IamCoder 2021 송년대회 풀이

Official Solutions

by

나는코더다 37기 문홍윤 박재민 송준혁

2021년 12월 30일

문제		의도한 난이도	출제자
A	아기 홍윤	**Easy**	mhy908
B	삼색 그래프	**Medium**	arnold518
C	돌의 정령 줄세우기	**Easy**	mhy908
D	기차 여행	**Medium**	SongC
E	두 트리	**Challenging**	mhy908
F	사진 촬영	**Hard**	arnold518
G	정기 모임 3	**Medium**	SongC
H	향수	**Challenging**	SongC
I	카카오 택시	**Medium**	arnold518
J	깔때기와 비커	**Hard**	mhy908
K	맥스웰의 악마	**Medium**	SongC
L	선형대수학	**Hard**	mhy908

A. 아기 홍윤

`slding_window`
출제진 의도 – **Easy**

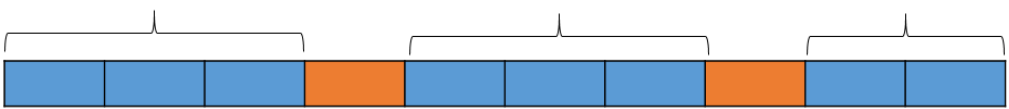
- 제출 68번, 정답 26팀 (정답률 44.118%)
- 처음 푼 팀: **콜라는 펩시지** (정석현, 김진휘, 김상혁), 5분
- 출제자: `mhy908`
- 가장 짧은 검수진 코드 : 342bytes

A. 아기 홍윤

- 어떤 수에 bitwise or 연산을 적용하면 크기는 항상 커집니다.
- 구간의 양 끝점이 s, e라 합시다.
- 구간 전체의 or값이 k보다 작으면 e를, 크면 s를 1씩 증가시키는 방법을 쓸 수 있습니다.
- 현재 상태에서 구간의 or값을 구하는 것은 각 비트별로 구간에 몇개가 켜져있는지를 관리하는 배열을 관리하면 됩니다.
- 이를 흔히 inchworm이라 부릅니다.

A. 아기 홍윤

- 다른 풀이도 있습니다.
- 어떤 수들의 or값이 k 이려면, 적어도 k 에서 꺼진 비트가 그 수들에서 켜져있어서는 안됩니다.
- 이는 각 수를 x 라 하면, $(x|k) <= k$ 를 확인하는 것으로 충분합니다.
- 절대로 포함되어서는 안되는 값들을 배열에서 전부 지우고 남은 구간 조각들만 직접 살펴봐도 답을 찾을 수 있습니다.



B. 삼색 그래프

`ternary_search`, `dijkstra`
출제진 의도 – **Medium**

- 제출 15번, 정답 3팀 (정답률 20.000%)
- 처음 푼 팀: **그날 본 코드의 오류를 우리는 아직 모른다** (최우현, 노현서, 김민욱), 249분
- 출제자: `arnold518`
- 가장 짧은 검수진 코드 : 950bytes

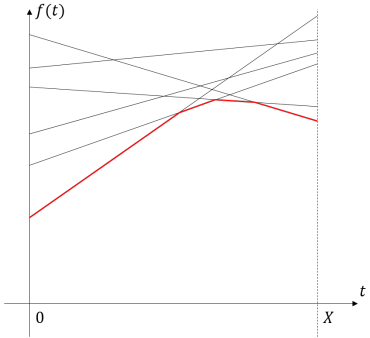
B. 삼색 그래프

- 1번 정점에서 N번 정점으로 향하는 임의의 경로를 생각해봅시다.
- 분말스프를 뿌리기 전 이 경로의 가중치 합을 w, 경로 위의 빨간색 간선의 개수 r, 파란색 간선의 개수 b개라 합시다.
- 만약 분말스프를 빨간색 간선에 t스푼, 파란색 간선에 $X-t$스푼 뿌렸다면 이 경로의 최종 길이는 $w+r\times t+b\times(X-t)$입니다.

B. 삼색 그래프

- 분말스프를 빨간색, 파란색 간선에 어떻게 나눌지를 결정하는 것이 문제의 가장 어려운 부분이니, t를 변수로 생각해 봅시다.
- 경로 (w, r, b)에서의 비용함수 $f(t) = w + bX + (r - b)t$로, t에 대한 일차함수입니다.
- 이제, 정해진 t에 대한 최단경로의 길이는 여러 개의 (w, r, b)에 따른 각자 다른 경로들에 대해 $f(t)$들의 최솟값입니다.
- t를 적당히 조절하여 $f(t)$들의 최솟값이 최대화될 수 있는 t를 찾아야 합니다.

B. 삼색 그래프



- 직선들을 그린 후, 최솟값만 생각하면 그림과 같이 Convex Hull의 형태를 띄게 됩니다.
- Convex 한 함수는 Unimodal하기 때문에, 삼분탐색을 활용하면 문제를 해결할 수 있습니다.

