# 2016 UCPC 풀이

제6회 전국 대학생 프로그래밍 대회 동아리 연합 여름 대회 풀이

2016. 08. 20

# A. 배열

정답 팀 : 3

가장 처음 푼 팀 : HanzoGak (김진표, 박상수, 박성민)

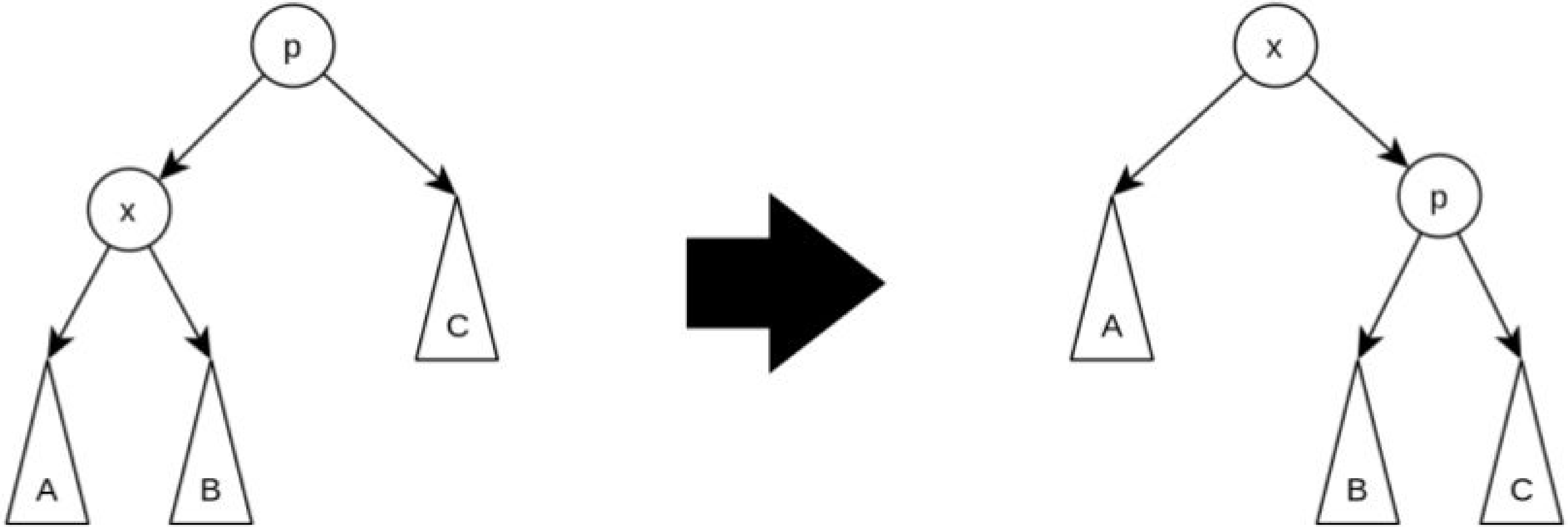
출제자 : myungwoo (전명우)
해설작성자 : myungwoo (전명우)
해설자 : myungwoo (전명우)

