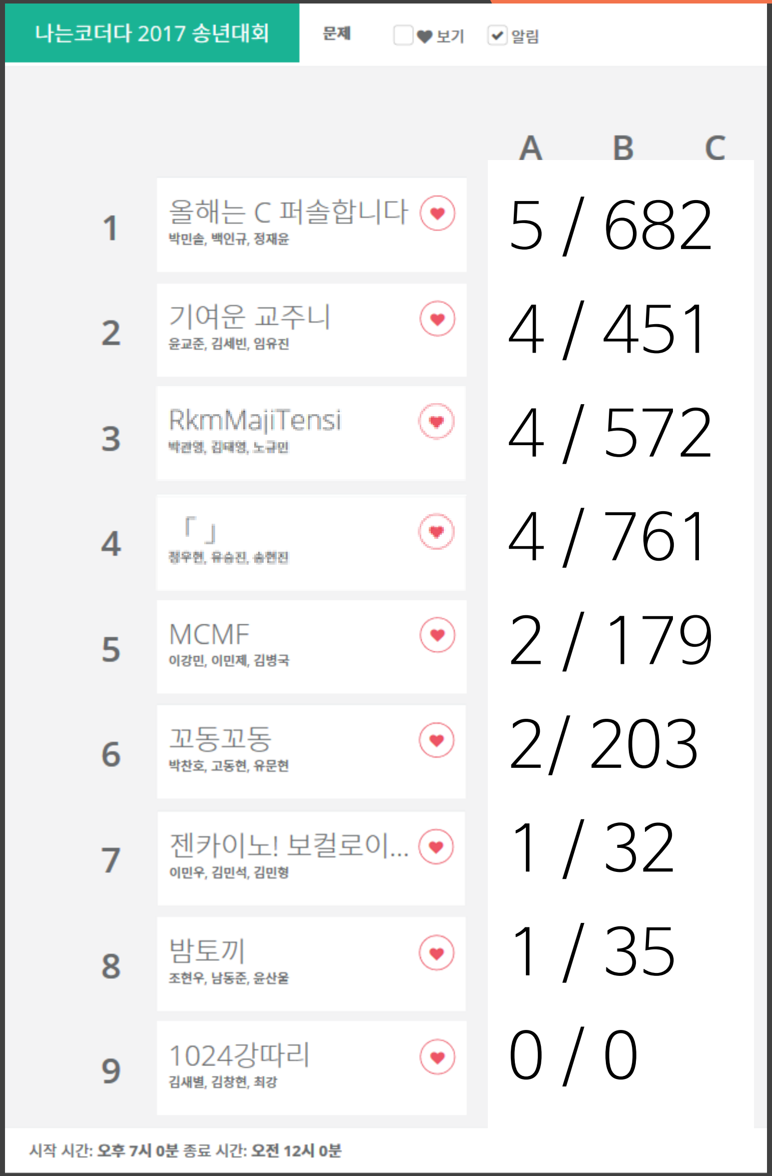
나는코더다 2017 송년대회 풀이 슬라이드

김동현, 김현수, 신승원

대회 결과

- Onsite (교내)

- 대회 진행 중 약간의 문제가 생겨서 스코어보드가 꼬였습니다 ㅠㅠ



- Open

나는코더다 2017 송년대회 Open Contest　문제　♥ 보기　✔ 알림　4시간 30분

		A	B	C	D	E	F	G	H	I	J	K	L	
1	koosaga -	1 99	1 224	1 238		1 56	1 206	1 73	1 8	1 71		6 274		9 1249
2	ainta -	1 215				3 232	1 202	2 190	2 209	1 187				6 1235
3	tlwpdus -		2 275			1 23		1 81	1 30	2 122	2			5 531
4	kajebiii -					1 49		1 62	1 38	3	7 278			4 427
5	doju -	2 148	3 233	1 222										3 603
6	alpah -					6 304			1 29					2 333
7	august14 -	1 155	2 230											2 385
8	moonrabbit2 -					6 353			1 110					2 463

9	junis3 -	3			1 16	1 16
10	poketred12 -				1 17	1 17
11	exqt -				1 21	1 21
12	rdd6584 -		1		1 24	1 24
13	klimmek55 -				1 27	1 27
14	onjo0127 -			1	1 29	1 29
15	veydpz -				1 30	1 30
16	kjarong -				1 33	1 33
17	kyaryunha -				2 53	1 53
18	jason9319 -				1 55	1 55

19	heehcs -			2 57		1 57
20	subinium -			1 78		1 78
21	happyleeyi -		2	2 103		1 103
22	nfysh73 -			1 121		1 121
23	sky1357 -			2 143		1 143
24	water159 -			4 179		1 179
25	jaydoubluel -				1 196	1 196
26	jdown -	1	8	7 243		1 243

(1문제 이상 푼 사람만)

필요한 배경 지식

- DP (A, B, I, K)
- 기하 - 시계/반시계 방향 체크 (A, D)
- Modulo Inverse (B, E, K)
- 기초적 조합론 (B)
- 누적합 배열 (C)
- Dual Graph의 개념과 코딩, Union-Find (D)
- Floyd-Warshall algorithm (D)
- Centroid Decomposition (K)
- 세그먼트 트리 (L)
- 트리에서 LCA 구하기(L)
- 절점 BCC 분할 (L)

A 경곽 침공

시간 : 5초

출제 : 신승원

문제

평면 상에 점들이 놓여 있다

(단, 세 점이 한 직선 위에 있는 경우는 없다)

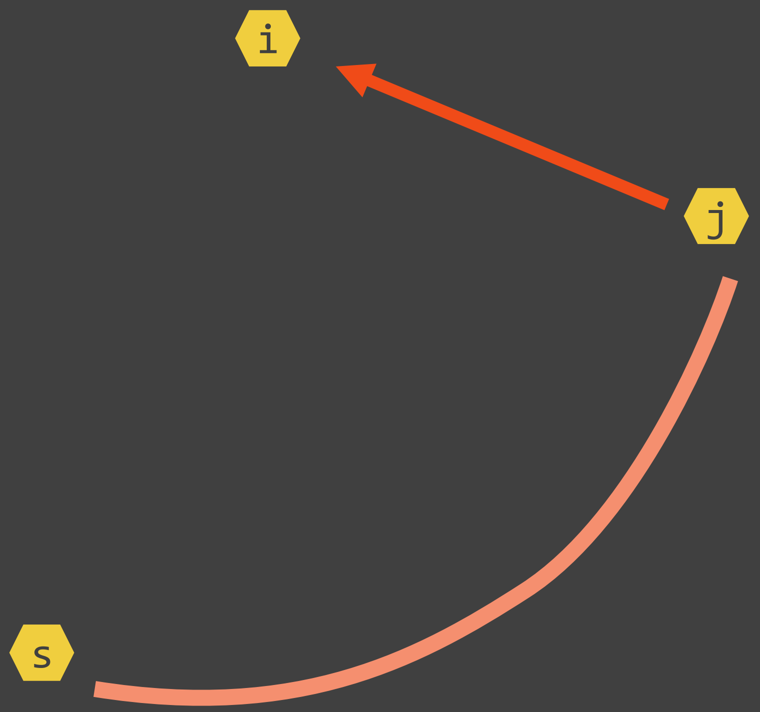
서로 다른 convex hull의 개수는?

$N \leq 300$

- DP로 해결해야 할 듯하다

- DP라면 조금씩 쌓아가야 하는 형태가 필요할 터

- 변을 하나씩 추가해 보자.



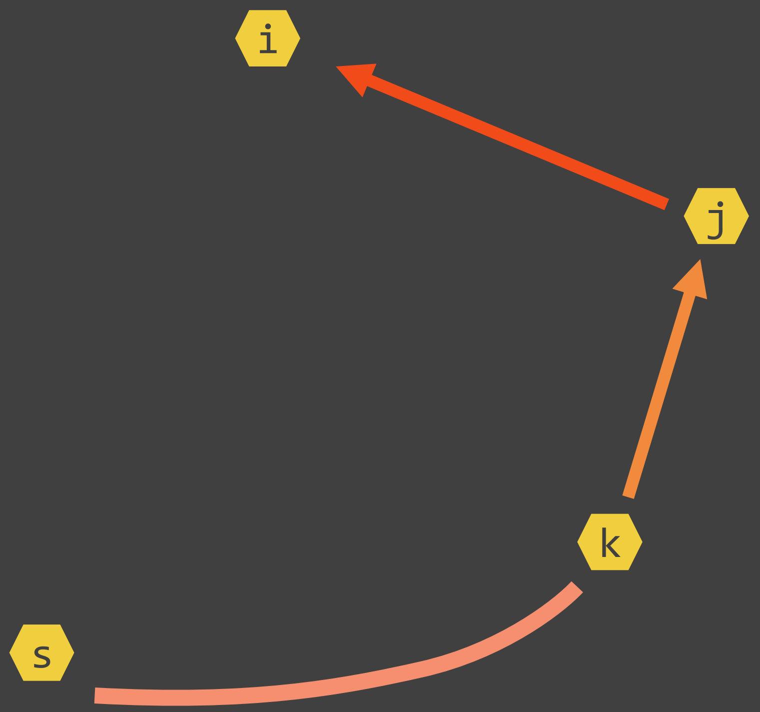
dp[s][i][j]
=(s에서 시작해 볼록하게 만든 '껍질의 일부' 중, 마지막 변이 j→i인 것의 개수)

```
dp[s][i][j]
=(s에서 시작해 볼록하게 만든 껍질의 일부 중,
마지막 변이 j→i인 것의 개수)

=sum(dp[s][j][k]) for all possible k
```

엥? 이런 식으로 하면 DP가 정의가 안 되지 않나?
(저렇게 쓰고 보면 dp가 서로 무한히 의존하지 않나?)

맞습니다.
적당한 제한과 순서를 잡아서 계산해야 할 것.



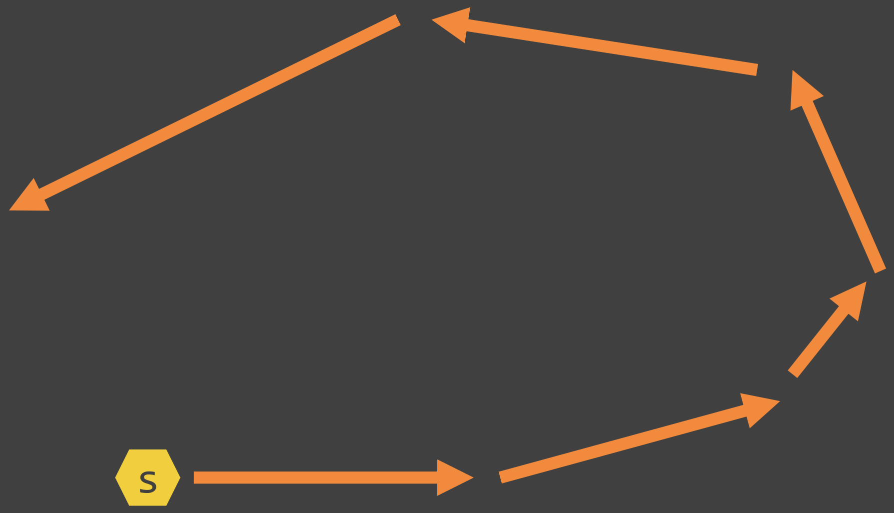
제가 임의로 정한 규칙은

"시작점은 항상 convex hull의 가장 아래,
그 중에서는 가장 왼쪽 점으로 잡는다"

+

"각도가 커지는 순서대로 간선을 추가해 나갈 것"

※ x축 양의 방향 기준,
0 이상 2π 미만의 각도

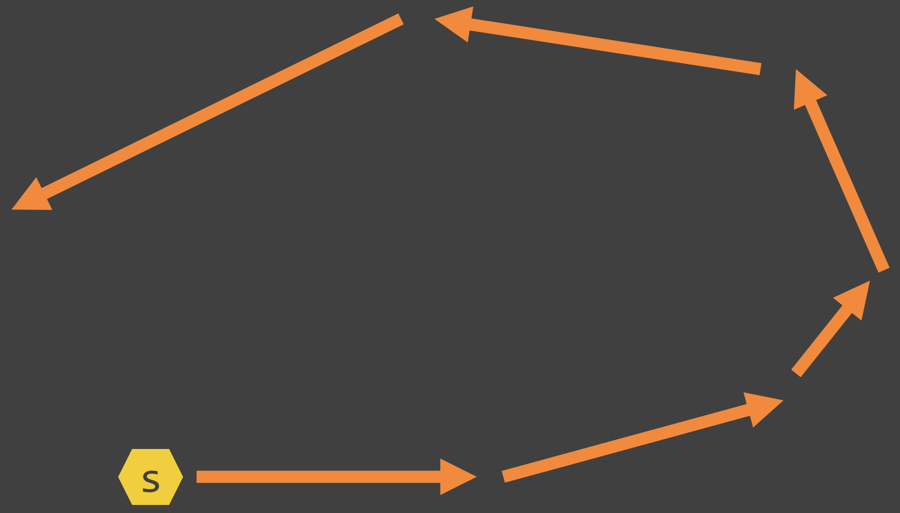


“각도가 커지는 순서대로 간선을 추가해 나갈 것”

덕분에 할 수 있는 접근
: 간선을 각도 순으로 정렬하여 그 순서대로 DP 갱신에 사용하기.

