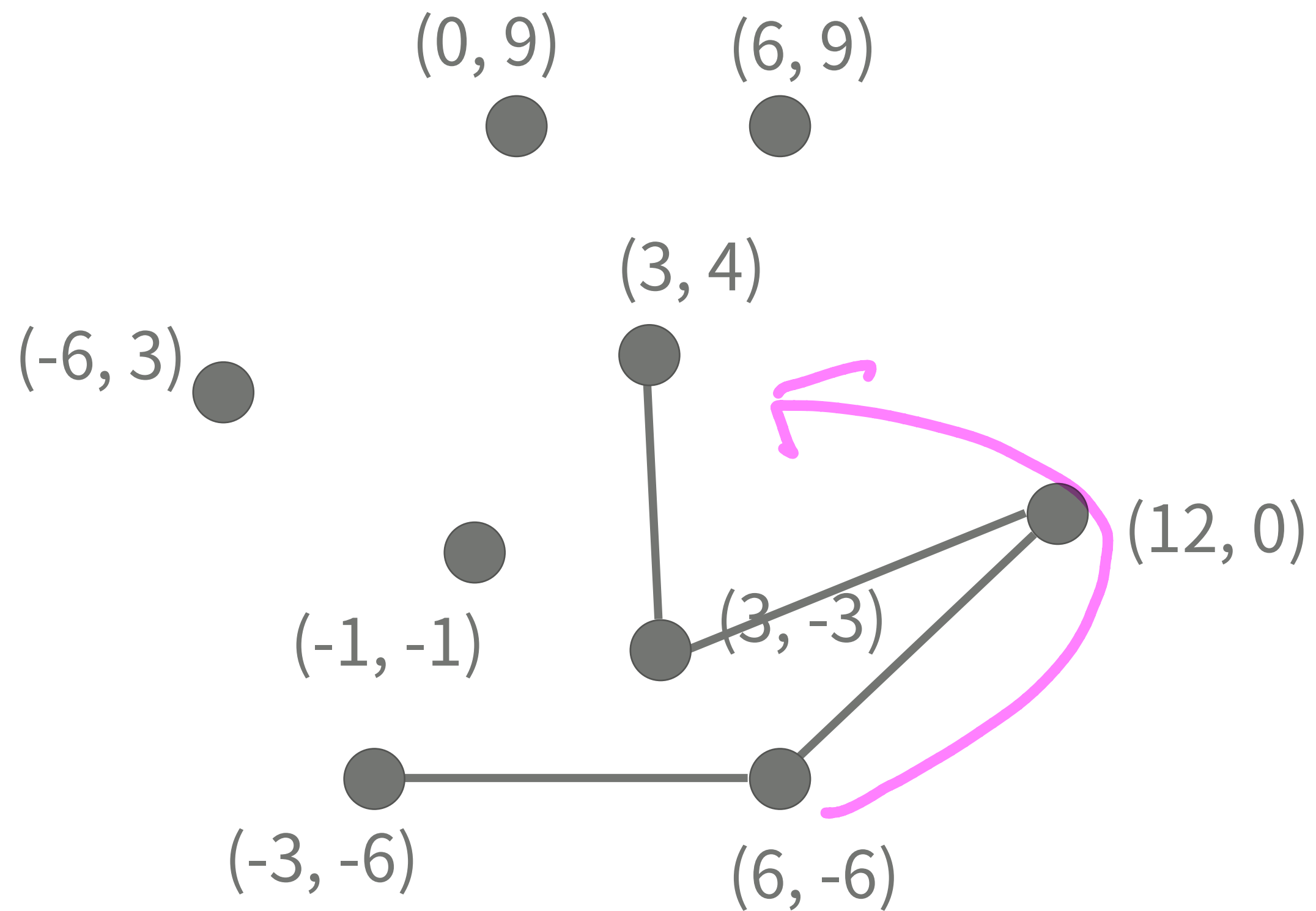


스택	
1. -3 -6	-3 -6
2. 6 -6	6 -6
3. 12 0	12 0
4. 3 -3	<del>3 -3</del>
5. 3 4	
6. 6 9	
7. -1 -1	
8. 0 9	
9. -6 3	

# 볼록 껍질

Convex Hull

47



스택

1. -3 -6

-3 -6

2. 6 -6

6 -6

3. 12 0

12 0

4. 3 -3

5. 3 4

6. 6 9

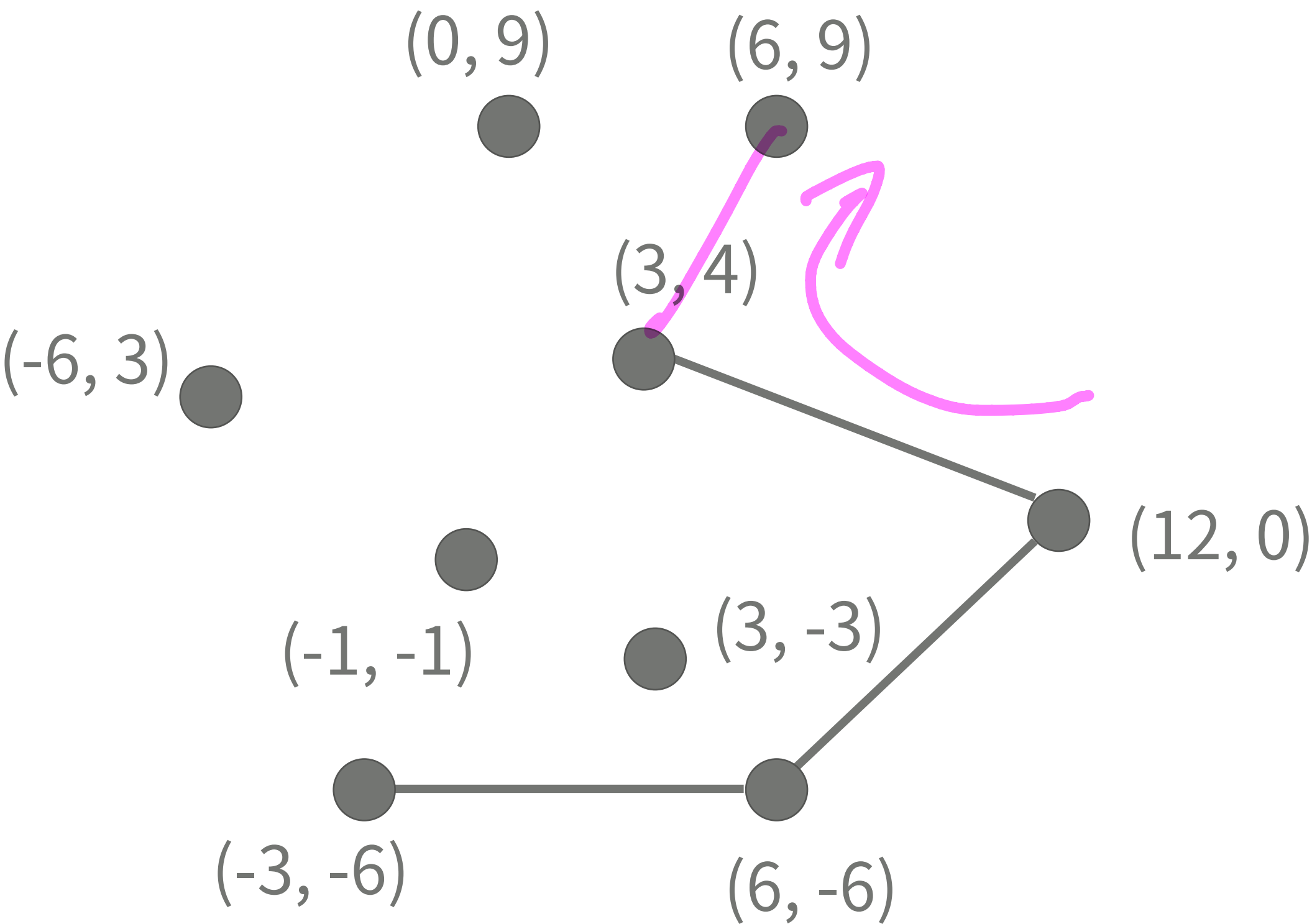
7. -1 -1

8. 0 9

9. -6 3

# 볼록 껍질

Convex Hull



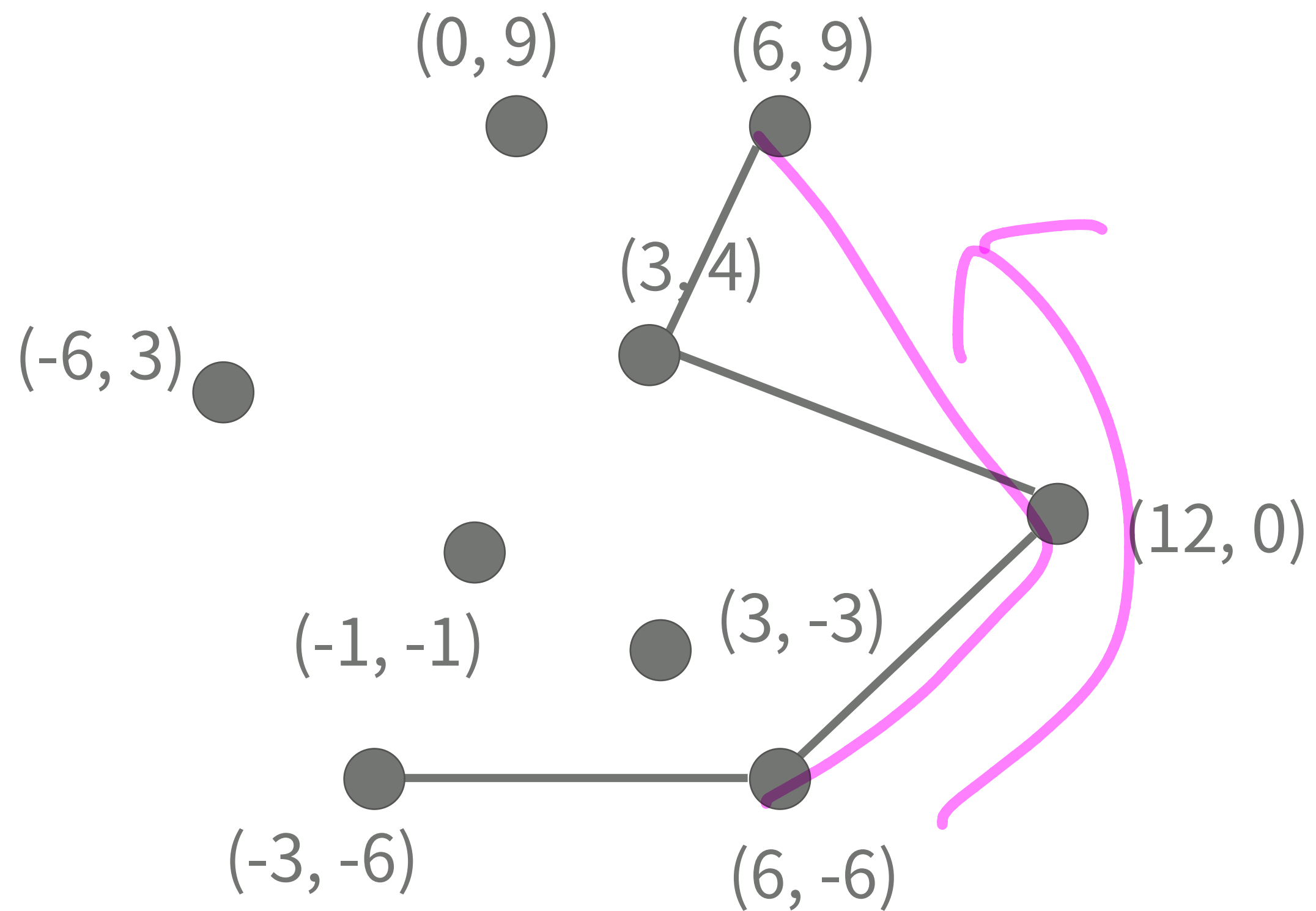
스택	
1. -3 -6	-3 -6
2. 6 -6	6 -6
3. 12 0	12 0
4. 3 -3	<del>3 4</del>
5. 3 4	
6. 6 9	
7. -1 -1	
8. 0 9	
9. -6 3	



