UCPC 2019 본선 풀이

2019년 8월 3일

B. 비트베리

- 제출 121회, 정답 48팀 (정답률 39.67%)
- 처음 푼 팀: 78+9 (김현수, 신승원, 지구이), 6분
- 출제자: 박수찬

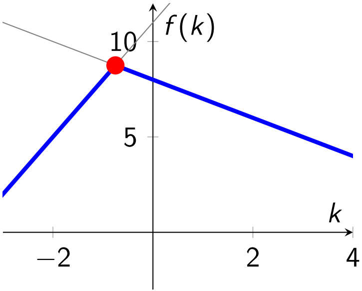
B. 비트베리

- 비트-코인, 베리-코인 간 교환은 수수료 없이 자유롭게 가능하기 때문에, 우선 주어진 모든 베리를 코인으로 바꾸는 것이 좋습니다.
- 베리 Q개는 코인 $\lfloor Q/C \rfloor \cdot D$개로 바꿀 수 있습니다.

B. 비트베리

- 이제는 비트 $X = P$개, 코인 $Y = \lfloor Q/C \rfloor \cdot D$개를 가지고 있을 때, 비트코인의 개수를 최대화하면 됩니다.
- 교환을 하면 비트 A개가 생기고 코인 B개가 사라지거나, 비트 A개가 사라지고 코인 B개가 생깁니다.
- 교환을 적절히 반복하면 어떤 정수 k에 대해 비트는 $X + A \cdot k$개, 코인 $Y - B \cdot k$개가 됩니다.
- k를 적당히 정해 $f(k) = \min(X + A \cdot k, Y - B \cdot k)$를 최대화하면 됩니다.

B. 비트베리



- 그래프를 그려 보면 $y = X + A \cdot k$와 $y = Y - B \cdot k$의 교점에서 $f(k)$가 최댓값을 가진다는 것을 알 수 있습니다.
- $k = \frac{Y-X}{A+B}$가 정수가 아닐 수도 있으므로, 그 부근의 정수 몇 개를 택해 대입해 보고 최댓값을 구하면 됩니다.

B. 비트베리

- k의 범위를 $[-10^4, 10^4]$ 정도로 착각하고 최댓값을 brute force로 찾으면 TLE를 받습니다.
- 예시: $P = 1$, $Q = 10^4$, $A = B = C = 1$, $D = 10^4$일 때, $X = 1$, $Y = \lfloor Q/C \rfloor \cdot D = 10^8$이고 최대 비트코인의 수는 $5 \cdot 10^7$입니다.

L. 대진표

- 제출 108회, 정답 48팀 (정답률 44.44%)
- 처음 푼 팀: Cafe Mountain (조승현, 박상수, 시제연), 12분
- 출제자: doju

L. 대진표

명제 1

모든 팀이 k개 이하의 경기를 치르고 우승할 수 있다면 팀의 수는 2^k 이하이다.

- 모든 팀이 k개 이하의 경기를 치르고 우승할 수 있다면 마지막 경기에서는 항상 $k-1$개 이하의 경기를 치르고 올라온 두 팀이 맞붙어야 합니다. 이를 귀납적으로 반복하면 됩니다.
- 대우 명제로, 팀의 수가 2^k+1 이상이라면 우승하기 위해 $k+1$경기 이상을 치러야 하는 팀이 반드시 존재합니다.

L. 대진표

- $2^{n-1} < N \leq 2^n$, 즉 슬롯의 수가 2^n개라고 둡니다.
- 한 라운드마다 슬롯의 수가 절반씩 줄어들므로 각 팀이 최대로 치를 수 있는 경기 수는 n입니다.
- 앞의 명제에 의해 우승하기 위해 n개 이상의 경기를 치러야 하는 팀이 반드시 존재합니다.
- 따라서 정확히 n경기를 치러야 하는 팀이 존재합니다.
- 즉 최소한으로 경기를 치르는 팀은 $n-1$개 이상의 경기를 치러야 합니다.

명제 2

모든 팀이 k개 이상의 경기를 치러야 우승할 수 있다면 팀의 수는 2^k 이상이다.

- 명제 1과 비슷하게 증명됩니다.
- 대우 명제로, 팀의 수가 2^k보다 작다면 k보다 적은 수의 경기를 치르고 우승할 수 있는 팀이 항상 존재합니다.
- 결승전에 진출한 두 팀은 $n-2$경기 이상을 치렀어야 하므로, 마지막 경기를 기준으로 양쪽 그룹에는 2^{n-2}개 이상의 팀이 있어야 합니다.

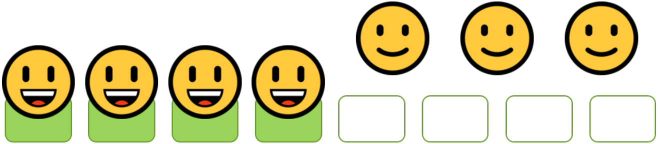
L. 대진표

- 최적의 전략은 오른쪽 그룹의 왼쪽 절반에 2^{n-2}개 팀을 배정하고, 나머지 팀을 왼쪽 그룹에 배정하는 것입니다.
  - 오른쪽 그룹에 속한 팀들은 모두 우승하기 위해 $n-1$경기를 치르게 됩니다.
  - 왼쪽 그룹에는 2^{n-2}보다 많은 수의 팀이 있으므로 우승하기 위해 n경기를 치러야 하는 팀이 반드시 존재합니다.
  - 오른쪽 그룹에 더 많은 팀을 배정할 경우 더 오른쪽의 슬롯을 채워야 하므로 손해입니다.

L. 대진표

- 그러나 팀 수가 너무 많다면, 정확히 말해 $N - 2^{n-2} > 2^{n-1}$ 이라면 이 전략을 쓸 수 없습니다.

L. 대진표



- 오른쪽 그룹에는 $N - 2^{n-1}$개 이상 2^{n-1}개 이하의 팀을 배정할 수 있습니다.
- 이때 $N - 2^{n-1}$개 팀을 배정하는 경우, 즉 왼쪽 그룹을 전부 채우는 경우가 최적이라면 편할 것 같습니다.

L. 대진표

명제 3

$2^{k-1} < a < b \leq 2^k$일 때, a에 대한 답은 b에 대한 답보다 좋다.

- 귀납법으로 증명합니다.
- $k = 2$일 때 `###.` $<$ `####`이므로 성립합니다.
- $k - 1$에서 명제가 성립한다고 가정해 봅시다.
  - 즉 $2^{k-2} < x \leq 2^{k-1}$ 범위에서 x가 커질수록 답이 나빠집니다.

L. 대진표

- a는 오른쪽 절반을 채울 수 있고 b는 불가능한 경우
  - 앞서 말했듯 오른쪽 그룹에 절반보다 많은 팀을 배정하면 더 뒤의 슬롯을 채워야 하므로 손해입니다.
  - 따라서 자명하게 성립합니다.

L. 대진표

- a는 오른쪽 절반을 채울 수 있고 b는 불가능한 경우
  - 앞서 말했듯 오른쪽 그룹에 절반보다 많은 팀을 배정하면 더 뒤의 슬롯을 채워야 하므로 손해입니다.
  - 따라서 자명하게 성립합니다.
- a와 b 모두 오른쪽 절반을 채울 수 있는 경우
  - 둘 다 오른쪽 그룹의 절반을 채울 것이므로 왼쪽 그룹의 답을 비교하면 됩니다.
  - a는 왼쪽에 $a - 2^{k-2}$팀을, b는 $b - 2^{k-2}$팀을 배정합니다.
  - 두 값은 모두 2^{k-2}보다 크고 2^{k-1} 이하이므로 가정에 의해 성립합니다.

L. 대진표

- 둘 다 오른쪽 절반을 채울 수 없는 경우
  - a가 오른쪽 그룹에 배정할 수 있는 팀 수는 $a - 2^{k-1}$ 이상 2^{k-1} 이하입니다.
  - 가정에 의해 더 많은 팀을 배정할수록 답이 나빠지므로, $a - 2^{k-1}$개의 팀을 배치하는 것이 최적입니다. b 역시 마찬가지입니다.
  - 역시 가정에 의해 $a - 2^{k-1}$에 대한 답이 $b - 2^{k-1}$에 대한 답보다 좋으므로 성립합니다.

L. 대진표

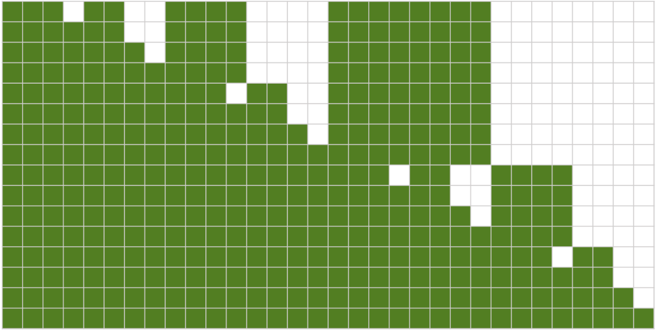
- 둘 다 오른쪽 절반을 채울 수 없는 경우
  - a가 오른쪽 그룹에 배정할 수 있는 팀 수는 $a - 2^{k-1}$ 이상 2^{k-1} 이하입니다.
  - 가정에 의해 더 많은 팀을 배정할수록 답이 나빠지므로, $a - 2^{k-1}$개의 팀을 배치하는 것이 최적입니다. b 역시 마찬가지입니다.
  - 역시 가정에 의해 $a - 2^{k-1}$에 대한 답이 $b - 2^{k-1}$에 대한 답보다 좋으므로 성립합니다.
- 따라서 $k - 1$에서 명제가 성립한다면 k에 대해서도 성립합니다.

L. 대진표

- 그러므로 다음의 전략을 재귀적으로 사용하면 됩니다.
  - 오른쪽 그룹의 절반을 채울 수 있다면 채우고 나머지 팀을 왼쪽에 배정한다.
  - 그렇지 않다면 왼쪽 그룹을 전부 채우고 나머지 팀을 오른쪽에 배정한다.

L. 대진표

- 작은 데이터를 직접 돌려 보면 앞의 내용을 증명하지 않더라도 **믿음**을 가지고 풀 수 있습니다.
- 아래는 $17 \leq N \leq 32$일 때의 답입니다.



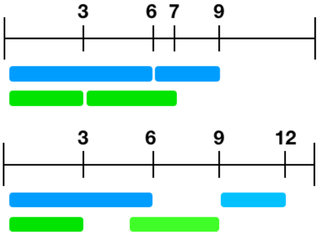
K. 옥토끼는 통신교육을 풀어라!!

- 제출 100회, 정답 46팀 (정답률 46.00%)
- 처음 푼 팀: 78-9 (hyea, ko_osaga, kdh9949), 15분
- 출제자: functionx

K. 옥토끼는 통신교육을 풀어라!!

- *Max*(옥토끼가 문제를 풀지 않은 시간)을 최소화하는 문제입니다.
- *Max*가 X 이하인 답이 존재한다면 $X + 1$ 이하인 답 역시 존재하므로 `Parametric Search`를 고려해봅시다.
- 옥토끼는 한 번에 두 개의 문제를 풀 수 있으므로 두 개의 라인에 스케쥴을 적절히 배치하는 형태를 생각할 수 있습니다.