이러면 여러 바퀴를 돈다든가 하는 문제에 있어 자유로우며,
모든 convex hull이 이 방법으로 표현 가능함.

이는 s를 가장 아래, 가장 왼쪽 점으로 잡았기 때문.
이 점을 기준으로 convex hull을 일주하면,
간선들이 항상 각도가 커지는 순서대로 나타남.

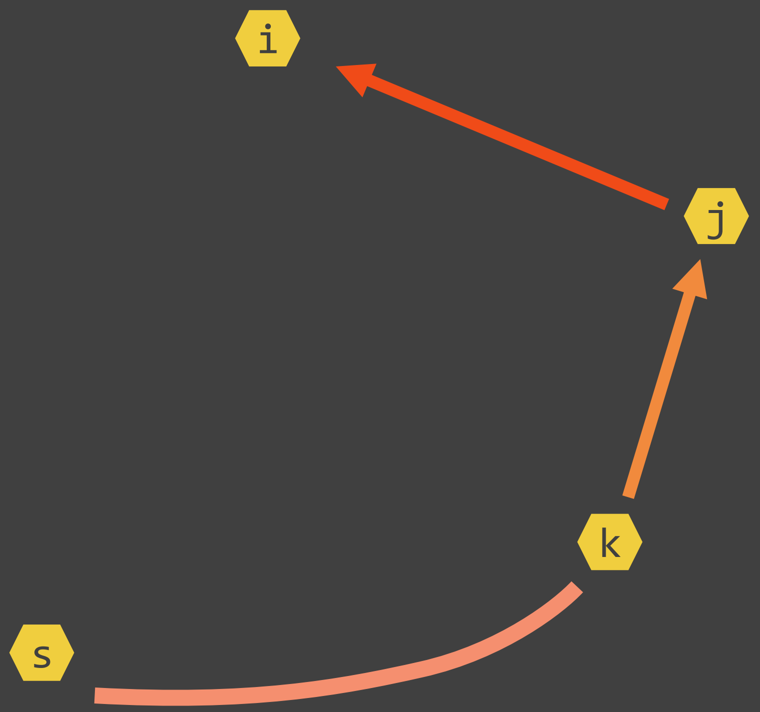


```
dp[s][i][j]
=(s에서 시작해 볼록하게 만든 껍질의 일부 중,
마지막 변이 j→i인 것의 개수)

=sum(dp[s][j][k]) for all possible k
```

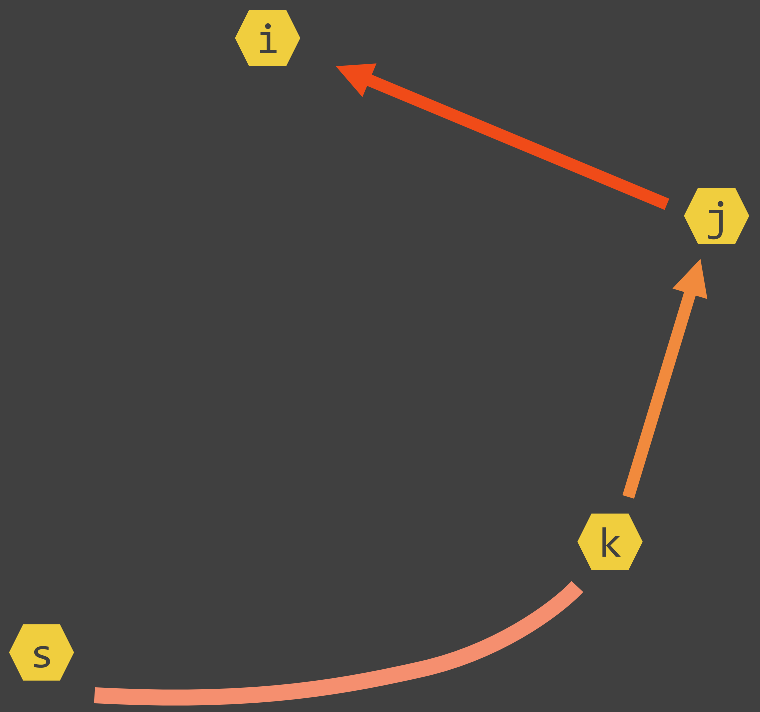
가능한 k의 조건은?

: (k → j)에서 (j → i)가 시계 방향이어야 함.



```
dp[s][i][j]
=(s에서 시작해 볼록하게 만든 껍질의 일부 중,
마지막 변이 j→i인 것의 개수)

=sum(dp[s][j][k]) for all possible k
```



계산량 추산

모든 s에 대해 - $O(N)$

모든 i,j에 대해 - $O(N^2)$

모든 가능한 k를 찾아 - $O(N)$

$= O(N^4)$

$N \leq 300$ 이므로, TLE가 날 듯 함.

(※ 사실은 계산 내용이 간단해서, 정해보다도 빨리 작동하는 AC가 잘만 나왔다고 합니다. 흑흑...)

```
dp[s][i][j]
=(s에서 시작해 볼록하게 만든 껍질의 일부 중,
마지막 변이 j→i인 것의 개수)

=sum(dp[s][j][k]) for all possible k
```

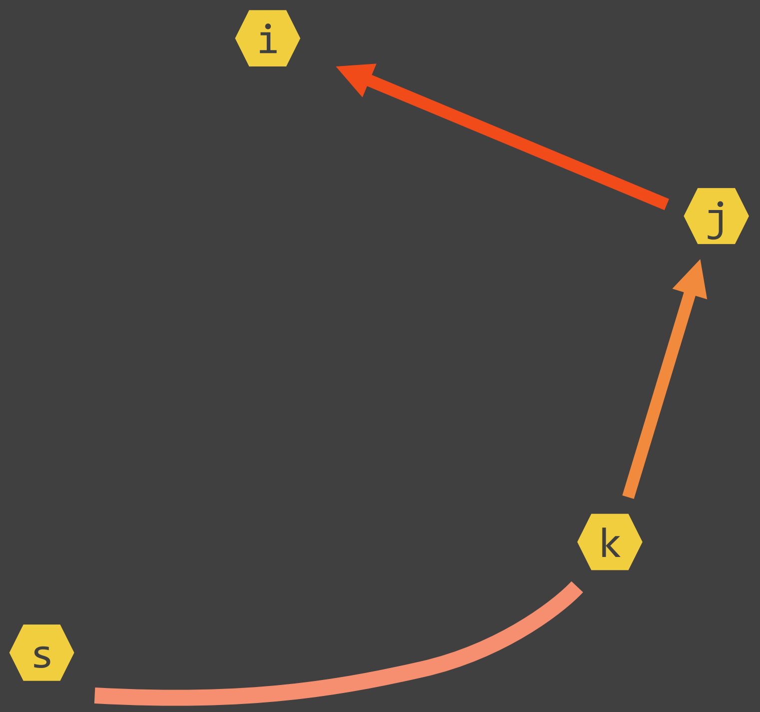
계산량 추산

모든 s에 대해 - $O(N)$

모든 i,j에 대해 - $O(N^2)$

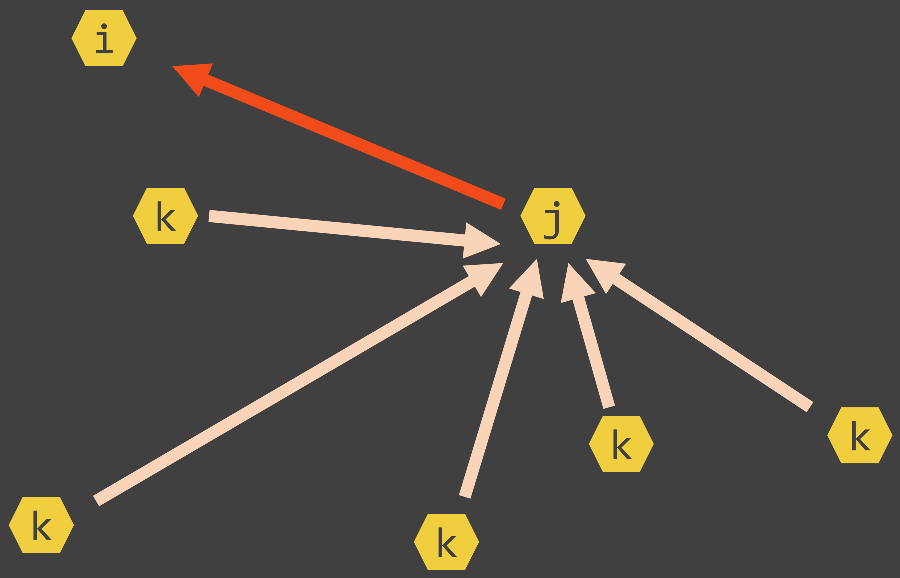
모든 가능한 k를 찾아 - $O(N)$

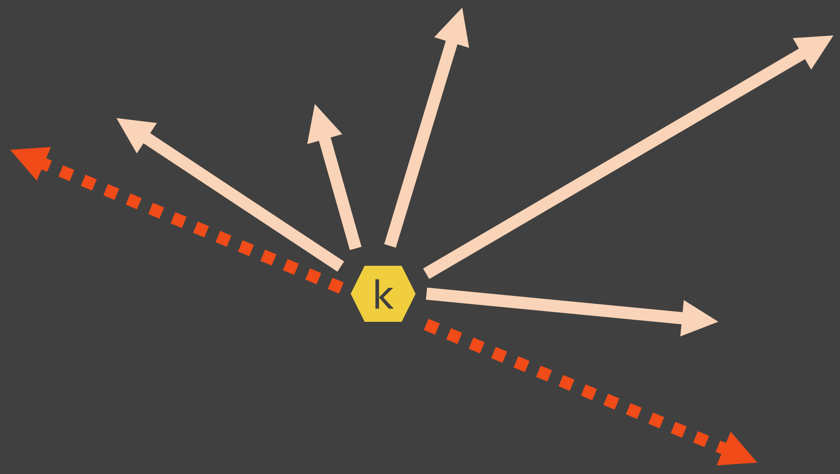
ㄴ꼭 그래야만 할까?

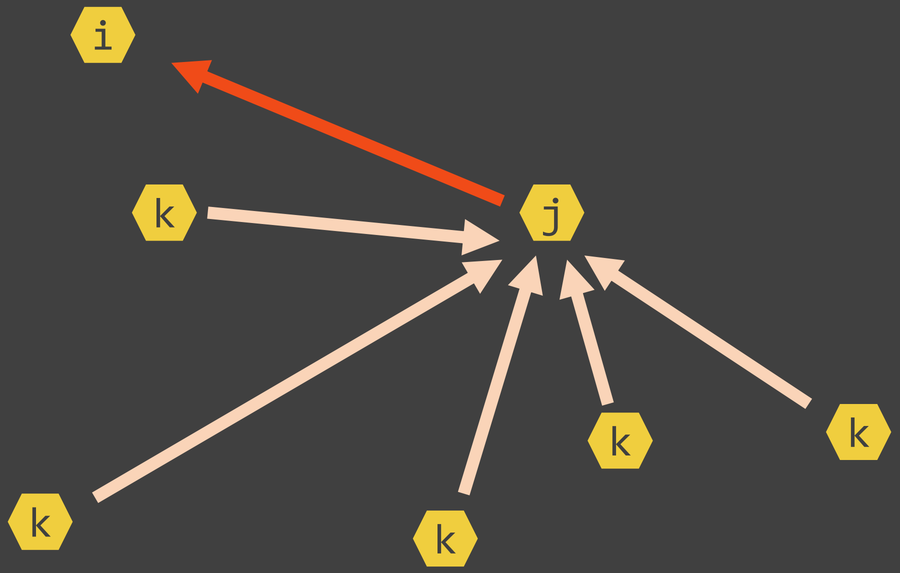


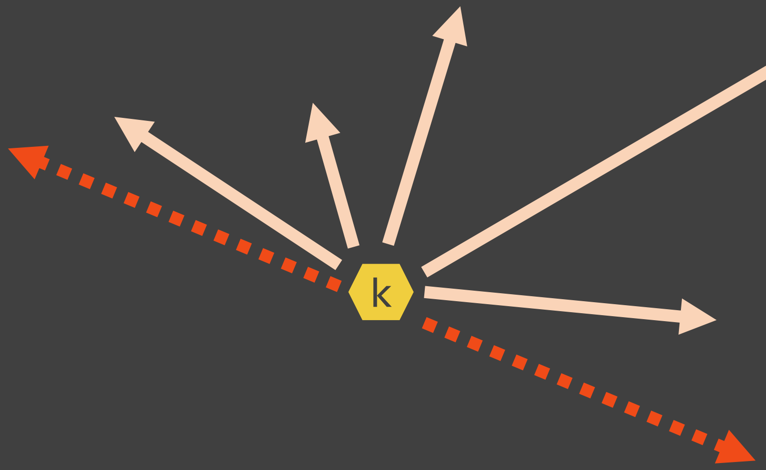
(k → j)에서 (j → i)가 시계 방향인 모든 k에 대해 `sum dp[s][j][k]` 이 모든 k에 대해서 (k → j)들은 각도의 구간을 이룬다.