# 볼록 껍질

Convex Hull

49



스택

1. -3 -6

-3 -6

2. 6 -6

6 -6

3. 12 0

12 0

4. 3 -3

5. 3 4

6. 6 9

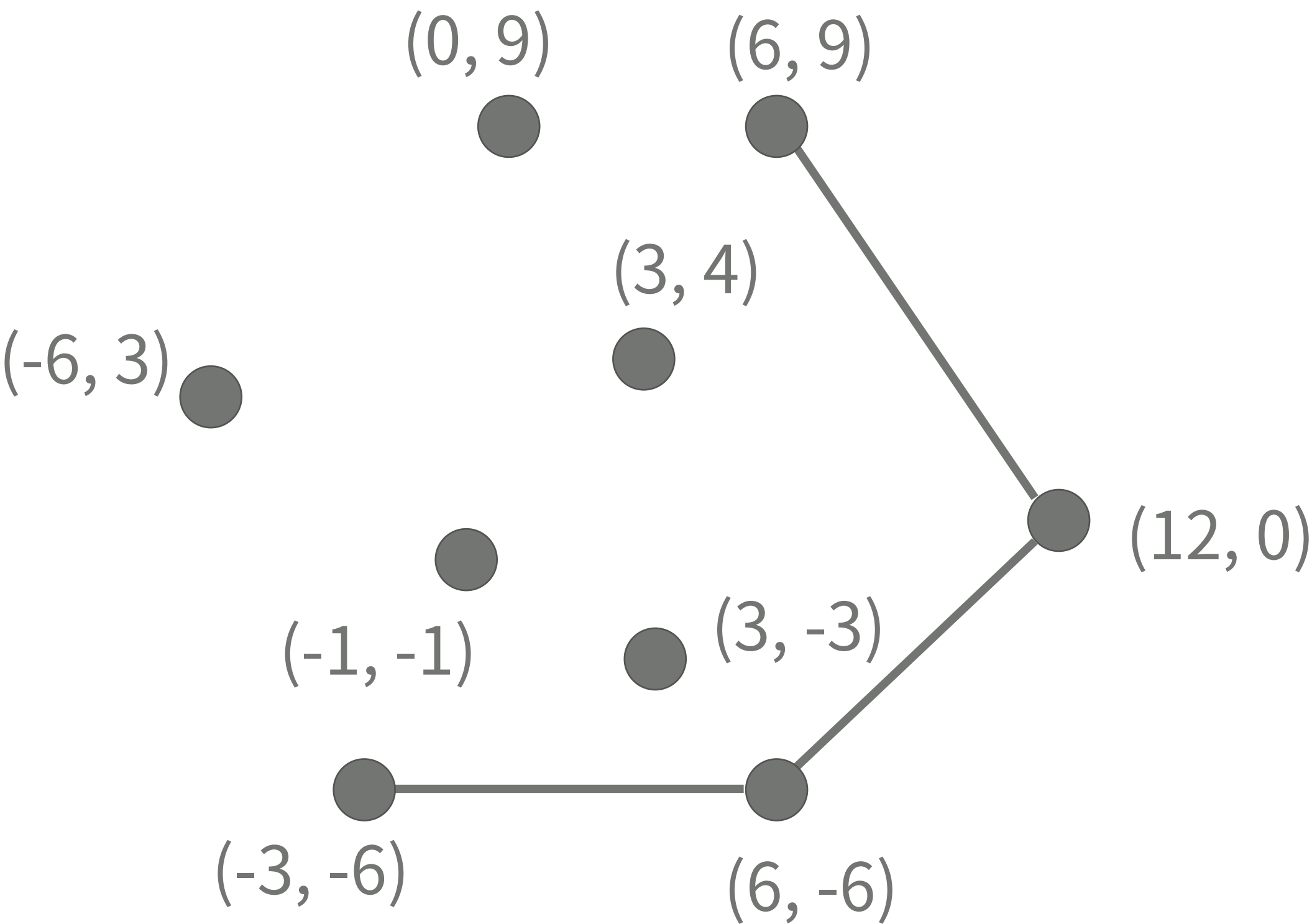
7. -1 -1

8. 0 9

9. -6 3

# 볼록 껍질

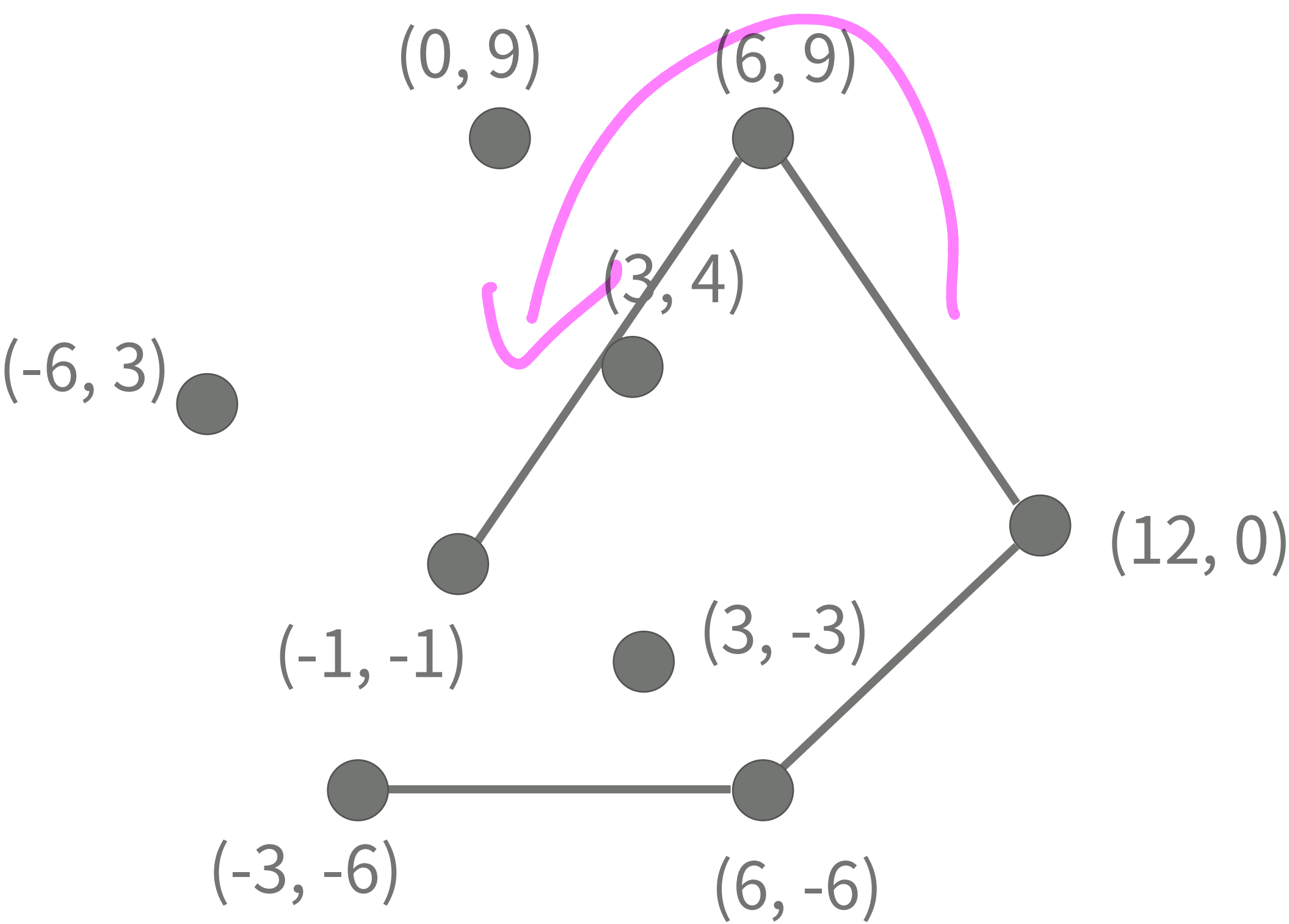
Convex Hull



스택	
1.	-3 -6
2.	6 -6
3.	12 0
4.	3 -3
5.	3 4
6.	6 9
7.	-1 -1
8.	0 9
9.	-6 3

# 볼록 껍질

Convex Hull

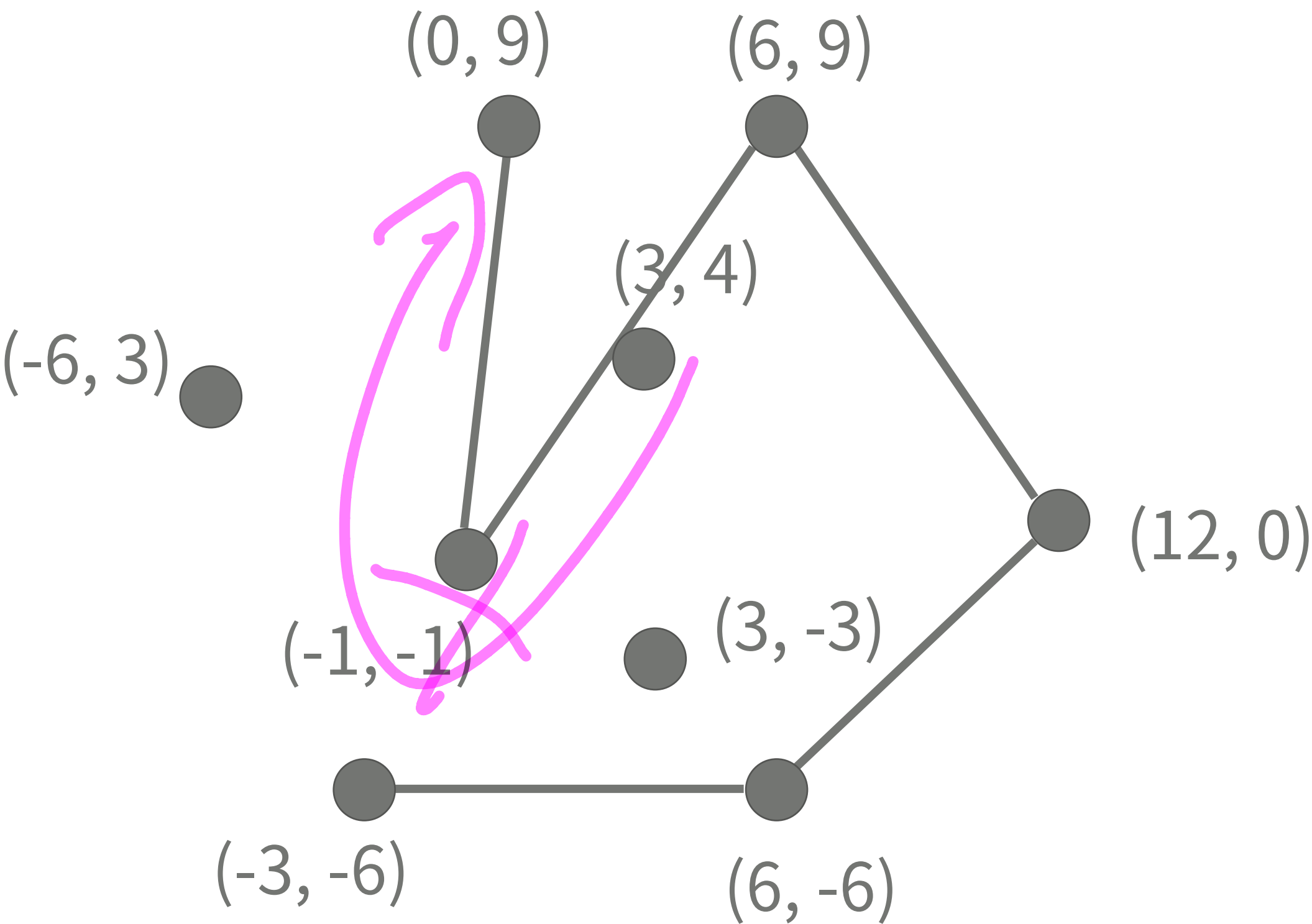


스택

- 1. -3 -6      -3 -6
- 2. 6 -6      6 -6
