2024 ICPC Sinchon SUAPC Winter Solution

신촌지역 대학생 프로그래밍 동아리 연합

2024년 2월 17일

문제		의도한 난이도	출제자
A	가상 검증 기술	Easy	유상혁 golazcc83
B	가장 짧은 높이	Hard	현지훈 qawbecrdtey
C	스펀지	Easy	이지훈 maker29
D	배열 제작의 달인	Challenging	김규진 snrnsidy
E	문자열-그래프 매칭	Medium	유호영 tkfkddl59323
F	호떡 뒤집기	Hard	임예준 edward3378
G	AND, OR, XOR 2	Medium	장래오 leo020630
H	신촌 통폐합 계획	Easy	박진한 jinhan814
I	최대공약수 게임	Hard	유상혁 golazcc83
J	토지 판매	Challenging	장래오 leo020630
K	신촌방위본부: 지하벙커의 비밀	Challenging	김도훈 dohoon
L	신촌 도로망 관리와 쿼리	Medium	노현제 hjroh0315
M	엘리스 트랙 매칭	Easy	권순호 tnsgh9603

A. 가상 검증 기술

`bruteforcing`

출제진 의도 – **Easy**

- ✓ 제출 92번, 정답 40팀 (정답률 43.478%)
- ✓ 처음 푼 팀: **AKARAKA**[연세대학교], 2분
- ✓ 출제자: 유상혁[golazcc83]

A. 가상 검증 기술

- ✓ 차종 B에 대한 가상 검증 수행은 도훈이만 할 수 있기 때문에 $T_B \times V_B$의 시간이 확정적으로 소요됩니다.
- ✓ 차종 A에 대한 가상 검증 항목을 도훈이가 X개, 상혁이가 $V_A - X$개 맡는다면 가상 검증 업무를 완료하는 데 소요되는 시간은 $max(X \times T_A + T_B \times V_B, (V_A - X) \times T_A)$ 입니다.
- ✓ V_A의 값이 크지 않기 때문에 $0 \leq X \leq V_A$의 모든 X에 대해 소요되는 시간을 구하고, 그 중 최솟값을 출력하면 됩니다.
- ✓ 총 시간 복잡도는 $\mathcal{O}(Q \times V_A)$ 입니다.

B. 가장 짧은 높이

`geometry`

출제진 의도 – **Hard**

- ✓ 제출 39번, 정답 1팀 (정답률 2.564%)
- ✓ 처음 푼 팀: **AKARAKA**[연세대학교], 166분
- ✓ 출제자: 현지훈[qawbecrdtey]

B. 가장 짧은 높이

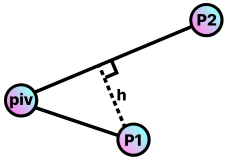
- ✓ $\binom{N}{3}$ 개 삼각형을 모두 확인하는 당연한 풀이는 시간복잡도가 $\mathcal{O}\left(N^3\right)$ 입니다.
- ✓ 불도저 트릭을 떠올렸다면, 일반적인 구현에 대해 공간 복잡도가 $\mathcal{O}\left(N^2\right)$ 입니다.
- ✓ 후보가 되는 높이를 가지는 삼각형만 효율적으로 확인할 방법을 찾아야 합니다.

B. 가장 짧은 높이

Algorithm.

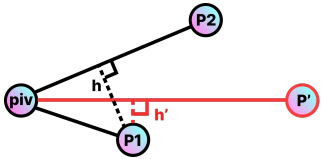
1. 한 정점 piv에 대해 그 정점을 제외한 $N-1$개의 점을 (반)시계방향으로 정렬합니다.
2. 이때 piv와 위의 정렬에서 이웃한 두 점으로 만들어지는 $N-1$개의 삼각형에서 높이의 최솟값을 찾습니다.
3. N개의 piv에 대해 1~2를 거쳐 나온 N개의 값 중 최솟값을 구하면 됩니다.

✓ 이를 통해 $\mathcal{O}\left(N^2 \log N\right)$ 시간 및 $\mathcal{O}(N)$ 공간 복잡도로 문제를 풀 수 있습니다.



B. 가장 짧은 높이

- ✓ 이 알고리즘으로 높이의 최솟값 h 를 찾지 못한다고 가정합시다.
- ✓ 모든 삼각형은 예각을 가지고 있으므로, $\angle piv$ 가 예각이 되도록 하는 piv 를 항상 잡을 수 있습니다.
- ✓ 알고리즘을 통해 해당 삼각형을 찾지 못했으므로, p_1 과 p_2 사이에 점 p' 이 반드시 존재합니다.
- ✓ 더 짧은 높이 h' 를 찾게 되었으므로, 귀류법에 의해 알고리즘이 올바름을 알 수 있습니다.