정확히는, $\vec{v}$=(j → i)일 때, 각도가 ($-\vec{v}$)보다 크고 $\vec{v}$보다 작은 (k → j)들에 대해서 그 k들이 후보가 될 수 있다.









모든 j에 대해서 (k → j)의 각도를 기준으로 k를 정렬해 놓으면,
dp[s][j][k]가 구해질 때마다 j에 대한 어떤 구조에 값을 갱신하고,
dp[s][i][j]를 구할 때 어떤 구조에 특정 구간의 합을 물어보면 된다.

segment tree를 사용해도 좋고,
물어보는 구간의 양 끝점이 항상 증가하므로 inchworm을 사용해도 좋다.

각도 구간에 대해 구간에 해당하는 점을 찾는 과정 등에서

이분탐색을 사용하면 $O(N^3 lgN)$이 소요.

$O(lgN)$에 작동하는 BIT나 세그먼트 트리를 사용해도 되고,

amortized $O(1)$인 inchworm을 사용해도 된다.

설명을 어렵게 했지만, 잘 하면 $O(N^3)$에도 해결 가능하다.

B 구몬 알고리즘

시간 : 1초

출제 : 김동현

순열 P에 대해, (i, P_i) 쌍들을 모두 이은 그래프를 $G(P)$라 하자. $G(P) = X$가 되는 길이 N짜리 순열 P가 l개 이상 r개 이하인 X의 개수를 구해라.

- 문제가 참 난해하다. 해석을 먼저 하자.
- 어떤 순열 하나에 대응되는 그래프는 사이클 여러 개로 이루어져 있다.

- 사이클 컴포넌트들은 각각 독립적 → 각 컴포넌트에 대응되는 순열의 개수를 보자.
- 각 컴포넌트에 포함된 원소의 수가 2개 이하이면 1가지, 3개 이상이면 2가지이다.

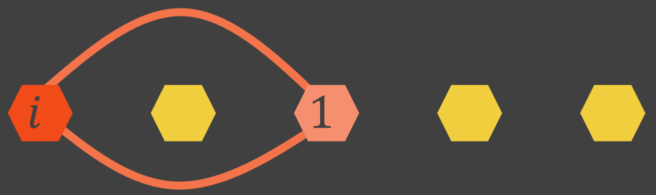
- 즉, 어떤 사이클로만 이루어진 그래프에서 컴포넌트 사이즈가 3 이상인 사이클의 수를 k라 할 때, 그 그래프에 대응되는 순열의 개수는 2^k개이다.
- l, r 모두 2^{31} 이하이므로 $[l, r]$ 구간에 포함되는 모든 2^k에 대해 해당하는 X의 개수를 각각 세 주면 될 것 같다!
- DP 항을 다음과 같이 정의하자.

$D[k][n] =$ 길이 n짜리 순열 P 중 $G(Q) = G(P)$인 순열 Q가 2^k개인 P의 갯수

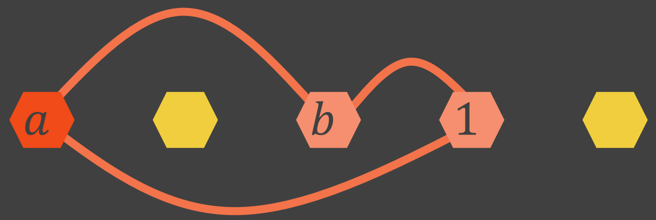
- 답을 구하려면 $l \leq 2^k \leq r$인 모든 k에 대해 $\frac{D[k][N]}{2^k}$을 더하면 된다.

- 수열의 첫 번째 원소가 포함되는 사이클의 컴포넌트 수에 따라 경우를 나누자.

1개 : $D[k][n-1]$



2개 : $(n-1)D[k][n-2]$



$i(\geq 3)$개 : $\frac{(n-1)!}{(n-1-i)!}D[k-1][n-i]$

- 결과적으로 다음과 같은 점화식이 나온다.

$$\boxed{D[k][n] = D[k][n-1] + (n-1)D[k][n-2] + \sum_{i=3}^{n-1} \frac{(n-1)!}{(n-1-i)!} D[k-1][n-i]}$$

- 이대로 계산하면 $O(N^2 \log_2 r)$이라 시간초과가 난다 ㅠㅠ
- $D[k][n]$과 $D[k][n-1]$을 계산할 때 더해지는 시그마 항을 비교해 보자.
- $S[k][n] = (D[k][n]$ 점화식에서 더해지는 Σ 항) 이라 하면

$$S[k][n] = \sum_{i=3}^{n-1} \frac{(n-1)!}{(n-1-i)!} D[k-1][n-i] = \sum_{i=2}^{n-2} (n-1) \frac{(n-2)!}{(n-2-i)!} D[k-1][n-1-i]$$

$$= (n-1)\left((n-2)D[k-1][n-3] + \sum_{i=3}^{n-2} \frac{(n-1)!}{(n-1-i)!} D[k-1][n-i] \right)$$

$$= (n-1)\big((n-2)D[k-1][n-3] + S[k][n-1]\big)$$

$$D[k][n] = D[k][n-1] + (n-1)D[k][n-2] + S[k][n]$$

- 시간복잡도는 $O(N \log_2 r)$.

C 민돌 투어

시간 : 1초
출제 : 김현수

- 0~i에서의 민돌 투어를 i-민돌 투어라고 하고, i-민돌 투어의 수를 C_i로 두자.
- 이때, C_{i-1}에 (i를 끼워 넣는 경우의 수)를 곱하는 방식으로 C_i를 구할 수 있다.

($i = 4$인 경우 예시) 0 - 1 - 3 - 4 - 2 - 0

- i의 뒤에 무엇이 올지는 상관이 없다. (i에서는 $0 \sim (i-1)$까지 모두 도달 가능)
- 따라서, $C_i = C_{i-1} \times (0 \sim (i-1)$ 중 i의 앞에 올 수 있는 트램펄린의 수).
- 답은 $C_N = \prod_{i=1}^{N} (0 \sim (i-1)$ 중 i에 도달할 수 있는 트램펄린의 수).

- 앞의 방식의 정당성을 보장하려면 모든 i−민돌 투어에서 i를 빼면 $(i-1)$−민돌 투어의 조건을 만족해야 한다는 전제가 붙는다.
- i−민돌 투어에서 i의 앞에 오는 것이 x, 뒤에 오는 것이 y라고 두자.

$$x \to i \to y$$

- x에서 도달 가능한 범위는 $[0, k]$ 꼴이므로, x에서 i에 도달 가능하다면 적어도 $[0, i]$ 내에 있는 모든 점에는 도달 가능하다.
- $y < i$이므로, x에서 i에 도달 가능하다면 x에서 y에도 도달 가능하다.
- 따라서, i를 빼더라도 민돌 투어의 조건을 만족하므로 $(i-1)$−민돌 투어가 된다.

D 비밀 요원

시간 : 2초

출제 : 신승원

문제

연결된 평면 그래프가 주어져 있다

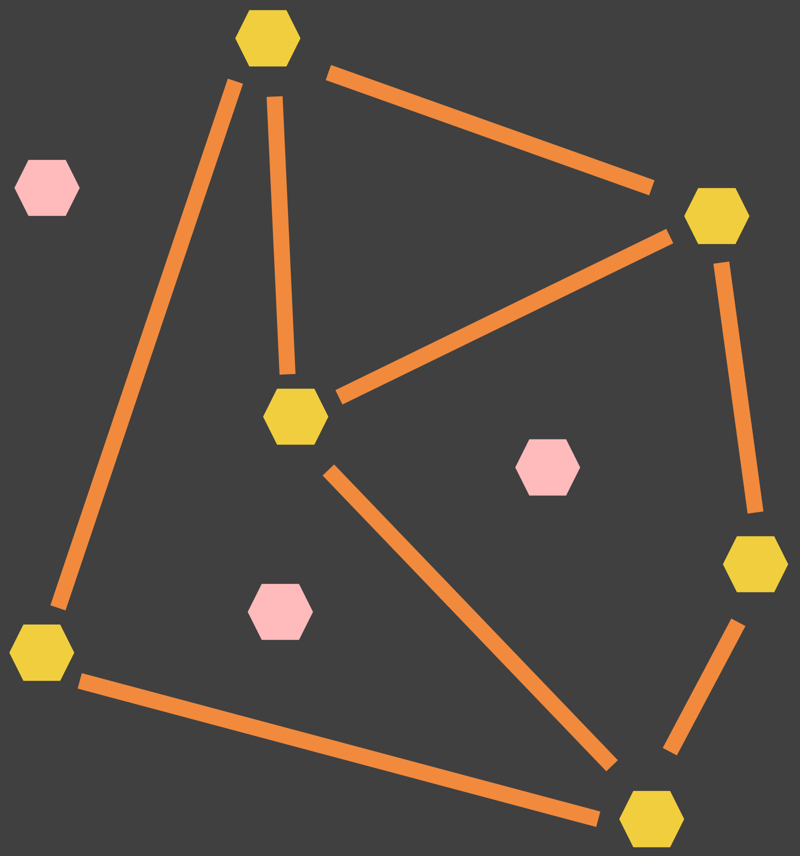
성벽이 가로막고 있는 가운데,

각 점 사이를 이동하는 데에 걸리는 시간을 구하자

$V \le 200{,}000$

$V-1 \le E \le V+100$

$Q \le 200{,}000$



점과 직선으로 이루어진 평면 그래프에서 다음의 식이 성립한다.

$$v - e + f = c$$

여기서

- v =점의 수
- e =모서리의 수
- f =면의 수(무한히 넓은 배경 면은 제외)
- c =컴포넌트의 수

연결 그래프이므로 $c = 1$이고,
조건에서 $v - 1 \leq e \leq v + 100$ 이므로,
$f \leq 101$.

무한히 넓은 배경 면까지 포함해도 최대 102개의 면이 존재함을 알 수 있다.

쿼리로 들어오는 점들은 이 최대 102개의 면 중 하나에 속할 것이다.

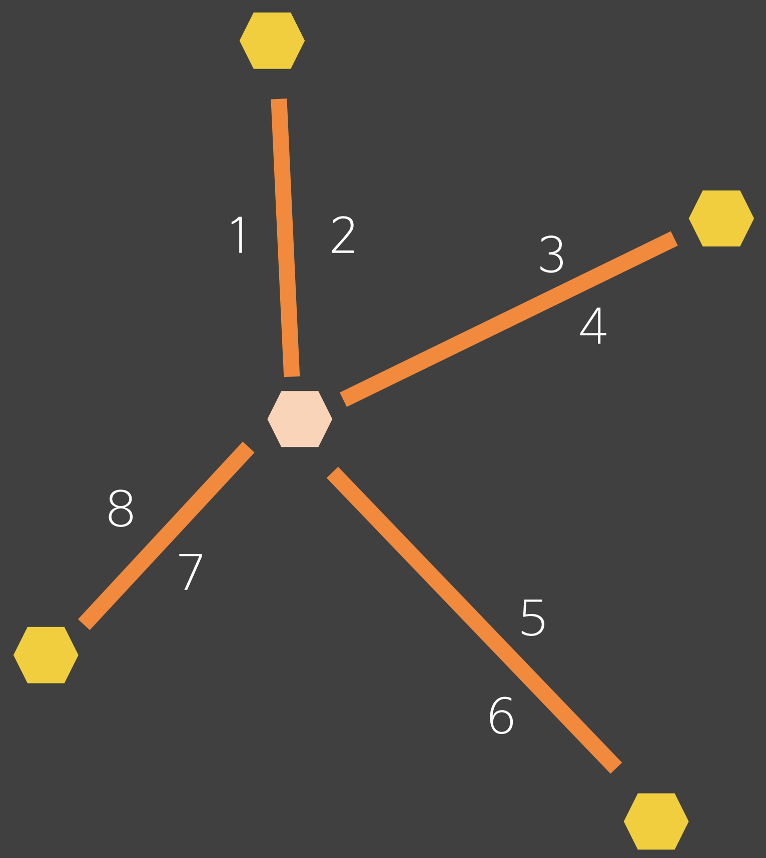
또한 어느 면에 속하는 지만 알면 점의 위치 자체는 중요하지 않음.
각각의 면을 오가는 최단 거리를 많이 답하는 문제로 바뀐다.