B. 삼색 그래프

- 삼분탐색 풀이의 시간복잡도는 $O((V+E)logV)$ 의 시간복잡도에 다익스트라, $O(logX)$ 의 시간복잡도에 삼분탐색을 통해 전체 $O((V+E)logVlogX)$ 입니다.
- 별해로, 어떤 t 값에서의 최단 경로 뿐만아니라 해당 직선의 기울기까지 알아낼 수 있습니다.
- 최대점의 왼쪽 직선들은 모두 기울기가 양수이고, 최대점의 오른쪽 직선들은 모두 기울기가 음수라는 점을 활용하면 삼분탐색이 아닌, 이분탐색을 이용하여 문제를 해결할 수도 있습니다.

C. 돌의 정령 줄세우기

`constructive`
출제진 의도 – **Easy**

- 제출 46번, 정답 29팀 (정답률 54.348%)
- 처음 푼 팀: **notorious coincidence** (박영우, 박종경, 이민욱), 11분
- 출제자: `mhy908`
- 가장 짧은 검수진 코드 : 392bytes

Special Thanks to ...

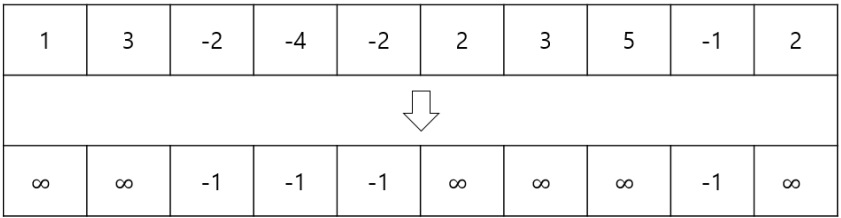
C. 돌의 정령 줄세우기

- ✓ 돌정령은 귀엽습니다.

C. 돌의 정령 줄세우기

- 다양한 풀이가 가능하지만, 가장 쉬운 풀이를 소개하겠습니다.

C. 돌의 정령 줄세우기

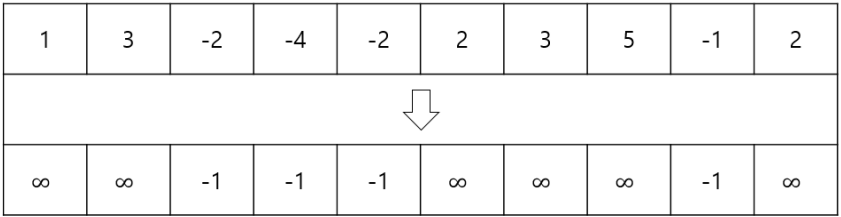


✓ 음수인 조건은 -1로, 양수인 조건은 ∞로 바꾸면 항상 조건이 강력해집니다.

C. 돌의 정령 줄세우기

- -1과 ∞로만 이루어진 문제는 다음과 같은 방법으로 항상 풀 수 있습니다.
- 먼저, N 번째 수가 음수라면 답은 항상 -1입니다.
- 그 이외의 경우, N 부터 1 까지 내림차순으로 적혀있는 순열을 만듭니다.
- -1인 구간이 $[s, e]$ 라면, $[s, e+1]$ 구간을 뒤집으면 답입니다.

C. 돌의 정령 줄세우기



✓ 위 상황의 답은 ’10 9 5 6 7 8 4 3 1 2’가 될 수 있습니다.

D. 기차 여행

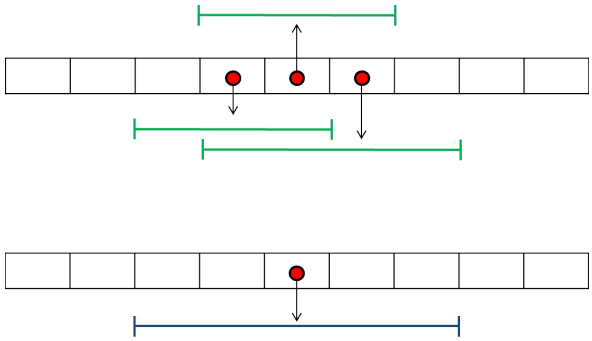
`sparse_table`, `segtree`
출제진 의도 – **Medium**

- 제출 41번, 정답 7팀 (정답률 17.073%)
- 처음 푼 팀: **부릉부릉 이환버스** (채이환, 문정후, 이동현), 35분
- 출제자: SongC
- 가장 짧은 검수진 코드 : 1450bytes

D. 기차 여행

- i번 도시에서 한 번 기차를 타고 이동할 수 있는 위치들은 i번 도시를 포함하는 하나의 구간으로 표현됩니다.
- i번 도시에서 두 번 기차를 타고 이동할 수 있는 위치들은 어떻게 될까요?
- i번 도시에서 한 번 기차를 타고 갈 수 있는 곳들로부터 한 번 기차를 타고 갈 수 있는 곳들의 합집합입니다.

D. 기차 여행



D. 기차 여행

- 여기서 관찰할 수 있는 사실은 몇 번을 타고 이동하더라도 가능한 위치의 집합은 항상 구간으로 나올 것이라는 점입니다.
- 한 번 이동 후 한 번 이동하는 위치가 두 번 이동하는 위치라면 그 이상의 이동에 대해서도 비슷하게 구간을 합쳐주면 어떨까요?