C. 스펀지

`math, constructive`

출제진 의도 – **Easy**

- ✓ 제출 105번, 정답 29팀 (정답률 27.619%)
- ✓ 처음 푼 팀: **AKARAKA**[연세대학교], 7분
- ✓ 출제자: 이지훈[maker29]

C. 스펀지

- ✓ 각 바이러스는 T초 후 (위쪽 or 아래쪽), (왼쪽 or 오른쪽) 방향으로 최대 T칸 이동 가능합니다.
- ✓ 바이러스는 스펀지 밖으로 나갈 수 없으니 T초 후에도 스펀지의 범위 내에 있어야 합니다.
- ✓ 각 바이러스의 이동 범위를 (X, Y)로 표현하면 $max(1, x_i - T) \leq X \leq min(x_i + T, W)$, $max(1, y_i - T) \leq Y \leq min(x_i + T, H)$ 이 됩니다.
- ✓ 각 바이러스가 이동할 수 있는 경우의 수는 (X, Y) 범위 직사각형의 면적이며, 이는 $(min(x_i + T, W) - max(1, x_i - T) + 1) \times (min(y_i + T, H) - max(1, y_i - T) + 1)$ 입니다.
- ✓ 각 바이러스는 모두 독립적으로 이동 가능하므로, 이들의 경우의 수를 모두 곱한 것을 mod로 나눈 값이 정답이 됩니다.

D. 배열 제작의 달인

`fft`, `dp`, `combinatorics`

출제진 의도 – **Challenging**

- ✓ 제출 7번, 정답 0팀 (정답률 0.000%)
- ✓ 처음 푼 팀: -
- ✓ 출제자: 김규진 `snrnsidy`

D. 배열 제작의 달인

- ✓ 먼저, 각 수가 앞으로 더 등장할 수 있는 갯수를 배열 cnt에 전처리 해둡니다.
- ✓ 즉, 문제 조건에 따라 $1 \leq i \leq N$에 대해 $cnt[i] = i$로 초기화합니다.
- ✓ 그 후, 입력으로 주어진 배열에 대해, 해당 배열의 원소 x가 0이 아닌 경우는 $cnt[x]$를 하나 빼줍니다.
- ✓ 만약, 해당 연산 이후에 $cnt[x] < 0$이 된다면 주어진 배열 자체가 조건에 만족할 수 없는 배열이기 때문에 0을 출력하고 종료합니다.
- ✓ 해당 배열의 원소 x가 0인 경우는 변수 M을 사용하여 세줍니다.

D. 배열 제작의 달인

- ✓ 이제 답이 될 수 있는 경우의 수의 형태를 생각해봅시다. M개 중에서, 조건에 위배되지 않도록 수를 더 뽑는 것으로 생각할 수 있습니다.
- ✓ 여기서, $1 \leq x \leq N$에 대해 x를 더 뽑은 횟수를 저장한 배열을 A라고 합시다.
- ✓ 그러면, 경우의 수는 $C(M, A[1]) * C(M - A[1], A[2]) * ... * (M - A[1] - A[2].. - A[N-1], A[N])$로 정리할 수 있습니다.
- ✓ 여기서 $C(a, b)$는 $(a)!/((a-b)!b!)$를 의미하고, $A[1] + ... + A[N] = M$를 만족합니다.
- ✓ 따라서, 식을 잘 정리하면 $M!/(A[1]!A[2]!...A[N]!)$이 되는 것을 볼 수 있습니다.

D. 배열 제작의 달인

- ✓ 그러므로, 나올 수 있는 배열 A 전부에 대해, 저 식을 계산하여 전부 합한 값이 정답이 되는 것을 알 수 있습니다.
- ✓ 이 때, 이를 효율적으로 구하기 위해 dp 를 사용합니다.
- ✓ 즉, $dp[i][j]$ 를 i 번째 수까지 봤고, j 개의 0 를 다른 수로 바꿨을 때 나올 수 있는 경우의 수로 정의합니다.
- ✓ 그러면, 위에서 정리한 내용에 따라 $dp[i][j]$ 는 $(0 \le k \le cnt[i])$ 에 대해 $(dp[i-1][j-k]/k!)$ 의 합으로 점화식을 표현할 수 있습니다.
- ✓ 이를 단순히 반복문으로 구하게 되는 경우 시간복잡도가 $O(N^3)$ 이기 때문에 시간 초과를 받게 됩니다.

D. 배열 제작의 달인

- ✓ 따라서, 이를 효율적으로 할 필요가 있는데 이를 위해 fft를 사용합니다.
- ✓ 즉, $C = 1/0!, 1/1!, ..., 1/cnt[i]!$와 $dp[i-1]$ 두 배열에 대해 fft를 수행하여 나온 배열을 $dp[i]$로 갱신해주면 됩니다.
- ✓ 따라서, 초기 $dp[0][0]$의 값을 $M!$로 초기화하고 위의 과정을 N번 반복해서 얻은 $dp[N][M]$가 최종적인 정답이 됩니다.

D. 배열 제작의 달인

- ✓ 참고자료: https://atcoder.github.io/ac-library/production/document_en/convolution.html
- ✓ 추천문제: https://www.acmicpc.net/problem/13184

E. 문자열-그래프 매칭

two_pointer, sweeping

출제진 의도 – **Medium**

- ✓ 제출 12번, 정답 9팀 (정답률 75.000%)
- ✓ 처음 푼 팀: **입대 전 라스트댄스**연세대학교, 24분
- ✓ 출제자: 유호영tkfkddl59323

E. 문자열-그래프 매칭

- ✓ 여기서 정점 집합은 알파벳 소문자 집합으로 고정되어 있으므로, 임의의 두 그래프가 같음은 간선의 집합이 같음을 의미합니다.
- ✓ 모든 부분 문자열에 대해 동일한 집합을 가지는 케이스가 몇 개 있는지 확인해야 하므로, 투 포인터를 활용하는 스위핑 풀이를 생각해볼 수 있습니다.
- ✓ 스위핑 전에, 각 간선별로 스위핑 중 등장한 횟수를 기록해두는 배열 $cnts$를 만든다고 합시다.
- ✓ 두 개의 포인터를 각각 l, r이라고 하고, 둘 다 0번 인덱스에서 시작합니다.