K. 옥토끼는 통신교육을 풀어라!!



- Max가 X 이하인 답이 존재한다면, 스케쥴을 적절히 뒤로 옮겨서 옥토끼가 문제를 푸는 시각이 $X, 2X, ..., NX$가 되게 할 수 있습니다.
- 옥토끼가 시각 $X, 2X, ..., NX$에 맞춰서 문제를 풀 수 있는지 구하면 됩니다.

K. 옥토끼는 통신교육을 풀어라!!

- 푸는 데 걸리는 시간이 T_i인 문제를 jX에 풀었다고 합시다. $kX\left(j - \left\lceil \frac{T_i}{X} \right\rceil + 1 \le k < j\right)$ 시각은 해당 문제가 점유하고 있으므로 다른 라인에서 풀어야 합니다.
- 언급한 $\left\lceil \frac{T_i}{X} \right\rceil$를 슬롯 크기라고 해봅시다.
- 만약 $k'X\left(j - \left\lceil \frac{T_i}{X} \right\rceil + 1 < k' < j\right)$ 시각에 슬롯 크기가 2 이상인 문제를 풀면 $(k'-1)X$ 시각에 문제를 풀 수 없습니다.
- 따라서 $k'X$ 시각에는 반드시 슬롯 크기가 1인 문제를 풀어야 합니다.

K. 옥토끼는 통신교육을 풀어라!!

- 슬롯 크기가 1인 문제들만 남아있다면 쉽게 스케줄을 짤 수 있습니다.
- 슬롯 크기가 2 이상인 문제들을 체인 형태로 배치하면 무조건 넣어야 하는 1짜리 문제들의 수가 최소가 됩니다.
- 1짜리 문제가 남으면 배치 가능, 부족하면 배치 불가능입니다.

A. 네힝

- 제출 159회, 정답 26팀 (정답률 16.35%)
- 처음 푼 팀: Cafe Mountain (조승현, 박상수, 시제연), 47분
- 출제자: functionx

A. 네힁

A. 녜힁

- 주어진 수열의 부분수열이 아닌 길이 2짜리 수열 중 사전 순으로 K번째 수열을 구해야 합니다.
- 수열 $\{X, Y\}$가 수열 A의 부분수열일 필요충분조건은 $(A\text{의 첫 등장위치}) < (B\text{의 마지막 등장위치})$입니다.

A. 녜힁

앞글자 구하기

앞글자	1	2	3	4	5	6	7
처음 위치	2	6	1	3	X	5	X
마지막 위치	4	6	1	7	X	5	X
뒷글자 개수	3	6	3	3	7	5	7

- 수열 : {`3 1 4 1 6 2 4`}
- 뒷글자 개수는 `Sliding Window` 알고리즘을 이용하여 $O(N+M)$으로 구할 수 있습니다.

앞글자 구하기

앞글자	1	2	3	4	5	6	7
뒷글자 개수	3	6	3	3	7	5	7
글자 순서	1-3	4-9	10-12	13-15	16-22	23-27	28-34

- 글자 순서는 뒷글자 개수의 누적합입니다.
- 앞글자를 구할 때, 이분탐색을 하거나 `lower_bound` 함수 등을 이용하여 쿼리 당 $O(logN)$으로 구할 수 있습니다.

A. 녜힁

뒷글자 구하기

- 앞글자가 수열에 등장하지 않는다면, 뒷글자를 구하는 것은 쉽습니다.
- 앞글자가 등장하는 경우를 생각해봅시다. 편의상 앞글자가 빨리 나오는 순서대로 숫자를 정렬합시다.

뒷글자 구하기

뒤＼앞	3	1	4	6	2
1	X	X	X		
2	X	X	X	X	
3					
4	X	X	X	X	X
5					
6	X	X	X		
7					

뒷글자 구하기

- 앞글자가 나중에 등장하는 쿼리가 앞에 오게 쿼리를 정렬합니다.
- 그러면 $\{1, 2, ..., M\}$이 있는 집합에서 아래 두 질의를 수행하는 문제를 풀면 됩니다.

질의의 종류

- 집합에서 특정 숫자 i를 삭제
- 집합의 K번째 원소를 구함

뒷글자 구하기

- 해당 질의는 `Segment Tree`를 이용하여 쿼리 당 $O(logN)$으로 구할 수 있습니다.
- 앞선 표를 `Persistent Segment Tree`에 저장하면 똑같은 시간복잡도를 가지고 온라인으로 쿼리를 처리할 수 있습니다.

E. Taxi

- 제출 162회, 정답 24팀 (정답률 14.81%)
- 처음 푼 팀: 애용애용김애용 (eaststar, mindol, cloge), 80분
- 출제자: bryan

E. Taxi

doju 8:16 AM
택시 풍선 40개는 좀 적게 잡은 것 아닌가요

bryan 11:41 AM
40개 제가 적은 건 아니지만

적당해보이는데요

박수찬 11:43 AM
🤔

40팀보다는 많이 풀 거 같은데요

bryan 11:43 AM
그런가요

너무 참가자를 과소평가했나

박수찬 11:43 AM
다른 문제들이 너무 어려워서

그거 풀 시간이 너무 많아요

moonrabbit2 11:53 AM
전 모두 풀 가능성 충분하다고 생각해요

bryan 12:02 PM
그럼 일단 55로 적을게요

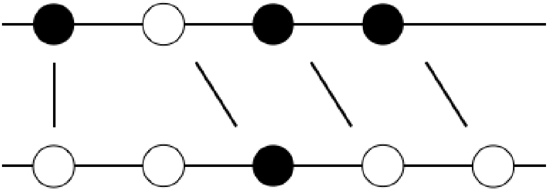
E. Taxi

- 설명대로 잘 구현하면 됩니다.
- double을 사용하면 틀리도록 만드는 게 의도였는데 eps를 잘 더해서 푼 팀이 많더라고요..

M. Uncrossed Knights' Tour

- 제출 48회, 정답 10팀 (정답률 20.83%)
- 처음 푼 팀: Cafe Mountain (조승현, 박상수, 시제연), 70분
- 출제자: functionx

M. Uncrossed Knights' Tour



- 위쪽 점과 오른쪽 점을 매칭하는데, 점의 색깔이 서로 달라야 하며, 매칭한 선이 서로 교차하면 안됩니다.
- 만약 아래쪽 점의 색깔을 반전시키면, `Longest Common Subsequence` 문제가 됩니다.

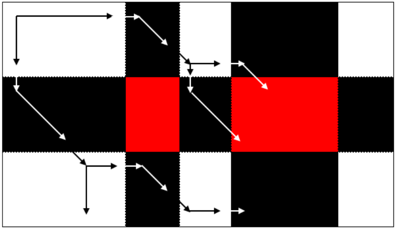
M. Uncrossed Knights' Tour

- 위쪽 점들의 수와 아래쪽 점들의 수가 모두 $O(N+M)$이므로 평범한 $O((N+M)^2)$ 알고리즘은 사용할 수 없다.
- N이 작으므로, 연속한 같은 점을 하나로 묶으면(RLE Encoding), 위쪽 점들의 수와 아래쪽 점들의 수가 $O(N)$ 이므로 이런 방향으로 생각하자.
- 일반적인 경우에는 압축한 길이가 N, M인 두 RLE Encoded String에서 LCS를 구하는 $O(NM \log(NM))$ 알고리즘(논문) 이 존재한다.
- 이 문제에서는 모든 기사를 매칭해야 하므로 $O(NM)$에 해결할 수 있다.

M. Uncrossed Knights' Tour

- 맨 앞 점끼리 비교하자.
- 만약 기사가 둘이라면 하나는 매칭할 수 없으므로 답이 없다.
- 만약 기사가 하나, 자연경관이 하나라면 이 둘을 매칭한다.
- 이런 방법을 반복하여 맨 앞 점이 자연경관이 되도록 한다.
- 똑같은 방식을 맨 뒤에서도 적용하여 맨 뒤 점도 자연경관이 되도록 한다.

M. Uncrossed Knights' Tour



- 위쪽 점과 아래쪽 점의 매칭 관계를 직사각형 격자로 나타내면 그림과 같습니다.
- 격자에서 DP를 한다고 하면, 검은 격자에서는 오른쪽 아래로, 흰 격자에서는 오른쪽 혹은 아래로, 빨간 격자는 지나면 안됩니다.

M. Uncrossed Knights' Tour

- 여기서 DFS로 답을 구합니다.
- 흰 격자에서 오른쪽 혹은 아래로 가는 선택지가 있는데, 무조건 아래로 먼저 갑시다.
- 아래로 가면 나오는 검은 격자(커다란 영역)는 체크해주어 나중에 다시 방문하지 않도록 합니다.
- 나중에 검은 격자 영역을 다시 방문하더라도 처음 방문했을 때보다 오른쪽 위치를 방문하기 때문에 처음 방문한 경우보다 더 적은 격자를 방문할 수밖에 없습니다.
- 체크하는 검은 격자 영역의 수가 $O(N^2)$이므로 총 시간복잡도는 $O(N^2)$입니다.

F. 공의 합집합

- 제출 72회, 정답 9팀 (정답률 12.50%)
- 처음 푼 팀: 화석 (삼엽충, 암모나이트, 실러캔스), 129분
- 출제자: 박수찬

F. 공의 합집합

복잡한 도형의 부피를 구해야 하므로, 아래와 같은 개념을 생각할 수 있습니다.

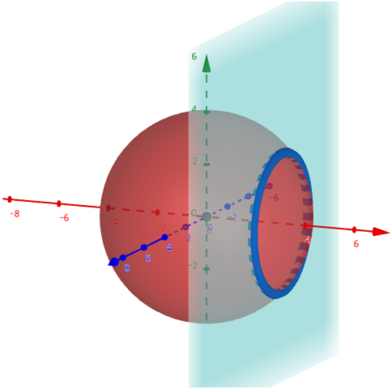
입체도형의 부피

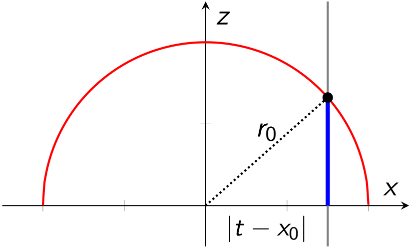
구간 $[a,\ b]$의 임의의 점 x에서 x축에 수직인 평면으로 자른 단면의 넓이가 $S(x)$인 입체도형의 부피 V는

$$V = \int_a^b S(x)dx \text{ (단, } S(x)\text{는 구간 } [a,\ b]\text{에서 연속)}$$

F. 공의 합집합

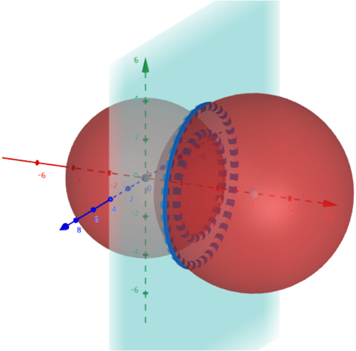
예를 들어 공 하나의 부피를 구해야 한다고 합시다. 중심이 $(x_0, 0, 0)$에 있고 반지름이 r_0인 공을 $x = t$ 단면으로 자르면 그 넓이는 $\{r_0^2 - (t - x_0)^2\}\pi$입니다.