- 3. 12 0      12 0
- 4. 3 -3      6 9
- 5. 3 4      -1 -1
- 6. 6 9
- 7. -1 -1
- 8. 0 9
- 9. -6 3

# 볼록 껍질

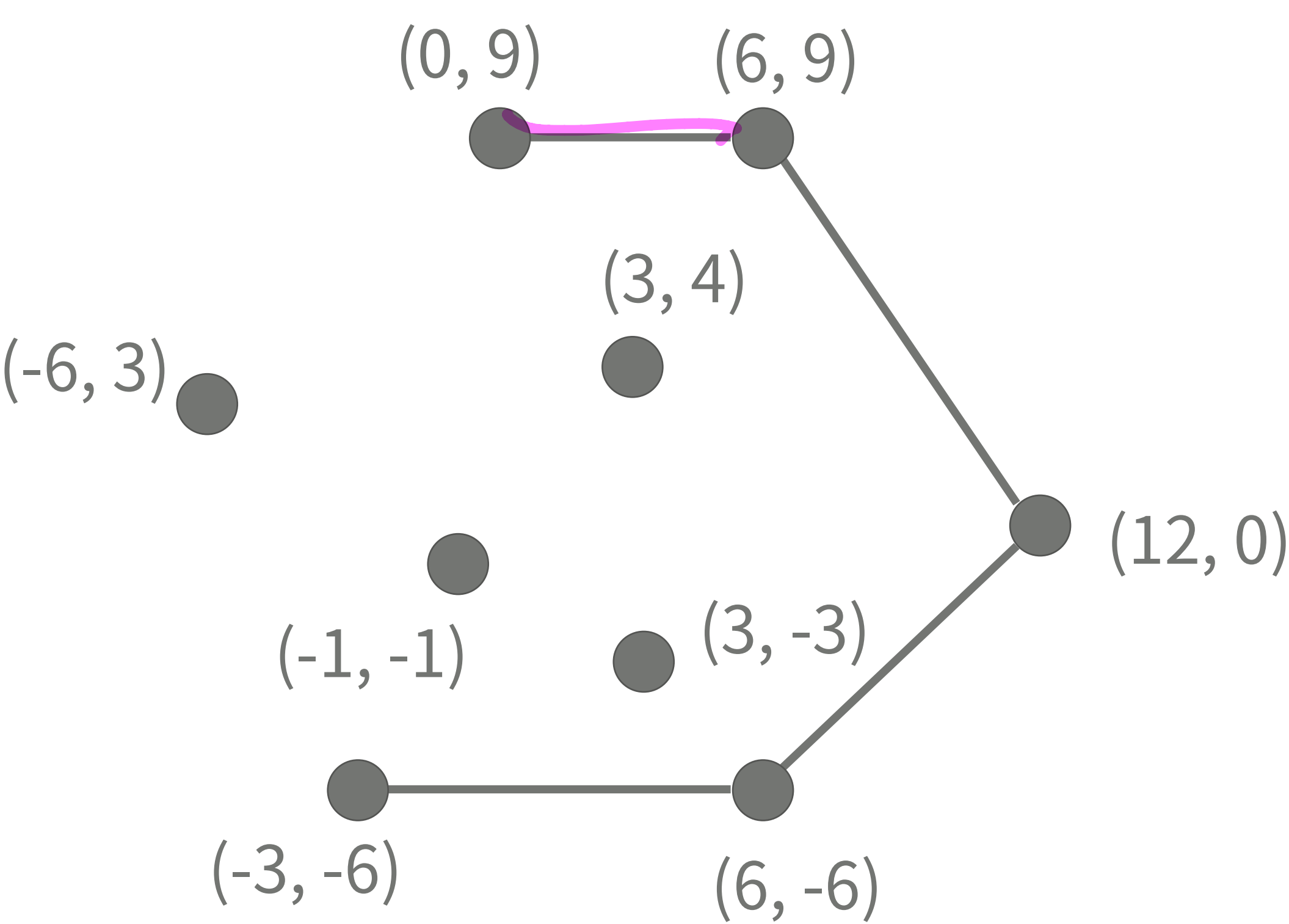
Convex Hull



	스택
1. -3 -6	-3 -6
2. 6 -6	6 -6
3. 12 0	12 0
4. 3 -3	6 9
5. 3 4	
6. 6 9	
7. -1 -1	
8. 0 9	
9. -6 3	

# 볼록 껍질

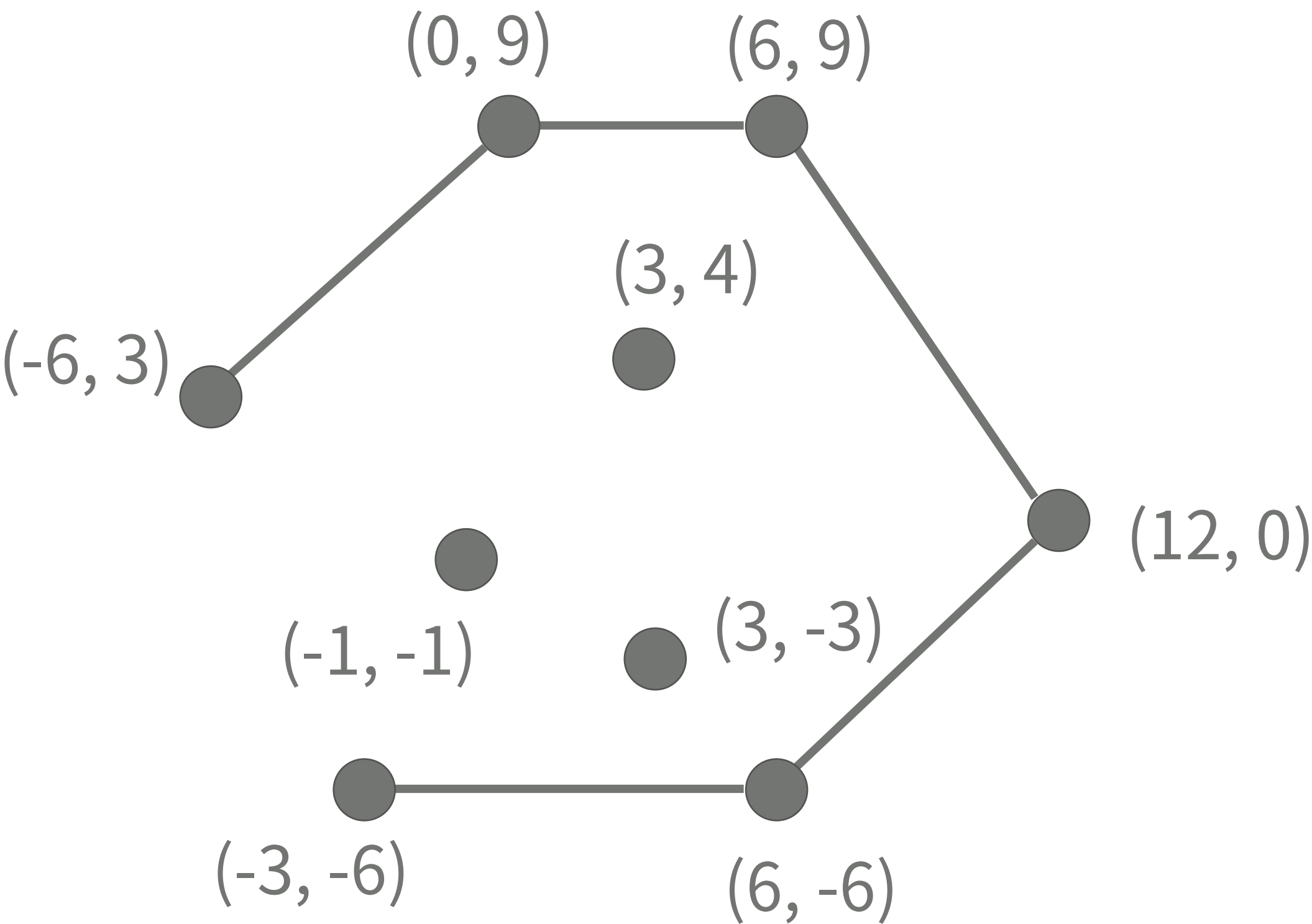
Convex Hull



	스택
1. -3 -6	-3 -6
2. 6 -6	6 -6
3. 12 0	12 0
4. 3 -3	6 9
5. 3 4	0 9
6. 6 9	
7. -1 -1	
8. 0 9	
9. -6 3	