E. 문자열-그래프 매칭

- ✓ 먼저 목표로 하는 집합이 완성될 때까지 r 포인터를 오른쪽으로 한 칸씩 움직이면서 $cnts$ 배열에 반영해줍니다. 이때 추가되는 간선 e 에 대해 $cnts[e]$ 가 0 에서 1 로 변하는 때마다 그 개수를 $edge\_size$ 같은 변수에 체크해 줍니다. 단, e 가 목표하는 집합에 없는 간선이면 무조건 멈춥니다.
- ✓ $edge\_size$ 변수가 목표로 하는 집합의 간선 개수와 같아지면 일단 r 포인터를 멈춥니다. 현재의 l 포인터를 기준으로 했을 때 r 포인터까지의 부분 문자열이 가장 짧은 우리가 원하는 부분 문자열임을 알 수 있습니다.

E. 문자열-그래프 매칭

- ✓ 이후 있으면 안되는 간선 전까지는 전부 세어야 하는 부분 문자열이므로, 추가로 rr 등의 포인터를 만들어 그러한 간선이 나올때까지 계속 오른쪽으로 한 칸씩 움직입니다. 이 때는 $cnts$ 배열에 반영하지 않습니다.
- ✓ 사실 굳이 포인터를 만들지 않고, 현재 r 포인터 보다 오른쪽에 있는 가장 왼쪽에 있는 존재하지 않는 간선의 위치를 찾기만 하면 되니, 다양한 방법을 사용할 수 있습니다.
- ✓ 존재하면 안 되는 간선이 나오면 rr 포인터를 멈추고 l 포인터에 대한 부분 문자열 개수를 답에 반영해줍니다.

E. 문자열-그래프 매칭

- ✓ 이후 l 포인터를 오른쪽으로 한 칸 옮기고, 그에 따라 원래 있던 간선이 하나 제거되므로 그걸 $cnts$ 배열에 반영해줍니다.
- ✓ 그에 맞게 다시 r 포인터를 오른쪽으로 움직이며 $cnts$ 배열에 반영해주며, rr 포인터가 r 포인터보다 오른쪽에 있는 동안은 그대로 답에 반영해줍니다.
- ✓ 만약 r 포인터가 rr 포인터를 추월하게 되면 l 포인터를 rr 포인터 바로 뒤까지 계속해서 움직이면서 $cnts$ 배열에서 하나씩 줄여줍니다.
- ✓ 이후 r 포인터를 l에 위치로 옮겨놓은 후 다시 처음부터 반복하면 됩니다.
- ✓ 이 순서로 스위핑을 완료하면 모든 부분 문자열의 개수를 셀 수 있습니다. 물론 각각의 포인터는 N번 이상 움직이지 않으므로 $O(N)$의 시간복잡도를 가집니다.

F. 호떡 뒤집기

`constructive, ad-hoc, case-work`

출제진 의도 – **Hard**

- ✓ 제출 64번, 정답 9팀 (정답률 14.062%)
- ✓ 처음 푼 팀: **AKARAKA**[연세대학교], 39분
- ✓ 출제자: 임예준[edward3378]

F. 호떡 뒤집기

- ✓ $N \leq 5 \times 10^5$ 이므로 모든 경우를 백트래킹으로 시도해보는 $\mathcal{O}(N!)$ 의 풀이는 너무 느립니다.
- ✓ 따라서, 일일이 모든 경우의 수의 작업을 수행하지 않고 원하는 색깔의 배열을 만드는 방법을 찾아야 합니다.

F. 호떡 뒤집기

- ✓ $N-1$ 개의 인접한 두 호떡 쌍에 대해 서로 다른 색깔의 인접한 두 호떡 쌍의 개수 D 를 각 작업마다 주목해봅시다.
- ✓ 먼저, 모든 호떡을 뒤집어도 D 는 변하지 않고 일정하다는 사실을 쉽게 관찰할 수 있습니다.
- ✓ 만약, 한 작업에서 x 번째와 $x+1$ 번째의 호떡 색깔이 같은 x 를 골랐다고 생각해봅시다.
- ✓ D 의 값은 작업하기 전에 비해 x 번째와 $x+1$ 번째의 호떡 색깔이 달라지는 횟수가 하나 추가되므로, 1 증가합니다.
- ✓ 따라서, i 번째 작업이 끝난 후 D 의 값은 i 가 된다는 것을 알 수 있습니다.

