SASA Programming Contest 2023 풀이

Thanks to - 대회를 만든 사람

- 운영자

  김경호 김민재 노준혁 박선호 성의현 이진명 윤의진
  이상민 이주윤 장민균 전하진 정시우 정희우 최영준 호준수

- 검수자

  kiwiyou lighter naeby qwerasdfzxcl rustiebeats
  shiftpsh swoon utilforever wjdclgns1217 yongwhan

Thanks to - 후원



주식회사 스타트링크 / 대회 플랫폼 제공



솔브드 / 대회 검수 및 상품 지원



넥슨재단 / 대회 상품 지원

Thanks to - 대회를 완성한 사람

- 참가자
  - **교내 대회** - **34**명
  - **Open Contest**
    Div. 1 - **289**명
    Div. 2 - **112**명
    Div. 1과 Div. 2를 모두 참가한 사람 - **48**명

문제		의도한 난이도	출제자
A/2A	관공... 어찌하여 목만 오셨소...	**Very Easy**	이상민
B/1A	방형구 탐색 (Easy)	**Easy**	최영준
C/1B	미역은 식물 아닌데요	**Easy**	정시우
D/2B	세과영엔 슬픈 전설이 있어	**Easy**	장민균
E/2C	SASA 마니또	**Medium**	최영준
F/1C	가위 가위 가위	**Medium**	정시우
G/2D	제목 추천 좀	**Medium**	이상민

문제		의도한 난이도	출제자
H/1D	고인물이싫어	**Hard**	정희우
I/2E	그래서 나는 코딩을 그만두었다	**Hard**	정시우
J/2F	토마에 함수	**Hard**	장민균
K/2G	조용히 완전히 영원히	**Hard**	정시우
L/1E	꼬치구이	**Hard**	이상민
M/1F	하이퍼 삼각형 자르기	**Very Hard**	장민균
N/1G	두근 어질	**Very Hard**	정시우
O/1H	방형구 탐색(Hard)	**Very Hard**	최영준

A/2A. 관공... 어찌하여 목만 오셨소...

`implementation, string`

출제진 의도 – **Very Easy**

- 본 대회 제출: 54번, 정답 23명 (정답률 51.852%)
- 본 대회에서 처음 푼 사람: **박범진**, 1분
- 오픈 대회 제출: 125번, 정답 111명 (정답률 88.800%)
- 오픈 대회에서 처음 푼 사람: **seawon0808**, 0분

A/2A. 관공... 어찌하여 목만 오셨소...

- 문제에서 제시한 대로 이름에 S가 들어가는 사람의 이름을 찾아 출력합니다.
- N 개의 문자열을 입력받을 때마다 하나의 문자씩 확인해 보면 되므로 시간복잡도 $\mathcal{O}(N)$에 문제를 해결할 수 있습니다.

B/1A. 방형구 탐색 (Easy)

`bruteforcing`
출제진 의도 – **Easy**

- 본 대회 제출: 34번, 정답 23명 (정답률 67.647%)
- 본 대회에서 처음 푼 사람: **박범진**, 5분
- 오픈 대회 제출: 350번, 정답 260명 (정답률 79.143%)
- 오픈 대회에서 처음 푼 사람: **riroan**, 1분

B/1A. 방형구 탐색 (Easy)

- 구간 안의 꽃을 확인하거나 없애는 두 가지 쿼리를 처리해야 합니다.
- Easy 문제는 단순히 한 칸씩 직접 꽃을 확인해 보거나 없애는 것으로 충분히 해결할 수 있습니다.
- 시간복잡도는 $\mathcal{O}(QN)$ 입니다.

C/1B. 미역은 식물 아닌데요

`implementation`
출제진 의도 – **Easy**

- 본 대회 제출: 52번, 정답 22명 (정답률 42.308%)
- 본 대회에서 처음 푼 사람: **김진형**, 6분
- 오픈 대회 제출: 406번, 정답 203명 (정답률 50.000%)
- 오픈 대회에서 처음 푼 사람: **moonrabbit2**, 4분

C/1B. 미역은 식물 아닌데요

- 실험 결과 광합성을 **하고** 운동성이 없는 생물은 **반드시** 식물입니다.
- 반대로 광합성을 하지 **않거나** 운동성이 있는 생물은 **반드시** 식물이 아닙니다.
- 식물의 최솟값은 **반드시** 식물인 생물의 수, 최댓값은 전체 생물 수에서 **반드시** 식물이 아닌 생물의 수를 뺀 값을 구하면 됩니다.
- 시간복잡도는 $\mathcal{O}(N+M)$ 입니다.