# 볼록 껍질

Convex Hull



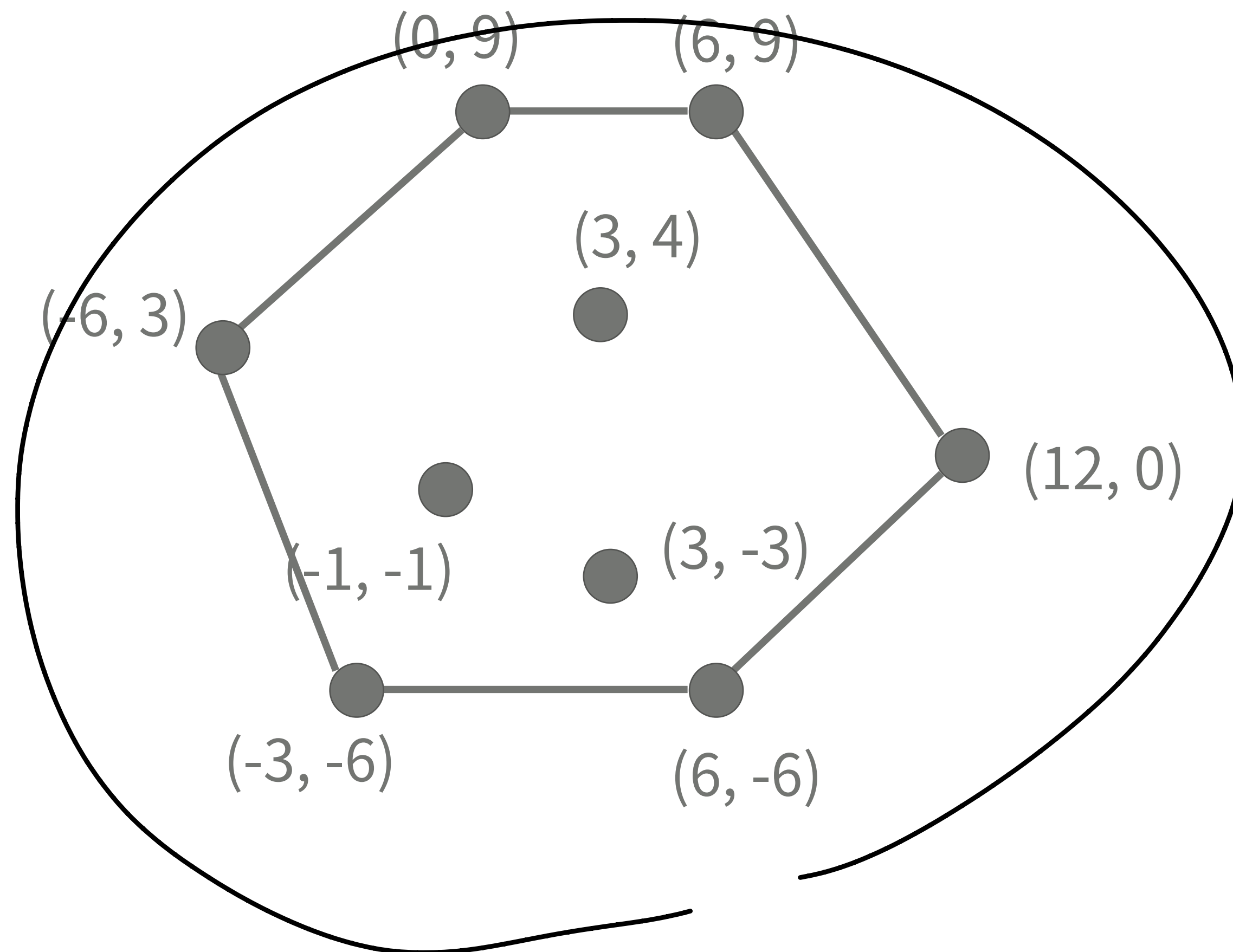
	스택
1. -3 -6	-3 -6
2. 6 -6	6 -6
3. 12 0	12 0
4. 3 -3	6 9
5. 3 4	0 9
6. 6 9	-6 3
7. -1 -1	
8. 0 9	
9. -6 3	

# 볼록 껍질

Convex Hull

$\frac{N \log N + N}{\text{점질}}$

$O(N \log N)$   
스택



1. -3 -6

-3 -6

2. 6 -6

6 -6

3. 12 0

12 0

4. 3 -3

6 9

5. 3 4

0 9

6. 6 9

-6 3

7. -1 -1

8. 0 9

9. -6 3

# 볼록 껍질

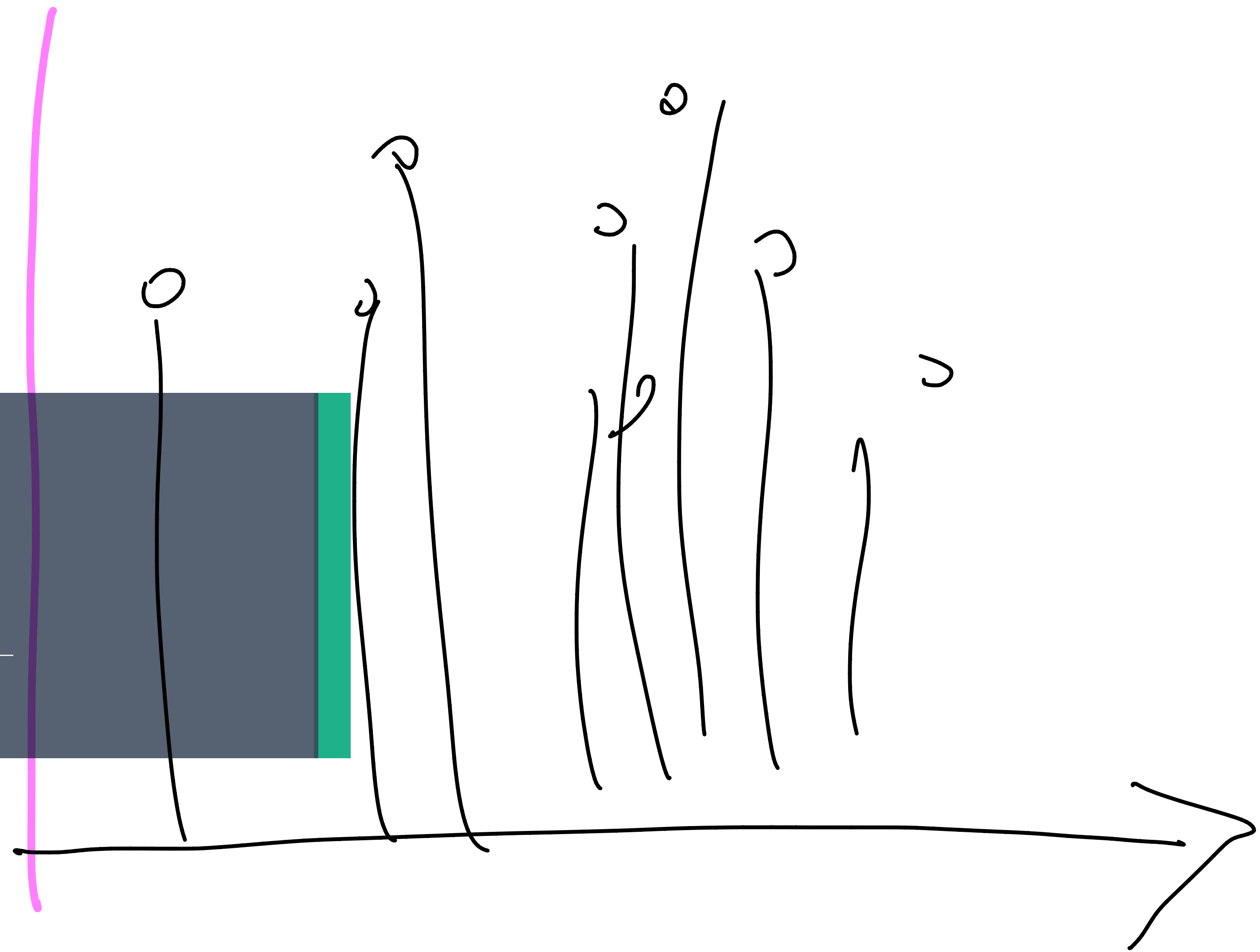
<https://www.acmicpc.net/problem/1708>

- 소스: <http://codeplus.codes/0519acc69a99497c88af21b86f65676e>



$N^2$   
 $\times \frac{1}{2} N^2$

# Line Sweep



# 가장 가까운 두 점

<https://www.acmicpc.net/problem/2261>

- 2차원 평면 위의 N개의 점 중에서 가장 가까운 두 점을 찾는 문제
- $2 \leq N \leq 100,000$

모든 점끼리의 쌍

거리

$O(N^2)$

Live Sweeping  
D&C

# 가장 가까운 두 점

<https://www.acmicpc.net/problem/2261>

N개

후보

정답이 될 수 있는 점

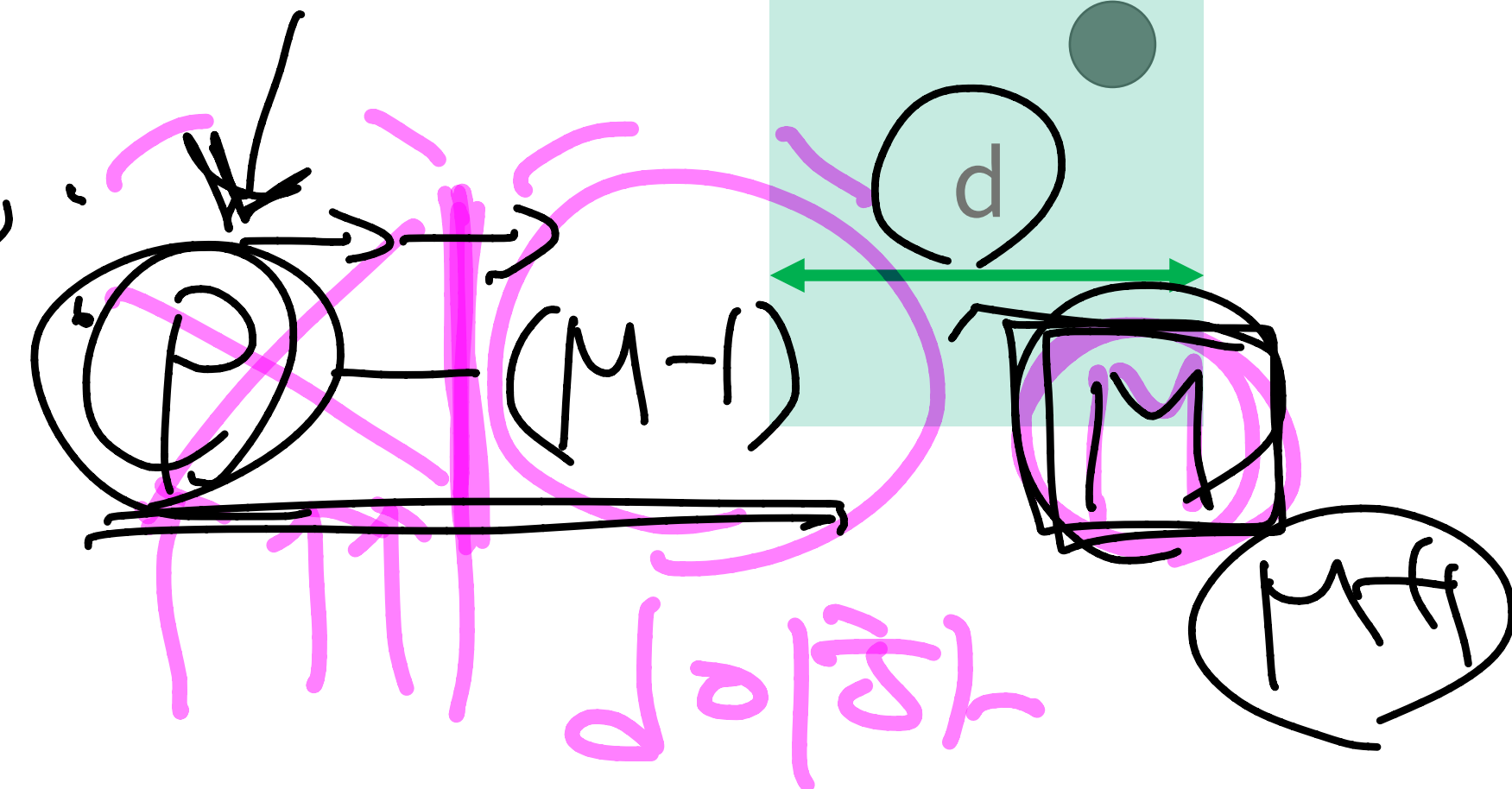
59

- 점을 x좌표가 증가하는 순으로 정렬을 한다.
- 1번부터 M-1번점까지 있을 때, 가장 가까운 점의 거리를 d라고 하자.
- M번점이 추가되었을 때, 가장 가까운 점을 구해야 한다.
- 지금까지 구한 정답이 d이기 때문에
- M번점과 x좌표의 차이가 d이하인 점만 후보가 될 수 있다.

$$dist = \sqrt{x^2 + y^2}$$

후보 거리

후보



# 가장 가까운 두 점

<https://www.acmicpc.net/problem/2261>

- 후보가 되는 점 중에서
- y좌표의 차이가 d이하인 점만
- 정답이 될 수 있다.