F. 호떡 뒤집기

- ✓ 편의상, 새로운 배열 T를 다음과 같이 정의하겠습니다.
- ✓ $T: S_i \neq S_{i+1}$ 인 모든 i의 값을 오름차순 정렬한 배열
- ✓ 만약 현준이가 원하는 배열을 만들 수 있다면, 이전 페이지에 따라 작업 횟수 K는 $|T|(\leq N-1)$라는 것을 알 수 있습니다.
- ✓ 또한, T의 원소 중 하나라도 작업에서 x로 사용하지 않는다면 원하는 배열을 만들 수 없습니다.
- ✓ 따라서, T를 적절히 재배열해서 작업 순서를 정해야 합니다.

F. 호떡 뒤집기

- ✓ 이제 $|T|$의 값에 따라 문제를 해결해봅시다.
- ✓ $|T| \leq 1$일 때는 모든 경우에 대해 조건 분기를 통해 답을 구합니다.
- ✓ $|T| \geq 2$일 때를 봅시다.
- ✓ T를 아무렇게나 재배열하여 수행할 경우, 원하는 배열의 모든 색깔이 뒤집힐 수 있으므로 마지막 원소인 S_N과 작업을 모두 수행했을 때 마지막 호떡 색깔을 같은 색으로 맞춥시다.

F. 호떡 뒤집기

- ✓ 먼저 순차적으로 $T_1, T_2, ..., T_{|T|-2}$의 값을 x로 하는 작업을 수행해봅시다.
- ✓ 여러 번 작업을 해도 맨 마지막인 N번째 호떡의 색깔은 처음 'W'로 변하지 않습니다.
- ✓ 만약 S_N이 'W'일 때, 추가로 $T_{|T|-1}, T_{|T|}$의 값을 x로 하는 작업을 수행합니다. 이 경우도 마찬가지로 N번째 호떡의 색깔이 'W'로 유지됩니다.
- ✓ 반대로 S_N이 'B'일 때, 순서만 바꿔서 추가로 $T_{|T|}, T_{|T|-1}$의 값을 x로 하는 작업을 수행합니다. 이때 N번째 호떡의 색깔이 'B'로 바뀌게 됩니다.
- ✓ 따라서 작업이 모두 끝난 후 N번째 호떡의 색깔과 S_N은 같으므로, 원하는 배열을 만들 수 있다는 사실을 알 수 있습니다.

F. 호떡 뒤집기

✓ 위 풀이를 코드로 작성하면 $\mathcal{O}(N)$ 의 시간복잡도로 이 문제를 해결할 수 있습니다.

G. AND, OR, XOR 2

`math`, `bitmask`

출제진 의도 – **Medium**

- ✓ 제출 26번, 정답 10팀 (정답률 38.462%)
- ✓ 처음 푼 팀: **Raa_FanClub**서강대학교, 42분
- ✓ 출제자: 장래오`leo020630`

G. AND, OR, XOR 2

- ✓ 우선, 각 원소를 2진법으로 나타낸 뒤 비트별로 생각해 봅시다.
- ✓ 한 비트에 대해서만 생각한다면 0과 1만으로 이루어진 수열을 얻을 수 있습니다.
- ✓ 이 수열에서의 답을 구할 수 있다고 가정하고, i번째 비트의 수열에서 구한 답을 $f(i)$라 합시다.
- ✓ 전체 문제의 정답은 $\sum_{i=0}^{29} 2^i f(i)$ 입니다.
- ✓ 이제 각 경우에 대해 0과 1만으로 이루어진 수열에서 문제를 해결하는 법을 알아봅시다.

G. AND, OR, XOR 2

1. AND

- ✓ 두 인덱스 $1 \leq x \leq y \leq N$에 대해, $(A_x \wedge A_{x+1} \wedge \cdots \wedge A_y)$가 1이려면 $A_x, A_{x+1}, \cdots, A_y$가 모두 1이어야 합니다.
- ✓ 따라서 1만으로 이루어진 연속된 부분수열들의 길이를 구한 뒤, 이를 $B_1, \cdots, B_k$라 합시다.
- ✓ $\sum_{i=1}^{k} \binom{B_i+1}{2}$가 문제의 답이 됩니다.

G. AND, OR, XOR 2

2. OR

- ✓ AND와 같은 방법으로 생각합시다.
- ✓ 두 인덱스 $1 \le x \le y \le N$에 대해, $(A_x \vee A_{x+1} \vee \cdots \vee A_y)$가 0이려면 $A_x, A_{x+1}, \cdots, A_y$가 모두 0이어야 합니다.
- ✓ 따라서 0만으로 이루어진 연속된 부분수열들의 길이를 구한 뒤, 이를 $C_1, \cdots, C_k$라 합시다.
- ✓ 해당 부분수열에 포함되는 경우가 아니라면 OR 연산한 값은 항상 1이 됩니다.
- ✓ 따라서 $\binom{N+1}{2} - \sum_{i=1}^{k} \binom{C_i+1}{2}$가 문제의 답이 됩니다.

G. AND, OR, XOR 2

3. XOR

✓ SUAPC 2022 Summer에 출제된 AND, OR, XOR 문제와 다르게 이번 문제는 XOR의 경우가 가장 어렵습니다.

✓ 수열의 누적 XOR 값을 저장한 배열 S를 $S_0 = 0, S_i = S_{i-1} \oplus A_i$로 정의합시다.