D/2B. 세과영엔 슬픈 전설이 있어

`sort`, `greedy`
출제진 의도 – **Easy**

- 본 대회 제출: 71번, 정답 13명 (정답률 18.310%)
- 본 대회에서 처음 푼 사람: **김진형**, 12분
- 오픈 대회 제출: 75번, 정답 23명 (정답률 41.333%)
- 오픈 대회에서 처음 푼 사람: **sehujeong**, 3분

D/2B. 세과영엔 슬픈 전설이 있어

- 세종이의 분노가 큰 날부터 순서대로 많은 돈이 든 자루를 배정하는 것이 최적입니다.
- 자루와 세종이의 분노를 크기 순으로 정렬에 각각 매칭해 주어 돈을 주는 것이 가능한 지 판정할 수 있습니다.
- 총 시간복잡도 $\mathcal{O}\left(N \log N\right)$에 해결할 수 있습니다.

E/2C. SASA 마니또

`case_work`
출제진 의도 – **Medium**

- 본 대회 제출: 76번, 정답 7명 (정답률 9.211%)
- 본 대회에서 처음 푼 사람: **이민석**, 37분
- 오픈 대회 제출: 60번, 정답 3명 (정답률 5.000%)
- 오픈 대회에서 처음 푼 사람: **wpqlks7**, 45분

E/2C. SASA 마니또

- 세종이의 번호 **s**에 대해 마니또가 이미 정해졌다면 NOJAM을 출력합니다.
- 세종이의 마니또가 정해지지 않은 상태라면 세종이 번호 **s**를 제외한 모든 학생들의 번호에 대해 세종이의 마니또가 될 수 있는지 조사합니다.
- 가능한 학생 수가 1이라면 NOJAM을 출력합니다.
- 가능한 학생 수가 2이고, 세종이가 이미 누구의 마니또일 때 한 학생이 세종이의 마니또가 된다면 혼자 남는 학생이 존재할 수 있습니다.
- 따라서 위의 상황에서 2명의 학생 중 마니또가 없으며 누구의 마니또도 아닌 학생이 존재할 경우 NOJAM을 출력합니다.
- 그 외의 경우에는 가능한 학생 수를 출력합니다.
- 시간복잡도는 $\mathcal{O}(N)$ 입니다.

F/1C. 가위 가위 가위

`constructive`

출제진 의도 – **Medium**

- 본 대회 제출: 25번, 정답 3명 (정답률 12.000%)
- 본 대회에서 처음 푼 사람: **곽정환**, 101분
- 오픈 대회 제출: 584번, 정답 88명 (정답률 15.240%)
- 오픈 대회에서 처음 푼 사람: **moonrabbit2**, 9분

F/1C. 가위 가위 가위

- 첫 손은 모두 가위로 결정되어 있으므로 실제로 손 모양을 골라서 대결할 수 있는 수는 최대 100회입니다.
- 한 번의 대결로 하나의 손 모양을 확정할 수 있어야 합니다.

F/1C. 가위 가위 가위

- 손 모양 하나를 가위에서 바위로 바꿔 대결을 진행해 봅시다.
- 머신의 손 모양에 따라 세 가지 결과가 나타납니다.
  - 머신이 가위를 낸 경우 세종이가 이긴 횟수는 1 증가합니다.
  - 머신이 바위를 낸 경우 세종이가 이긴 횟수는 그대로입니다.
  - 머신이 보를 낸 경우 세종이가 이긴 횟수는 1 감소합니다.
- 따라서 첫 대결과 다른 하나의 대결의 결과를 토대로 이기는 손 모양 하나를 **반드시** 알아낼 수 있습니다.