↓ :

1-1번까지 가까이

가능한 가장  
가까운 두 점의  
거리

가능성이 있다

크냐

× 좌표의 차이가  
 $\sqrt{d}$  이하

60

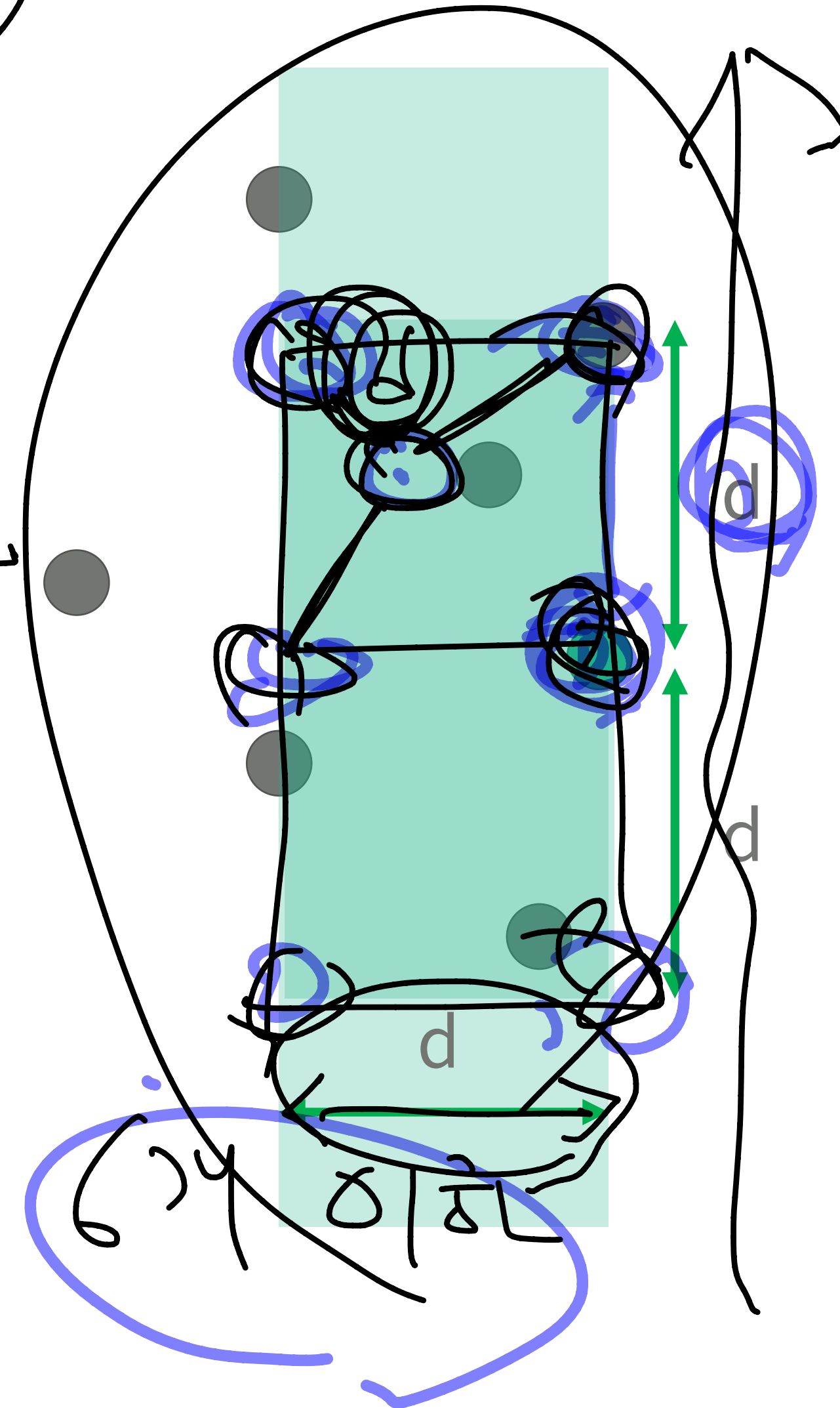
가능성이 있다 2

(y좌표의 차이가  
 $\sqrt{d}$  이하)

정확히 1-1번까지 같은  
거리

정답이 개수

$\Theta(6)$



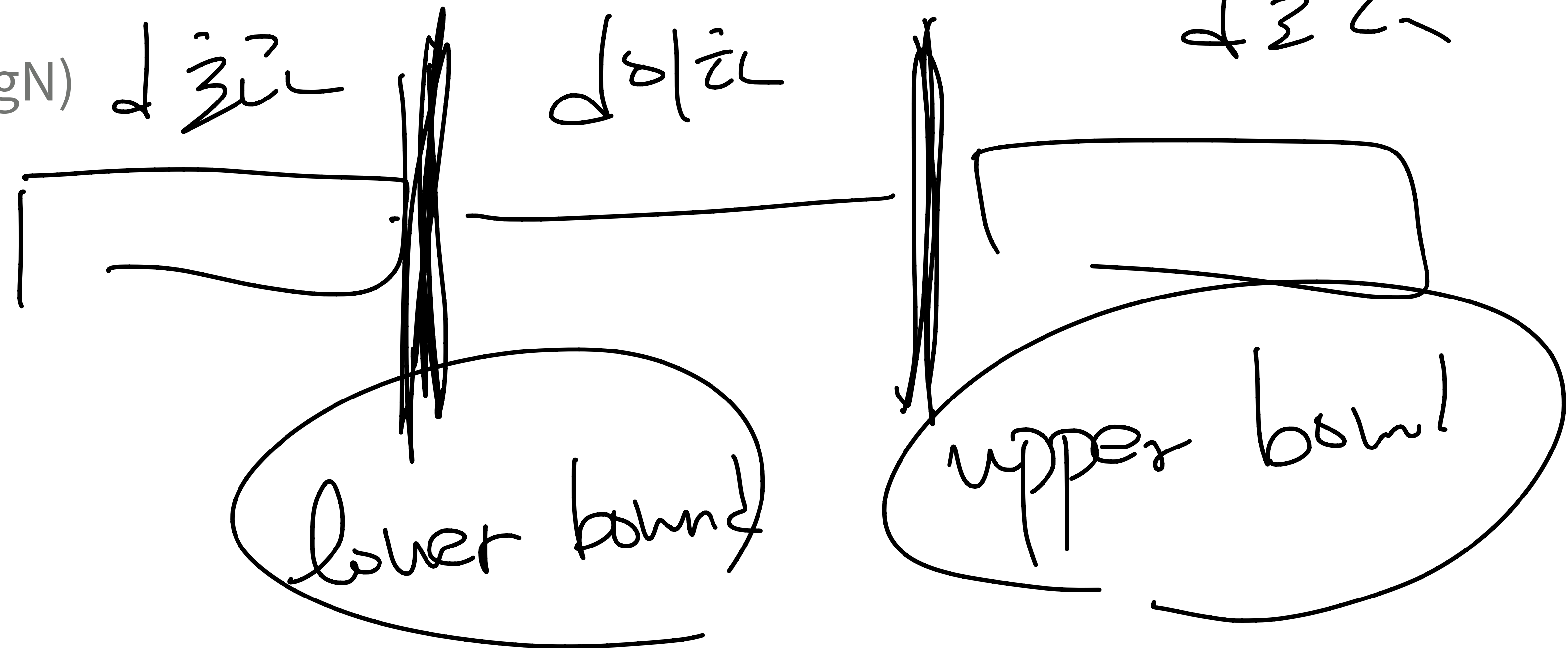
# 가장 가까운 두 점

61

<https://www.acmicpc.net/problem/2261>

$O(N \lg N)$

- M번째 점이 추가되었을 때
- 후보가 되는 점은 k번째 점부터 M-1번째 점까지와 같이 번호가 연속된 형태이다.
- 후보가 되는 점은 y좌표로 정렬하면 y좌표의 차이가 d이하인 점을  $O(\lg N)$ 에 찾을 수 있다.
- y좌표의 차이가 d이하인 점은 최대 6개이기 때문에
- 총 시간 복잡도는  $O(N \lg N)$