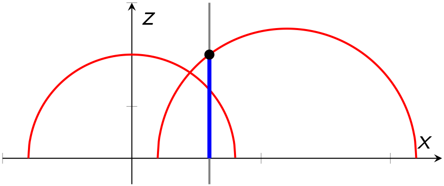




F. 공의 합집합

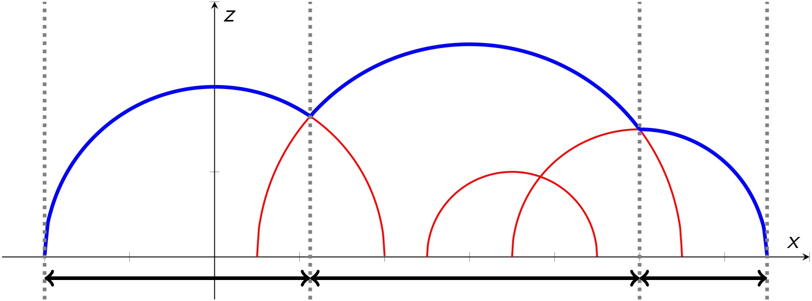
여러 개의 구가 같은 단면 위에 있다면, 작은 원이 반드시 큰 원 안에 포함되므로 그 중 반지름이 가장 큰 것을 구하면 됩니다.





F. 공의 합집합

각각의 공 i마다 $r_i^2 - (x - x_i)^2 = max_j\{r_j^2 - (x - x_j)^2\}$를 만족하는 x는 존재하지 않거나 구간 $[a_i, b_i]$로 나타납니다. 이 구간은 공 i가 합집합의 부피에 기여하는 구간을 의미합니다.



F. 공의 합집합

이러한 구간을 모두 구할 수 있다면, 답은

$$\sum_i \int_{a_i}^{b_i} \left\{ r_i^2 - (x - x_i)^2 \right\} \pi dx$$
$$= \sum_i \left[r_i^2 \cdot (b_i - a_i) - \frac{1}{3} \left\{ (b_i - x_i)^3 - (a_i - x_i)^3 \right\} \right] \pi$$

입니다.

F. 공의 합집합

구간 $[a_i, b_i]$를 어떻게 구할 수 있을까요? 아래 부등식의 해를 구해야 합니다.

$$r_i^2 - (x - x_i)^2 \geq r_j^2 - (x - x_j)^2$$

F. 공의 합집합

구간 $[a_i, b_i]$를 어떻게 구할 수 있을까요? 아래 부등식의 해를 구해야 합니다.

$$r_i^2 - (x - x_i)^2 \geq r_j^2 - (x - x_j)^2$$

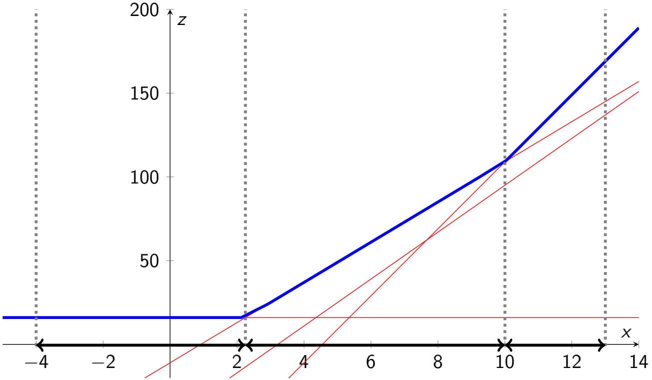
양변에 x^2를 더한 후 정리하면,

$$2x_ix + r_i^2 - x_i^2 \geq 2x_jx + r_j^2 - x_j^2$$

과 같이 되어, 결국 여러 직선들이 주어질 때 직선 i가 최대인 구간을 구하는 문제가 됩니다.

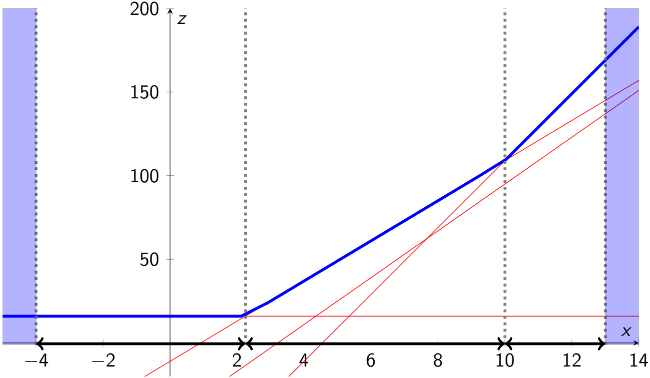
F. 공의 합집합

이는 모든 직선의 upper hull을 구하면 간단하게 알 수 있습니다.



F. 공의 합집합

다만 직선 i가 최대인 구간이 공 i의 x좌표 범위를 벗어나는 경우 적분구간에 포함하면 안 됩니다.



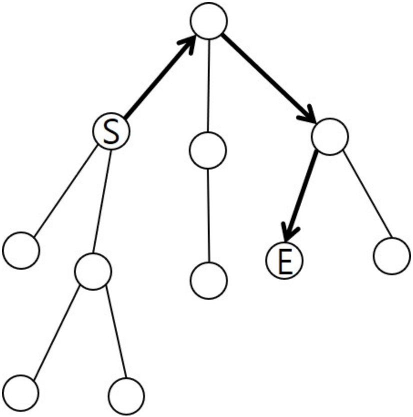
D. 룰렛

- 제출 21회, 정답 4팀 (정답률 19.05%)
- 처음 푼 팀: 고풍당당 (jihoon, OnionPringles, rhrnald), 118분
- 출제자: moonrabbit2

D. 룰렛

- 주어지는 놀이공원은 놀이기구를 정점으로 해 그래프로 보면 간선이 $N - 1$개인 연결그래프이므로 트리입니다.
- 1번 정점을 루트로 잡아 정점들의 부모/자식 관계를 설정할 수 있으니, 이를 가정하고 진행합니다.

D. 룰렛



- 트리의 중요한 특징은 어떠한 두 정점을 잇는 경로는 하나밖에 없다는 것입니다.

D. 룰렛

- S에서 출발해 E에서 집으로 갈 때, S에서 E까지의 경로 상의 모든 정점을 한번 이상 방문합니다.
- 최초 방문 시점은 경로 상의 위치가 앞일수록 빠릅니다.

D. 룰렛

- S에서 E까지의 경로를 $V_1 \to V_2 \to \cdots \to V_K$라고 합시다. $(S = V_1, E = V_K)$
- 최초 방문 시점이 경로 상의 위치가 앞일수록 빠르기에 V_2를 처음 방문한 순간 $V_3, \cdots, V_K$는 방문한 적이 없습니다.
- 그러므로 V_1에서 출발해 V_K에서 정지하는 문제는 V_1에서 V_2를 처음 방문한 후에는 V_2에서 출발해 V_K에서 정지하는 문제와 동치임을 알 수 있습니다.

D. 룰렛

- 이제 정점 i에서 출발해 인접한 정점 j로 이동할 확률을 구하는 것을 목표로 합시다.
- 이는 최종적으로 j에서 집으로 갈 확률이 아니라, j에 도달하기만 하면 이후 경로는 무시할 때의 확률입니다.
- 정점 i에서 출발해 i에서 집으로 갈 확률도 구해야 합니다.

D. 룰렛

- 각 룰렛마다 어떤 그림이 등장할 확률은 그림이 그려진 칸 수에 비례합니다.
- 이를 이용해 다음 세 가지 값을 알 수 있습니다.
- A_i : i의 부모의 룰렛에서 i로 가는 그림이 등장할 확률
- U_i : i의 룰렛에서 i의 부모로 가는 그림이 등장할 확률
- E_i : i의 룰렛에서 집으로 가는 그림이 등장할 확률

D. 룰렛

- 다음 세 가지 값 또한 정의합니다.
- D_i : i의 부모에서 출발해 i로 이동할 확률
- P_i : i에서 출발해 i의 부모로 이동할 확률
- S_i : i에서 출발해 i에서 집으로 갈 확률
- 앞서 언급했던 것처럼 한번 도달한 후의 경로는 무시할 때의 확률입니다.

D. 룰렛

P_i 구하기

- i에서 j로 내려갔을 경우, 다시 i로 돌아와야 i의 부모로 갈 수 있습니다.
- j로 내려간 후(A_j) 다시 i로 올라갈(P_j) 확률은 A_jP_j입니다.
- 그러므로

$$V_i = \sum_{j=child(i)} A_jP_j$$

  라고 할 때, V_i는 i에서 한 번 i의 자식으로 내려간 후 다시 돌아올 확률입니다.
- $P_i = U_i + U_iV_i + U_iV_i^2 + \cdots$ 이므로, $P_i = \frac{U_i}{1-V_i}$입니다.

D. 룰렛

D_i 구하기

- i의 부모를 p라고 합시다.
- $$V_p = \sum_{j=child(p)} (A_j P_j) - A_i P_i + D_p U_p$$
  라고 할 때, V_p는 p에서 한 번 i가 아닌 연결된 정점으로 이동한 후 다시 돌아올 확률입니다.
- $D_i = A_i + A_i V_p + A_i {V_p}^2 + \cdots$ 이므로, $D_i = \frac{A_i}{1-V_p}$입니다.

D. 룰렛

S_i 구하기

- $$V_i = \sum_{j=child(i)} (A_j P_j) + D_i U_i$$
  라고 할 때, V_i는 i에서 한 번 연결된 정점으로 이동한 후 다시 돌아올 확률입니다.
- $S_i = E_i + E_i V_i + E_i {V_i}^2 + \cdots$ 이므로, $S_i = \frac{E_i}{1-V_i}$입니다.

D. 룰렛

- 위의 값들은 모두 $X \cdot X^{-1} \equiv 1 \pmod{10^9+7}$인 역원 X^{-1}를 구할 수 있기 때문에 계산이 가능합니다.
- 페르마의 소정리에 따라 소수 p에 대해 $X^{-1} \equiv X^{p-2} \pmod{p}$가 성립합니다.
- 10^9+7이 소수이므로 10^9+5제곱을 구하면 되며, 이는 $O(\log(10^9+7))$에 가능합니다.

D. 룰렛

- S에서 E의 경로를 $V_1 \to \cdots \to V_L \to \cdots \to V_K$라고 합시다. $(S = V_1, E = V_K, V_L = LCA(S, E))$
- 앞에서의 논리를 적용하면 S에서 출발해 E에서 집으로 갈 확률은 $P_{V_1} \cdots P_{V_{L-1}} D_{V_{L+1}} \cdots D_{V_K} S_{V_K}$입니다.
- 해당 값은 LCA를 $O(\log N)$에 구할 수 있으므로 미리 부분 곱 배열을 만들어 놓으면 $O(\log N + \log P)$에 해결 가능하며, LCA를 구할 때 구간 곱 Sparse Table을 만들어 놓았다면 $O(\log N)$에도 가능합니다.