✓ XOR 연산은 자기 자신을 역원으로 가지기 때문에, 두 인덱스 $1 \leq x \leq y \leq N$에 대해 $(A_x \oplus A_{x+1} \oplus \cdots \oplus A_y) = S_y \oplus S_{x-1}$이 성립합니다.

G. AND, OR, XOR 2

3. XOR

✓ 따라서 모든 $1 \le j \le N$에 대해, $0 \le i < j, S_i \oplus S_j = 1$를 만족하는 i의 개수 합이 답이 됩니다.

✓ 두 배열 Z_i, O_i를 관리합시다. $Z_i = 0 \le j \le i, S_j = 0$인 j의 개수, $O_i = 0 \le j \le i, S_j = 1$인 j의 개수입니다.

✓ $\sum_{i=1}^{N} \begin{cases} O_{i-1}, & \text{if } S_i = 0 \\ Z_{i-1}, & \text{if } S_i = 1 \end{cases}$ 가 답이 됩니다.

✓ 최종 시간 복잡도는 $\mathcal{O}(N \log A)$ 입니다. A는 A_i의 최댓값입니다.

H. 신촌 통폐합 계획

`dfs`, `linked-list`

출제진 의도 – **Easy**

- ✓ 제출 263번, 정답 25팀 (정답률 9.506%)
- ✓ 처음 푼 팀: **AKARAKA**연세대학교, 13분
- ✓ 출제자: 박진한`jinhan814`

H. 신촌 통폐합 계획

- ✓ dfs, linked-list를 이용한 $\mathcal{O}(N + \sum |s_i|)$ 풀이와 small to large를 이용한 $\mathcal{O}(\sum |s_i| logN)$ 풀이 등 여러 풀이가 존재하는 문제입니다.
- ✓ 여기선 dfs를 이용한 풀이를 알아보겠습니다.

H. 신촌 통폐합 계획

- ✓ 다음의 명제가 성립합니다.
- ✓ 1. i, j가 주어질 때마다 $i \to j$ 간선을 추가해 그래프를 구성하면 만들어지는 그래프는 트리입니다.
- ✓ Proof. i, j를 고를 때 s_i, s_j가 빈 문자열이 아니라는 조건에 의해 j로 들어오는 간선은 많아야 하나입니다. 따라서 입력되는 j는 중복되지 않고, $N-1$개의 j에 대응되는 i가 정확히 하나씩 존재합니다. j에서 i로 이동하는 과정을 반복하면 사이클에 빠지거나 더 이상 대응되는 i가 없는 정점에 도달할텐데, 전자를 가정하면 모순이 생깁니다. 따라서 만들어지는 그래프는 사이클이 없는 연결 그래프이니 트리입니다.

H. 신촌 통폐합 계획

- ✓ 2. 간선이 추가된 순서대로 정점을 방문하는 dfs로 x번 정점에서부터 트리를 순회하며 문자열을 이어붙인 결과를 $f(x)$라 정의할 때, 문제의 정답은 마지막으로 주어진 i에 대한 $f(i)$입니다.
- ✓ Proof. 정점의 깊이에 대한 수학적 귀납법을 이용합니다. j가 트리의 리프 노드인 경우는 $f(j) = s_j$이고, j가 트리의 리프 노드가 아니면서 자식 노드에 대한 명제가 성립한다면 자식 노드를 문자열을 이어 붙이는 순서대로 방문하니 마찬가지로 명제가 성립함을 보일 수 있습니다.
- ✓ 따라서 그래프를 구성한 뒤, dfs를 이용해 마지막으로 주어진 i에 대한 $f(i)$를 구해주면 $\mathcal{O}(N + \sum |s_i|)$에 문제를 해결할 수 있습니다.

I. 최대공약수 게임

`number-theory`, `game-theory`, `dp`

출제진 의도 – **Hard**

- ✓ 제출 27번, 정답 0팀 (정답률 0.000%)
- ✓ 처음 푼 팀: -
- ✓ 출제자: 유상혁 `golazcc83`

I. 최대공약수 게임

- ✓ 칠판에 처음 적은 수가 X일 때, 게임을 진행하면서 칠판에 적을 수 있는 수는 X의 약수입니다.
- ✓ $X \leq 10^9$일 때 가능한 약수의 개수는 최대 $1\,344$개 입니다.
- ✓ 따라서 게임이 진행되는 동안 칠판에 적힐 수 있는 최대 $1\,344$개의 정수에 대해서만 고려해도 됩니다.

I. 최대공약수 게임

- ✓ 현재 칠판에 적힌 수가 G일 때, 지금까지 게임을 진행하면서 선택한 모든 카드들은 G의 배수입니다.
- ✓ G의 배수가 아닌 카드는 게임이 진행되는 동안 선택되지 않은 카드이고, 선택하면 칠판에 적힌 수가 바뀝니다.
- ✓ G의 배수인 카드는 게임이 진행된 턴의 수만큼 소모되었습니다. 이 카드는 선택해도 칠판에 적힌 수가 바뀌지 않기 때문에, 게임이 진행되는 동안 선택한 카드가 정확하게 어떤 카드인지 관리할 필요가 없습니다.
- ✓ 위의 사실로 현재 칠판에 적힌 수와 게임이 진행된 턴의 수만으로 모든 게임판의 상태를 표현할 수 있음을 관찰할 수 있습니다.