G/2D. 제목 추천 좀

`exponentiation_by_squaring`
출제진 의도 – **Medium**

- 본 대회 제출: 46번, 정답 3명 (정답률 6.522%)
- 본 대회에서 처음 푼 사람: **김진형**, 82분
- 오픈 대회 제출: 8번, 정답 5명 (정답률 62.500%)
- 오픈 대회에서 처음 푼 사람: **hsh8086**, 50분

G/2D. 제목 추천 좀

- 원형으로 앉아있는 경우를 그대로 다루는 것은 복잡합니다.
- 1 번 학생이 쓴 수를 고정하고 나머지 학생들이 선형으로 앉아 있는 것처럼 생각할 수 있습니다.
- a_n, b_n 을 아래와 같이 정의합시다.
  - a_n: n 명의 학생이 선형으로 앉아있을 때, 1 번 학생이 1 을 쓰고 나머지 학생이 이웃한 학생끼리 쓴 수의 합이 0 이 되지 않도록 하는 경우의 수
  - b_n: n 명의 학생이 선형으로 앉아있을 때, 1 번 학생과 n 번 학생이 1 을 쓰고 나머지 학생이 이웃한 학생끼리 쓴 수의 합이 0 이 되지 않도록 하는 경우의 수
- 이제 구해야 하는 답을 $4 \times (a_n - b_n)$ 으로 나타낼 수 있습니다.

G/2D. 제목 추천 좀

- a_{n+1} 을 구해 봅시다.
- $n+1$ 명이 앉아있는 경우는 n 번째 학생이 쓴 수가 1 인 경우와 아닌 경우로 나눌 수 있습니다.
- 먼저 n 번째 학생이 1 을 적은 경우 $n+1$ 번째 학생은 -1 을 제외한 나머지 세 수가 올 수 있으므로 가능한 경우의 수는 $3b_n$ 입니다.
- n 번째 학생이 적은 수가 1 이 아닌 경우 2 번째 수부터 n 번째 수까지 이전 수와의 합이 0 이 되지 않도록 하는 경우의 수는 3^{n-1} 이고, 여기서 n 번째 학생이 쓴 수는 1 이 아니어야 하므로 b_n 을 뺍니다.
- $n+1$ 번째 수에는 -1 과 n 번째 수의 덧셈의 역원을 제외한 2 개의 수가 올 수 있습니다.

G/2D. 제목 추천 좀

- 따라서 $a_{n+1} = 2 \times \left(3^{n-1} - b_n\right) + 3b_n = 2 \times 3^{n-1} + b_n$ 입니다.
- $b_{n+1} = a_n, a_1 = 1, a_2 = 3$ 이므로, 이로부터 최종적으로 다음의 답을 얻을 수 있습니다.
  - n 이 짝수인 경우 $a_n = \dfrac{3^n + 3}{4}$.
  - n 이 홀수인 경우 $a_n = \dfrac{3^n + 1}{4}$.
- 이제 분할정복을 통한 거듭제곱을 이용하여 답을 구할 수 있습니다.
- 시간복잡도는 $\mathcal{O}\left(\log N\right)$ 입니다.

H/1D. 고인물이싫어

`bfs`, `prefix_sum`
출제진 의도 – **Hard**

- 본 대회 제출: 35번, 정답 7명 (정답률 20.000%)
- 본 대회에서 처음 푼 사람: **이강규**, 82분
- 오픈 대회 제출: 310번, 정답 78명 (정답률 26.452%)
- 오픈 대회에서 처음 푼 사람: **ainta**, 12분

H/1D. 고인물이싫어

- 문제를 풀기 위해 두 가지 과정을 거쳐야 합니다.
  - 물이 고인 칸의 위치 확인
  - 발을 디딜 수 있는 방법들 각각에 대해 물이 고인 칸이 있는지 확인
- 물이 고인 칸의 위치는 하수구에서부터 BFS 등을 이용해 물이 빠진 칸을 찾아주는 것으로 처리할 수 있습니다.
- 물이 빠진 칸의 개수로 2차원 누적 합 배열을 만들면 가능한 모든 발을 딛는 경우의 수에 대해 그중 물이 고인 칸이 있는지 빠르게 판정할 수 있습니다.
- 따라서 총 시간복잡도 $\mathcal{O}(NM)$에 문제를 해결할 수 있습니다.