D. 룰렛

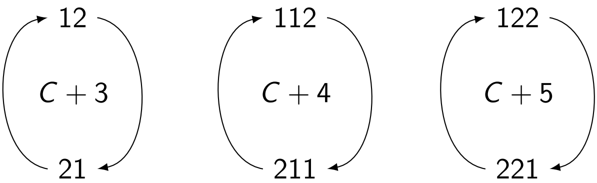
- 모든 쿼리의 분모와 분자가 $10^9 + 7$의 배수가 아님이 보장되므로, D_i, P_i, S_i는 모두 0이 아니라는 것을 알 수 있습니다.
- 그러니 안심하고 역원과 구간 곱을 구해도 문제가 없습니다.
- 총 시간복잡도 $O((N + Q)(\log N + \log P))$ 등에 해결 가능합니다. ($P = 10^9 + 7$)

C. 컨테이너

- 제출 4회, 정답 1팀 (정답률 25.00%)
- 처음 푼 팀: Cafe Mountain (조승현, 박상수, 시제연), 197분
- 출제자: kriii

C. 컨테이너

부분 문자열을 바꾸는 세 가지 연산을 잘 사용해서 문자열을 바꾸는 문제입니다.

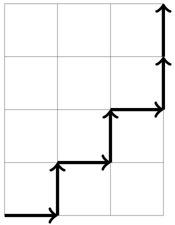


C. 컨테이너

갑작스럽지만, 문자열을 격자 위의 경로 하나로 나타냅니다.

$(0,0)$에서 시작해서 문자열 왼쪽에서 부터 보면서

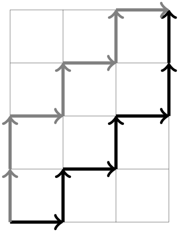
1이면 x좌표를, 2이면 y좌표를 1씩 늘립니다.



1212122의 상상도

1212122를 2212121로 바꾸고 싶습니다.

둘 다 그려봅니다.



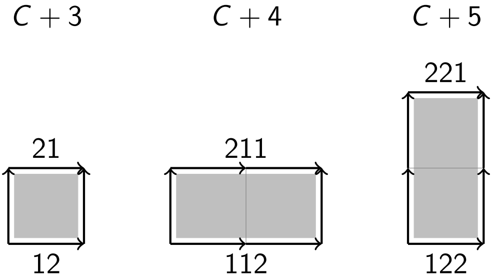
1212122가 2212121이 되는 상상함

C. 컨테이너

이렇게 경로를 그리면 대체 무엇이 남나요?

이렇게 경로를 그리면 대체 무엇이 남나요?

연산을 1×1, 1×2 크기의 타일로 볼 수 있게 됩니다.



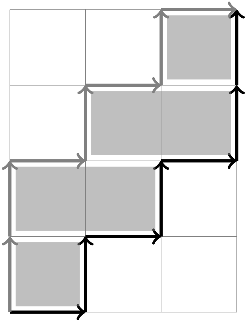
C. 컨테이너

그래서 이제 뭘 해야 하나요?

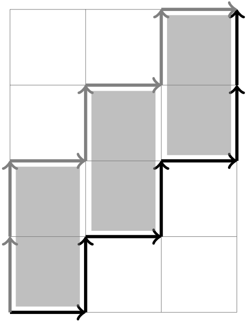
C. 컨테이너

그래서 이제 뭘 해야 하나요?

두 경로 사이를 타일로 겹치지 않게 채우면 됩니다.



$C = 0$



$C = 2$

C. 컨테이너

이렇게 두 경로 사이를

$$(C+3)\cdot\#(\square)+(C+4)\cdot\#(\square\square)+(C+5)\cdot\#\left(\begin{smallmatrix}\square\\\square\end{smallmatrix}\right)$$

가 최소가 되도록 타일로 배치한 것이 최적해가 됩니다.

C. 컨테이너

이렇게 두 경로 사이를

$$(C+3)\cdot\#(\square)+(C+4)\cdot\#(\square\!\square)+(C+5)\cdot\#\left(\begin{smallmatrix}\square\\\square\end{smallmatrix}\right)$$

가 최소가 되도록 타일로 배치한 것이 최적해가 됩니다.

왜?

C. 컨테이너

이렇게 두 경로 사이를

$$(C+3)\cdot\#(\square)+(C+4)\cdot\#(\square\square)+(C+5)\cdot\#\left(\begin{smallmatrix}\square\\\square\end{smallmatrix}\right)$$

가 최소가 되도록 타일로 배치한 것이 최적해가 됩니다.

왜? 는 나중에 알아봅니다.

기본적으로 모든 칸에 를 넣고,

로 대체하면 $(C+4) - 2 \cdot (C+3) = -(C+2)$,

로 대체하면 $(C+5) - 2 \cdot (C+5) = -(C+1)$

의 비용을 얻는 것으로 봅니다.

격자는 이분그래프라서, Min Cost Flow로 모델링 할 수 있습니다.

이제 $O(N^4 \lg N)$의 시간에 문제를 해결할 수 있습니다.

C. 컨테이너

$O(N^4 \lg N)$은 당연히 느립니다.

C. 컨테이너

$O(N^4 \lg N)$은 당연히 느립니다.

Min Cost Flow의 argument path의 길이는 단조 증가 하므로,
를 처음에 최대한 많이 채워 넣어도 됩니다.

C. 컨테이너

$O(N^4 \lg N)$은 당연히 느립니다.

Min Cost Flow의 argument path의 길이는 단조 증가 하므로,
를 처음에 최대한 많이 채워 넣어도 됩니다.

이제 각 행마다 최대 한 개의 칸이 남기 때문에,
앞으로 $O(N)$번만 플로우를 흘리면 되고,
$O(N^3 \lg N)$의 시간에 문제를 해결할 수 있습니다.

C. 컨테이너

$O(N^4 \lg N)$은 당연히 느립니다.

Min Cost Flow의 argument path의 길이는 단조 증가 하므로,
를 처음에 최대한 많이 채워 넣어도 됩니다.

이제 각 행마다 최대 한 개의 칸이 남기 때문에,
앞으로 $O(N)$번만 플로우를 흘리면 되고,
$O(N^3 \lg N)$의 시간에 문제를 해결할 수 있습니다.

★역추적은 위상정렬로 적절히 합시다.★

이제 두 경로 사이를 침범하지 않아도, 타일을 겹치지 않아도 된다는 증명이 필요합니다.

처음의 문자열에서, 다른 문자열을 향한 최소 비용을 구했을 때 위에서 구한 비용으로 모든 전이가 relaxing되어 있으면 됩니다.

C. 컨테이너

Case1: , , 추가

각각 $C + 3$, $C + 4$, $C + 5$의 비용이 추가되는데,

해는 더 최적화 될 수 있어서 같거나 더 작은 비용입니다.

Case2: 제거

$C + 3$의 비용이 추가되는데,

원래 해에 속할 수 있었다면 해의 비용은 $-(C + 3)$낮습니다.

속할 수 없었다면, 나 를 하나 덜어내고 를 넣습니다.

그로인해, 비용이 $1 \sim 2$줄어든 다음 최적화를 거쳐 더 줄어듭니다.

C. 컨테이너

Case3: , 제거

각각 $C + 4$, $C + 5$의 비용이 추가되는데,

원래 해에 속할 수 있었다면 해의 비용은 그만큼 낮습니다.

속할 수 없었다면, 겹치는 나 를 덜어내고 를 넣습니다.

그로인해, 비용이 줄어든 다음 최적화를 거쳐 더 줄어듭니다.

Case4: 추가, 제거

각각 $C + 4$, $C + 5$의 비용이 추가되는데,

Case1과 Case2를 합친 경우입니다. 그러므로 비용이 줄어듭니다.

I. 미로

- 제출 8회, 정답 0팀 (정답률 0.00%)
- 처음 푼 팀: 없음
- 출제자: functionx

I. 미로

- 방향 그래프에서 경로를 찾는 문제니까 우선 격자를 강연결요소(SCC)로 묶어서 위상정렬을 해놓고 풀어야 될 것 같아 보입니다.
- 원본 그래프가 격자 그래프의 부분 그래프이므로 하나의 SCC 는 격자에서 하나의 연결 요소입니다.

성질 1

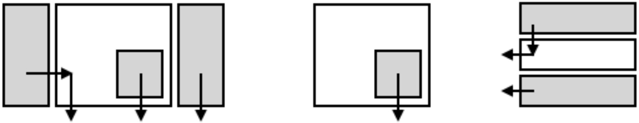
위상정렬 순위가 높은 K개의 SCC를 모으면, 격자에서는 서로 만나지 않는 여러 직사각형의 집합 형태가 됩니다.

- 위상정렬 순위가 낮은 SCC에서는 순위가 높은 SCC로 갈 수 없습니다.
- 어떤 격자에서 인접한 격자로 갈 수 없다면, 그 격자의 내용은 하나로 고정됩니다. 만약 갈 수 없는 인접한 격자가 2개 이상이라면 해당 격자의 내용을 정할 수 없어 모순입니다.
- 만약 위상정렬 순위가 높은 SCC를 모았을 때 직사각형이 아니라면, 내용을 정할 수 없는 격자가 반드시 생깁니다.

성질 2

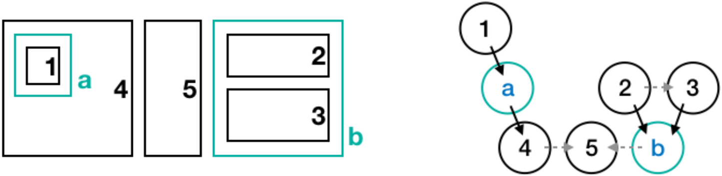
성질 1의 여러 직사각형의 집합에서 직사각형 안의 임의의 격자를 선택할 때, 격자에서 해당 직사각형의 위 및 아래쪽으로 나갈 수 있거나 왼쪽 및 오른쪽으로 나갈 수 있습니다.

- (수학적 귀납법) 새로운 SCC가 추가되면 그림과 같은 세 가지 형태가 됩니다. 세 경우 모두 **성질 2**를 만족합니다.



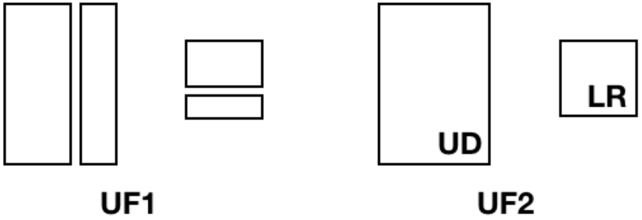
모델링

- SCC들을 가지고 트리(실선)와 형제 노드끼리 있는 일직선 그래프(점선)를 구축해야 합니다. 필요에 의해 새로운 가상 노드를 만들 수도 있습니다.