I. 최대공약수 게임

- ✓ $C_G =$ 아무것도 선택하지 않은 카드 뭉치에서 G의 배수가 적힌 카드의 수라고 정의합니다.
- ✓ $dp[T][G] = T$번째 턴에 현재 칠판에 적힌 수가 G일 때 승패 여부라고 정의합니다.
- ✓ 아래의 경우 중 상대방이 패배하는 경우를 선택할 수 있다면 $dp[T][G]$ 일 때 승리합니다.
  - G의 배수가 아닌 카드를 선택하여 칠판에 새로 적히는 수가 G' 일 때, $G' \neq 1$ 이면서 $dp[T+1][G']$ 가 패배하는 경우
  - $T \leq C_G$ 일 때, G의 배수인 카드를 선택하여 $dp[T+1][G]$ 가 패배하는 경우

I. 최대공약수 게임

- ✓ 정수 X의 약수의 개수를 $d(X)$라 할 때, 모든 G에 대한 C_G는 $\mathcal{O}(d(X) \times N)$에 전처리할 수 있습니다.
- ✓ dp의 상태는 $d(X) \times N$개이고 각 상태마다 최대 N개의 카드에 대해 최대공약수를 계산합니다.
- ✓ 총 시간복잡도는 HashMap을 잘 사용했을 때 $\mathcal{O}(d(X) \times N^2 \log X)$ 입니다. 상수 커팅이 어느정도 적용되므로 해당 시간복잡도로 이 문제를 풀 수 있습니다.
- ✓ 현재 턴의 기우성에 대한 게임판의 상태를 관찰한다면 $\mathcal{O}(d(X) \times N \log X)$의 시간복잡도로 문제를 해결할 수 있습니다.

J. 토지 판매

`math`, `floor_sum`

출제진 의도 – **Challenging**

- ✓ 제출 0번, 정답 0팀 (정답률 0.000%)
- ✓ 처음 푼 팀: -
- ✓ 출제자: 장래오 `leo020630`

J. 토지 판매

✓ 2차원 격자는 너무 넓으므로 차원을 줄여 봅시다.

✓ $f(i) = pi + q, g(j) = rj + s$로 정의합시다.

✓ $a \leq i \leq c$에 대해 $f(i)$는 전체 합에 $(d - b + 1) \bmod B$번 더해집니다.

✓ 같은 이유로 $b \leq j \leq d$에 대해 $g(j)$는 $(c - a + 1) \bmod B$번 더해집니다.

J. 토지 판매

- ✓ 이제 우리의 목표는 다음과 같습니다.
- ✓ $\bigoplus_{i=a}^{c} f(i) \otimes ((d-b+1) \bmod B), \bigoplus_{j=b}^{d} g(j) \otimes ((c-a+1) \bmod B)$ 을 구하면 됩니다.
- ✓ $a \otimes b$ 는 a 를 b 번 $\oplus$ 연산한 수를 의미합니다.
- ✓ $f(i), g(j)$ 는 등차수열이지만 $\oplus$ 연산은 $B = 2$ 일 때의 Bitwise XOR과 비슷하게 동작하기 때문에 이를 한 번에 구하는 것은 어렵습니다.
- ✓ 답의 각 자릿수에 해당하는 숫자를 따로 구해봅시다.

J. 토지 판매

- ✓ 첫 자리는 자명하게 $(\sum_{i=a}^{c} f(i) + \sum_{j=b}^{d} g(j)) \bmod B$ 입니다.
- ✓ 이는 등차수열의 합 공식을 통해 어렵지 않게 구할 수 있습니다.
- ✓ 어려워지는 것은 두 번째 자리부터입니다.
- ✓ 두 번째 자리에 해당하는 수는 $(\sum_{i=a}^{c} \lfloor \frac{f(i)}{B} \rfloor + \sum_{j=b}^{d} \lfloor \frac{g(j)}{B} \rfloor) \bmod B$ 가 됩니다.
- ✓ 마찬가지 이유로 k 번째 자리에 해당하는 수는 $(\sum_{i=a}^{c} \lfloor \frac{f(i)}{B^{k-1}} \rfloor + \sum_{j=b}^{d} \lfloor \frac{g(j)}{B^{k-1}} \rfloor) \bmod B$ 입니다.
- ✓ ... 이 값을 어떻게 구하면 될까요?

J. 토지 판매

- ✓ $\sum_{x=0}^{n} \lfloor \frac{ax+b}{c} \rfloor$ 꼴의 문제를 floor sum이라 하고, $\mathcal{O}(\log a)$ 시간복잡도에 해결할 수 있음이 알려져 있습니다.
- ✓ 자세한 계산 과정은 다음 페이지에 첨부하였습니다.
- ✓ $\sum_{i=a}^{c} \lfloor \frac{f(i)}{B^{k-1}} \rfloor = \sum_{i=0}^{c} \lfloor \frac{f(i)}{B^{k-1}} \rfloor - \sum_{i=0}^{a-1} \lfloor \frac{f(i)}{B^{k-1}} \rfloor$ 임을 이용해 floor sum으로 원하는 값을 계산해줄 수 있습니다.
- ✓ 최종 시간복잡도는 $\mathcal{O}(K \log \max(pN, rM) \log \max(p, r))$ 입니다.