# 배열

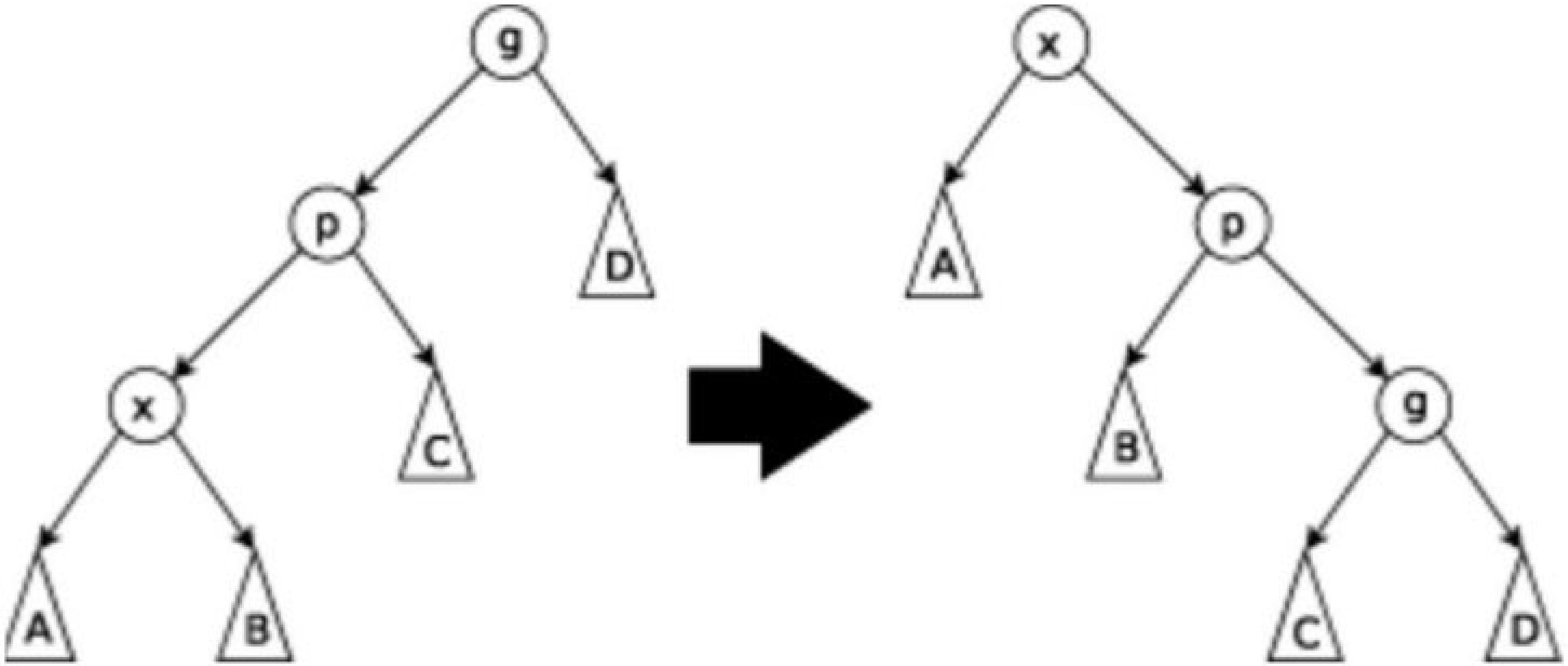
- Are you good at using Splay Tree?

- Splay Tree
  - 이진 탐색 트리
  - 하지만 특별한 연산 한 개가 있다…
  - splay(i)
    - 노드 i를 이진 탐색 트리의 루트로 만든다.
  - 응용: splay(j, i)
    - 노드 j를 노드 i의 바로 아래에 붙인다.