D. 기차 여행

- 현재의 구간과 각 도시별로 k번 이동했을 때의 구간을 모두 알고 있다면 현재 도시에서 k번 더 이동했을 때의 구간을 알 수 있을까요?
- 이 부분은 세그먼트 트리를 이용해서 할 수 있습니다.
- 구간 내에서 k번 이동시의 가장 왼쪽 점과 가장 오른쪽 점을 RMQ로 구하면 충분합니다.

D. 기차 여행

- 그렇다면 모든 도시에서 한 번 움직여 갈 수 있는 구간, 두 번 움직여 갈 수 있는 구간, $\cdots 2^k$ 번 움직여 갈 수 있는 구간을 모두 구할 수 있습니다.
- 8번 움직일 때의 구간은 4번 움직인 구간 내에서 4번 움직인 구간들을 합집합한 것이기 때문입니다.
- 이런 방식을 흔히 **Sparse Table**이라고 부릅니다.

D. 기차 여행

- 2^k 번의 이동들에 대해 각각 세그먼트 트리를 구성했다면 이제 이분 탐색을 할 수 있습니다.
- 현재 가능한 위치들에서 2^k 번 더 이동하면 목적지를 포함하는지 아닌지를 가지고 이분 탐색으로 도달에 필요한 최소 이동 수를 결정할 수 있습니다.
- 총 시간 복잡도는 $O((N+Q)log^2N)$ 입니다.

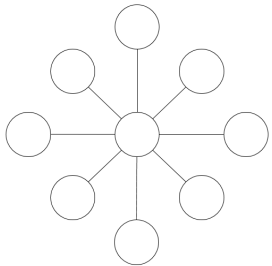
E. 두 트리

`constructive, ad_hoc`
출제진 의도 – **Challenging**

- 제출 6번, 정답 0팀 (정답률 0.000%)
- 처음 푼 팀: **??** (?, ?, ?), ?분
- 출제자: `mhy908`
- 가장 짧은 검수진 코드 : 4550bytes

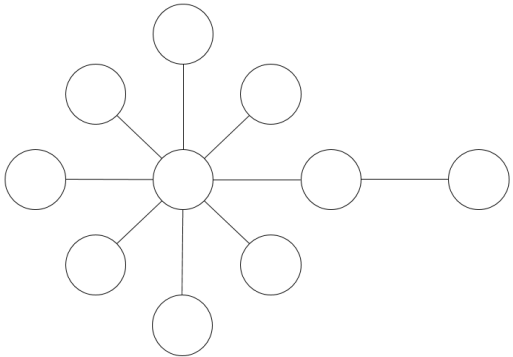
E. 두 트리

- 답이 -1인 경우부터 생각해 봅시다.
- 어느 한쪽 트리가 '성게'라면 항상 간선이 적어도 하나 겹칠 수 밖에 없습니다.

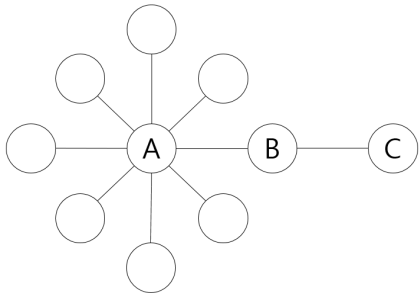


E. 두 트리

- 그렇다면 성게가 되기 바로 직전의 'semi-성게'에 대한 답을 구해봅시다.



E. 두 트리



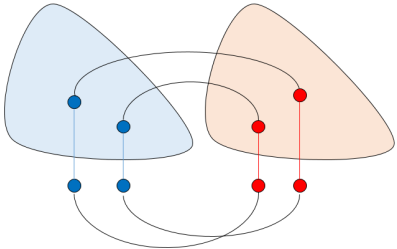
- A에 다른 트리의 리프를 연결합니다.
- 연결한 리프와 인접한 정점을 C에 연결합니다.
- 연결한 리프와 거리가 3 이상인 아무 정점을 B에 연결합니다.
- 남은 정점은 아무렇게나 연결해도 됩니다.

E. 두 트리

- 귀납을 사용합니다.
- 만약 크기 N 인 두 트리의 일대일 대응이 존재한다면,
- 각 트리별로 정점을 두 개씩 선택해서 리프를 두 개씩 새로 단 크기 $N+2$ 인 두 트리의 일대일 대응도 존재합니다.

E. 두 트리

- 파란 트리에 추가한 리프를 a, b라 하고,
- 빨간 트리에 추가한 리프를 c, d라 하면,
- a-c/b-d와, a-d/b-c 중 적어도 하나는 조건을 만족합니다.



E. 두 트리

- 이제 문제를 풀 준비가 전부 끝났습니다.
- 거꾸로, 각 트리에서 거리가 3 이상인 리프 쌍을 계속해서 뽑아봅시다.
- 리프를 뽑는 도중, 어느 한 트리가 'semi-성게'가 된다면 즉시 종료하고 위에서 언급한 방법으로 대응을 시킵니다.
- 뽑은 리프 쌍들의 역순으로 다시 대응을 시킵니다.