모델링: 사전 설명

- 총 2개의 `Union-Find` 자료구조를 만듭니다. 하나는 후술할 `Group(i)`으로 묶인 직사각형의 집합, 다른 하나는 붙어있는 SCC를 모두 하나로 묶어서 만든 직사각형의 집합입니다.
- 노드 하나가 직사각형 하나를 의미합니다.



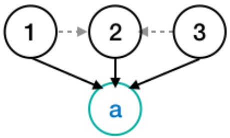
I. 미로

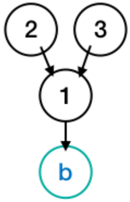
모델링: 사전 설명

Group(X)

새로운 가상 노드를 만들어서 두 Union-Find 자료구조에 적용.

- 첫 번째 자료구조에서는 X번 노드에서 점선 간선을 통해 순회한 모든 노드와 새로운 가상 노드를 묶습니다.
- 두 번째 자료구조에서는 X번 노드와 새로운 가상 노드를 묶습니다.





모델링: 방법

- 위상정렬 순위가 높은 SCC부터 하나씩 추가하면서 그래프를 구축합니다.
- SCC가 감싸는 직사각형들을 먼저 처리하고, 양옆 및 위아래에 붙어있는 직사각형을 처리합니다.

모델링: 방법

- 추가할 SCC : a, 인접한 SCC : b
- 두 자료구조에서 b가 속한 노드 : B1, B2
- 맨 처음에 새로운 노드 A1를 만들어서 두 Union-Find 자료구조에 적용합니다.
- 만약 A1과 B2가 묶여있다면, 이미 b를 작업했다는 의미이므로 넘어갑니다.
- 작업을 할 때 b의 위치에 따라 적절히 처리해줍니다.
- 작업이 끝나면 A1과 B2를 묶습니다.

모델링: 방법

감싸는 직사각형

- 우선 Group(B1)를 합니다. 그 과정에서 만들어지는 새로운 가상 노드를 GB라 합시다.
- **트리 간선** GB -> A1을 추가하고, 두 Union-Find 자료구조에서 GB와 A1을 묶습니다.
- 모든 직사각형을 다 감싸면 하나의 직사각형으로 합쳐지는데, 이 직사각형에서는 상하좌우로 모두 빠져나갈 수 있습니다.

모델링: 방법

양옆으로 붙은 직사각형

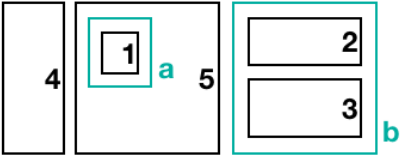
- 만약 B2가 **상하로 빠져나갈 수 없는 직사각형**이면 Group(B2)를 합니다. 그 과정에서 만들어지는 새로운 가상 노드를 GB라 합시다.
- **일직선 그래프 간선** GB -> A1을 추가하고, 두 번째 Union-Find 자료구조에서만 GB와 A1을 묶습니다.
- 다른 경우의 B2에 대해서는 GB가 아니라 B2라고 생각하고 간선을 추가합니다.
- 모든 직사각형을 다 붙이면 하나의 직사각형으로 합쳐지는데, 이 직사각형에서는 상하로만 빠져나갈 수 있습니다.

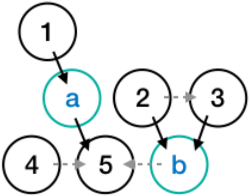
모델링: 방법

위아래로 붙은 직사각형

- 양옆으로 붙은 직사각형과 비슷한 방식으로 처리합니다.
- 여기서는 Group(B2)를 하는 조건이 B2가 **좌우로 빠져나갈 수 없는 직사각형**일 때입니다.
- 모든 직사각형을 다 붙이면 하나의 직사각형으로 합쳐지는데, 이 직사각형에서는 좌우로만 빠져나갈 수 있습니다.

모델링: 방법





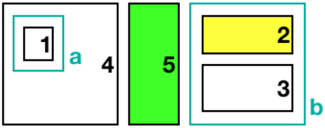
쿼리 처리

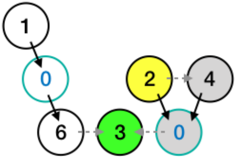
- SCC 안의 노드끼리는 서로 드나들 수 있으므로 모델에서 SCC의 가중치는 $|SCC|$입니다.
- 모든 가상노드는 가중치가 0입니다.
- 노드 a에서 노드 b로 가는 경로에 포함될 수 있는 노드의 가중치 합을 구해야 합니다.
- 경로는 **일직선 그래프 간선**을 통해 형제 노드로 가고, **트리 간선**을 통해 부모 노드로 가고, ...을 반복하는 형태입니다.

I. 미로

쿼리 처리

- 모든 노드에 대하여 **일직선 그래프 간선**을 통해 갈 수 있는 노드의 가중치 합을 미리 구합니다.
- 쿼리의 정답은 트리 경로 및 도착점의 형제 노드에서 도착지까지 가는 경로에 있는 노드의 가중치의 합입니다.





I. 미로

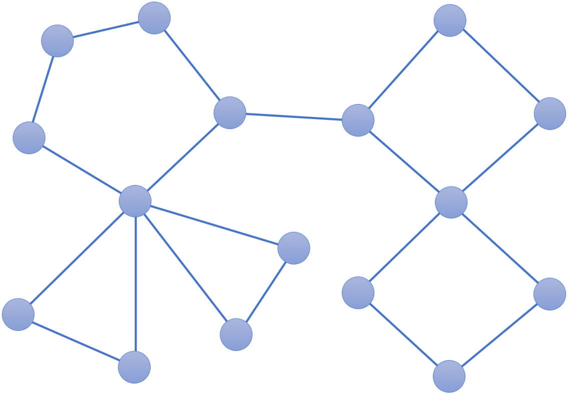
- 모델링에 필요한 시간은 $O(NM\alpha(NM))$입니다.
- `Sparse Matrix`로 쿼리를 처리했다면, 전처리에 필요한 시간은 $O(NM \log(NM))$, 쿼리 처리에 필요한 시간은 $O(\log(NM))$입니다.
- `Heavy-Light Decomposition`으로 쿼리를 처리했다면, 전처리에 필요한 시간은 $O(NM)$, 쿼리 처리에 필요한 시간은 $O(\log(NM))$입니다.

H. 죽은 선인장의 사회

- 제출 7회, 정답 0팀 (정답률 0.00%)
- 처음 푼 팀: 없음
- 출제자: kriii

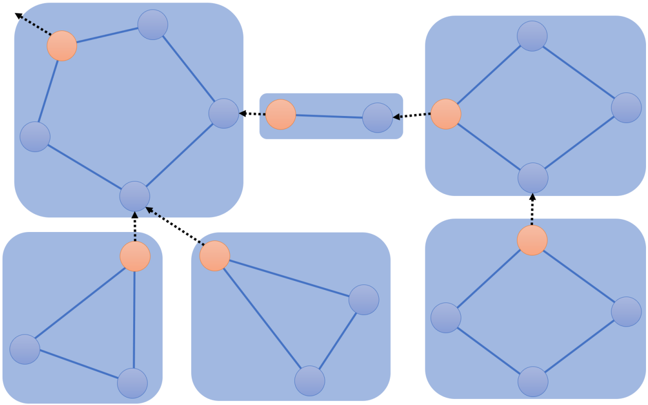
H. 죽은 선인장의 사회

선인장입니다. 용서할 수 없습니다.



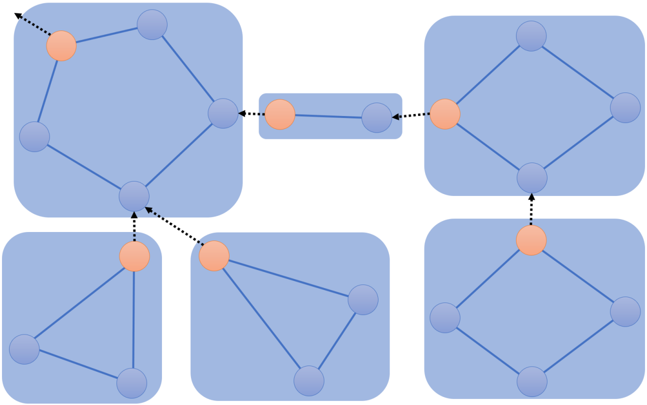
H. 죽은 선인장의 사회

가차없이 간선과 사이클 단위로 쪼개 버립니다.



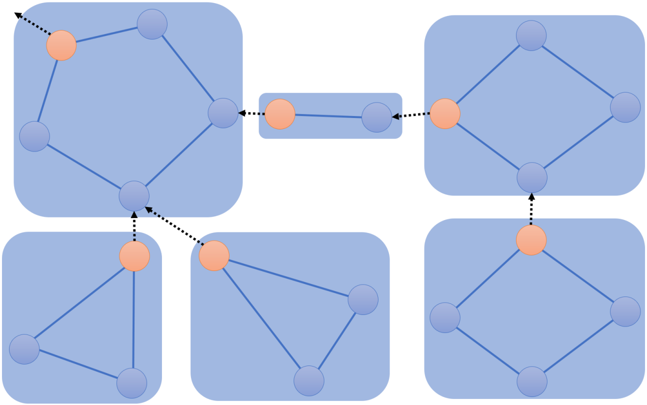
H. 죽은 선인장의 사회

BCC 구하는 법을 응용해 DFS 한번이면 됩니다.



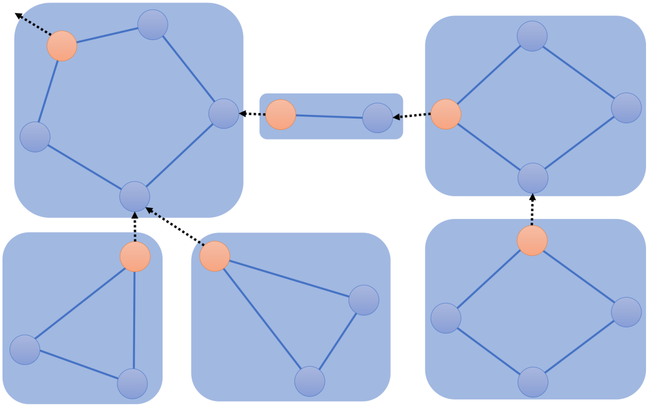
H. 죽은 선인장의 사회

방향선은 탐색 순서를 나타내는 DAG입니다.



H. 죽은 선인장의 사회

이렇게 분해하고 나면 연결 관계는 트리와 유사합니다.

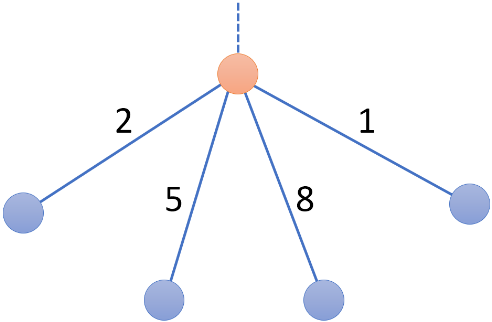


H. 죽은 선인장의 사회

- 이제, 사이클에 있는 간선을 하나씩 없애 트리를 만들어 보고, 지름의 최솟값을 구하면 됩니다.
- 새로 자라는 간선은 일단 무시합니다.
- 그런데, 트리의 지름은 어떻게 구하나요?