I/2E. 그래서 나는 코딩을 그만두었다

`binary_search`, `greedy`

출제진 의도 – **Hard**

- 본 대회 제출: 1번, 정답 0명 (정답률 0.000%)
- 본 대회에서 처음 푼 사람: —, XX분
- 오픈 대회 제출: 7번, 정답 0명 (정답률 0.000%)
- 오픈 대회에서 처음 푼 사람: —, XX분

I/2E. 그래서 나는 코딩을 그만두었다

- 일단 K 번의 휴식을 취하기로 했다고 하고 생각해 봅시다.
- 멘탈 수치를 높일 수 있는 방법이 있다면 최대한 빨리 사용하는 것이 유리합니다.
- 따라서 첫 K 일 동안 연속해서 쉬는 것이 항상 최적입니다.

I/2E. 그래서 나는 코딩을 그만두었다

- 이제 문제를 K일 동안 쉰 후 나머지 $N-K$일 동안 대결을 할 수 있는지 판단하는 문제로 바꿀 수 있습니다.
- 휴식을 많이 취할 수록 멘탈 수치는 높아지고 대결의 수는 줄어듭니다.
- 따라서 어떤 K에 대해 K일 동안 휴식을 취한 후 대결을 진행하는 것이 가능하다면, 그보다 큰 모든 K에 대해서도 역시 대결이 가능함을 알 수 있습니다.
- 즉 K에 대한 이분 탐색을 통해 $N-K$일 동안 대결이 가능해지는 K의 최솟값을 찾아낼 수 있습니다.

I/2E. 그래서 나는 코딩을 그만두었다

- 이제 $N - K$ 일 동안의 최적의 대결 순서를 생각해 봅시다.
- 대결에서 지는 경우 멘탈 수치가 감소하고, 이는 세종이의 퍼포먼스를 감소시킵니다.
- 따라서 세종이가 이기는 대결을 먼저 하는 것이 항상 유리합니다.
- 또한 세종이보다 실력이 높은 사람을 이기면 그 실력 차이만큼 멘탈 수치를 얻을 수 있으므로 세종이보다 실력이 높은 사람을 먼저 이기는 것이 유리합니다.
- 세종이가 이기기 위해서는 상대의 퍼포먼스가 자신보다 낮아야 하므로, 세종이보다 실력이 높은 사람을 퍼포먼스가 낮은 순으로 정렬해 그 순서대로 대결하는 것이 세종이가 이기는 대결을 최대한 많이 만들 수 있습니다.

I/2E. 그래서 나는 코딩을 그만두었다

- 이제 세종이보다 실력이 높은 사람 중 더 이상 이길 수 있는 사람이 없다고 해 봅시다.
- 그러면 더 이상 세종이의 멘탈 수치는 증가할 수 없음을 관찰할 수 있습니다.
- 즉 이때가 세종이의 퍼포먼스가 가장 높을 때이므로, 지금 세종이가 이기거나 비길 수 있는 사람들과의 대결을 모두 하는 것이 유리합니다.

I/2E. 그래서 나는 코딩을 그만두었다

- 세종이가 이기거나 비길 수 있는 상대가 더 남지 않았다고 합시다.
- 이제 세종이의 멘탈 수치는 대결을 진행할 때마다 감소합니다.
- 세종이보다 실력이 낮은 사람에게 지는 경우 멘탈 수치가 추가로 감소하기 때문에 멘탈 수치를 최대한 적게 깎이기 위해서는 실력이 높은 사람들과 먼저 대결하는 것이 유리합니다.
- 따라서 남은 대결 상대를 실력 기준 내림차순 정렬해 그 순서대로 대결하는 것이 최적입니다.
- 이를 통해 $\mathcal{O}\left(N \log N\right)$에 한 K에 대한 대결 가능성을 판단할 수 있고, 이분 탐색을 포함해 총 시간 복잡도 $\mathcal{O}\left(N \log^2 N\right)$에 문제를 해결할 수 있습니다.