J. 토지 판매

✓ let $\sum_{x=0}^{n} \lfloor \frac{ax+b}{c} \rfloor = f(a,b,c,n)$

✓ if $a \geq c$ or $b \geq c$

– $f(a,b,c,n) = \sum_{x=0}^{n} \left\lfloor \frac{(\lfloor \frac{a}{c} \rfloor c + (a \bmod c))x + \lfloor \frac{b}{c} \rfloor c + (b \bmod c)}{c} \right\rfloor$

$= \sum_{x=0}^{n} \left(\lfloor \frac{a}{c} \rfloor x + \lfloor \frac{b}{c} \rfloor + \left\lfloor \frac{(a \bmod c)x + (b \bmod c)}{c} \right\rfloor \right)$

$= \frac{n(n+1)}{2} \lfloor \frac{a}{c} \rfloor + (n+1) \lfloor \frac{b}{c} \rfloor + f(a \bmod c, b \bmod c, c, n)$

J. 토지 판매

✓ then $a, b < c$

✓ $f(a,b,c,n) = \sum_{x=0}^{n} \lfloor \frac{ax+b}{c} \rfloor = \sum_{y=0}^{\lfloor \frac{an+b}{c} \rfloor} (0 \le x \le n, y < \lfloor \frac{ax+b}{c} \rfloor$ 를 만족하는 x의 개수$)$

✓ $y < \lfloor \frac{ax+b}{c} \rfloor \Rightarrow y+1 < \frac{ax+b}{c} \Rightarrow \lfloor \frac{cy+c-b-1}{a} \rfloor < x$

J. 토지 판매

- ✓ let $m = \dfrac{an + b}{c}$
- ✓ $f(a, b, c, n) = \sum_{y=0}^{m-1} n - \left\lfloor \frac{cy+c-b-1}{a} \right\rfloor = mn - f(c, c-b-1, a, m-1)$
- ✓ $c > a$이기 때문에 직후 $f(c \bmod a, c-b-1, a, m-1)$로 간주됩니다.
- ✓ 유클리드 호제법과 같은 원리로 $\mathcal{O}(\log a)$ 시간복잡도가 보장됩니다.

K. 신촌방위본부 : 지하 벙커의 비밀

ad-hoc, centroid, divide and conquer, tree isomorphism

출제진 의도 – **Challenging**

- ✓ 제출 2번, 정답 0팀 (정답률 0.000%)
- ✓ 처음 푼 팀: -
- ✓ 출제자: 김도훈dohoon

K. 신촌방위본부 : 지하 벙커의 비밀

✓ 해설에서는 트리, 정점, 간선으로 용어를 통일합니다.
✓ 벙커가 있는 건물도 '목표정점'이라고 부르겠습니다.

K. 신촌방위본부 : 지하 벙커의 비밀

Naive solution 1: 트리의 크기가 30 이하일 경우

- ✓ 목표정점만 검은색으로 색칠하고 나머지는 전부 하얀색으로 색칠하는 간단한 전략이 성립합니다.
- ✓ 후에 나올 전략은 트리의 크기가 30 이하일 경우에 곤란해지므로, 트리 크기가 30 이하라면 미리 예외 처리를 해둡시다.

K. 신촌방위본부 : 지하 벙커의 비밀

지식 1. 동형의 트리라면 변치 않는 것

✓ 센트로이드는 트리마다 고유하게 두 개 이하의 센트로이드를 갖는다는 사실을 알면 좋습니다.

K. 신촌방위본부 : 지하 벙커의 비밀

Naive solution 2: 루트를 센트로이드로 고정했다면

✓ 루트에서 목표정점까지의 경로를 전부 검은색으로, 경로를 둘러싸는 정점들은 전부 하얀색으로 색칠하는 전략을 생각할 수 있습니다.

✓ 다만 K가 최악의 경우 N에 달하므로 줄여야 합니다.

K. 신촌방위본부 : 지하 벙커의 비밀

지식 2: 센트로이드 트리

- ✓ 기존의 트리에서 센트로이드를 반복적으로 따와서 새로 구성한 트리를 센트로이드 트리라고 부릅니다.
- ✓ 센트로이드 트리는 깊이가 로그 스케일이라는 성질이 있습니다.
- ✓ 이에 더해, 문제 상황에 따라 센트로이드 트리는 최대 세 개의 자식 정점을 갖습니다.
- ✓ 센트로이드 트리에서 Naive solution 2를 잘 적용하면 문제를 해결할 수 있을 것 같습니다.

K. 신촌방위본부 : 지하 벙커의 비밀

걸림돌 1. 센트로이드가 2개일 경우

✓ 일단 루트를 잡고 진행하는 과정에서는 루트와 가까운/먼 센트로이드만 반복적으로 잡는 것으로 센트로이드를 특정하여 걸림돌을 해결 가능합니다.

✓ 전체 트리의 루트를 잡을 때에는 적당한 규칙으로 구분해주면 됩니다.