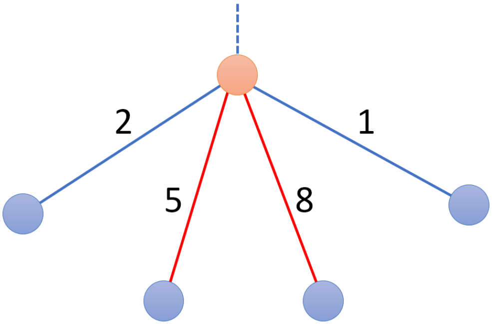
트리의 지름 구하기

DFS 순서로 트리를 탐색하면서, 각 정점을 지나는 경로의 길이 중 가장 긴 것을 구합니다.



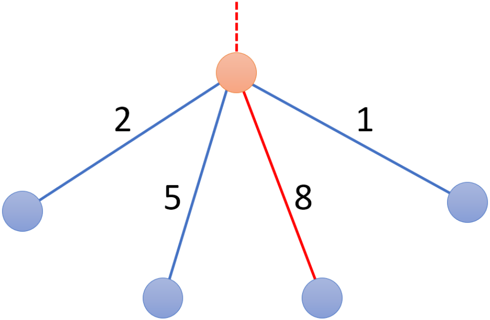
H. 죽은 선인장의 사회
트리의 지름 구하기

지름은 두 방향으로 뻗으면서 가장 길어야 합니다. 그러니 아래로 뻗을 수 있는 가장 긴 길이와 두 번째로 긴 길이를 고릅니다.



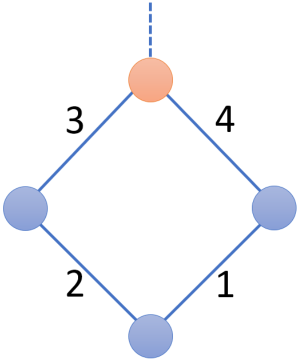
H. 죽은 선인장의 사회
트리의 지름 구하기

부모로 돌아갈 때는 아래로 뻗은 가장 긴 길이를 넘겨주면 조상 노드에 대한 경로 후보를 구할 수 있습니다.



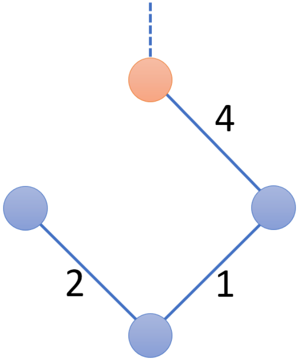
사이클 잘라보기

사이클의 간선들을 자르면서 비슷한 과정을 해봅니다.



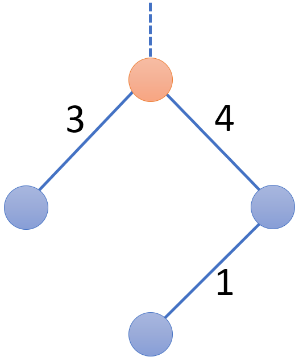
사이클 잘라보기

지름 $= 7$, 최대 길이 $= 7$



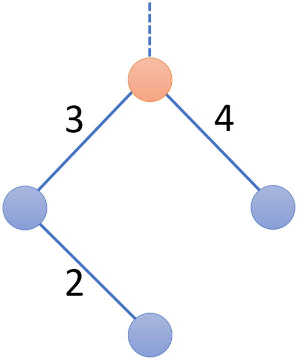
사이클 잘라보기

지름 $= 8$, 최대 길이 $= 5$



사이클 잘라보기

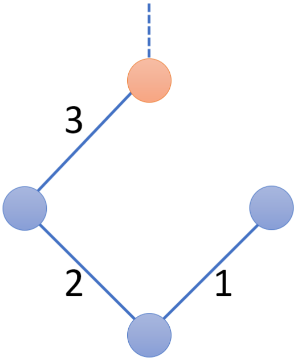
지름 $= 9$, 최대 길이 $= 5$



H. 죽은 선인장의 사회

사이클 잘라보기

지름 $= 6$, 최대 길이 $= 6$



H. 죽은 선인장의 사회

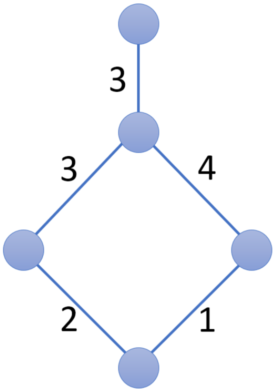
- (지름, 최대 길이)의 쌍이 $(7, 7)$, $(8, 5)$, $(9, 5)$, $(6, 6)$ 넷 나왔습니다.
- 지름을 최소화하기 위해 의미있는 쌍은 $(8, 5)$, $(6, 6)$입니다.
- 둘 중 무엇을 선택해야 할까요?

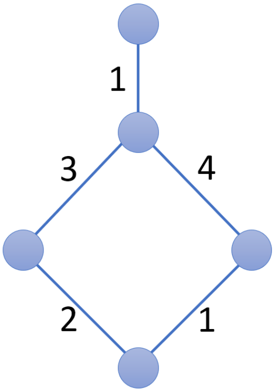
H. 죽은 선인장의 사회

- (지름, 최대 길이)의 쌍이 $(7, 7)$, $(8, 5)$, $(9, 5)$, $(6, 6)$ 넷 나왔습니다.
- 지름을 최소화하기 위해 의미있는 쌍은 $(8, 5)$, $(6, 6)$입니다.
- 둘 중 무엇을 선택해야 할까요?
- 이후 상황에 따라 무엇을 선택해야 하는 지 다를 수 있습니다.

H. 죽은 선인장의 사회

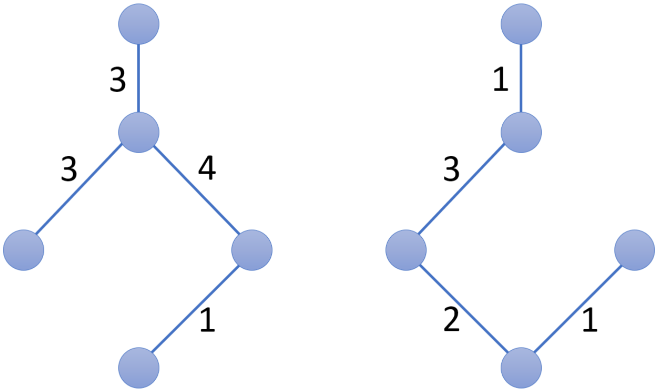
예를 들면 이 둘은





H. 죽은 선인장의 사회

각각이 최선입니다.



H. 죽은 선인장의 사회

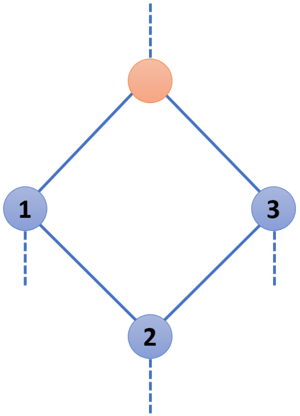
- 그러므로, 모든 정보를 저장하면서 부모에게 넘겨줘야 합니다.
- 하지만 사이클이 많아지면 지수적으로 정보의 개수가 증가하므로 좀 더 효율적인 방법을 찾아야 합니다.
- 이건 또 어떻게 할까요?

H. 죽은 선인장의 사회

- 문제를 결정 문제로 바꿉니다.
- 지름이 R이하인 트리를 만들 수 있는지 검사합니다.
- 정보들을 검사한 다음 지름이 R이하인 것 중에서 최대 길이가 가장 작은 것만 알면 됩니다.

H. 죽은 선인장의 사회

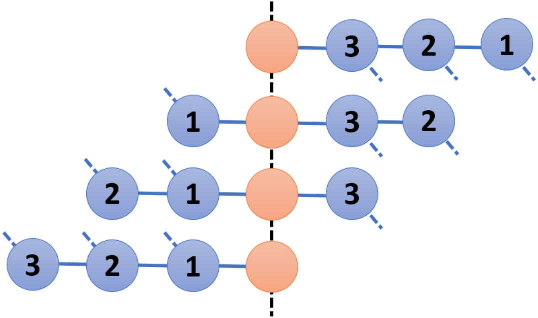
앞으로 이런 형태의 사이클을 처리해야 합니다.



H. 죽은 선인장의 사회

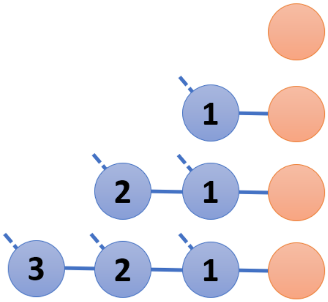
주황색 정점을 기준으로 다음과 같이 경우가 나뉩니다.

왼쪽과 오른쪽을 나눠 해결한 뒤 합칩니다.



H. 죽은 선인장의 사회

정점을 하나씩 추가할 때마다, 지름과 최대 길이를 $O(1)$ 시간에 구할 수 있습니다.

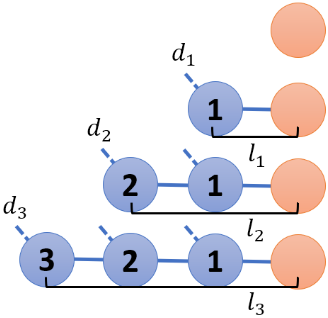


H. 죽은 선인장의 사회

주황색 정점에서 i번 정점 까지의 거리를 l_i,

i번 정점 밑의 최대 길이를 d_i 라고 합시다.

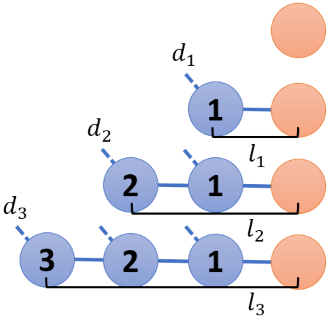
주황색 정점은 0번으로 보고, $l_0 = d_0 = 0$ 입니다.



i번 정점까지 봤을 때

지름 $R_i = \max_{0 \le p < q \le i} [(d_p - l_p) + (d_q + l_q)]$,

최대 길이 $D_i = \max_{0 \le p \le i} (d_p + l_p)$ 입니다.



H. 죽은 선인장의 사회

i번 정점까지 봤을 때

지름 $R_i = \max_{0 \le p < q \le i} [(d_p - l_p) + (d_q + l_q)]$,

최대 길이 $D_i = \max_{0 \le p \le i}(d_p + l_p)$ 입니다.