J/2F. 토마에 함수

`euler_phi`

출제진 의도 – **Hard**

- 본 대회 제출: 44번, 정답 3명 (정답률 6.818%)
- 본 대회에서 처음 푼 사람: **전민성**, 56분
- 오픈 대회 제출: 9번, 정답 5명 (정답률 55.556%)
- 오픈 대회에서 처음 푼 사람: **nflight11**, 41분

J/2F. 토마에 함수

- 문제에서처럼 $x = \frac{p}{q}$ 라고 합시다. 그러면 $q \le \frac{Q}{P}$ 인 모든 양의 정수 q에 대해 $\frac{p}{q}$ 가 기약분수가 되도록 하는 순서쌍 (p, q)의 개수를 구하는 문제와 같습니다.
- 하나의 q에 대한 p의 개수는 오일러 피 함수의 정의와 동치입니다. 따라서 답은 오일러 피 함수의 값을 합해서 구할 수 있습니다.
- $\sqrt{\frac{Q}{P}}$ 까지의 소수를 구해 두면 피 함수를 한 번 실행하는 데 $\mathcal{O}\left(\sqrt{\frac{Q}{P}}\right)$ 정도 시간이 걸립니다.
- 총 시간 복잡도는 $\mathcal{O}\left(\left(\frac{Q}{P}\right)^{3/2}\right)$ 입니다.
- $x = 0$일 때 $f(x) = 1$임에 유의해 주세요.

K/2G. 조용히 완전히 영원히

`lazyprop`
출제진 의도 – **Hard**

- 본 대회 제출: 4번, 정답 0명 (정답률 0.000%)
- 본 대회에서 처음 푼 사람: —, XX분
- 오픈 대회 제출: 2번, 정답 1명 (정답률 50.000%)
- 오픈 대회에서 처음 푼 사람: **jh05013**, 99분

K/2G. 조용히 완전히 영원히

- 우선 세종이가 원래 풀고 있던 문제부터 풀어 봅시다.
- 구간에 $\min$ 연산을 적용하는 쿼리를 처리해야 합니다.
- 이 문제는 느리게 갱신되는 세그먼트 트리를 이용해 해결할 수 있습니다.

K/2G. 조용히 완전히 영원히

- 이제 마지막으로 업데이트시킨 쿼리의 번호를 같이 관리해 봅시다.
- 이는 구간에 $\max$ 연산을 적용하는 것과 비슷하게 처리할 수 있습니다.
- 느리게 갱신되는 세그먼트 트리를 하나 더 만들어서, 세종이의 문제에서 주어진 업데이트를 할 때마다 쿼리의 번호를 같이 전파해 줍시다.
- 총 시간복잡도는 $\mathcal{O}\left((N+Q)\log N\right)$ 입니다.
- 이외에도 누적 합을 이용해 그냥 세그먼트 트리로 해결하거나, 쿼리를 정렬해 분리 집합으로 해결하는 등의 풀이가 있습니다.

L/1E. 꼬치구이

`bipartite_matching`
출제진 의도 – **Hard**

- 본 대회 제출: 1번, 정답 0명 (정답률 0.000%)
- 본 대회에서 처음 푼 사람: —, XX분
- 오픈 대회 제출: 70번, 정답 23명 (정답률 32.857%)
- 오픈 대회에서 처음 푼 사람: **moonrabbit2**, 23분

L/1E. 꼬치구이

- 문제를 요약하면 조건을 만족하는 S와 G를 간선으로 이은 그래프에서의 최대 매칭을 구하는 문제가 됩니다. 여기서 플로우 문제임을 관찰한 후 그냥 호프크로프트-카프를 이용해 해결하는 것도 가능합니다.
- 그런데 S와 S, G와 G를 직접 잇는 간선이 없기 때문에, 이 문제에서 모델링되는 그래프는 S와 G에 대한 **이분 그래프**입니다.
- 따라서 단순 이분 매칭을 구현하는 것으로 시간 복잡도 $\mathcal{O}\left(N^2M^2\right)$에 문제를 해결할 수 있습니다.