E. 두 트리

- 주의할 점이 있습니다.
- semi-성게를 거치지 않고 바로 성게가 되는 케이스가 있습니다.
- 가장 큰 정점의 디그리가 $N-2$이고, 거리가 2인 정점이 두개 있는 모양이 반례인데,
- 이는 단순히 거리가 4인 리프 쌍만 뽑지 않으면 해결 가능합니다.
- 다만 $N=5$ 인 일직선 트리의 경우 그럴 수 없기 때문에 이 케이스만 별도로 완전탐색을 통해 답을 구할 수 있습니다.

F. 사진 촬영

dp

출제진 의도 – **Hard**

- 제출 4번, 정답 1팀 (정답률 25.000%)
- 처음 푼 팀: **부릉부릉 이환버스** (채이환, 문정후, 이동현), 100분
- 출제자: `arnold518`
- 가장 짧은 검수진 코드 : 1180bytes

F. 사진 촬영

- 첫 번째로, 모든 swap 연산을 실행한 이후에 개인사진이나, 단체사진을 찍는 것이 항상 최적의 방법입니다.
- 만약 중간에 사진을 한번 찍고 swap 연산으로 이동시키는 최적해가 존재한다면, 모든 사진촬영을 swap 이후로 이동시켜서 swap 연산 이후에 사진촬영이 오는 최적해로 만들 수 있습니다.
- 그렇다면, swap 연산들을 다 수행한 이후에 개인사진으로 찍히는 학생에는 A, 단체사진으로 찍히는 학생들에는 B를 써 봅시다. 또, 같은 단체사진으로 찍히는 학생들은 B_1, B_2와 같이 단체사진의 번호를 붙여 구분합시다.

F. 사진 촬영

- 이렇게 최종 상태의 배열에 A, B를 적어놓고, 모든 swap 연산을 수행시키기 전의 상태로 돌아가면 전체 배열은 A, B가 섞여 있는 형태가 될 것입니다.
- 이 배열에서 관찰할 수 있는 점은 B들만 뽑아서 보았을 때 위치상 연속한 값들이 찍힌 단체사진의 번호도 같을 수 있다는 점입니다.
- 예를 들어, $AAB_1B_2AB_1B_2AB_1B_2B_2AA$와 같은 배치는 최적이 될 수 없고, $AAB_1B_1AB_1B_1AB_2B_2B_2AA$와 같은 배치는 최적이 될 수 있습니다.

F. 사진 촬영

- 이제 문제를 거꾸로 생각해서, 초기 배열에 A, B_i 가 모두 적혀 있다면 이를 swap해서 B_i 들이 인접하도록 하는 최소 비용을 먼저 계산해 봅시다.
- $BAABABBAAB$ 등의 배열 하나에서 B 가 모두 인접하도록 최적으로 swap하기 위해서는 B 의 중앙값 위치를 기준으로 모아야 합니다.
- 이는 먼저 swap을 통해 모든 B 가 정확히 중앙값의 위치에 존재하도록 이동시킨 이후, 다시 일렬로 퍼뜨리는 비용을 빼는 방식으로 생각할 수 있습니다.

F. 사진 촬영

- 이 문제는 수직선 위의 몇개의 점을 한 점으로 모으기 위한 최소 비용으로 생각할 수 있으며, 중앙값으로 모으는 것이 최적임이 알려져 있습니다.
- 구체적인 비용을 계산하기 위해서는 중앙값 기준으로 왼쪽의 점들의 좌표를 빼주고, 오른쪽 점들의 좌표를 더하는 방식으로 이동 비용을 계산할 수 있습니다.
- 따라서 원래 문제로 돌아가서 B가 모두 인접하도록 하는 최소 swap의 횟수은 중앙값 기준으로 왼쪽 점들의 좌표는 빼주고, 오른쪽 점들의 좌표는 더해준 후, 일렬로 퍼뜨리는데 드는 추가 비용을 빼주면 됩니다.
- 이 때 빼줘야 할 추가 비용은 오로지 B의 개수에만 영향을 받습니다.

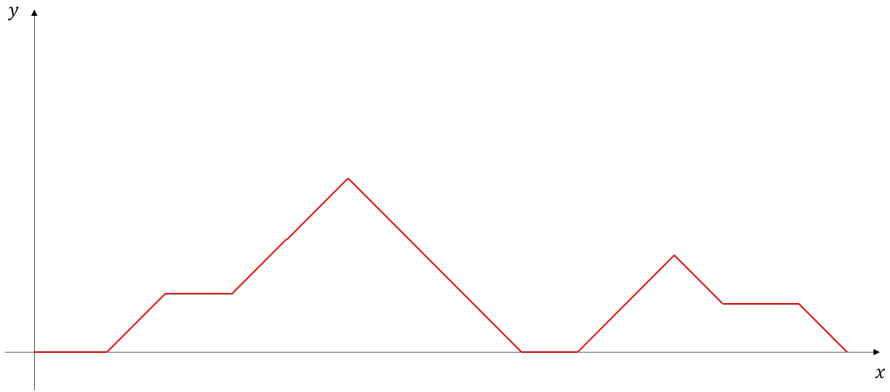
F. 사진 촬영

- ✓ 지금까지의 관찰들을 바탕으로 원래 문제는 초기 배열에 A, B_i 를 적어가며 Dynamic Programming을 이용하여 해결합니다.
- ✓ B_i 를 사용할 수 있는 개수에 제한이 걸려 있고, 위 계산 과정에서 B_i 를 더하고 빼는 위치를 결정해야 하기 때문에, 배열의 위치 x 와는 별개로 현재까지 쌓아올리고 있는 B 의 개수를 나타내는 추가 변수 y 가 필요합니다.