$D_i = \max(D_{i-1},\ d_i + l_i)$ 이고,

$X_i = \max_{0 \le p \le i}(d_p - l_p) = \max(X_{i-1},\ d_i - l_i)$ 라고 하면,

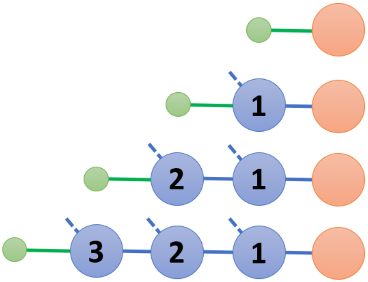
$R_i = \max\left(R_{i-1},\ X_{i-1} + (d_i + l_i)\right)$ 입니다.

H. 죽은 선인장의 사회

- 똑같이 반대쪽도 구해줍니다.
- 두 부분을 합치는 것도 비슷한 과정의 반복이고, 지름이 R을 넘을 때는 무시하면 됩니다.
- 그러므로, 총 $O\left(N \lg(N \times X)\right)$의 시간에 문제를 해결할 수 있습니다. X는 간선 길이의 상한입니다.

H. 죽은 선인장의 사회

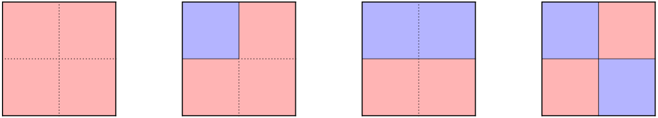
- 간선을 잘라낸 부분에 새로운 간선이 자라면 다음과 같은 형태가 됩니다.
- 어렵지 않은 일반화이므로 생략합니다.



G. 땅다람쥐

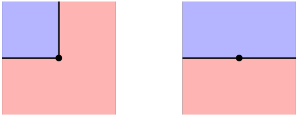
- 제출 5회, 정답 0팀 (정답률 0.00%)
- 처음 푼 팀: 없음
- 출제자: doju

G. 땅다람쥐



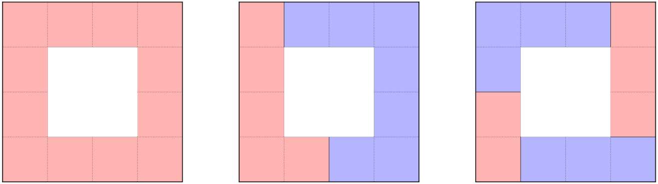
- 2×2 사각형을 채우는 경우를 생각해 봅시다.
- 첫 번째 경우는 이미 사이클이 생겼으므로 등장할 수 없습니다.
- 네 번째 경우는 파란색 칸을 연결하고 나면 빨간색 칸을 연결할 수 없게 되므로 등장할 수 없습니다.

G. 땅다람쥐



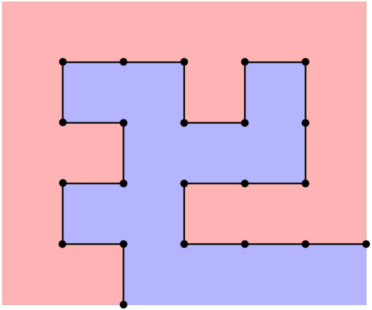
- 따라서 앞의 그림에서 두 번째와 세 번째 경우만 답에 등장할 수 있습니다.
- 두 경우 모두 두 구역 사이의 경계를 가운데의 점을 지나는 길로 볼 수 있습니다.
- 이는 답에 등장하는 모든 2×2 부분 사각형이 갖는 성질임에 주목합시다.

G. 땅다람쥐



- 다음으로 답의 테두리를 생각해 봅시다.
- 첫 번째 경우는 이미 사이클이 생겼으므로 불가능합니다.
- 세 번째 경우와 같이 같은 색의 구역이 여러 번 등장하는 경우는 파란색 칸을 연결하고 나면 빨간색 칸을 연결할 수 없게 되므로 불가능합니다.

G. 땅다람쥐



- 앞에서 나온 사실들을 조합해 보면, 답에서 두 구역의 경계는 테두리의 한 점에서 들어와 내부의 모든 점을 한 번씩 지나고 다시 테두리의 다른 점으로 빠져나가는 경로의 형태가 됩니다.

G. 땅다람쥐

- 이제 불가능한 경우를 찾을 준비가 되었습니다!

G. 땅다람쥐

- 이제 불가능한 경우를 찾을 준비가 되었습니다!

불가능한 경우

땅의 크기가 다음 조건을 만족할 때 땅을 완전히 덮지 못하도록 땅다람쥐를 배치할 수 있다.

G. 땅다람쥐

- 이제 불가능한 경우를 찾을 준비가 되었습니다!

불가능한 경우

땅의 크기가 다음 조건을 만족할 때 땅을 완전히 덮지 못하도록 땅다람쥐를 배치할 수 있다.

1. 한쪽 길이는 2이고 다른 쪽 길이는 4 이상인 경우

G. 땅다람쥐

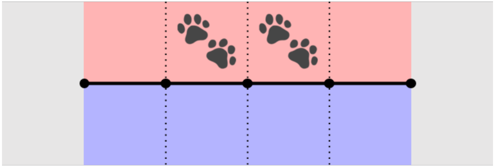
- 이제 불가능한 경우를 찾을 준비가 되었습니다!

불가능한 경우

땅의 크기가 다음 조건을 만족할 때 땅을 완전히 덮지 못하도록 땅다람쥐를 배치할 수 있다.

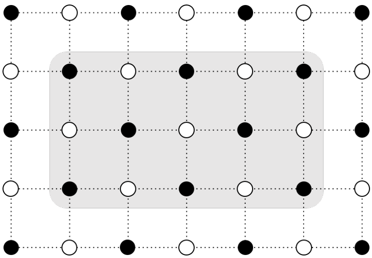
1. 한쪽 길이는 2이고 다른 쪽 길이는 4 이상인 경우
2. 세로 길이와 가로 길이가 모두 4 이상의 짝수인 경우

G. 땅다람쥐



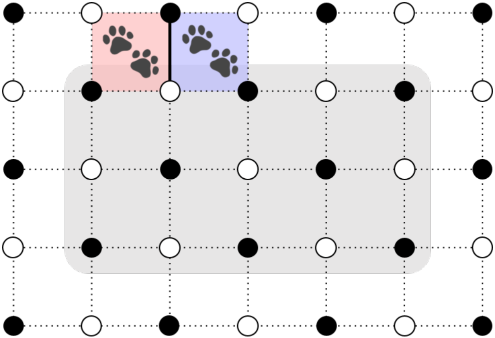
- 한쪽 길이가 2일 때 내부의 모든 점을 지나는 경로는 단 하나뿐입니다.
- 따라서 같은 방향에 땅다람쥐 두 마리를 올려놓으면 절대로 땅을 전부 덮을 수 없습니다.

G. 땅다람쥐



- 모든 점을 홀짝성에 따라 칠하면, 내부에는 검은색 점이 하얀색 점보다 하나 많습니다.
- 따라서 테두리의 검은색 점에서 출발할 경우 절대로 내부의 모든 점을 지나는 경로를 만들 수 없습니다.

G. 땅다람쥐



- 경로의 시작점을 강제하려면 땅다람쥐를 위와 같이 놓으면 됩니다.

G. 땅다람쥐

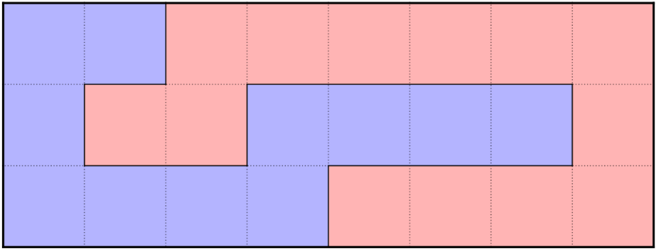
- 이 예외들 외에는 전부 가능할까요?

G. 땅다람쥐

- 이 예외들 외에는 전부 가능할까요?
- **그렇습니다!**
- 나머지는 여러분의 땀과 눈물에 맡깁니다.
- 출제자의 답을 감상하세요.

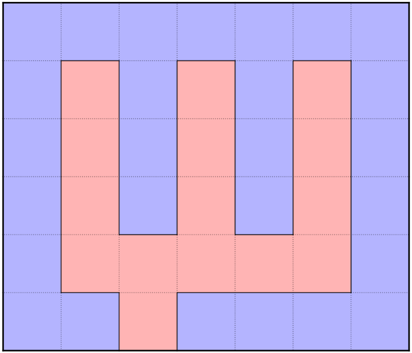
G. 땅다람쥐

- Case #1



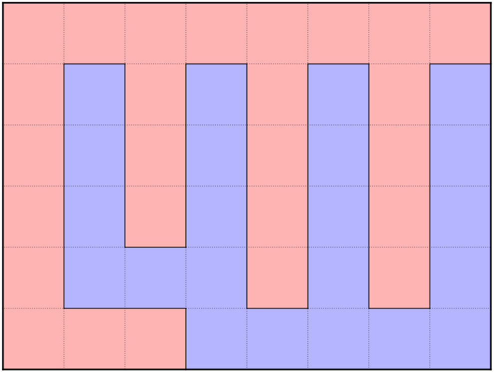
G. 땅다람쥐

- Case #2



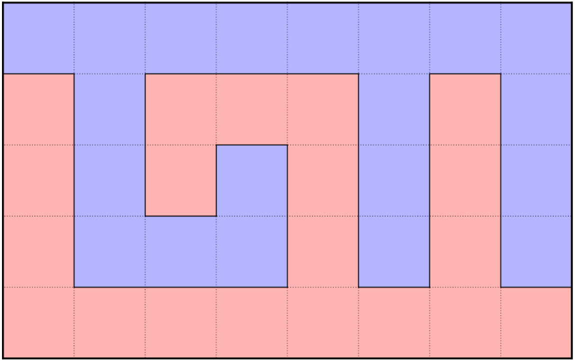
G. 땅다람쥐

- Case #3



G. 땅다람쥐

- Case #4



J. 격자 속의 숫자

- 제출 0회, 정답 0팀 (정답률 0.00%)
- 처음 푼 팀: 없음
- 출제자: 박수찬

J. 격자 속의 숫자

편의상 이 풀이에서는 0-based index를 사용하도록 하겠습니다.