면이 102개 이하이므로,
$O(F^3)$의 Floyd-Warshall 알고리즘을 적용할 수 있을 것으로 보인다.

이를 위해서 필요한 것은
① 각각의 면들을 구분해 내고, 각각의 성벽의 양쪽 면을 확인
② 쿼리의 점들이 어떤 면에 속하는지 알아내기

① 각각의 면들을 구분해 내고, 각각의 성벽의 양쪽 면을 확인

각각의 간선(성벽)에 대해 양쪽의 면들에 번호를 붙여보자



① 각각의 면들을 구분해 내고, 각각의 성벽의 양쪽 면을 확인

각각의 간선(성벽)에 대해 양쪽의 면들에 번호를 붙여보자

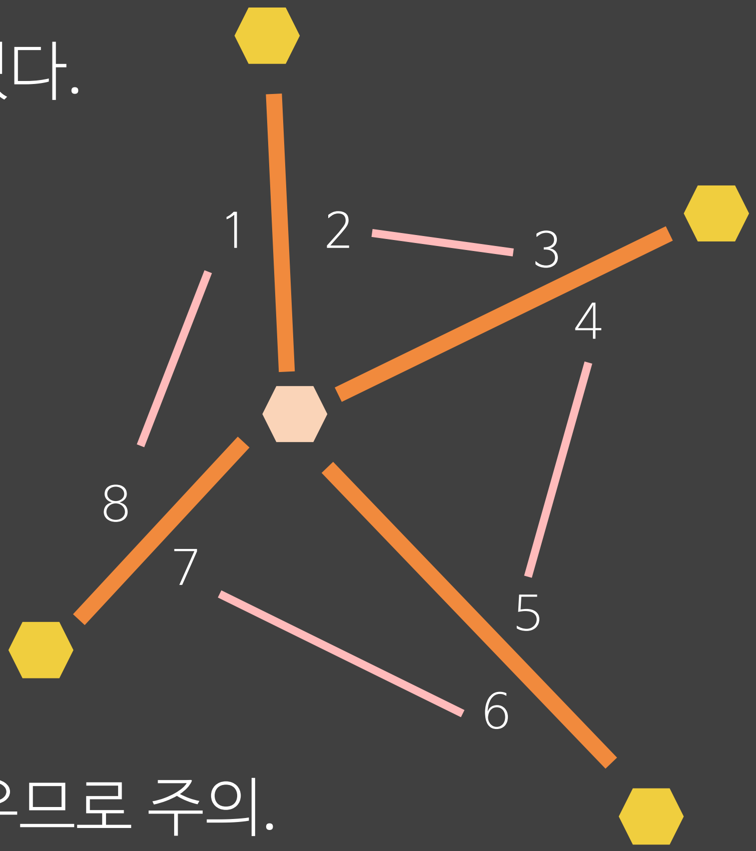
어떤 점을 기준으로 봤을 때, 서로 같은 영역들이 있다.

이들을 합쳐준다. (union-find 이용)

자세히는, 각 점을 기준으로 연결된 점들을

각도 순으로 정렬한 후,

인접한 간선끼리 일치하는 영역을 이어준다.



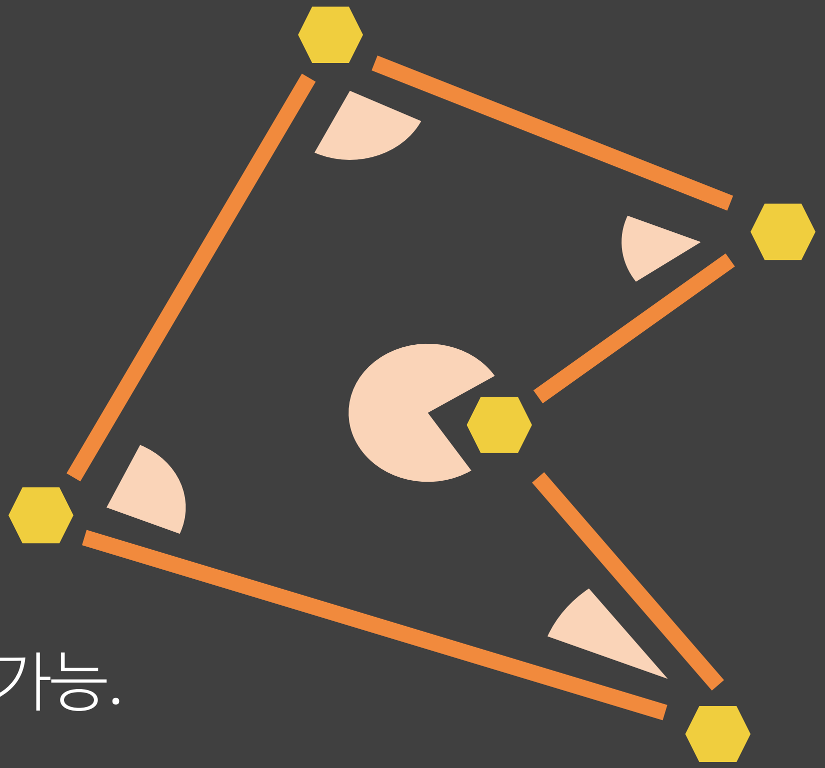
간선의 어느 쪽인지를 판별하는 것이 약간 까다로우므로 주의.

① 각각의 면들을 구분해 내고, 각각의 성벽의 양쪽 면을 확인

즉, 각 점을 기준으로 연결된 점들을 각도 정렬 후,
인접한 영역끼리 이어준다.
이렇게 하면 각각의 면이 하나의 영역으로 묶인다.

또한 바깥쪽에 해당하는 영역도 하나로 묶이는데,
어떤 것인지 미리 파악해 두면 좋다.

'가장 왼쪽, 그 중에서 가장 아래쪽 점의,
기울기가 가장 큰 간선의 시계 반대 방향 쪽' 등으로 파악 가능.



② 쿼리의 점들이 어떤 면에 속하는지 알아내기

x축으로 sweeping하면서 이분탐색을 사용하면 해결 가능.

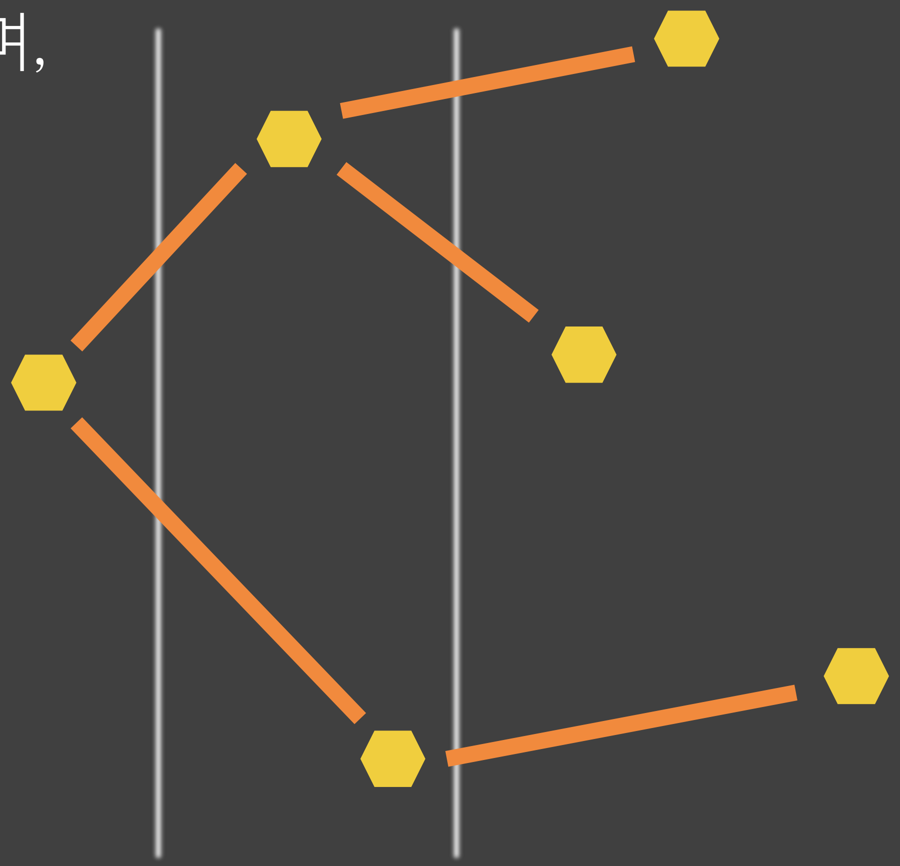
`std::set`에 현재 x좌표에서의 간선들을 저장하며,

정렬 기준은 그 x좌표에서의 y값으로 하자.

간선의 왼쪽 점을 만났을 때 넣고,

오른쪽 점을 만났을 때 빼기로 하자.

(x축에 수직한 간선은 무시하기로 하자.)



② 쿼리의 점들이 어떤 면에 속하는지 알아내기

쿼리 점을 처리할 때가 되면,

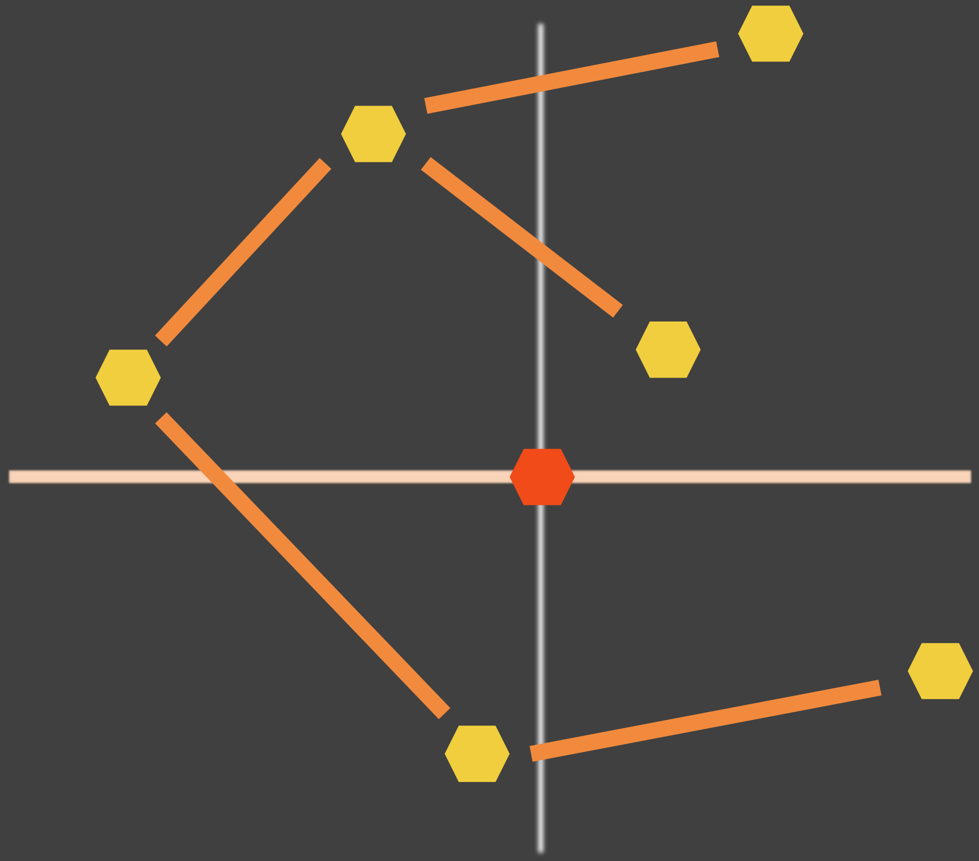
그 점에 해당하는 직선을 잡아서

`std::set`에 `lower_bound`를 물을 수 있다.

위나 아래의 직선을 찾았다면

그를 바탕으로 쿼리 점이 속하는 영역도

알 수 있다.



이렇게 필요한 정보를 모두 구하고 나면

Floyd-Warshall 알고리즘을 이용해 처리하면 된다.

시간 복잡도는 $O(ElgV + (E-V)^3)$.