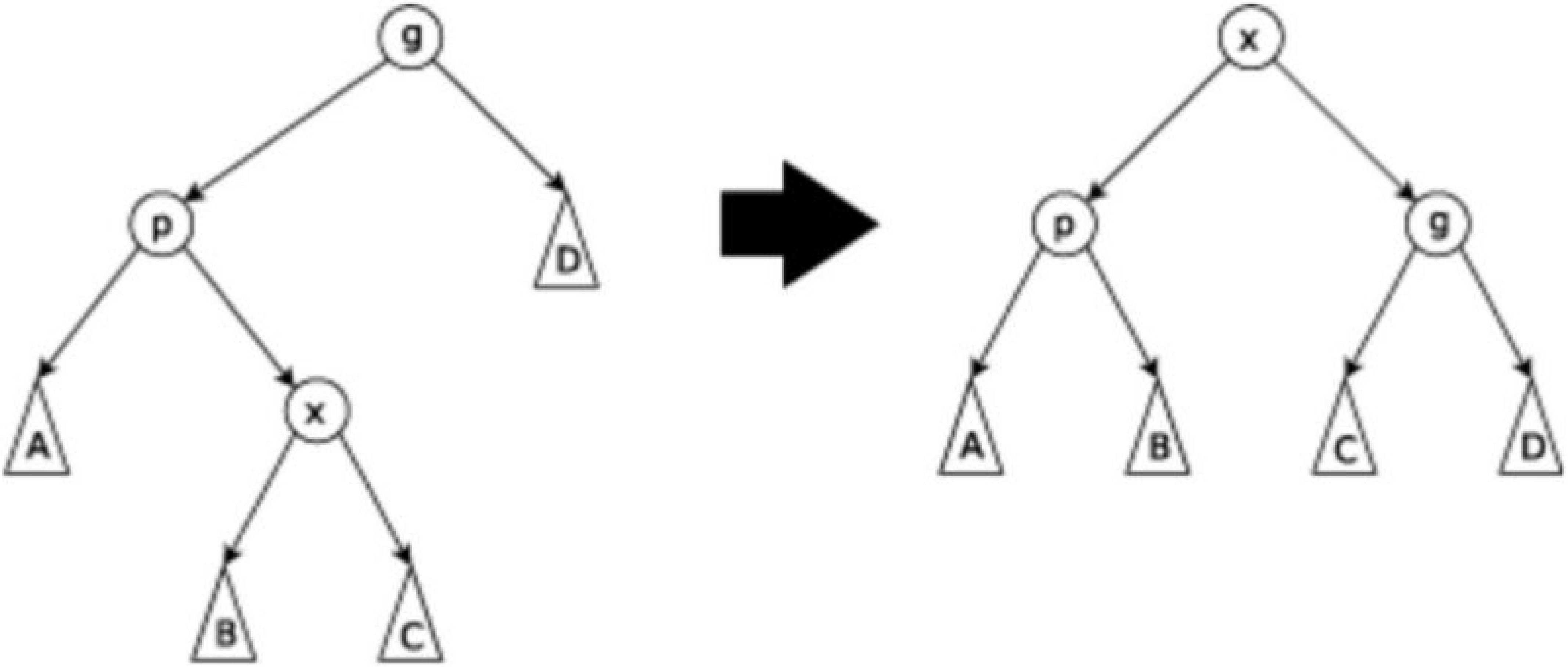
# 1) Zig step



# 2) Zig-zig Step



# 3) Zig-zag Step



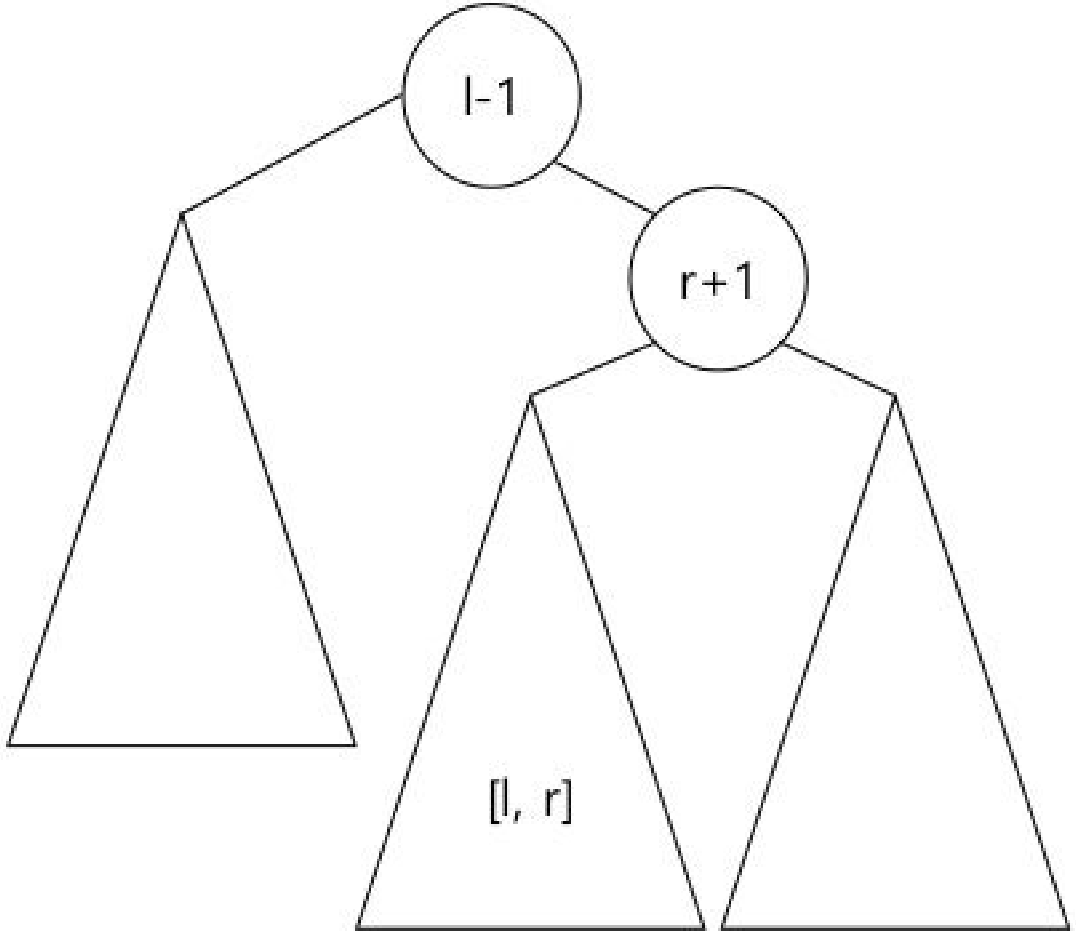
# 배열

- 이제 splay를 적절히 이용해서 flip, shift를 구현하면 된다!
- Subarray의 shift는 flip 3번으로 구현이 가능하다.

# 배열

- Flip은 어떻게 구현할까?
  - Flip(l, r)이 이루어진다고 하자
  - splay(l−1), splay(r+1, l−1)

- [l, r]에 해당하는 부분 이진탐색 트리를 만들고, lazy propagation으로 구현



# 배열

- 3번은 이진탐색트리에서 i번째 수를 구하는 연산

- 4번은 각 수에 해당하는 노드 포인터를 미리 배열로 가지고,
  해당 노드를 splay , 그러면 왼쪽 서브트리 크기+1이 답.

# 배열

- Splay 트리의 시간복잡도는?
  - 앞서 설명한 방식으로 splay를 구현하면 모든 연산이 amortized O(lg n)라는 것이 증명됨
  - 다만 splay를 자주해줘야함
    - 예를 들어, 이진탐색트리에서 search한 이후 search로 나온 노드를 splay
    - 예를 뜰어, 이진탐색트리에서 insert한 이후 insert한 노드를 splay

- 최소, 최대, 합의 구현은?
  - 각 노드별로 노드를 루트로 했을 때 서브트리의 최소, 최대, 합을 저장
  - Zig, zig-zig, zig-zag 과정에서 관련 자료를 실수없이 계산