M/1F. 하이퍼 삼각형 자르기

`combinatorics`

출제진 의도 – **Very Hard**

- 본 대회 제출: 1번, 정답 0명 (정답률 0.000%)
- 본 대회에서 처음 푼 사람: —, XX분
- 오픈 대회 제출: 48번, 정답 4명 (정답률 8.333%)
- 오픈 대회에서 처음 푼 사람: **gs22059**, 115분

M/1F. 하이퍼 삼각형 자르기

- 고차원의 하이퍼 삼각형을 문제 지문대로 그려서 잘라보는 것은 복잡합니다.
- 하이퍼 삼각형을 보다 다루기 쉬운 형태로 변형시켜 봅시다.
- 하이퍼 삼각형의 모서리 하나를 다른 모서리에 수직하도록 만드는 선형 변환을 생각해 봅시다.
- 그러면 그 변환을 벡터공간 $\mathbb{R}^M$ 전체를 정의역으로 갖는 전단사 사상으로 표현할 수 있습니다.
- 이는 변환 전후의 두 도형이 위상 동형임을 의미합니다.
- 이러한 변환을 여럿 결합해도 여전히 도형은 위상 동형입니다.

M/1F. 하이퍼 삼각형 자르기

- 따라서 하이퍼 삼각형을 적절히 비틀어서 어떤 한 점에서 만나는 M 개의 모서리가 각각 서로 수직하도록 만들어도 좋습니다. 이는 한 꼭짓점을 시점으로, 나머지 M 개의 꼭짓점을 종점으로 갖는 M 개의 벡터를 기저로 설정한 벡터공간을 생각하는 것과 같습니다. 이때 원래의 공간에서 하이퍼 삼각형의 형태를 변환한 형태도 편의 상 하이퍼 삼각형으로 부르겠습니다.
- 수직하는 M 개의 모서리가 만나는 점을 원점으로 잡고, 나머지 M 개의 꼭짓점을 각각 하나의 축 위에 올라가도록 합시다.
- 이제 하이퍼 삼각형을 잘라 얻은 조각의 꼭짓점은 모두 음이 아닌 정수 좌표를 갖습니다. 각 점의 좌표를 $(x_1, x_2, \ldots, x_M)$ 으로 둡시다.
- 이때 한 점의 좌푯값을 모두 더한 값은 항상 N 이하입니다. 이 조건을 만족하는 음이 아닌 정수 좌표는 **모두** 하나 이상의 조각의 꼭짓점입니다.

M/1F. 하이퍼 삼각형 자르기

- 방금 전 변환에 의해 M 개의 모서리가 서로 다른 M 개의 축 위에 올라갑니다.
- 이는 즉 모든 가능한 하이퍼 삼각형 조각의 $(M-1)$ 차원 면은 $x_i = t$ 또는 $\sum_i x_i = t$ 꼴로 나타낼 수 있는 면에 포함됨을 의미합니다.
- 따라서 어떤 하이퍼 삼각형에 대해 M 개의 모서리가 서로 수직으로 만나는 점을 지나는 모서리의 길이는 모두 같고 $(M-1)$ 차원 도형인 $\sum_i x_i = t$ 와 만나야 합니다.
- 이를 만족하기 위해서는 M 개의 모서리가 한 점으로부터 **모두** 양의 방향으로 뻗거나 **모두** 음의 방향으로 뻗어야 합니다.

M/1F. 하이퍼 삼각형 자르기

- 이제 문제를 처음 하이퍼 삼각형의 경계 또는 내부에 있는 모든 정수점들에 대해 음의 방향으로 뻗을 수 있는 거리와 양의 방향으로 뻗을 수 있는 거리의 총합을 계산하는 문제로 바꿀 수 있습니다.
- 한 정수점에서 음의 방향으로 뻗을 수 있는 거리는 각 점의 M 개 축에 대한 좌표 중 최솟값과 같습니다.
- 즉 음의 방향으로 K 만큼 뻗을 수 있으려면 모든 좌표가 K 이상이어야 합니다. 이는 모든 축 방향으로 K 만큼을 더하고 나머지 $N - MK$ 를 M 개의 서로 다른 차원에 나누어 주거나 일부는 나누어 주지 않는 경우의 수와 같습니다.

M/1F. 하이퍼 삼각형 자르기

- 수를 버리기 위해 하나의 축을 임시로 만들었다고 생각하면, 음의 방향으로 K 이상 뻗을 수 있는 정수점의 개수를 중복조합을 이용해 ${}_{M+1}\mathrm{H}_{N-M\times K}$ 로 나타낼 수 있습니다.
- 따라서 모든 음의 방향으로의 하이퍼 삼각형의 개수는 $\sum_{K=1}^{\lfloor N/M \rfloor} {}_{M+1}\mathrm{H}_{N-M\times K}$ 입니다.