F. 사진 촬영

- 앞에서부터 x를 증가시키고 한 칸씩 A, B를 배정해 나가며
- A를 배정할 때는 y를 유지
- B_i를 배정할 때 B_i가 중간값보다 앞쪽에 존재해 나중에 계산될 때 -로 작용하면 y를 1 증가
- B_i를 배정할 때 B_i가 중간값보다 뒤쪽에 존재해 나중에 계산될 때 +로 작용하면 y를 1 감소
- 의 형태로 DP를 전이시킵니다.
- 이 때, B의 개수가 홀수일 경우에 주의합시다.

F. 사진 촬영



A, B를 배정한 한 가지 예시입니다.

F. 사진 촬영

- $dp[x][y]$를 전이시킬 때
- A를 배정할 때는 A_i를 더해주고
- B를 배정하며 y를 1 증가시킬 때는 $C\times$(현재 좌표값)을 빼주고
- B를 배정하며 y를 1 감소시킬 때는 $C\times$(현재 좌표값)을 더해주고
- 곡선이 상승 방향에서 하향 방향으로 바뀌는 지점, 즉 산의 꼭대기에서 위에서 언급됐던 B를 다시 퍼트리기 위한 $C\times$(추가 비용)을 빼주면 됩니다. 이것이 가능한 이유는 꼭대기에서 B의 개수가 결정되며 추가 비용은 구간의 길이에 의해서만 결정되기 때문입니다.

F. 사진 촬영

- 구현 과정은 매우 단순하며 곡선이 0일때, 상승할 때, 하강할 때의 3가지 케이스를 모두 $O(N^2)$ DP로 관리하면 됩니다.
- 추가로 $K = 1$일 때는 $min(A_i, B)$의 합으로 계산할 수 있습니다.
- 또한, $K = 2$일 때의 DP 전이 과정에서의 주의가 필요합니다.

G. 정기 모임 3

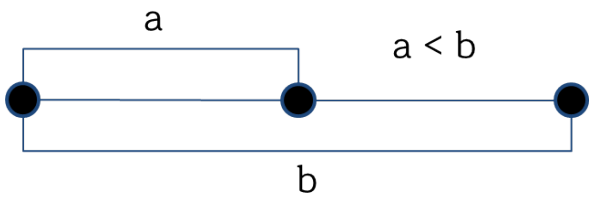
`dp_tree`, `dfs`

출제진 의도 – **Medium**

- 제출 34번, 정답 5팀 (정답률 14.706%)
- 처음 푼 팀: **notorious coincidence** (박영우, 박종경, 이민욱), 95분
- 출제자: SongC
- 가장 짧은 검수진 코드 : 1220bytes

G. 정기 모임 3

- 사람들이 배치된 모양의 형태를 관찰해봅시다.
- 사람과 사람 사이의 경로에 다른 사람이 있을 수 있을까요?
- 모든 사람 사이의 거리가 같기 때문에 답은 불가능하다입니다.

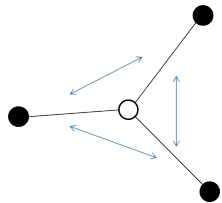


G. 정기 모임 3

- 여기서 X 의 홀짝성을 나누어 생각해봅시다.
- 먼저 홀수일 때를 살펴봅시다.
- 답이 3 이상이 될 수 있을까요?

G. 정기 모임 3

- 정답은 **불가능하다** 입니다.



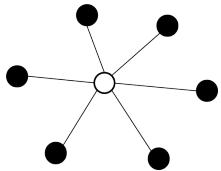
- 표시된 경로 쌍 셋이 모두 길이의 기우성이 달라야 하고, 이는 불가능합니다.
- 따라서 X 가 홀수일 때의 답은 X 가 지름보다 작거나 같을 때 2, 아닐 때 1입니다.

G. 정기 모임 3

- X 가 짝수일 때는 어떨까요?
- 어떤 두 사람 사이의 경로와 그 바깥의 한 사람 사이의 관계를 봅시다.
- 그 밖의 한 사람과 가장 가까운 경로 위의 점은 경로의 중점이어야 합니다.
- 그렇지 않다면 두 사람의 바깥 사람과의 거리가 서로 달라지므로 불가능합니다.

G. 정기 모임 3

- 따라서 답은 그림처럼 하나의 중심점으로부터 $K/2$씩 떨어진 점의 집합입니다.
- 이때 각 점은 중심점을 기준으로 다른 서브트리에 있어야 합니다.