평면 그래프에서는 이렇게 면을 정점으로 취급하고,

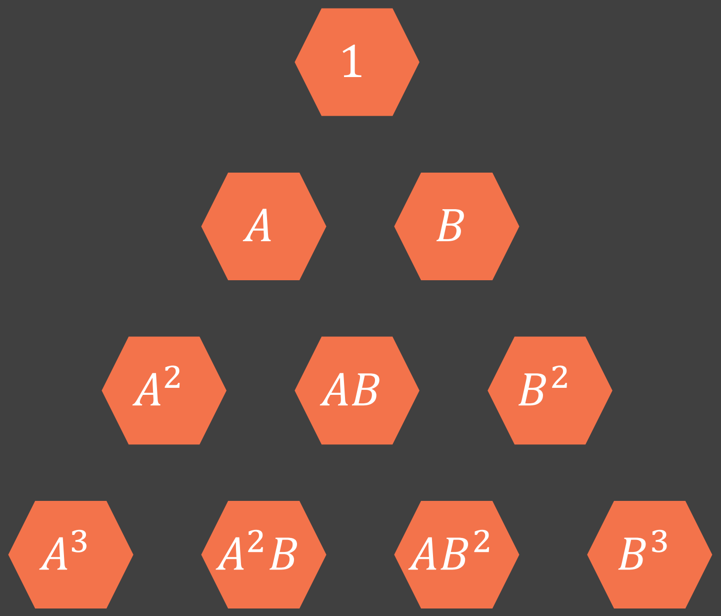
간선을 양쪽 면을 잇는 간선으로 취급하는 “dual graph”라는 개념이 있다.

출제자는 개인적으로 코딩이 굉장히 힘들었다.

질 좋은 평면 그래프와 쿼리 점들을 만드는 과정 자체가 힘들다ㅠㅠ

E 스킬 트리

시간 : 2초
출제 : 김동현



- 커다란 삼각형 모양 격자에서 특정 영역의 합을 빠르게 구하는 문제이다.
- 좌표 범위가 매우 커서 단순히 이중 for문으로 합하면 당연히 시간 초과.
- 등비수열의 합을 적절히 활용해 보자!

- 영역 내 모든 값을 위쪽 꼭지점의 값으로 나누면, 맨 위 꼭지점에 적힌 값이 1이라고 일반화할 수 있다.
- i번째 행의 값의 합을 등비수열의 합으로 표현하고, 이것을 $i = 1$부터 $i = m$까지 합해 보자.
- 등비수열의 합 공식을 이용하면 각 쿼리를 상수 번의 거듭제곱 연산으로 처리할 수 있다.
- 정수의 거듭제곱은 지수를 반씩 줄여나가는 방법으로 $O(\log_2(\text{지수}))$ 시간복잡도에 계산 가능하다. 따라서 총 시간복잡도는 $O(N\log_2(10^{18}))$.
- 모듈러 처리, A나 B가 1일 때 처리 등 예외 처리가 굉장히 많은 문제이다. 구현 시 유의하자.

F 아티스트

시간 : 1초
출제 : 신승원

$$\min\ (\Sigma w_i) \times (\Sigma h_i)$$

$1 \le K \le N \le 3{,}000$

먼저, N개에서 K개를 고를 수 있는 모든 방법들에 대해서

$(\Sigma w_i, \Sigma h_i)$ 라는 점을 평면 상에 찍었다고 하자.

같은 점을 중복해서 세어보면 당연히 ${}_nC_k$개일 것이다.

여기서 $x \times y$ = t가 최소로서 답이 되는 점 A가 있다고 하자.

나머지 모든 점들은 A를 기준으로 xy = t 곡선의 위쪽에 존재하므로,

A에서의 xy = t 곡선의 접선의 위쪽에 존재한다.

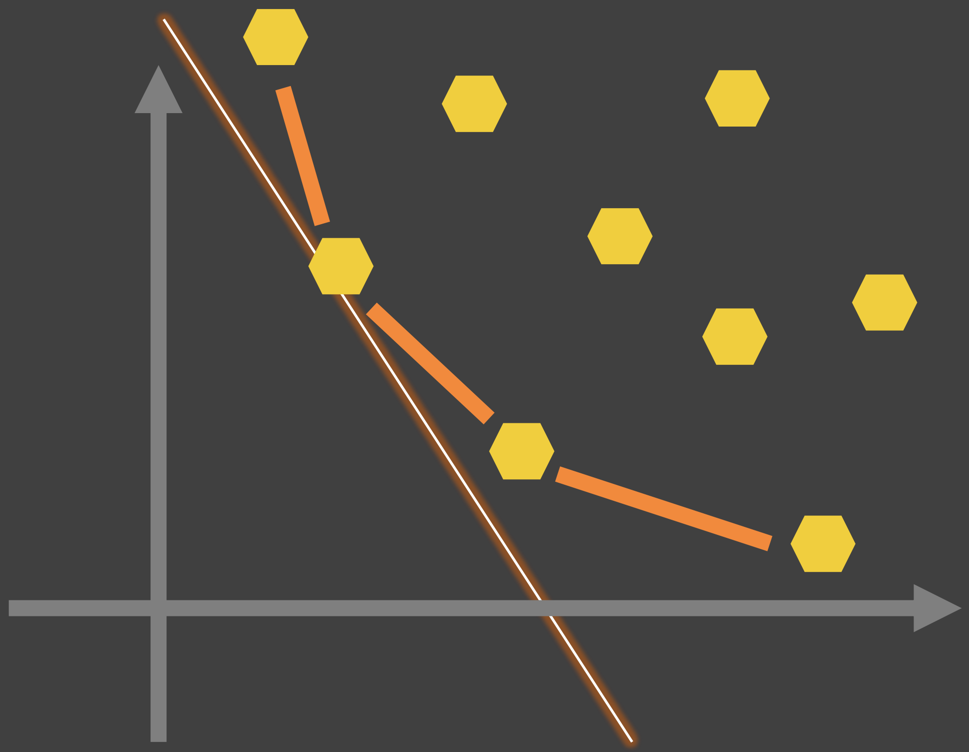
즉, 답이 되는 점 A는 어떤 기울기 (-a/b)에 대해

ax + by의 값이 가장 작은 점에 해당.

(그렇게 되도록 하는 a, b가 존재한다)

그 경우, 그를 구성하는 블록들은

$aw_i + bh_i$ 로 정렬했을 때 작은 것부터 k개를 고르면 된다.



즉 모든 (a, b) 에 대해, 주어진 블록을 $aw_i + bh_i$ 로 정렬하고
작은 것부터 k개를 골라보면, 그중 하나는 답이 된다.

좌표 범위가 너무 커서 가능한 모든 (a, b)를 시도해볼 수는 없다.
하지만, 의미가 있는 기울기는 주어진 점들 사이에 만들어지는 $O(N^2)$개의 기울기 뿐이다.

이 기울기들을 순서대로 보면, 블록의 정렬 순서가 바뀌는 것은
어떤 기울기를 지나갈 때, 그 기울기를 만든 두 블록이 서로 바뀌는 경우만 존재.

따라서 이를 유지하면서 각 순간마다 가장 작은 k개의 Σw_i 및 Σh_i를 유지하며
답을 갱신하면 해결 가능하다.

☆ 별해

이 점들의 아래쪽 convex hull (기울기가 증가하는 부분만)을 잡아 보면,

$x \times y$가 최소가 되는 점은 항상 그 위에 있음을 증명할 수 있다.

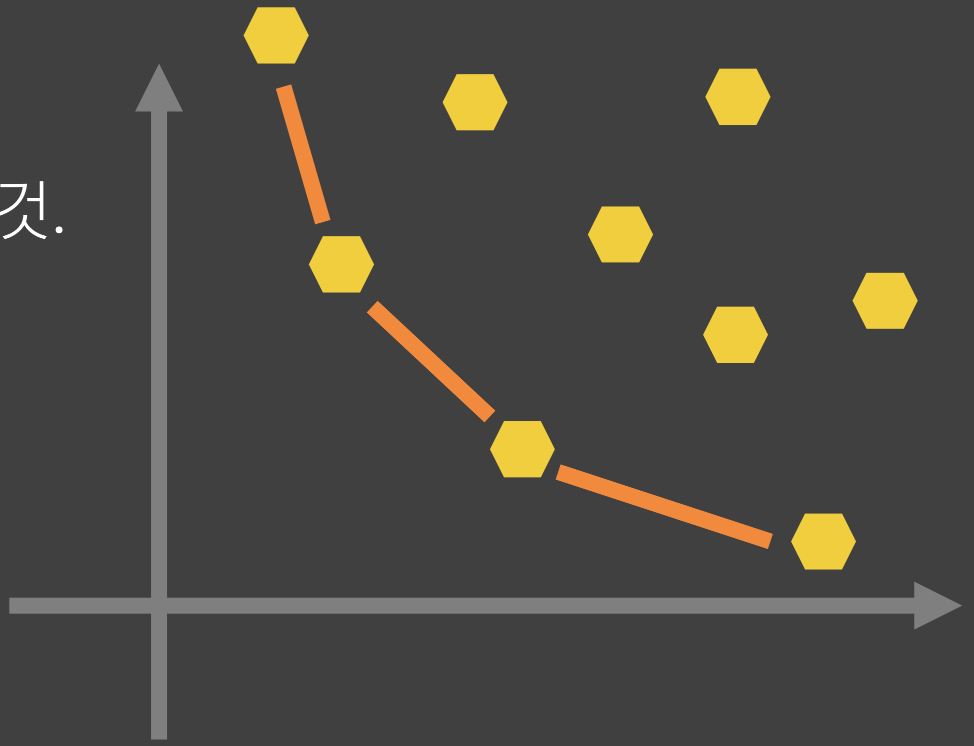
이 hull 위의 모든 점들을 찾을 수 있다면,

각각에 대해서 $x \times y$를 구해 그중 최솟값을 출력하면 될 것.

→ 중요한 것

① 어떻게 모든 점을 찾지?

② 점이 너무 많지 않나?



① hull 위의 모든 점 찾기

앞에서 설명한 것과 유사하게, 특정 기울기의 직선이 hull과 접하는 점을 찾을 수 있다.

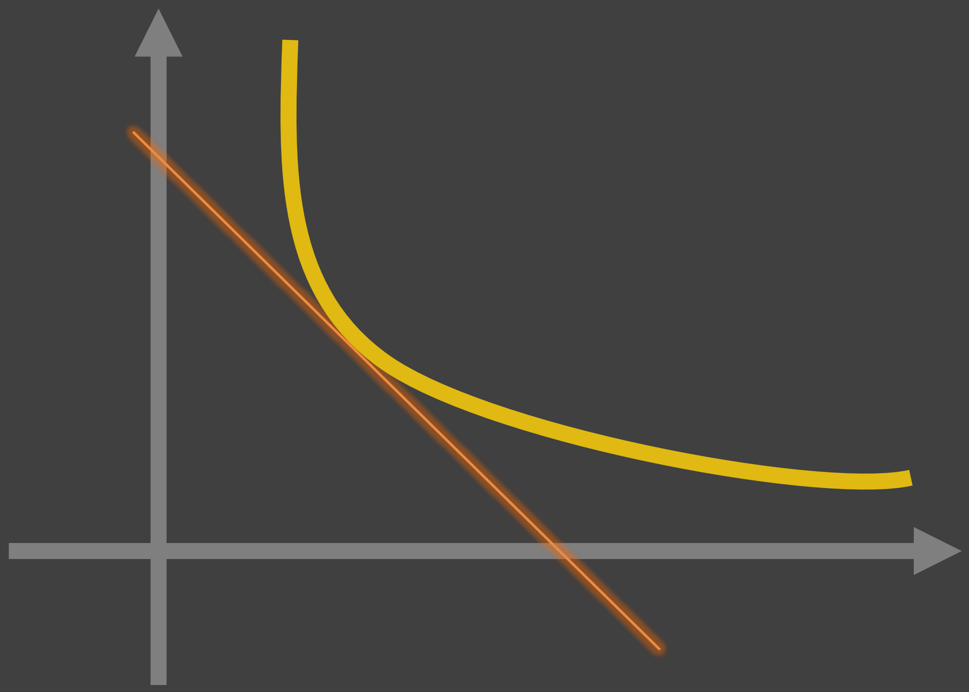
이 직선이 $ax + by + c = 0$ 이라고 하면,

그 점은 모든 점 중 $ax + by$가 최소가 되는 점이기 때문.

$a \cdot \Sigma w_i + b \cdot \Sigma h_i$ 가 최소가 되는 점?