M/1F. 하이퍼 삼각형 자르기

- 한 정수점에서 양의 방향으로 뻗을 수 있는 거리는 그 점의 M 개 축에 대한 좌푯값을 모두 더한 값을 S 로 두면 $N - S$ 와 같습니다. 왜냐하면 모서리를 1 만큼 뻗은 끝점의 좌푯값의 합은 원래의 점의 좌푯값의 합보다 반드시 딱 1 증가하기 때문입니다.
- 따라서 K 개의 양의 방향으로의 하이퍼 삼각형이 만들어지는 꼭짓점의 개수는 $N - K$ 를 M 개의 축에 나누어주는 방법의 수와 같으므로 중복조합을 이용해 ${}_M\mathrm{H}_{N-K}$ 로 표현할 수 있습니다.
- 따라서 모든 양의 방향으로의 하이퍼 삼각형의 개수는 $\sum_{K=1}^{N} {}_M\mathrm{H}_{N-K}$ 입니다.

M/1F. 하이퍼 삼각형 자르기

- 팩토리얼 값과 그 모듈로 곱셈 역원의 값을 전처리해 두면 각 연산은 $\mathcal{O}(1)$에 처리할 수 있습니다.
- 따라서 총 시간복잡도는 $\mathcal{O}(N+M)$ 입니다.

N/1G. 두근 어질

`game_theory`, `smaller_to_larger`
출제진 의도 – **Very Hard**

- 본 대회 제출: 1번, 정답 0명 (정답률 0.000%)
- 본 대회에서 처음 푼 사람: —, XX분
- 오픈 대회 제출: 83번, 정답 14명 (정답률 16.867%)
- 오픈 대회에서 처음 푼 사람: **kyo20111**, 71분

N/1G. 두근 어질

- 두 사람의 목표는 마지막 날의 게임에서 이기는 것입니다.
- 마지막 날에는 꽃이 있는 꽃집이 하나밖에 남지 않기 때문에 선공이 항상 이깁니다.
- 즉 마지막 게임을 이기려면 그 전 게임에서 반드시 승리해야 합니다.

N/1G. 두근 어질

- 이를 확장해 마지막 날부터 거꾸로 확인하면서, 마지막 게임에서 이기기 위해서 i 번째 날에 이겨야 할지 혹은 져야 할지를 판단할 수 있습니다.
- 각 날의 게임은 이겨야 하는지 져야 하는지에 따라 각각 님 게임 또는 Misère 님 게임으로 볼 수 있습니다.
- 두 종류의 게임은 스프라그-그런디 정리를 통해 선후공 중 누가 필승인지 알아낼 수 있습니다.

N/1G. 두근 어질

- 꽃집의 꽃을 합치는 과정에서 각 꽃집의 꽃의 종류를 모두 고려해 주어야 합니다.
- 이를 빠르게 수행하기 위해 작은 집합에서 큰 집합으로 합치는 테크닉을 사용할 수 있습니다.
- 시간복잡도는 $\mathcal{O}\left(N \log^2 N\right)$ 입니다.
- 그런데...

N/1G. 두근 어질

- **이 문제를 $\mathcal{O}(N)$에 풀 수 있습니다!**
- 각 꽃집에 있는 꽃의 종류의 수를 각 돌무더기에 있는 돌의 개수로 바꿔 생각합시다.
- 모든 꽃의 종류가 같은 경우 합치는 과정을 생략할 수 있습니다. 따라서 $\mathcal{O}(N)$에 문제를 풀 수 있습니다.
- 이제 종류가 다른 꽃이 있는 경우를 생각해 봅시다.