K. 신촌방위본부: 지하 벙커의 비밀

걸림돌 2. 30번의 색깔 반전 횟수 제한

- ✓ 여기까지 잘 왔다면 대략 $3 \times \lfloor \log_2 N \rfloor$ 번 정도의 색깔 반전이 필요하다는 것을 알 수 있습니다.
- ✓ 이는 대략 60 정도의 값인데, 원하는 제한 30의 두 배 가량입니다.
- ✓ 두 배라는 점에 착안하면, 색칠이 필요한 영역에서 색칠하는 것/색칠하지 않는 것을 비교해서 더 작은 쪽을 색칠하는 방식으로 횟수를 절반으로 줄일 수 있습니다.
- ✓ 그렇다면 색깔은 무엇으로 고정되어야 하는가? 라는 질문이 자연스럽게 떠오릅니다.
- ✓ 트리의 크기가 31 이상이므로 남는 정점이 무조건 있습니다. 그걸 반전시키던지, 냅두던지 함으로써 전체 색깔 XOR 값을 원하는 값으로 제어할 수 있습니다.

K. 신촌방위본부 : 지하 벙커의 비밀

엄밀한 상한

✓ 출제/검수진들의 풀이는 디테일에 따라 $\max K$가 28에서 29를 오갑니다.

✓ 계산식은 $\lfloor 1.5 \times \lfloor \log_2 N \rfloor \rfloor + 1$ or 2입니다.

K. 신촌방위본부 : 지하 벙커의 비밀

마.. 많이풀어주세요!!

L. 신촌 도로망 관리와 쿼리

`mst`, `bruteforcing`

출제진 의도 – **Medium**

- ✓ 제출 56번, 정답 12팀 (정답률 21.429%)
- ✓ 처음 푼 팀: **Raa_FanClub**[서강대학교], 16분
- ✓ 출제자: 노현제[hjroh0315]

L. 신촌 도로망 관리와 쿼리

- ✓ 문제 조건에 의해 관리된 도로로 구성된 부분 그래프는 연결되어 있어야 합니다.
- ✓ 따라서 관리 비용을 최소화할 경우 관리된 도로가 구성하는 부분 그래프는 MST가 됩니다.
- ✓ $Q \leq 2 \times 10^4$ 개의 쿼리에 대해 각각 MST의 가중치 합을 구해야 합니다.
- ✓ 그러나 당연하게도 쿼리당 $\mathcal{O}(M \log N)$ 은 너무 느립니다.

L. 신촌 도로망 관리와 쿼리

- ✓ MST의 중요한 특성을 크루스칼 알고리즘에서 관찰할 수 있습니다.
- ✓ 크루스칼 알고리즘이 반환하는 결과는 가중치의 값 자체보다는 가중치의 순서에 의존합니다.
- ✓ 다시 말해, 가중치의 값이 달라도 순서가 같다면 MST는 달라지지 않습니다.
- ✓ 또한, 가중치가 같은 간선이 있다면 어떤 순서로 추가해도 MST의 가중치는 같습니다.

L. 신촌 도로망 관리와 쿼리

- ✓ 이제 가중치의 값이 최대 5 가지이므로, 유의미한 순서는 총 $5! = 120$ 개입니다.
- ✓ 각각에 대해 크루스칼 알고리즘을 $\mathcal{O}(M\alpha(N))$ 또는 $\mathcal{O}(M \log N)$ 에 수행할 수 있습니다.
- ✓ 또는, 프림 알고리즘을 $\mathcal{O}(5M)$ 또는 $\mathcal{O}(M)$ 에 수행할 수 있습니다.
- ✓ 가중치의 가짓수를 c 라고 하면, 총 (MST를 구하는 시간) $\times\ \mathcal{O}(c!)$ 에 전처리가 가능합니다.

L. 신촌 도로망 관리와 쿼리

- ✓ 전처리 과정에서 각 스패닝 트리에 대해, 각 학교에서 포함된 도로 수만 저장하도록 합니다.
- ✓ 이렇게 하면 한 스패닝 트리에 순서쌍을 대입한 후의 가중치 합을 $\mathcal{O}(c)$에 구할 수 있습니다.
- ✓ 주어진 관리비 순서쌍을 정렬하여 순서를 찾으면 각 쿼리를 $\mathcal{O}(c \log c)$에 해결할 수 있습니다.

L. 신촌 도로망 관리와 쿼리

✓ 이제 전체 문제를 $\mathcal{O}(c!M\alpha(N) + Qc\log c)$ 등의 시간 복잡도로 해결할 수 있습니다.

M. 엘리스 트랙 매칭

`implementation, string`

출제진 의도 – **Easy**

- ✓ 제출 54번, 정답 48팀 (정답률 88.889%)
- ✓ 처음 푼 팀: **AKARAKA**[연세대학교], 1분
- ✓ 출제자: 권순호 `tnsgh9603`

M. 엘리스 트랙 매칭

- ✓ 세 번째 줄에 주어지는 알파벳이 두 번째 줄에 주어지는 알파벳들에 몇 번 등장하는지 세어 출력하면 됩니다.
- ✓ 총 시간 복잡도는 $\mathcal{O}(N)$ 입니다.