$= \Sigma(aw_i + bh_i)$가 최소가 되는 점

$= (aw_i + bh_i)$로 정렬하여 greedy하게 k개 선택



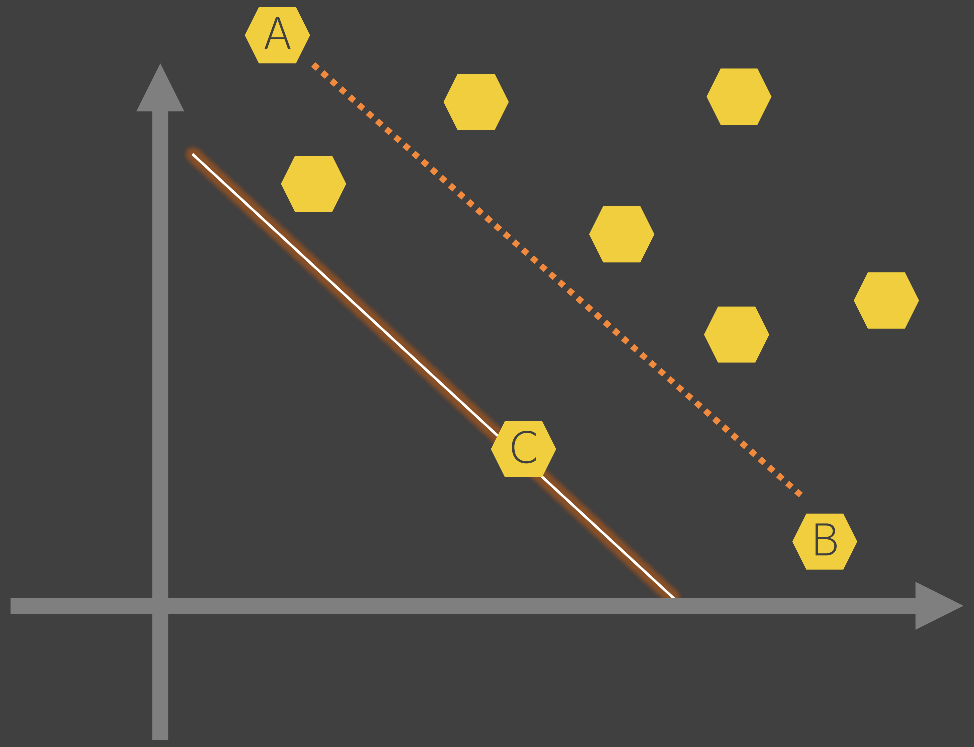
① hull 위의 모든 점 찾기

각 축에 평행한 직선에 접하는 점(그림에서 A, B)은 hull의 양쪽 끝 점.

이 사이에 있는 모든 점들을 찾아보자.

이 둘을 잇는 직선으로 접하는 점을 찾아보자.

(그림에서 C)



① hull 위의 모든 점 찾기

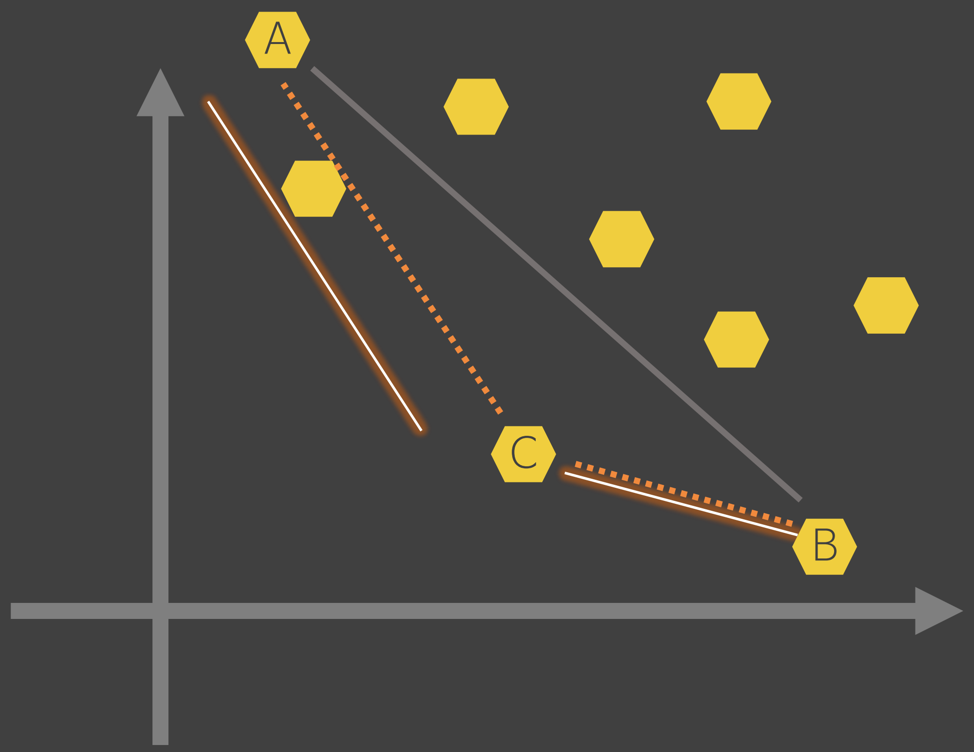
이제 (A, C)와 (C, B)에 대해

같은 작업을 재귀적으로 반복해주기만 하면 된다.

만일 그렇게 찾은 점이 A나 B라면 멈춘다.

이렇게 하면 hull 위의 모든 점을

간단하게 찾을 수 있다.



② 점의 개수에 대한 고민

“hull 위의 점이 너무 많으면 어떡하지?”

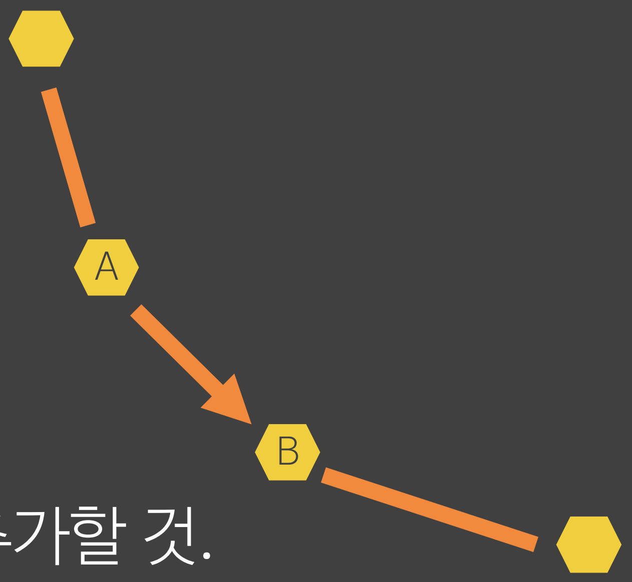
hull 위의 인접한 점들 사이의 관계를 생각해보자.

A상태에서 B상태로 갈 때,

고른 블록들의 집합에 변화가 있을 것이다.

어떤 개수만큼 없애고, 그 개수만큼 새로운 블록을 추가할 것.

이 때, 그 개수가 두 개 이상이라면?



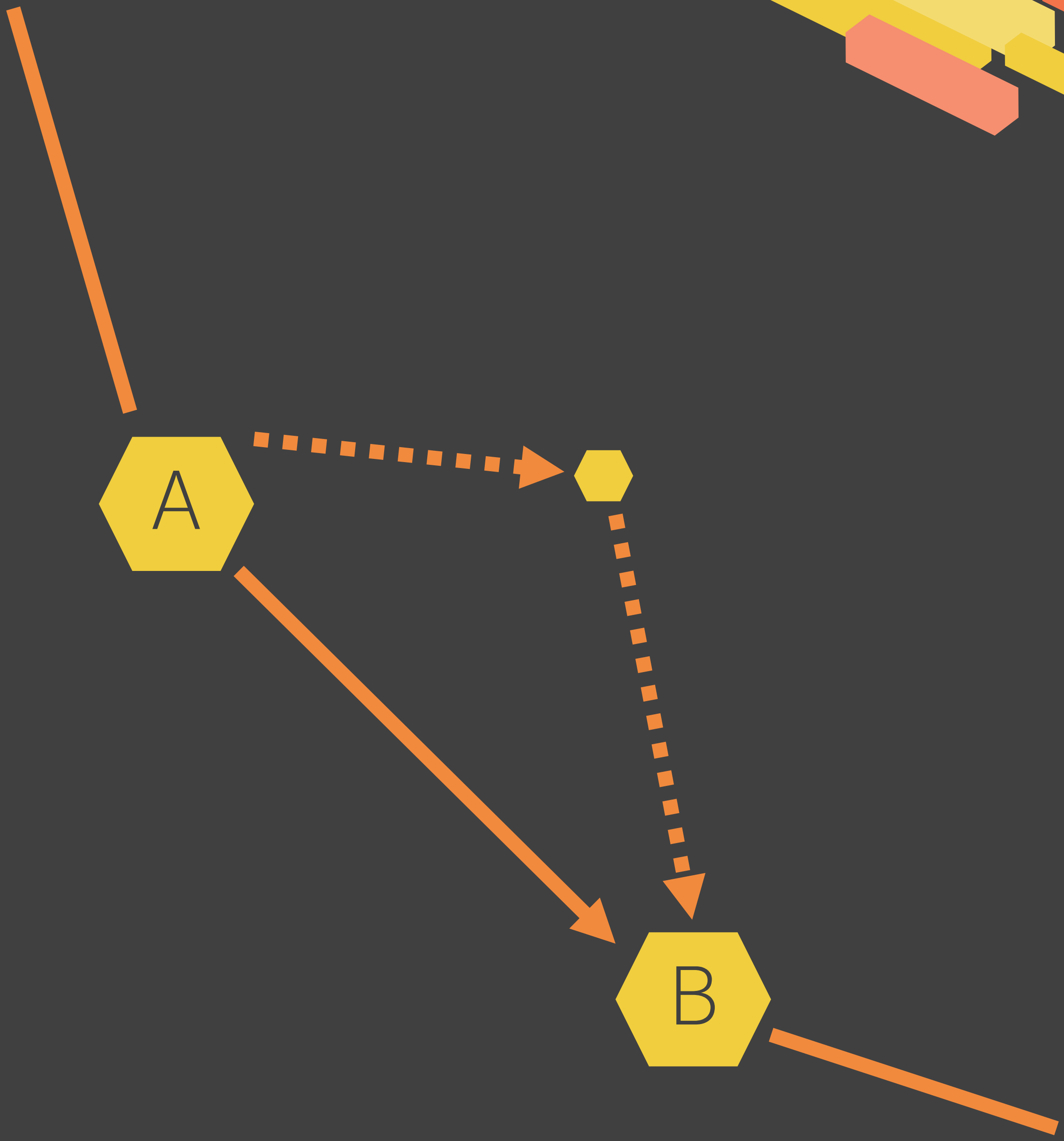
② 점의 개수에 대한 고민

‘하나 빼고, 하나 넣고’의 조합으로

쪼갤 수 있을 것이다.

그러면 A→B의 경로를

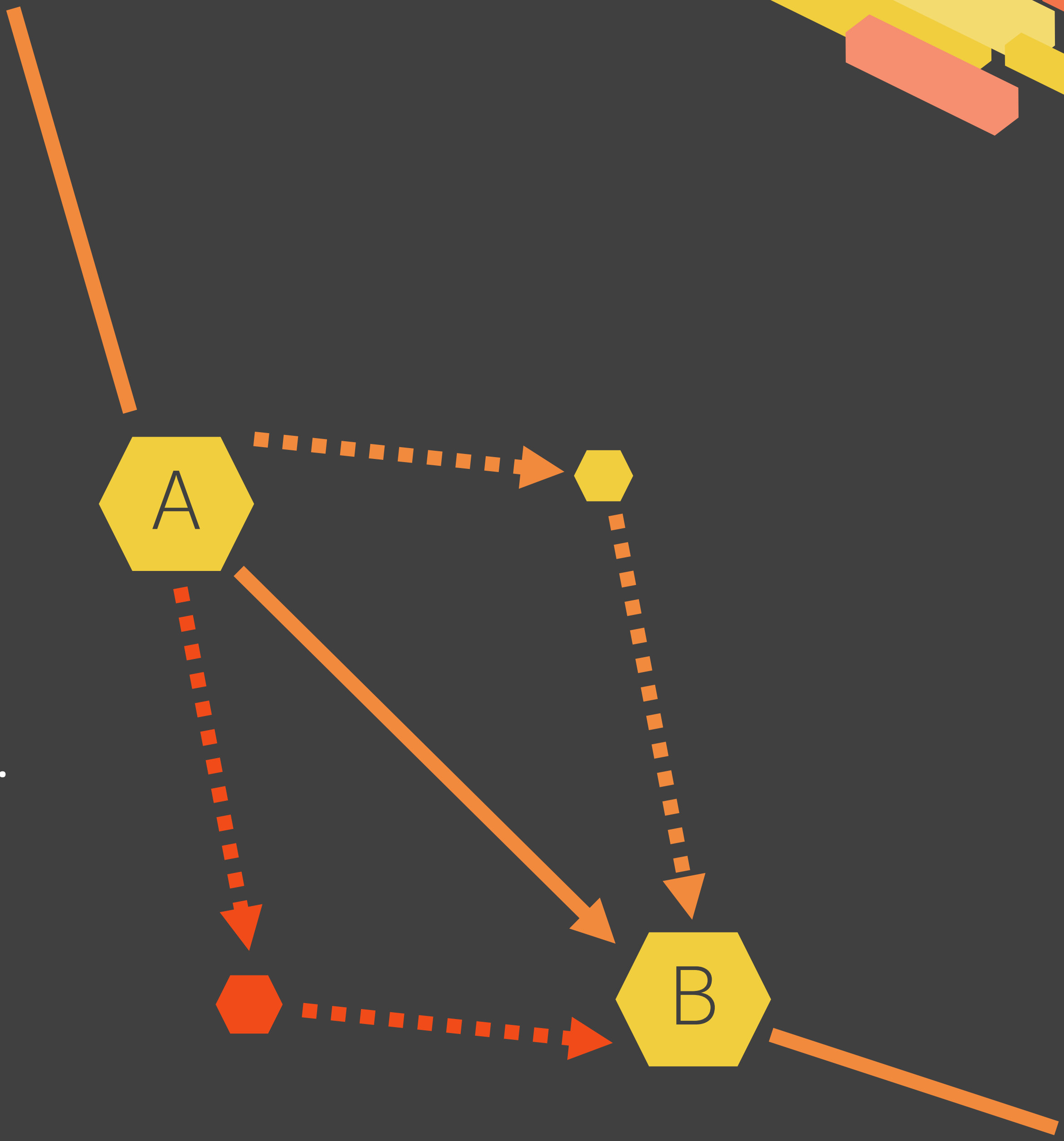
몇 개의 점을 거치는 것으로 쪼갤 수 있다.