J. 격자 속의 숫자

$$\sum_{i=r_1}^{r_2} \sum_{j=c_1}^{c_2} A_{i,j}$$

J. 격자 속의 숫자

- $f(x) = S[0] + S[1] + \cdots + S[x]$로 정의합시다.

$$
\begin{aligned}
\sum_{i=r_1}^{r_2} \sum_{j=c_1}^{c_2} A_{i,j} &= \sum_{i=r_1}^{r_2} \{f(i \cdot M + c_2) - f(i \cdot M + c_1 - 1)\} \\
&= \sum_{i=r_1}^{r_2} f(i \cdot M + c_2) - \sum_{i=r_1}^{r_2} f(i \cdot M + c_1 - 1)
\end{aligned}
$$

J. 격자 속의 숫자

- $f(x) = S[0] + S[1] + \cdots + S[x]$로 정의합시다.
- $\texttt{arithmetic_prefix_sum}(a, n) = \sum_{k=0}^{n} f(a + k \cdot M)$으로 정의합시다.

$$\begin{aligned}
\sum_{i=r_1}^{r_2} \sum_{j=c_1}^{c_2} A_{i,j} &= \sum_{i=r_1}^{r_2} \{f(i \cdot M + c_2) - f(i \cdot M + c_1 - 1)\} \\
&= \sum_{i=r_1}^{r_2} f(i \cdot M + c_2) - \sum_{i=r_1}^{r_2} f(i \cdot M + c_1 - 1) \\
&= \texttt{arithmetic_prefix_sum}(r_1 \cdot M + c_2, r_2 - r_1) \\
&\qquad - \texttt{arithmetic_prefix_sum}(r_1 \cdot M + c_1 - 1, r_2 - r_1)
\end{aligned}$$

J. 격자 속의 숫자

- $f(x) = S[0] + S[1] + \cdots + S[x]$
- $\texttt{arithmetic\_prefix\_sum}(a, n) = \sum_{k=0}^{n} f(a + k \cdot M)$
- $\texttt{arithmetic_prefix_sum}(a, n)$을 적당히 빨리 구할 수 있다면 문제를 해결할 수 있습니다.

J. 격자 속의 숫자

자연수의 길이를 통일할 수 있다면 계산이 편리할 것 같습니다.

S를 길이별로 ‘123456789’, ‘101112...979899’, ‘100101102103...997998999’, ...과 같이 나누어 봅시다.

각각의 그룹마다 해당 그룹의 위치 범위 안에 $a + k \cdot M$이 들어있다면 답에 $f(a + i * M)$을 더해줄 계획입니다.

예를 들어 $L = 4$글자짜리 그룹을 계산하고 있다고 합시다. 아래 그림은 $M = 17$일 때 행렬 A의 일부분을 가져온 것입니다.

7	2	2	7	7	3	2	7	7	4	2	7	7	5	2	7	7
6	2	7	7	7	2	7	7	8	2	7	7	9	2	7	8	0
2	7	8	1	2	7	8	2	2	7	8	3	2	7	8	4	2
7	8	5	2	7	8	6	2	7	8	7	2	7	8	8	2	7
8	9	2	7	9	0	2	7	9	1	2	7	9	2	2	7	9
3	2	7	9	4	2	7	9	5	2	7	9	6	2	7	9	7
2	7	9	8	2	7	9	9	2	8	0	0	2	8	0	1	2

관찰 1

자연수 x와 $x+M$는 항상 같은 열을 차지하며 행 번호의 차이는 L 입니다.

7	2	2	7	7	3	2	7	7	4	2	7	7	5	2	7	7
6	2	7	7	7	2	7	7	8	2	7	7	9	2	7	8	0
2	7	8	1	2	7	8	2	2	7	8	3	2	7	8	4	2
7	8	5	2	7	8	6	2	7	8	7	2	7	8	8	2	7
8	9	2	7	9	0	2	7	9	1	2	7	9	2	2	7	9
3	2	7	9	4	2	7	9	5	2	7	9	6	2	7	9	7
2	7	9	8	2	7	9	9	2	8	0	0	2	8	0	1	2

관찰 1

자연수 x와 $x+M$는 항상 같은 열을 차지하며 행 번호의 차이는 L입니다.

M개의 수는 $M \cdot L$글자이고, 이는 M의 배수이기 때문입니다.

7	2	2	7	7	3	2	7	7	4	2	7	7	5	2	7	7
6	2	7	7	7	2	7	7	8	2	7	7	9	2	7	8	0
2	7	8	1	2	7	8	2	2	7	8	3	2	7	8	4	2
7	8	5	2	7	8	6	2	7	8	7	2	7	8	8	2	7
8	9	2	7	9	0	2	7	9	1	2	7	9	2	2	7	9
3	2	7	9	4	2	7	9	5	2	7	9	6	2	7	9	7
2	7	9	8	2	7	9	9	2	8	0	0	2	8	0	1	2

J. 격자 속의 숫자

관찰 2

$a + k \cdot M$에 해당하는 위치를 $0 \leq k \leq n$에 대해 모두 표시해 주면 같은 열을 차지합니다.

7	2	2	7	7	3	2	7	7	4	2	7	7	5	2	7	7
6	2	7	7	7	2	7	7	8	2	7	7	9	2	7	8	0
2	7	8	1	2	7	8	2	2	7	8	3	2	7	8	4	2
7	8	5	2	7	8	6	2	7	8	7	2	7	8	8	2	7
8	9	2	7	9	0	2	7	9	1	2	7	9	2	2	7	9
3	2	7	9	4	2	7	9	5	2	7	9	6	2	7	9	7
2	7	9	8	2	7	9	9	2	8	0	0	2	8	0	1	2

관찰 2

$a + k \cdot M$에 해당하는 위치를 $0 \le k \le n$에 대해 모두 표시해 주면 같은 열을 차지합니다.

$a + k \cdot M$의 간격이 M이기 때문입니다.

7	2	2	7	7	3	2	7	7	4	2	7	7	5	2	7	7
6	2	7	7	7	2	7	7	8	2	7	7	9	2	7	8	0
2	7	8	1	2	7	8	2	2	7	8	3	2	7	8	4	2
7	8	5	2	7	8	6	2	7	8	7	2	7	8	8	2	7
8	9	2	7	9	0	2	7	9	1	2	7	9	2	2	7	9
3	2	7	9	4	2	7	9	5	2	7	9	6	2	7	9	7
2	7	9	8	2	7	9	9	2	8	0	0	2	8	0	1	2

J. 격자 속의 숫자

색칠된 열에 걸쳐 있는 자연수들을 강조해서 표시해 보면, 관찰 1에 의해 같은 형태가 L줄마다 반복된다는 사실을 알 수 있습니다.

7	2	2	7	7	3	2	7	7	4	**2**	**7**	**7**	**5**	2	7	7
6	2	7	7	7	2	7	7	8	**2**	**7**	**7**	**9**	2	7	8	0
2	7	8	1	2	7	8	2	2	7	8	3	**2**	**7**	**8**	**4**	2
7	8	5	2	7	8	6	2	7	8	7	**2**	**7**	**8**	**8**	2	7
8	9	2	7	9	0	2	7	9	1	**2**	**7**	**9**	**2**	2	7	9
3	2	7	9	4	2	7	9	5	**2**	**7**	**9**	**6**	2	7	9	7
2	7	9	8	2	7	9	9	2	8	0	0	**2**	**8**	**0**	**1**	2

또한 관찰 1에 의해, 걸쳐 있는 위치가 동일한 자연수는 간격이 M 이라는 것도 알 수 있습니다.

7	2	2	7	7	3	2	7	7	4	**2**	**7**	**7**	**5**	2	7	7
6	2	7	7	7	2	7	7	8	**2**	**7**	**7**	**9**	2	7	8	0
2	7	8	1	2	7	8	2	2	7	8	3	**2**	**7**	**8**	**4**	2
7	8	5	2	7	8	6	2	7	8	7	**2**	**7**	**8**	**8**	2	7
8	9	2	7	9	0	2	7	9	1	**2**	**7**	**9**	**2**	2	7	9
3	2	7	9	4	2	7	9	5	**2**	**7**	**9**	**6**	2	7	9	7
2	7	9	8	2	7	9	9	2	8	0	0	**2**	**8**	**0**	**1**	2

J. 격자 속의 숫자

또한 각각의 줄마다 $f(a+k\cdot M)$의 값을 구하기 위해 다음과 같은 정보를 얻을 수 있습니다.

- $g(n,i)$: $1, 2, \cdots, n-1$을 모두 나열했을 때 숫자 합
  $+$ n의 첫 i자리 숫자 합

$\cdots$	7	4	2	7	7	**5**	2	7	7	$g(2775,3)$
$\cdots$	8	2	7	7	9	2	7	8	0	$g(2779,4)$
$\cdots$	2	7	8	3	2	**7**	**8**	**4**	2	$g(2784,1)$
$\cdots$	7	8	7	2	7	**8**	**8**	2	7	$g(2788,2)$
$\cdots$	9	1	2	7	9	**2**	2	7	9	$g(2792,3)$
$\cdots$	5	2	7	9	6	2	7	9	7	$g(2796,4)$
$\cdots$	2	8	0	0	2	**8**	**0**	**1**	2	$g(2801,1)$

관찰 1에 의해 L줄 간격으로 x가 M 차이씩 나고 i는 값이 똑같기 때문에, 결국 다음 정보를 L개 얻을 수 있습니다.

- $h(x, k, i)$: $g(x, i) + g(x + M, i) + \cdots + g(x + k \cdot M, i)$의 값을 답에 더해야 함

⋯	7	4	2	7	7	**5**	2	7	7	$g(2775, 3)$
⋯	8	2	7	7	9	2	7	8	0	$g(2779, 4)$
⋯	2	7	8	3	2	**7**	**8**	**4**	2	$g(2784, 1)$
⋯	7	8	7	2	7	**8**	**8**	2	7	$g(2788, 2)$
⋯	9	1	2	7	9	**2**	2	7	9	$g(2792, 3)$
⋯	5	2	7	9	6	2	7	9	7	$g(2796, 4)$
⋯	2	8	0	0	2	**8**	**0**	**1**	2	$g(2801, 1)$

J. 격자 속의 숫자

- $g(n, i)$: $[1, n-1]$ 숫자 합 + n의 첫 i자리 숫자 합
- $h(x, k, i)$: $g(x, i) + g(x + M, i) + \cdots + g(x + k \cdot M, i)$의 값을 답에 더해야 함

각각의 $h(x, k, i)$ 쌍은 $O(L \cdot 10)$ 시간에 계산할 수 있습니다. 아래와 같은 상태로 DP를 해서 적절히 전처리를 한 후, 앞에서부터 한 자리씩 결정해 나가면서 적절히 숫자들의 합을 구할 수 있습니다.

- "$[1, n-1]$ *숫자 합*": (수의 길이, 수 mod M)
- "*n의 첫 i자리 숫자 합*": (수의 길이, 고려하는 접두사의 길이, 수 mod M)

J. 격자 속의 숫자

$O(MAXL^2)$개의 h 쌍이 나오고 각 쌍은 $O(L \cdot 10)$에 풀리므로, 질의당 시간복잡도는 $O(MAXL^3 \cdot 10)$입니다.
전처리의 시간복잡도는 $O(MAXL^2 \cdot 10 \cdot M)$이며, 공간복잡도는 $O(MAXL \cdot 10 \cdot M)$입니다.

J. 격자 속의 숫자

검수진에 의하면 이 문제는 $M \leq 10^9$일 때도 해결할 수 있습니다.

답을

$$F(x_0, k_1, k_2, b, c) = \sum_{0 \leq x \leq x_0} \sum_{k_1 \leq k \leq k_2} \left\lfloor \frac{k + b \cdot x}{c} \right\rfloor$$

함수의 값 $O(MAXL^3)$개로 나타낼 수 있고, 이 함수의 값은 빠르게 구할 수 있기 때문입니다.

A4용지 3장 정도만 쓰시면 됩니다.