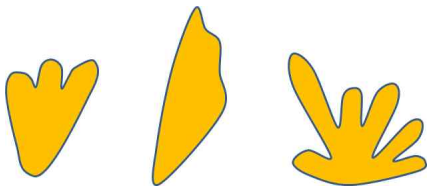
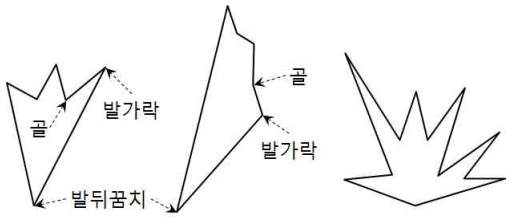
# 공룡 발자국

고대 공룡들이 남쪽에서 북쪽 방향으로 걸어간 발자국들이 발견되었다. 아래 그림은 발견된 공룡 발자국들의 다양한 예이다.



발자국을 분석한 결과 모든 발자국은 하나의 발뒤꿈치와 $k$개의 발가락 ($k \geq 2$)을 가지며, 두 발가락 사이마다 골이 존재한다. 이를 바탕으로 위 그림의 발자국을 단순화시켜보면 아래와 같이 $2k$각형으로 표현이 된다.

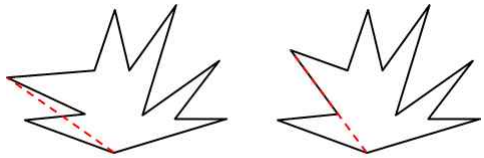



발자국의 다각형에서 발뒤꿈치를 시작으로 반시계 방향으로 돌면 항상 첫 번째 발가락에서 좌회전, 첫 번째 골에서 우회전, 두 번째 발가락에서 좌회전, 두 번째 골에서 우회전, ……, $k-1$번째 골에서 우회전, $k$번째 발가락에서 좌회전해서 발뒤꿈치로 돌아오게 된다. 또한 발뒤꿈치와 발가락을 선분으로 잇는 경우 다음 조건을 항상 만족한다.

① 해당 선분은 발자국의 다각형을 벗어나지 않는다.

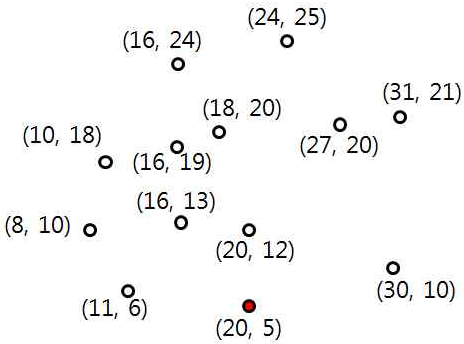
② 해당 선분은 골을 지나지 않는다.

다음 그림은 발자국이 될 수 없는 다각형들의 예이다.

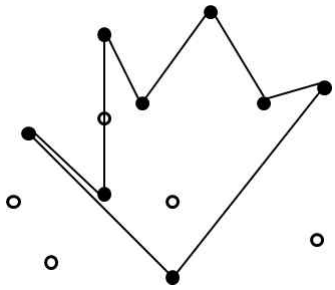


조건 ① 위배 조건 ② 위배

발자국이 발견된 일부 지역에서는 불행히도 정확한 발자국이 남아있지 않고 발자국 다각형의 꼭짓점이 될 수 있는 점들만 남아있다. 심지어 여기에는 발자국과 관련 없는 점들도 같이 남아있을 수 있다. 각 점의 위치는 좌표 $(x, y)$로 표현되며, $x$값은 서쪽에서 동쪽으로 증가하고 $y$값은 남쪽에서 북쪽으로 증가한다. 다행히도 점들 중 가장 남쪽에 있는 점은 유일하며 발뒤꿈치가 된다. 다음 그림의 예에서 보면 $(20, 5)$ 점(붉은 점)이 가장 남쪽에 위치하므로 발뒤꿈치이다.




주어진 점들로 만들 수 있는 가장 많은 발가락을 가진 발자국을 찾고 싶다. 아래는 위 그림에서 찾은 가장 많은 발가락을 가진 발자국의 한 예이다.



점들의 좌표를 입력으로 받아서, 가장 많은 발가락을 가진 발자국을 하나 찾은 뒤 찾은 다각형의 꼭짓점의 개수와 각 꼭짓점의 좌표를 출력하는 프로그램을 작성하라.

소스파일의 이름은 footprint.c 또는 footprint.cpp를 권장하지만, 서버에 제출하는 데는 다른 이름도 상관없다.

## 입력 형식

표준 입력으로 다음 정보가 주어진다. 첫 번째 줄에는 점의 수를 나타내는 정수 $N$이 주어진다. 다음 $N$개의 각 줄에는 한 점의 $x$좌표와 $y$좌표를 나타내는 두 정수가 주어진다. 입력으로 주어진 점들은 모두 서로 다르다는 것이 보장되고, $y$좌표가 가장 작은 점이 유일하다는 것도 보장된다.

## 출력 형식

표준 출력으로, 첫 번째 줄에는 가장 많은 발가락을 가진 발자국 다각형의 꼭짓점의 개수 $T$를 출력한다. 다음 $T$개의 각 줄에는 발뒤꿈치부터 시작하여 반시계 방향으로 돌면서 각 꼭짓점의 $x$좌표와 $y$좌표를 출력한다. 그러한 발자국이 여러 개가 있다면 그 중 하나의 발자국만 출력한다. 발자국이 존재할 수 없는 경우 0을 출력한다.

## 부분문제의 제약 조건

모든 부분문제에서 $-10^8 \leq x, y \leq 10^8$이다.

- **부분문제 1:** 전체 100점 중 14점에 해당하며 $4 \leq N \leq 3{,}000$, $T = N$이다.
- **부분문제 2:** 전체 100점 중 12점에 해당하며 $4 \leq N \leq 10$이다.
- **부분문제 3:** 전체 100점 중 29점에 해당하며 $4 \leq N \leq 300$이다.
- **부분문제 4:** 전체 100점 중 45점에 해당하며 $4 \leq N \leq 3{,}000$이다.

## 입력과 출력의 예

입력(1)

```
6
3 0
5 10
4 9
3 10
2 9
1 10
```

출력(1)

```
6
3 0
5 10
4 9
3 10
2 9
1 10
```

입력(2)

```
6
1 20
2 18
4 17
3 10
5 5
0 0
```

출력(2)

```
6
0 0
5 5
3 10
4 17
2 18
1 20
```

입력(3)

```
13
20 12
27 20
10 18
16 24
8 10
24 25
20 5
18 20
16 19
30 10
11 6
31 21
16 13
```

출력(3)

```
8
20 5
31 21
27 20
24 25
18 20
16 24
16 13
10 18
```