② 점의 개수에 대한 고민

그 거치는 점들이 hull의 아래쪽에 있다면,
convex hull의 정의에 모순.

그 거치는 점들이 hull의 위쪽에 있다면,
그 방법을 거꾸로 적용하면
A에서 hull의 아래쪽으로도 갈 수 있게 된다.
이 역시 정의에 모순.



② 점의 개수에 대한 고민

따라서 hull의 인접한 점 사이에는,

하나를 빼고 하나를 추가하는 조작만이 존재한다.

…

이를 바탕으로 점이 N개 이하임을 증명해 나갈 수 있을 것 같았는데 잘 안 됐다ㅠㅠ

경험적으로는 N개 이하인데, 심심하면 각자 증명 or 반증해 보자.

G 약 팔기

시간 : 1초
출제 : 김동현

- 100만 이하의 모든 자연수를 구간 합으로 가지는 길이 2000 이하의 배열을 만들자.
- $2000 = 1000 + 1000$
- $1000 \times 1000 = 100$만
- 100만 미만의 모든 수는 1000진법 수 두 자리로 표현할 수 있다…?
- 1의 자리는 1 1000개, 1000의 자리는 1000 1000개…?
- 1 천 개, 1000 천 개를 순서대로 출력하면 된다!
- $k = 1000q + r$ $(0 \le q \le 1000,\ 0 \le r < 1000)$일 때
  $[1001 - r, 1000 + q]$ 구간의 합이 k가 된다.
- 사실 1 한 개는 빼도 되지만 '일천천천'이 뭔가 어감이 예쁘다 (?)

H 이름 궁합

시간 : 1초

출제 : 김동현

- 길이 4000 이하의 정수 배열이 있을 때, 인접한 두 원소를 합해서 10으로 나눈 나머지를 순서대로 쓰는 작업을 계속 했을 때 마지막에 남는 두 원소를 출력하라.

- 하라는 그대로 구현하면 된다 ^^
- 2차원 배열을 써도 되고, 메모리를 절약하고 싶다면 1차원 배열 두 개를 왔다갔다하면서 계산해도 된다. (이 기법을 토글링이라고 한다)
- 단순 구현으로 시간복잡도 $O(|A|^2)$. ($|A|$ = 문자열 길이)

정과프 해적단

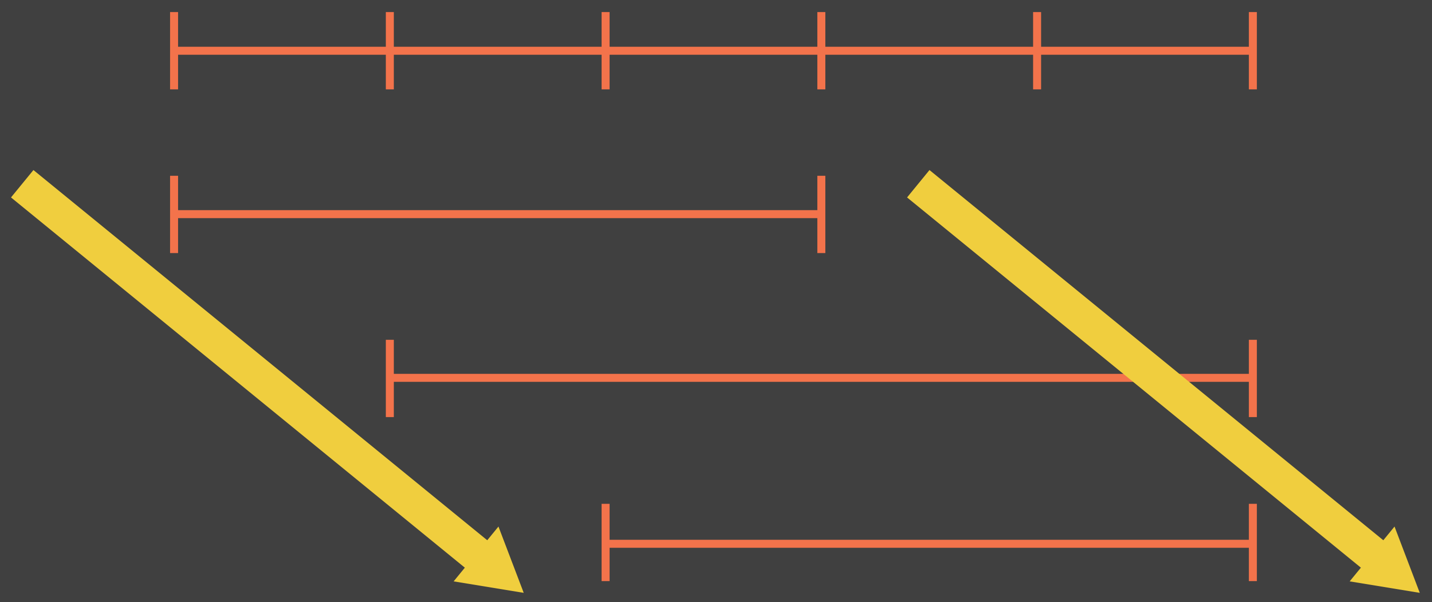
시간 : 1초

출제 : 김동현

- 최적해는 어떤 일정한 강도 범위 안에 있는 보물만 가져간 것으로 생각할 수 있다.
- 보물이 총 N개 있을 때 답이 될 수 있는 [최소 강도, 최대 강도]의 후보는 두 보물 간 가치 차이, 즉 $O(N^2)$개 밖에 없다는 사실을 일단 알 수 있다.
- [최소 강도, 최대 강도] 쌍을 정해 놓았다면, 그 범위 안에 들어오는 섬들만 고려했을 때 얻을 수 있는 가치의 최댓값을 구하면 된다.

- 문제 조건에 의해, 섬들 간에 오고 갈 수 있는 관계는 DAG이다.
- $D[i]=$(섬 i에 도달했을 때까지 얻을 수 있는 최대 가치) 로 정의하자.
- 점화식은 $D[i]=\max_{x_j\le x_i, y_j\le y_i\text{인 } j} D[j]+v_i$로 간단하게 주어진다.
- 점들을 미리 x좌표 오름차순 (같을 때는 y좌표 오름차순) 으로 정렬해 놓았다면 세그먼트 트리를 사용해 $O(N\log_2 N)$ 만에 모든 $D[i]$를 구할 수 있다.

- 이대로면 아직 $O(N^3 \log_2 N)$이다 ㅠㅠ
- 최소 강도 l이 정해져 있을 때, M 이상의 가치를 가져갈 수 있도록 하는 최소의 최대 강도 $f(l)$을 살펴보자.
- 1. l이 증가하면, $f(l)$은 감소하지 않을 것이다.
  - $l < l'$일 때 $f(l') = t$라면 강도가 $[l', t]$ 구간에 속하는 섬만 고려해도 M 이상 벌 수 있으므로 $[l, t]$ 구간에 속하는 섬만 고려하면 자명히 M 이상 벌 수 있음. 즉 $f(l) \leq t = f(l')$.
- 2. 임의의 l에 대해 '$[l, t]$ 구간에 속하는 섬만 고려해도 M 이상 벌 수 있다'는 '$t \geq f(l)$이다'와 동치.



- 이 두 가지 사실로부터 inchworm 기법이 사용 가능!
  - 어떤 l에 대해 $f(l)$을 구했다면, 그 다음 l'에 대해 $f(l')$을 구할 때는 $f(l)$에서 시작해 탐색하면 된다.
- DP 계산을 해야 하는 후보가 $O(N)$개로 줄었으니 시간복잡도는 $O(N^2 \log_2 N)$.
- Inchworm이 아닌 Parametric search를 쓰면 시간초과가 나도록 했다 ^^;;

J 쿵! 쿵!

시간 : 2초
출제 : 신승원

문제에서 요구하는 내용을 그대로 따르면 충분히 AC를 받을 수 있다.
도형을 하나씩 추가하며, 이전에 있던 도형과의 서로 다른 교점을 센 후…

도형이 직선인 경우, 이 직선은 (교점 수)+1 개의 조각으로 나뉘며,
각각의 조각은 이전에 하나였던 영역을 두 개로 나누므로 그만큼 영역이 추가된다.
단 지금까지 직선이 나온 적이 없다면 양쪽 끝의 직선은 같은 영역을 나누므로 1 적다.
단 모든 도형을 통틀어서 이 것이 첫 번째 도형이라면 위와 같이 1 줄여서는 안 된다.

도형이 원인 경우, 이 원은 (교점 수)개의 조각으로 나뉘며,
마찬가지로 그만큼의 영역이 추가된다.
단 이 원이 첫 도형인 경우 무조건 한 개의 영역이 추가된다.
(두 번째부터는 무조건 원점이라는 교점이 있으므로 문제 없음)

라는 방법도 있지만…

do you know...

...inversive geometry?

Inversive geometry

원점을 제외한 모든 점에 대해, 원점과의 거리가 r이라고 할 때,
같은 방향이고 거리가 1/r인 점으로 옮기는 변환을 취한다.

원점을 지나는 직선은 자기 자신으로,
원점을 지나는 원은 원점을 지나지 않는 직선으로,
인접한 영역은 인접한 영역으로
각각 옮겨짐을 알 수 있다.

이렇게 옮겨도 영역의 개수는 변하지 않는다.

하지만 모든 도형이 직선으로 바뀌므로, 코딩이 약간 더 간편하다.

이 경우, 앞에서는 원과 원, 원과 직선의 교점의 식을 구해야 했던 것과 달리,

직선과 직선의 교점의 식을 구해야 하게 된다.

참고로 원점과 (a, b)를 지름으로 하는 원은 $ax + by = 1$라는 직선으로 옮겨진다.

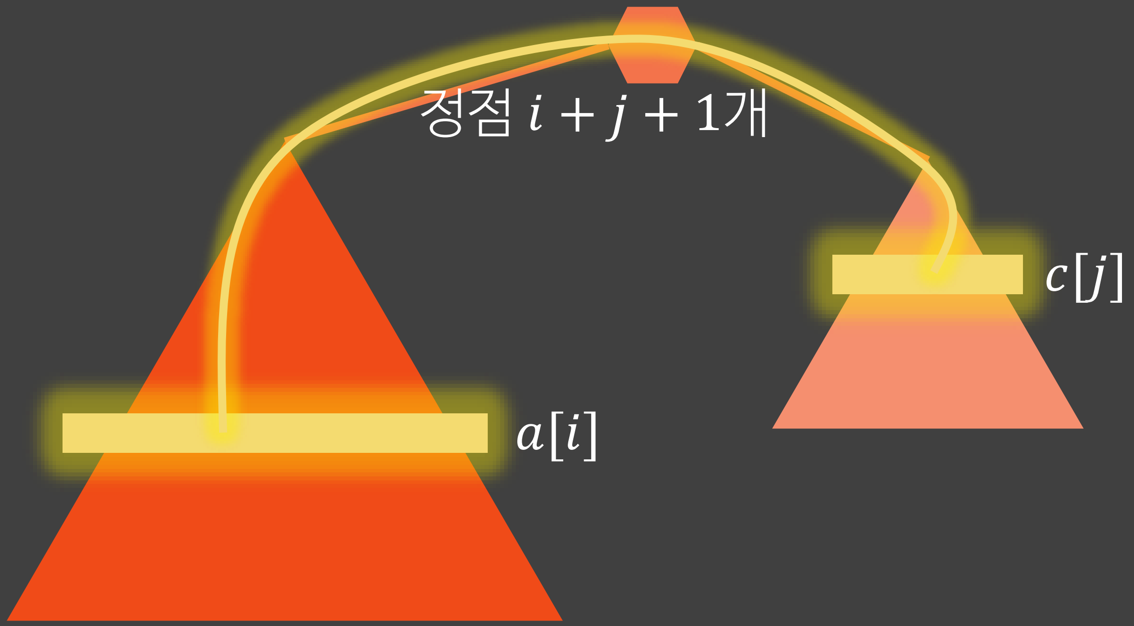
배경 지식이 있으면 약간 더 편리하지만

없다고 해도 난이도 차이가 크지 않은 문제.