# B. 최대 클리크 구하기

정답 팀 : 45

가장 처음 푼 팀 : Anti-ACG (박범수, 박서홍, 박성관)

출제자 : myungwoo (전명우)
해설작성자 : kriii (김경근)
해설자 : kriii (김경근)

# 구간 그래프

두 정점 $i, j$사이에 간선이 있으려면, $[S_i, E_i]$와 $[S_j, E_j]$사이에 공통된 부분이 있어야 합니다.

즉, $S_i \le x \le E_i$와 $S_j \le x \le E_j$를 동시에 만족하는 $x$가 있으면 됩니다.

조금 간단하게 적으면 $\max\left(S_i, S_j\right) \le x \le \min(E_i, E_j)$를 만족하는 $x$가 존재하면 됩니다.

# 구간 그래프의 클리크

$n$개의 정점 $i_1, \dots, i_n$사이의 모든 두 정점간에 간선이 있으려면, 아까 전의 식을 모두 연립해 볼 때,

$$\max\left(S_{i_1}, \dots, S_{i_n}\right) \le x \le \min\left(E_{i_1}, \dots, E_{i_n}\right)$$

를 만족하는 $x$가 존재하면 됩니다.

# 구간 그래프의 클리크

이제 역으로 어떤 $x$를 정했을 때, $S_i \le x \le E_i$를 만족하는 모든 $i$를 찾으면 그것이 클리크를 이룬다는 것을 알 수 있습니다.

주어진 구간의 좌표를 압축하고 $x$가 증가하는 순서로 스위핑을 하여 최대한 많은 구간을 통과하는 $x$를 찾아 $O(NlgN)$에 문제를 해결할 수 있습니다.