N/1G. 두근 어질

- 가장 처음 상황에서 돌무더기의 돌의 수는 모두 1이고 이때는 세종이와 아름이에게 가능한 선택의 수가 한 가지 뿐이므로 직접 계산을 통해 승패를 알 수 있습니다.
- 그러다 어느 순간 종류가 다른 꽃이 있는 꽃집끼리 합쳐지면, 돌무더기의 돌의 수는 하나의 2와 여러 개의 1로 구성됩니다.
- 돌이 2개 이상 있는 돌무더기가 존재하는 경우, 그런디 넘버가 0이 아니라면 님 게임과 Misère 님 게임 모두 선공이 이길 방법이 반드시 존재하고, 반대로 그런디 넘버가 0이면 후공이 이길 방법이 반드시 존재합니다.
- 그런데 처음으로 돌이 2개 있는 돌무더기가 만들어지는 상황에서의 그런디 넘버는 항상 2 이상이므로 선공이 마지막 꽃을 가져갈 지 선택할 수 있습니다.
- 따라서 이후의 게임의 결과를 **이때의 선공 플레이어가 마음대로 정할 수 있습니다.**

N/1G. 두근 어질

- 결론적으로 처음 다른 종류의 꽃을 가진 꽃집끼리 합쳐질 때까지 승패 판정만 해주면 합치는 과정 없이 마지막 게임의 승자를 알 수 있습니다.
- 돌무더기에 돌이 1개씩 밖에 없는 게임은 $\mathcal{O}(1)$에 승패를 판정할 수 있으므로 총 시간복잡도는 $\mathcal{O}(N)$입니다.

O/1H. 방형구 탐색 (Hard)

`data_structure, segtree`

출제진 의도 – **Very Hard**

- 본 대회 제출: 4번, 정답 0명 (정답률 0.000%)
- 본 대회에서 처음 푼 사람: —, XX분
- 오픈 대회 제출: 148번, 정답 25명 (정답률 16.892%)
- 오픈 대회에서 처음 푼 사람: **xiaowuc1**, 29분

O/1H. 방형구 탐색 (Hard)

- 문제에서 입력되는 꽃의 종류 배열을 편의상 A라고 하겠습니다.
- A를 꽃의 종류에 대해 오름차순으로 정렬합니다.
- 이때 꽃의 종류가 같다면 꽃이 위치한 인덱스에 대해 오름차순으로 정렬합니다.
- 이를 통해 A에서 같은 종류의 꽃끼리 붙어있게 만들 수 있습니다.
- 정렬된 전과 후의 꽃들의 인덱스 변화를 배열을 통해 저장합니다.
- 정렬 후 A에 대해 밟히지 않은 꽃의 갯수로 세그먼트 트리를 구축할 수 있습니다.

1번 쿼리

- 꽃의 종류에 따라 정렬 전 A에 그 종류를 가진 꽃들의 인덱스들을 set을 통해 관리합니다.
- 꽃의 종류가 k인 꽃들의 인덱스들을 저장한 set에서 이분 탐색을 사용할 수 있습니다. 이를 통해 정렬 전 A의 배열에서 l번 칸과 r번 칸 사이에 있는 종류가 k인 꽃의 인덱스들 중 최소와 최대를 $\mathcal{O}(\log N)$으로 구할 수 있습니다. 이를 s와 e라고 부르겠습니다.
- 정렬 후 A에서 s와 e에 대응되는 인덱스에 대해 밟히지 않은 꽃들을 세그먼트 트리로 구한다면 1번 쿼리를 $\mathcal{O}(\log N)$에 해결할 수 있습니다.

2번 쿼리

- 밟히지 않은 꽃들의 정렬 전 A에서의 인덱스들을 set을 통해 관리합니다.
- 밟히지 않은 꽃들의 인덱스들을 저장한 set에서 이분 탐색을 사용할 수 있습니다. 이를 통해 정렬 전 배열 A에서 l 이상 r 이하 인덱스를 가진 꽃들 중 밟히지 않은 꽃들을 모두 알 수 있습니다. 이때 시간복잡도는 $\mathcal{O}(\log N)$ 입니다.
- 해당하는 꽃들을 세그먼트 트리에서 값을 업데이트합니다. 이때 시간복잡도는 $\mathcal{O}(\log N)$ 입니다.
- 이후 해당하는 꽃들을 모두 set에서 삭제합니다. N 개의 인덱스들은 모두 최대 한 번씩만 삭제 및 업데이트되므로, 전체 시간복잡도는 $\mathcal{O}(Q \log N + N \log N)$ 입니다.

O/1H. 방형구 탐색 (Hard)

- 이외에도 제곱근 분할법이나 PBDS를 이용해서도 문제를 해결할 수 있습니다.
- 제곱근 분할법을 이용하는 경우 버킷의 크기를 최적화하는 것이 좋습니다.