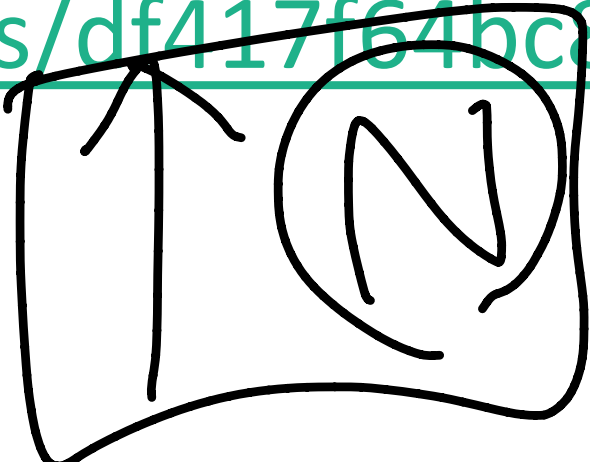


# 가장 가까운 두 점 $DT[CLS] =$

<https://www.acmicpc.net/problem/2261>

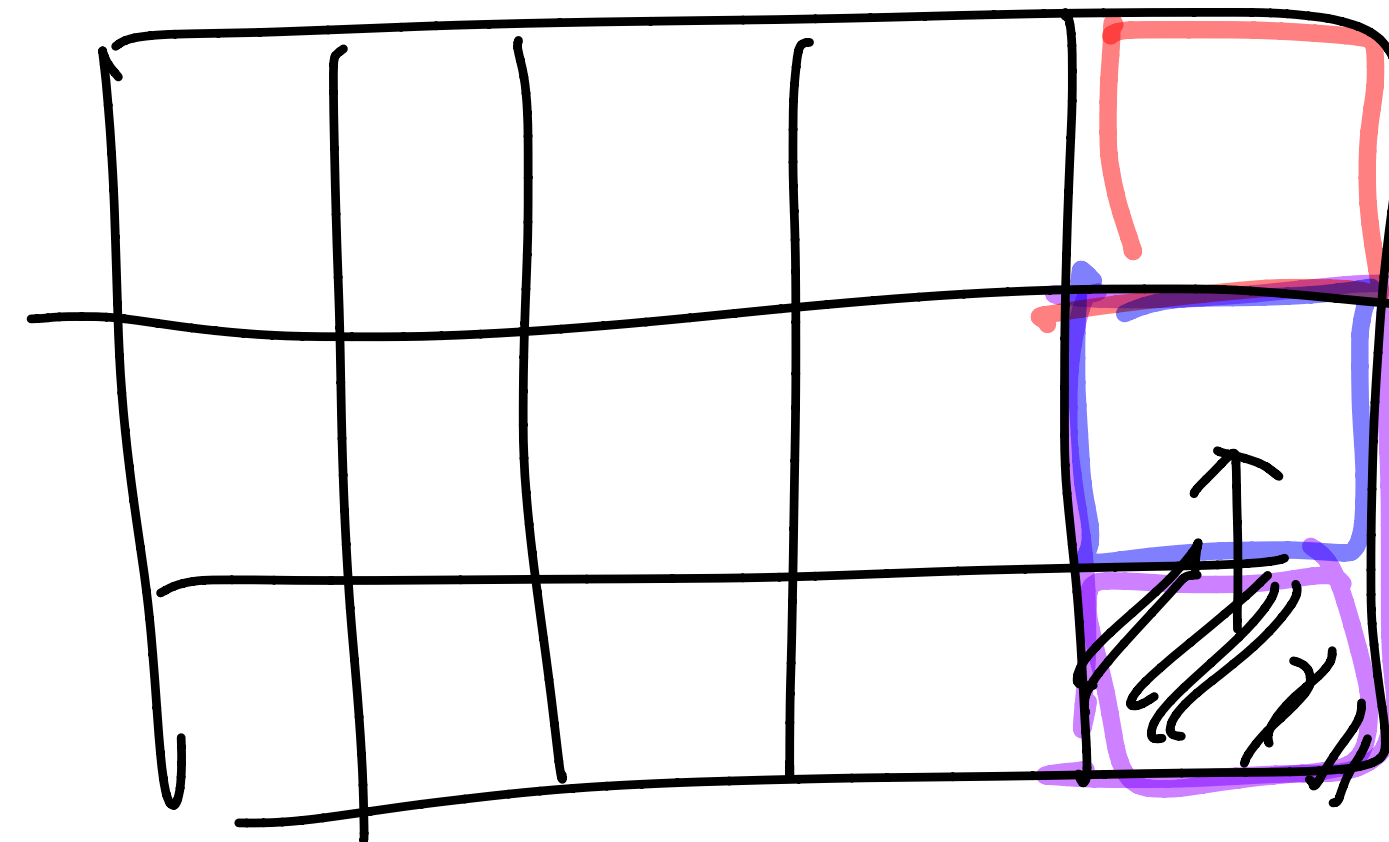
소스: <http://codeplus.codes/df417f64bc8344628af20980ee823954>

(1,5) 가 오른쪽  
위쪽과  
10이 연속되는 4  
회씩  
31씩

$\rightarrow O(N^2) \times$  

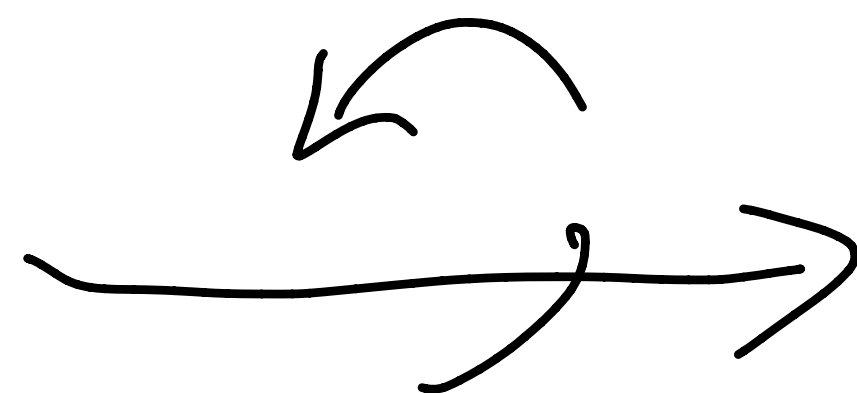
1	0	1	1	1
1	1	1	1	1
0	1	1	1	1

1	0	1	2	3
1	2	3	4	5
0	1	2	3	4



401

401	:	4	:	4
402	:	4	:	8
403	:	3	:	9



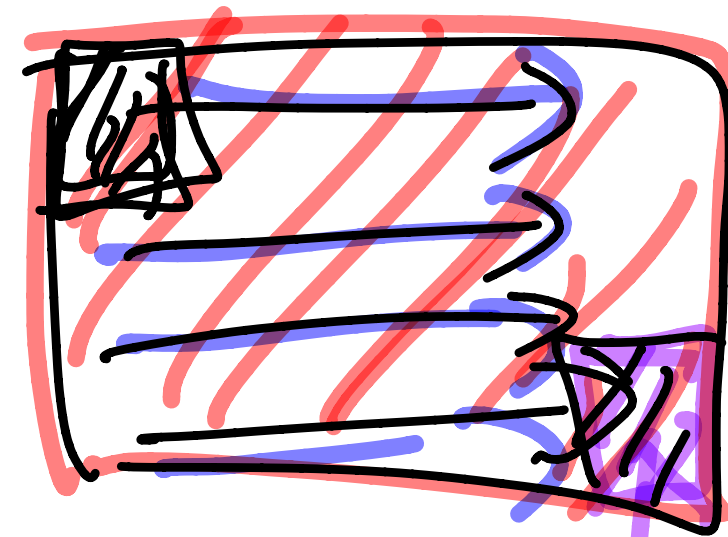
# 최대 직사각형

<https://www.acmicpc.net/problem/11873>

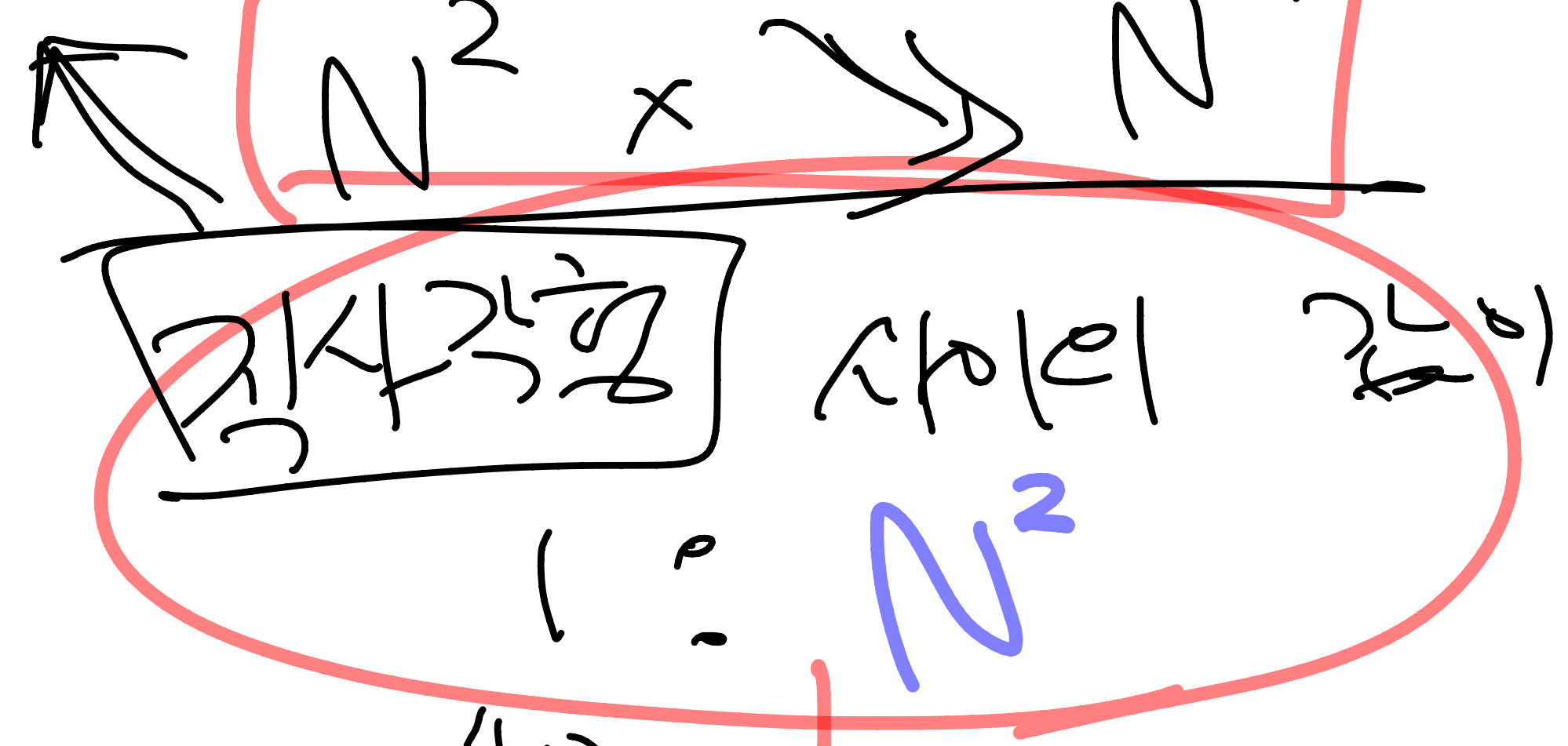
63

- $N \times M$  크기의 행렬이 주어졌을 때
- 1로 이루어진 가장 큰 직사각형을 찾는 문제
- $1 \leq N, M \leq 1,000$