G. 정기 모임 3

- 이제 각 점이 중심일 때 거리별로 가능한 최대 개수를 구해봅시다.
- 이 부분은 트리 DP를 하면 알 수 있습니다.
- 루트를 잡고 아래쪽 서브 트리의 최대 깊이들을 모두 구합니다.
- 이후 한 번 더 DFS를 돌려서 부모 방향 서브 트리의 최대 깊이를 계산합니다.
- 모든 정점의 차수 합은 $O(N)$ 이니 적절히 답 배열에 넣고 누적합 등으로 풀어주면 최종 시간 복잡도 $O(N)$ 에 답을 구할 수 있습니다.

H. 향수

`dp`, `alien`, `monotone_queue_optimization`, `pst`

출제진 의도 – **Challenging**

- 제출 5번, 정답 0팀 (정답률 0.000%)
- 처음 푼 팀: **???** (?, ?, ?), ?분
- 출제자: SongC
- 가장 짧은 검수진 코드 : 2420bytes

H. 향수

- 일단 좌표압축을 하고 생각합시다.
- 가장 단순한 DP 풀이를 떠올려 봅시다.
- $DP[i][j]$ 를 i 위치에 마지막 향수를 놓았고 j 개의 향수를 놓았을 때의 최대 비용이라고 정의합시다.
- 이렇게 정의하면 마지막 위치가 x 고 y 에 새로운 향수를 놓았을 때의 추가되는 행복도는 시작 지점이 $(x, y]$ 에 있고 끝 지점이 $[y, \inf)$ 에 있는 구간들의 가중치 합입니다.

H. 향수

- ✓ $DP[i][j]$ 가 $DP[k][j-1]$ 을 참조하고 비용함수 계산을 생각하면 이 DP는 $O(N^4)$ 입니다.
- ✓ 정말 말도 안되는 것 같지만 만점 풀이는 이 DP를 발전시켜서 만듭니다.

H. 향수

- 이 DP의 비용 함수는 **시작 지점이** $(x, y]$ **에 있고 끝 지점이** $[y, \inf)$ **에 있는 구간들의 가중치 합**입니다.
- 다시 말해 2차원 평면 상에서 직사각형 영역 내의 점의 가중치 합입니다.
- 따라서 비용 함수는 PST를 이용해 $O(logN)$ 에 계산됩니다.
- 이제 풀이는 $O(N^3logN)$ 입니다. 아직 갈 길이 멉니다.

H. 향수

- 이 DP의 비용 함수는 **시작 지점이** $(x, y]$ **에 있고 끝 지점이** $[y, \inf)$ **에 있는 구간들의 가중치 합**입니다.
- 그리고 이 비용 함수는... monge합니다. (사각부등식을 만족합니다.)
- 여기서 우리는 K에 대해 답이 볼록하다는 사실을 알 수 있고....
- Alien trick을 적용할 수 있습니다. 자세한 설명은 생략합니다.
- 이제 풀이는 $O(N^2 log N log X)$ 입니다. 아직도 갈 길이 멉니다.

H. 향수

- Alien을 적용했다면 향수의 개수 차원이 떨어져 1차원 DP가 되었습니다.
- 그리고 역시 비용 함수는... 여전히 monge합니다.
- 따라서 Monotone Queue opt를 적용할 수 있습니다. 자세한 설명은 생략합니다.
- 이제 풀이는 $O(Nlog^2NlogX)$ 입니다.
- 이제 거의 선형입니다! 하지만 로그 세제곱은 아직 TLE죠.

H. 향수

- 지금 계산되는 세 개의 로그는 각각,
- Alien
- Monotone Queue opt의 Cross 계산
- PST를 쓰는 비용 함수의 계산
- 에서 나옵니다.
- 마지막 두 로그를 합칠 수 있을 것 같지 않나요?

H. 향수

- 실제로 그렇습니다!
- cross 계산은 결국 특정 부등식을 만족하는 최소 인덱스를 찾는 작업이기 때문에 PST 내에서 세그 이분탐색으로 로그를 합쳐버릴 수 있습니다.
- 최종 시간 복잡도는 $O(NlogNlogX)$ 입니다. 수고 많았습니다.
- 풀이가 이해가 안된다면 하나씩 공부해 보는 것도 좋을 것 같습니다.

I. 카카오 택시

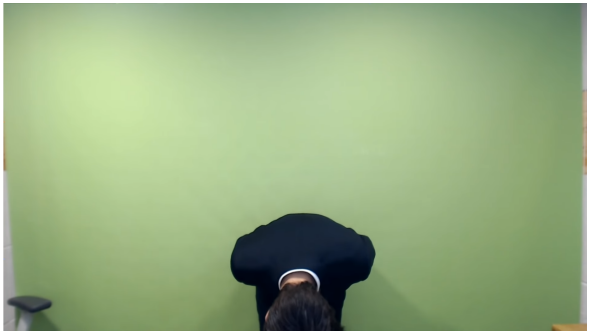
`implementation`, `graph_traversal`
출제진 의도 – **Medium**

- 제출 11번, 정답 5팀 (정답률 45.455%)
- 처음 푼 팀: **그날 본 코드의 오류를 우리는 아직 모른다** (최우현, 노현서, 김민욱), 102분
- 출제자: `arnold518`
- 가장 짧은 검수진 코드 : 3330bytes

Special Thanks to ...