K 태풍의 아들 KDH

시간 : 1초
출제 : 김동현

- $q = 1 - p$라 하자. 임의의 두 점을 잇는 경로가 '살아 남을' 확률은 경로상의 모든 마을이 살아남을 확률이므로, 그 경로에 대한 교통량의 기댓값은 두 점 사이의 거리를 d라 할 때 dq^{d+1}이다.
- 기댓값은 가산성이 성립하므로 결국 구하고자 하는 답은 $\sum_{i \neq j} d(i,j)q^{d(i,j)+1}$이다.
- 단순히 계산하면 $O(N^2)$.
- 트리 상의 경로에 대해 고려할 때는 Centroid Decomposition을 한 번 쯤 생각해 볼 수 있다.
- 어떤 트리의 루트를 지나는 모든 경로에 대한 기댓값의 합을 $O(\text{트리크기})$ 만에 구할 수 있다면 문제가 $O(N \log_2 N)$ 만에 해결된다.



$a[i] \times c[j] \times q^{i+j+1}$

- 서브트리를 하나씩 '합쳐 나가며' 생각하자.
- $a[i]$ = (지금까지 본 서브트리들에서 깊이가 i인 노드의 수),
  $c[i]$ =(이번에 보는 서브트리에서 깊이가 i인 노드의 수) 라 하자.
- (이전에 본 서브트리) – (루트) – (이번에 보는 서브트리) 의 형태로 지나는 모든 경로에 대한 기댓값의 합은 $\sum(i+j)a[i]c[j]q^{i+j+1}$.

- $S = \sum ia[i]q^i$, $T = \sum a[i]q^i$라 두면 앞의 식을 $\sum(S + jT)q^{j+1}$로 쓸 수 있다.
- 각 서브트리에 대해 O(서브트리크기) 만에 기댓값 가산, $a[i], S, T$ 값 갱신이 모두 가능하다!
- 루트를 한 끝점으로 하는 경로들은 S와 T의 초기값에 ‘깊이가 0인 노드’ 하나의 정보를 넣어 주면 같이 계산할 수 있다. (즉, $S_0 = 0, T_0 = 1$)
- Centroid Decomposition으로 $O(N \log_2 N)$.

★별해★

- Centroid Decomposition을 쓰지 않고 각 간선에 대해 그 간선에 더해지는 고통량의 기댓값을 모두 더하는 방식으로도 가능하다.
- 각 간선에 더해지는 기댓값은 서브트리 두 개에 대한 특정 값의 곱으로 나타낼 수 있다.
- 임의의 트리의 서로 다른 서브트리의 수는 $2N-2$개이기 때문에, 각 서브트리에 해당하는 값을 DP를 사용해 구하면 $O(N)$에 문제를 해결할 수 있다.

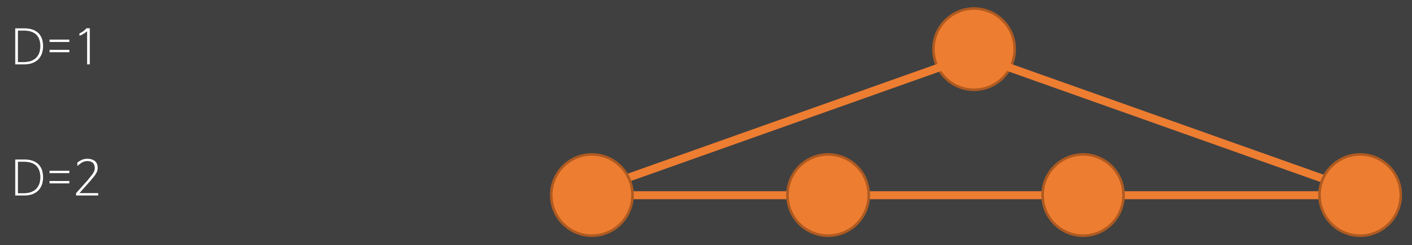
L 현수시티

시간 : 1초
출제 : 김현수

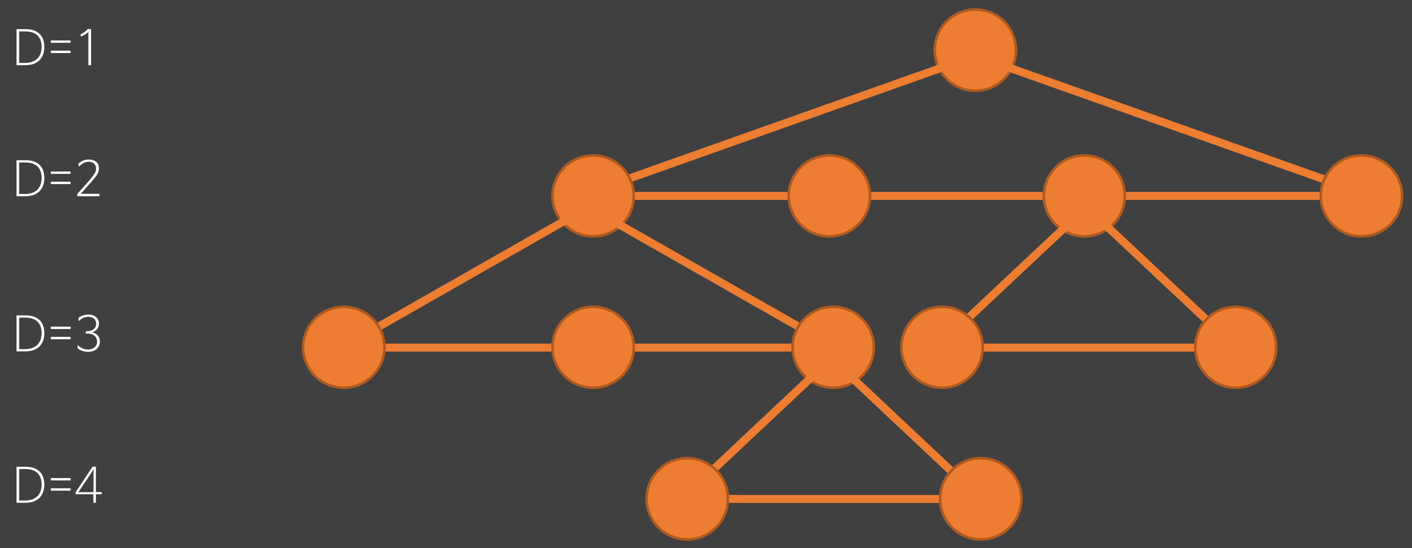
- 설명의 편의를 위해 모든 간선이 정확히 하나의 단순 사이클에 포함되었다고 가정하겠으나, 사이클에 포함되지 않은 간선이 있어도 크게 달라지는 것은 없다.
- 먼저, 입력을 받은 그래프에서 임의로 루트를 잡고 깊이를 1로 둔다.

D=1

- 루트를 포함하는 단순 사이클 중 추가되지 않은 것을 추가한다.
- 새로 추가된 정점의 깊이는 (마지막으로 추가된 정점의 깊이 + 1)로 한다.



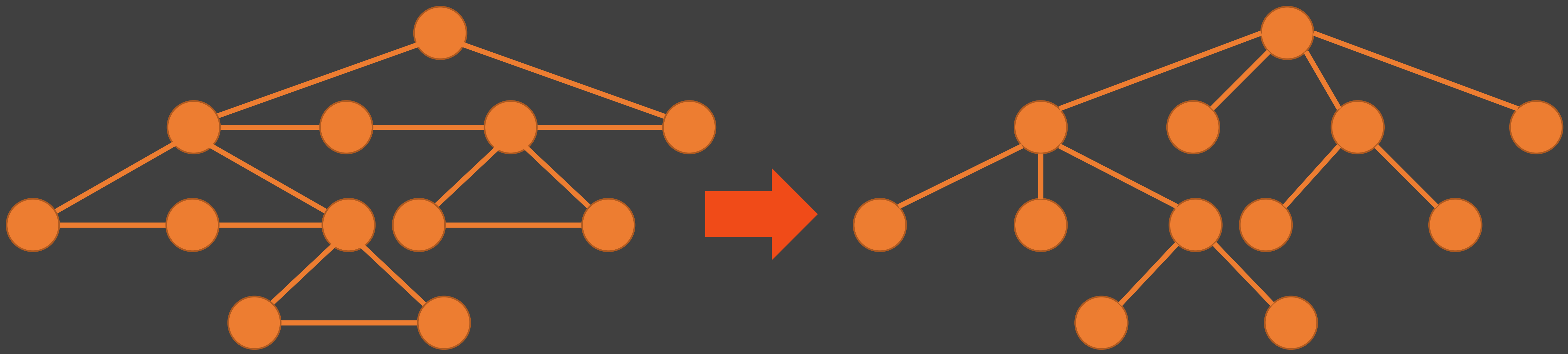
- 비슷한 과정을 모든 간선/정점이 포함될 때까지 반복한다.



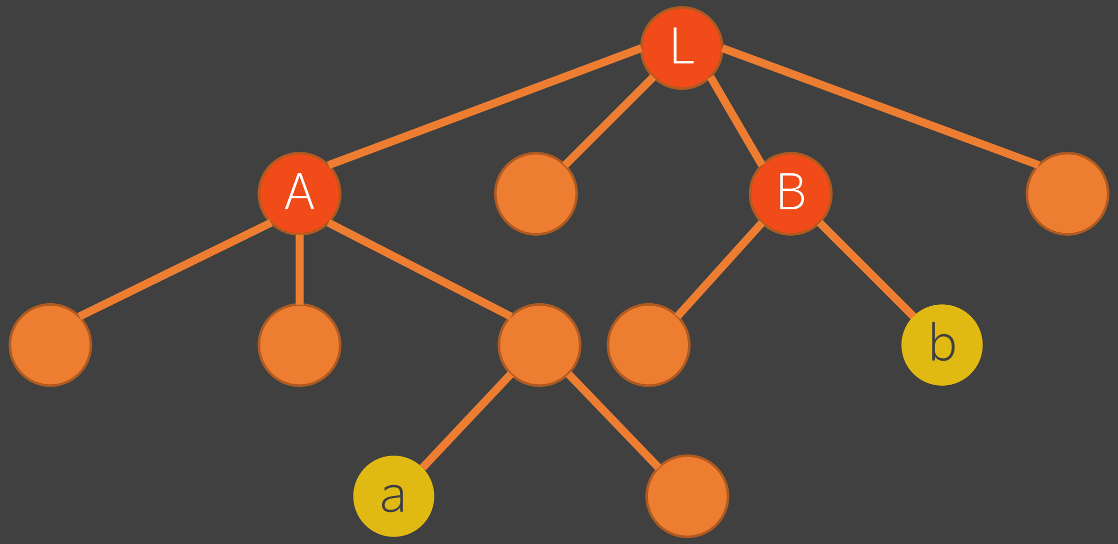
- 변환된 그래프에서, 깊이가 2 이상인 정점에 대하여 이를 포함하는 단순 사이클 중 단 하나만 깊이가 자신보다 작은 정점을 포함한다. (깊이가 자신보다 작은 정점은 깊이가 정확히 1만큼 작음이 보장된다.)
- 각각의 정점에 대해, 그 단순 사이클 내부의 간선만을 이용하여 깊이가 1만큼 작은 정점에 도달할 수 있는지의 여부를 간선으로 하는 새로운 그래프를 생각하자.

- 예를 들면, 하얀색으로 표시된 정점에서의 새로운 간선은 경로 1과 경로 2 중 적어도 하나의 경로에서 가로등이 모두 정상 작동하는가의 여부를 나타낸다.

- 아까 만들었던 그래프를 변환하면 다음과 같은 트리가 된다.
- 트리 간선의 활성화 여부는 세그먼트 트리를 통해 관리할 수 있다.



- a에서 b로 갈 수 있는지 물어보는 쿼리를 처리할 때, a와 b의 LCA인 정점 L을 찾고 그 바로 아래 있는 정점 A와 B를 찾는다.



- 경로 a-A와 b-B가 활성화되어 있는지를 변환된 트리에서 찾고, A에서 B로 갈 수 있는지는 원래 그래프에서 따로 찾아야 한다.
- The proof is left as an exercise to the reader.