$N \times N$   
 $N \leq 1000$



①  $\theta(N^6)$



②  $O(N^4)$

③  $O(N^3)$

④  $O(N^2)$

X

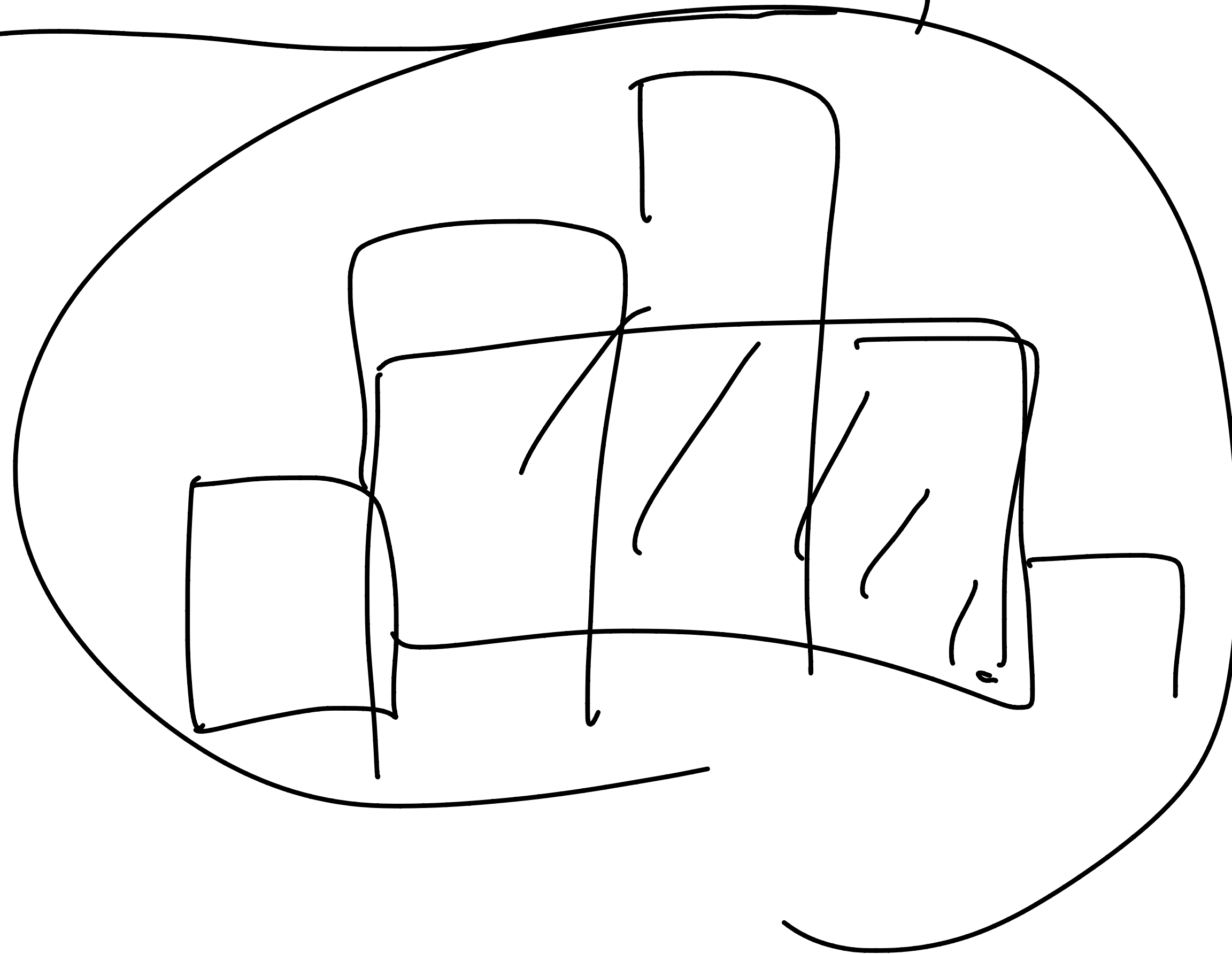
# 최대 직사각형

<https://www.acmicpc.net/problem/11873>

$O(N)$

64

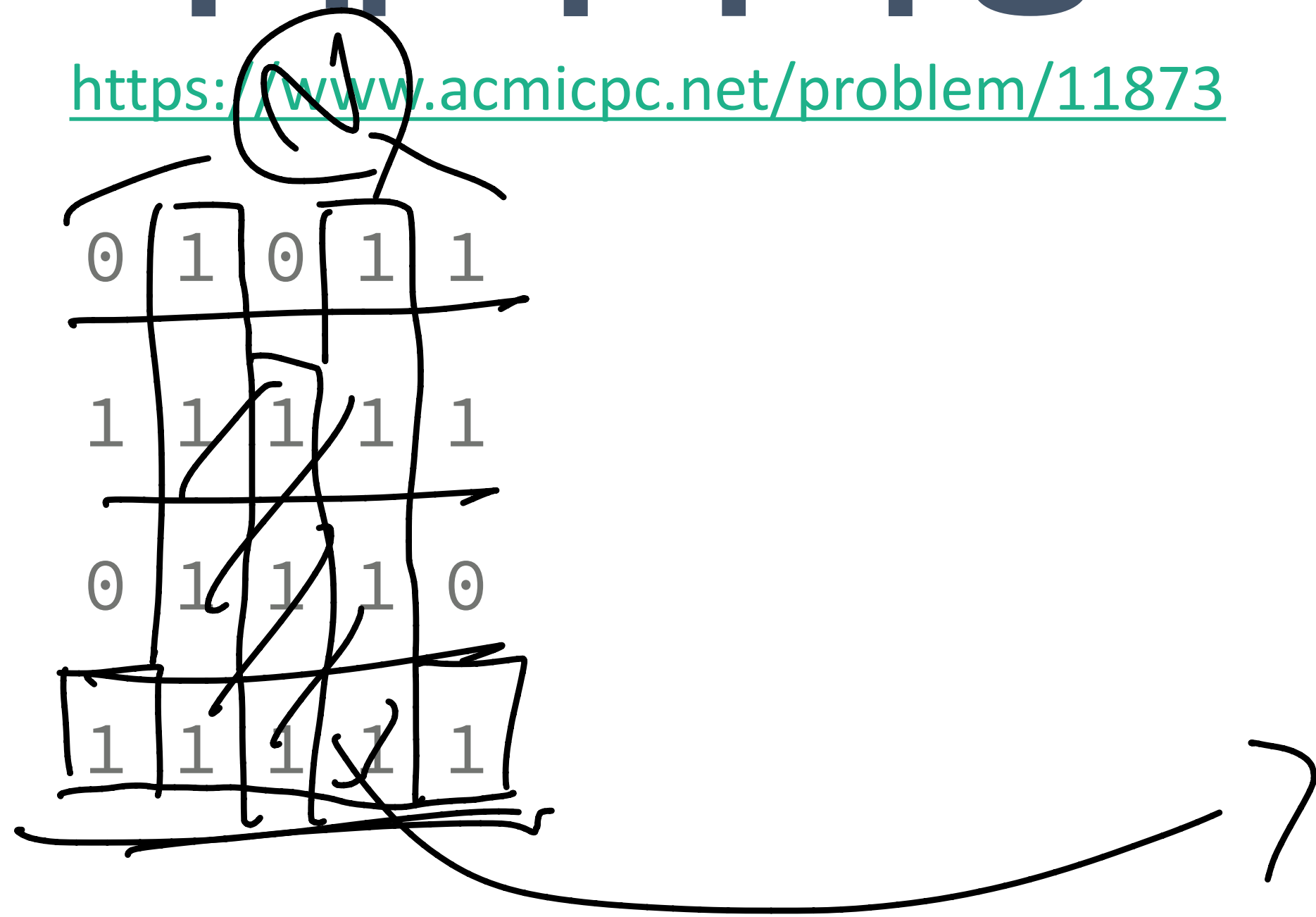
- 이 문제는 6549번 히스토그램에서 가장 큰 직사각형 문제와 같다



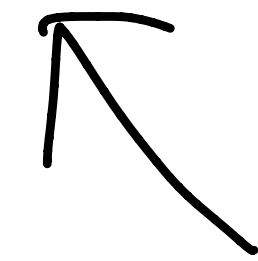


# 최대 직사각형

<https://www.acmicpc.net/problem/11873>

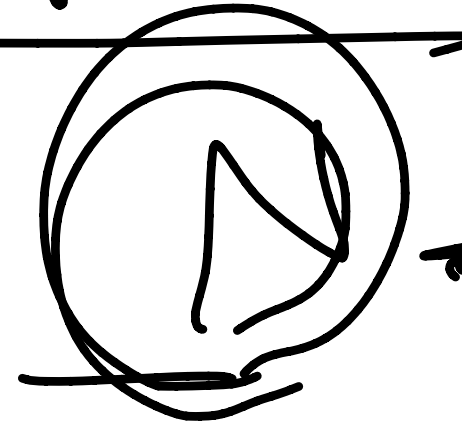


직사각형

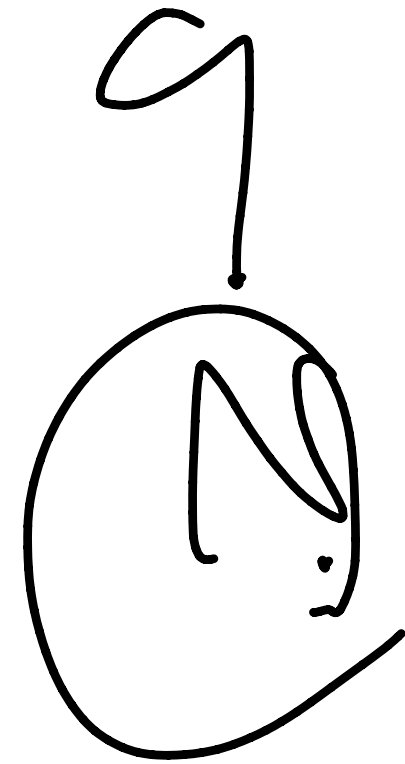


$N^2$   
 $N^4$   
 $N^3$

0/2/4



~~2/2~~



$O(N^2)$

# 최대 직사각형

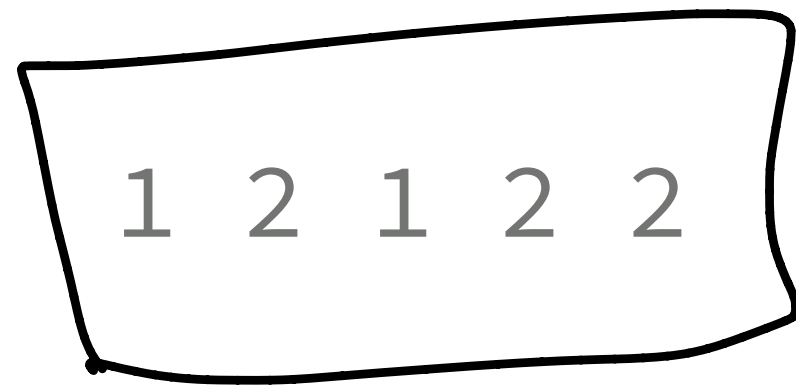
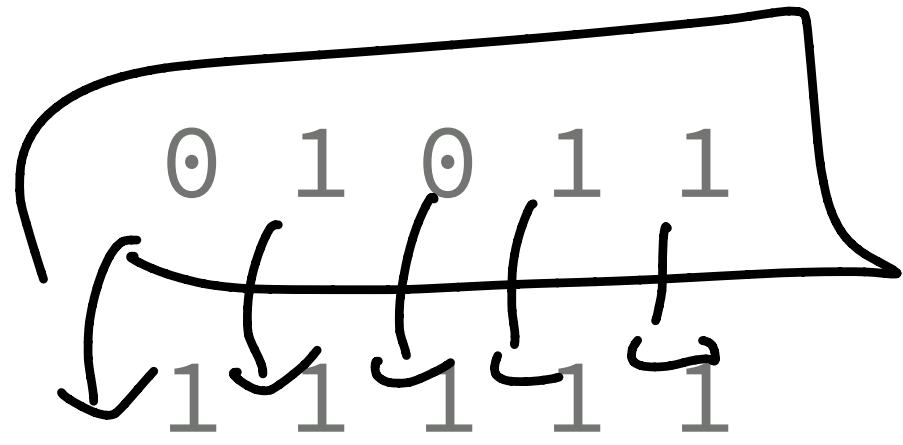
<https://www.acmicpc.net/problem/11873>