I. 카카오 택시

- ✓ 출제자 코드 길이 : 3245B
- ✓ 검수진 최대 코드 길이 : 4298B
- ✓ 죄송합니다.



I. 카카오 택시

- ✓ 먼저 교차로들을 입력으로 받은 후, 존재하는 모든 도로들을 생성해줍니다.
- ✓ 만약 T 가 작다면 T 초 동안의 카카오 택시의 움직임을 하나하나 시뮬레이션해주면 $O(T)$ 에 문제를 해결할 수 있습니다.

- 어떤 두 상황이 같은 도로에서, 같은 방향으로 차량이 위치하고, 그 시간이 K 로 나눈 나머지가 같다면?
- 이후의 모든 신호체계 변화가 두 상황에서 같기 때문에 두 상황에서 같은 이동을 합니다.
- 따라서 이미 방문했던 도로를, 전에 방문한 것과 같은 방향, 시간을 K 로 나눈 나머지가 같다면 이후에는 일정한 패턴이 반복됩니다.

- 전체 상태의 개수를 도로와 방향, 시간을 K 로 나눈 나머지로 나타낼 수 있고, 그 개수는 $O(NK)$ 입니다.
- 따라서 시뮬레이션을 진행하다, 방문한 상태에 도달하면 남은 시간과 발견한 사이클의 길이를 활용하여 바로 답을 구할 수 있습니다.
- 시간 복잡도는 $O(NK)$, 구현에 map 이나 set 등을 활용하면 $O(NKlogN)$ 입니다.

- 추가로, 이와 같이 현재 상태에서 다음 상태로 전이할 간선이 하나로 정해져 있는 그래프를 functional graph라고 합니다.
- 이 그래프의 구조적인 특징을 활용하면 입력이 쿼리로 주어질 때도 문제를 해결할 수 있습니다.
- 또한, Sparse Table을 활용해도 쿼리가 주어질 때 문제를 해결할 수 있습니다.

J. 깔때기와 비커

`offline_queries`, `sparse_table`, `segtree`
출제진 의도 – **Hard**

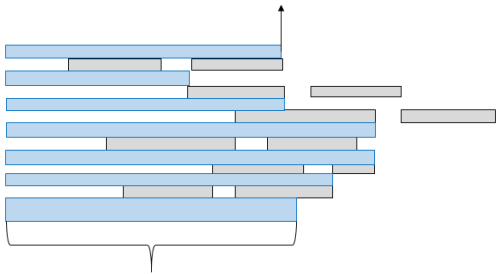
- 제출 9번, 정답 0팀 (정답률 0.000%)
- 처음 푼 팀: **???** (?, ?, ?), ?분
- 출제자: mhy908
- 가장 짧은 검수진 코드 : 2510bytes

J. 깔때기와 비커

- 어떤 구간에 떨어지는 물방울의 초기 좌표 범위는 항상 구간을 이룹니다.
- 만약 비커의 위치가 $[x, y]$ 라면, 출력해야 하는 값은 $[0, y]$ 의 답에서 $[0, x]$ 의 답을 뺀 것입니다.

J. 깔때기와 비커

- 각 쿼리에 대한 답을 $O(N)$ 에 구해봅시다.



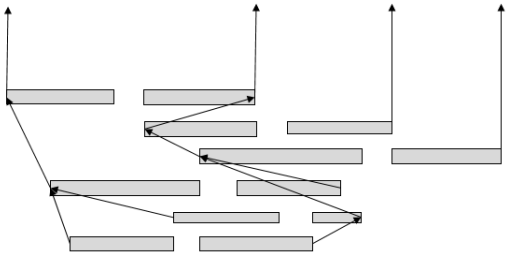
- 위 그림처럼 물방울들의 경로를 역추적하면 규칙이 보입니다.

J. 깔때기와 비커

- $[0, x]$에 대한 쿼리를 풀기 위해서는,
- x 좌표에서 $+y$ 방향으로 레이저를 쏩니다.
- 깔때기를 만나면 멈추는데, 왼쪽 부분에 맞았다면 가장 왼쪽으로, 오른쪽 부분에 맞았다면 가장 오른쪽으로 좌표를 이동합니다.
- 마지막 순간의 레이저의 좌표가 답입니다.
- 이 과정을 자료구조를 통해 $O(lgN)$ 정도로 최적화하면 문제가 풀릴 거 같습니다.
- $O(lg^2N)$ 풀이도 다양하게 존재하지만, 아마도 시간 제한 안에 작동하지 않을 것입니다.

J. 깔때기와 비커

- 2차원 평면상에서 사실 유의미한 점은 깔때기의 양 끝점과 비커의 양 끝 좌표 정도입니다.



- 그 점들에서 레이저가 지나가는 모습은 위의 그림처럼 포레스트의 형태를 띕니다.

J. 깔때기와 비커

- 또한, 쿼리로 들어오는 비커를 오프라인으로 처리하여, 그 위치를 저장해 둘 수 있습니다.
- 위 포레스트는 세그먼트 트리 등을 활용하여 sweeping을 해주면서 $O((N+Q)lg(N+Q))$ 정도에 구축할 수 있습니다.
- 각 비커에 대한 쿼리는 포레스트에 sparse table을 적용하면 쿼리당 $O(lgN)$ 에 해결할 수 있습니다.