0 1 0 1 1

# 최대 직사각형

<https://www.acmicpc.net/problem/11873>

67



# 최대 직사각형

68

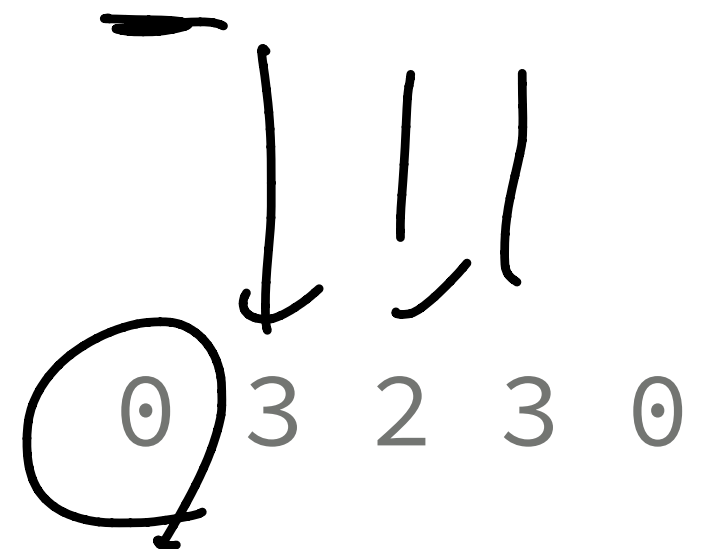
<https://www.acmicpc.net/problem/11873>

0 1 0 1 1

1 1 1 1 1

0 1 1 1 0

0 3 2 3 0



# 최대 직사각형

<https://www.acmicpc.net/problem/11873>

0 1 0 1 1

1 1 1 1 1

0 1 1 1 0

1 1 1 1 1

1 4 3 4 1

$$O(N) \quad \textcircled{\leq \epsilon_1}$$

$$\textcircled{N} \times N \quad \textcircled{\neq} \Theta(N^2)$$

# 최대 직사각형

70

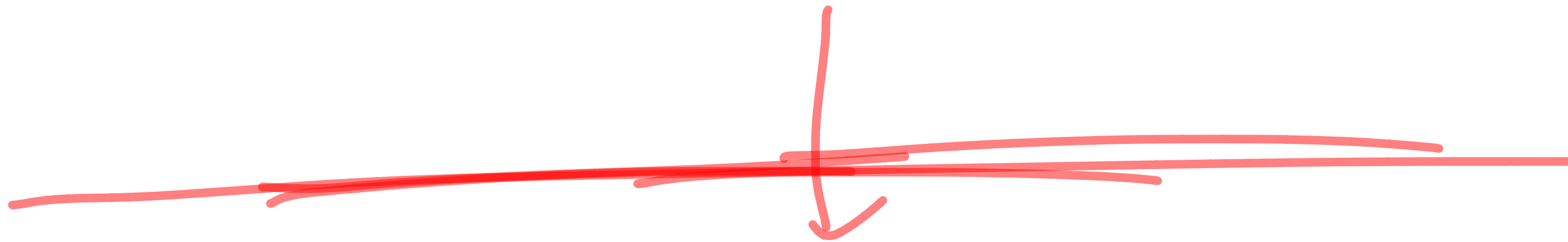
<https://www.acmicpc.net/problem/11873>

- 소스: <http://codeplus.codes/982fa7797fd3460997df282129475009>

# 겹치는 선분

<https://www.acmicpc.net/problem/1689>

- 1차원 좌표계에 선분 N개가 있을 때, 최대로 겹쳐있는 부분의 겹친 선분 개수를 구하는 문제
- $1 \leq N \leq 1,000,000$



# 겹치는 선분

<https://www.acmicpc.net/problem/1689>

72

- 선분 6개가 있고

• [0, 6]

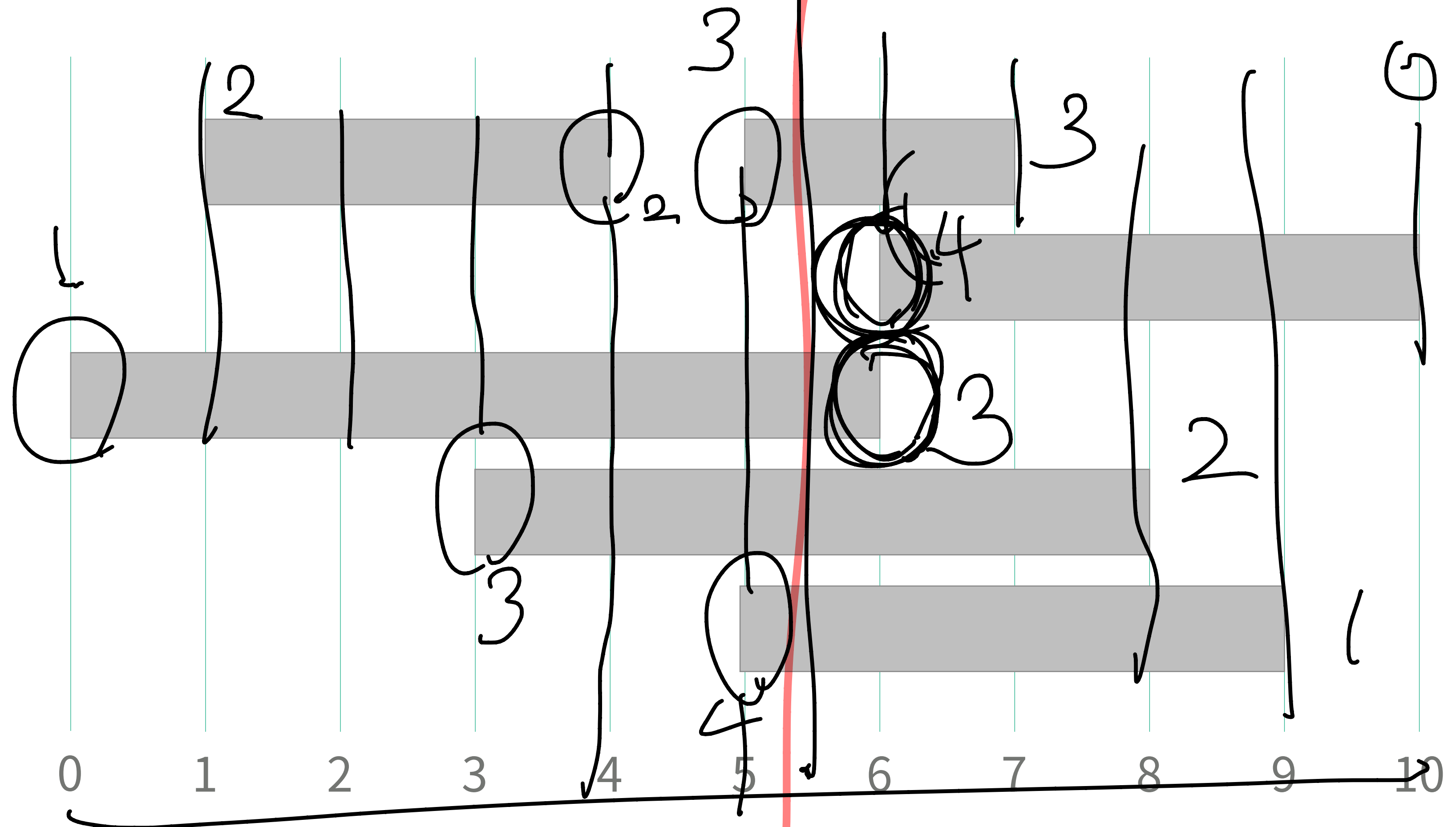
• [1, 4]

• [3, 8]

• [5, 7]

• [6, 10]

• [5, 9]





# 겹치는 선분

<https://www.acmicpc.net/problem/1689>

- 선분은 점 2개로 이루어져 있다
- 시작점과 끝점으로 나누어서
- 시작점 = 선분의 개수 1 증가
- 끝점 = 선분의 개수 1 감소

# 겹치는 선분

<https://www.acmicpc.net/problem/1689>

- 소스: <http://codeplus.codes/7d40cdcbacfb4642afc560f881ae9697>

# 선 긋기

75

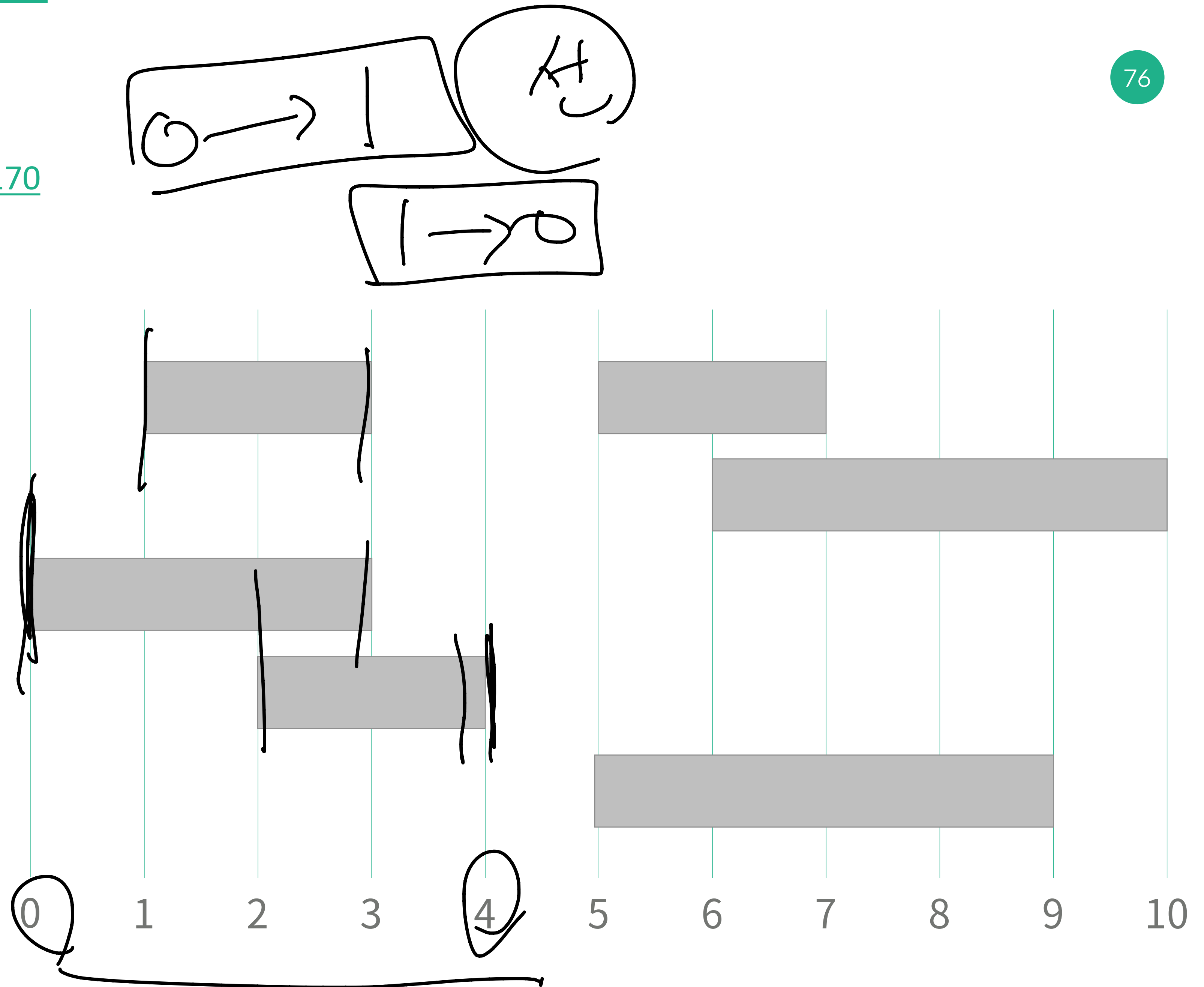
<https://www.acmicpc.net/problem/2170>

- 선분 N개가 있을 때, 총 길이를 구하는 문제
- $1 \leq N \leq 1,000,000$

# 선 긋기

<https://www.acmicpc.net/problem/2170>

- 선분 6개가 있고
- $[0, 3]$
- $[1, 3]$
- $[2, 4]$
- $[5, 9]$
- $[6, 10]$
- $[5, 7]$



# 선긋기

<https://www.acmicpc.net/problem/2170>

- 겹치는 선분 문제와 비슷하게 해결할 수 있다.

# 선긋기

78

<https://www.acmicpc.net/problem/2170>

- 소스: <http://codeplus.codes/8a32db5315514f9782862ba10ec5a356>