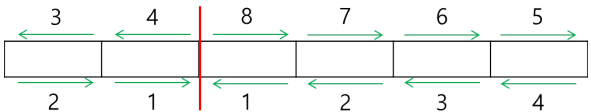
K. 맥스웰의 악마

`number_theory, greedy`
출제진 의도 – **Medium**

- 제출 28번, 정답 9팀 (정답률 32.143%)
- 처음 푼 팀: **본선도 좋은 결과가 있으면 좋을 것 같습니다** (최다니엘, 김용준, 박민철), 38분
- 출제자: SongC
- 가장 짧은 검수진 코드 : 789bytes

K. 맥스웰의 악마

- 우선 칸막이를 항상 닫아놓고, 매 순간 칸막이에 동시에 부딪혀오는 입자들의 위치를 서로 바꿀 수 있다고 생각해봅시다.
- 그리고 어떤 입자들끼리 서로 바꿀 수 있는지를 한 번 생각해봅시다.



K. 맥스웰의 악마

- 그림에서 서로 바꿀 수 있는 양쪽 입자의 번호를 나열해볼까요?
- $(1,1), (2,2), (3,3), (4,4), (1,5), (2,6), (3,7), (4,8), \cdots$
- ...규칙성을 찾을 수 있을 것 같습니다!

K. 맥스웰의 악마

- 일반화 된 결론을 생각해보면 그림과 같이 넘버링 했을 때, $gcd(2L, 2R)$ 로 나눈 나머지가 같은 두 입자들 사이를 항상 바꿀 수 있습니다.
- 그런 위치들 사이에서는 항상 바꿀 수 있으므로 $gcd(2L, 2R)$ 로 나눈 나머지를 기준으로 입자를 분류한 뒤, 값이 큰 것부터 오른쪽이 수용할 수 있는 만큼 뽑아 더하면 됩니다.
- 주의할 점은 같은 위치의 입자들은 처음에 하나로 합쳐줘야 한다는 구현 상의 문제가 있습니다.

L. 선형대수학

`geometry, segtree`
출제진 의도 – **Hard**

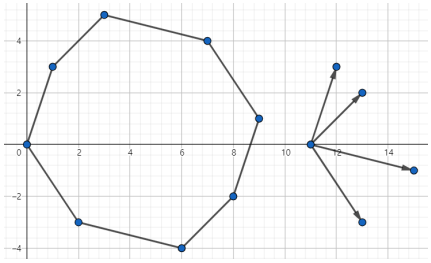
- 제출 1번, 정답 0팀 (정답률 0.000%)
- 처음 푼 팀: **???** (?, ?, ?), ?분
- 출제자: mhy908
- 가장 짧은 검수진 코드 : 3960bytes

- 문제를 재해석 해봅시다.
- 연립방정식의 해가 있다는 것은,
- 주어진 (P, Q) 벡터에 x를 곱한 뒤, 서로 더해서 (A, B)에 도달할 수 있는지와 동치입니다.
- 여기서 x는 $[0, 1]$ 내부의 실수입니다.

- ✓ 일단 편의를 위해 각 벡터의 x 성분이 0 또는 양수인 것만 생각해 봅시다.
- ✓ x 성분이 음수인 경우는 그만큼 전체 평면을 평행이동 시키고, 그 벡터의 역방향 벡터를 대신 추가하는 것으로 생각할 수 있기 때문입니다.

L. 선형대수학

- ✓ $[0, 1]$ 사이에 해가 존재하는 범위를 2차원 평면 상에 그려봅시다.



- ✓ 답이 YES인 (A, B)는 주어진 벡터들로 이루어진 볼록 도형 내부(경계포함)입니다.
- ✓ 이는 '민코프스키 합'에 대해 아시는 분들이라면 직관적으로 알 수 있고, 몰라도 몇번의 직접 그림을 그려보면 증명도 자연스럽게 될 거라 생각합니다.

- ✓ 결론적으로 지금 주어진 벡터들을 각도순으로 정렬하고, 순서대로 이어붙인 모양의 도형 내부에 어떤 점이 있는지 판별하면 문제가 풀립니다.
- ✓ 또, 벡터가 추가/삭제됨에 따라 볼록 도형을 dynamic하게 관리해야 합니다.

- 볼록 도형을 dynamic하게 관리하는 것은 다음과 같이 segment tree를 구현하면 됩니다:
- 먼저 벡터들을 각도순으로 정렬한 뒤, 순서대로 번호를 매깁니다.
- 벡터가 삽입될 때, 해당 위치의 segment tree 리프에 삽입합니다. 이때 segment tree는 x 성분과 y 성분 각각의 합을 관리합니다.
- 내부에 있는지 판별하는 부분은 세그먼트 트리 위에서 이분탐색을 통해 바로 아래/위에 있는 선분의 좌표를 구하면 됩니다.
- 시간복잡도는 $O((N+Q)lgN)$ 입니다.