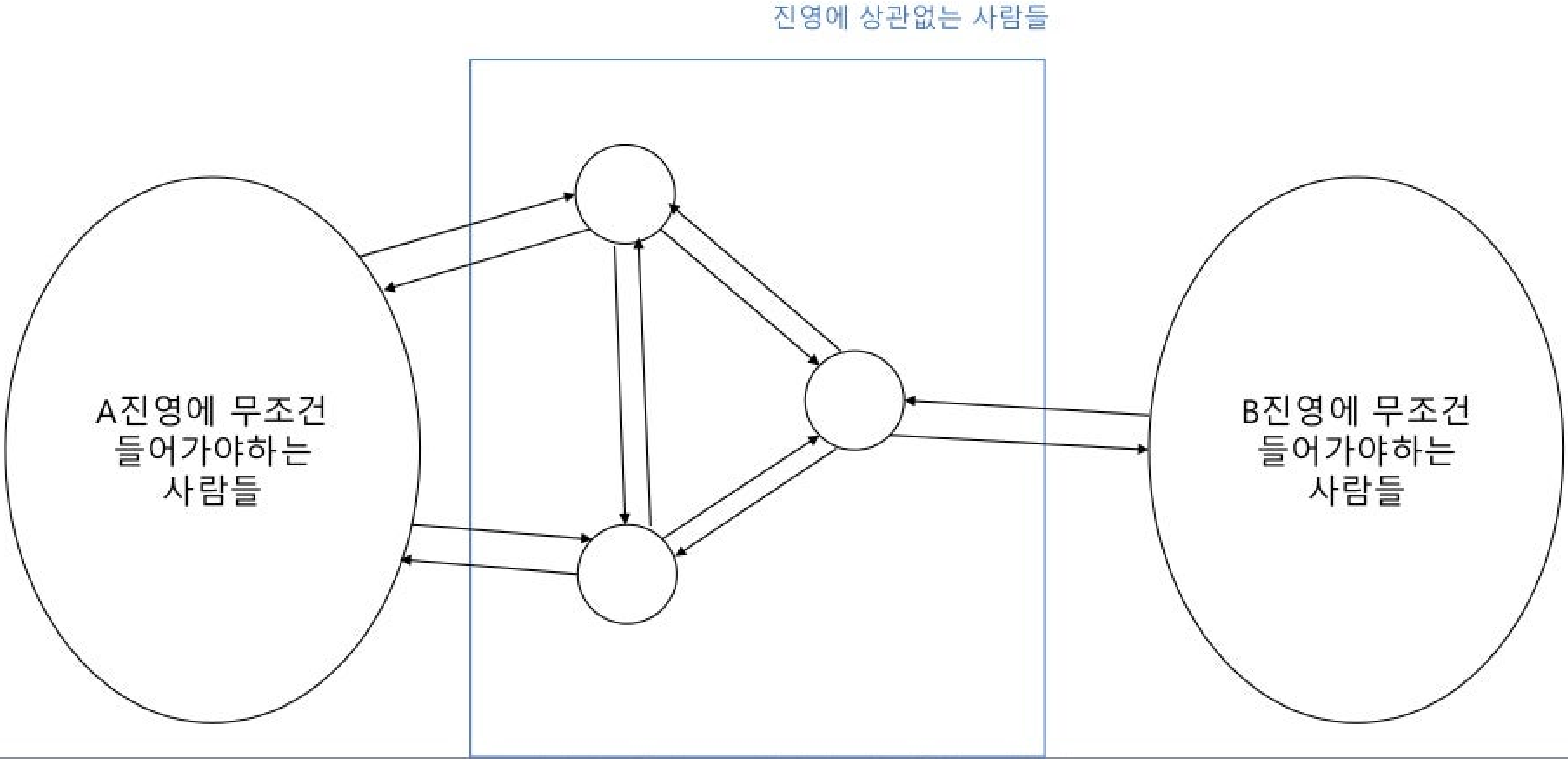
# C. 분단의 슬픔

정답 팀 : 2

가장 처음 푼 팀 : ACG (최석환, 윤지학, 조승현)

출제자 : myungwoo (전명우)
해설작성자 : myungwoo (전명우)
해설자 : myungwoo (전명우)

# 분단의 슬픔



# 분단의 슬픔

- 위와 같이 그래프를 만들고 max-flow를 구하면 된다

- Max-flow = min-cut

- 위 그래프에서 cut하나는 필연적으로 하나의 분단 구성을 의미하고, 그 때 cut 비용이 슬픔 정도의 합이기 때문!

# 분단의 슬픔

- 여기서, 2가지를 더 신경써줘야한다

1. Running time: Ford-Fulkerson TLE, Dinic 0.48초, Optimal 0.02초

2. (일부러 함정을 판 건 아닌데…) newline…
   - 출력 형식에 3개의 줄을 출력하고, 진영에 속한 사람이 없으면 빈 줄을 출력하라고 명시
   - "1₩n2₩n"은 두 개의 줄로 판단… (POSIX 정의에서 한 줄의 끝은 newline…)
   - 이걸로 고생한 팀이 있어서 죄송할 따름입니다…

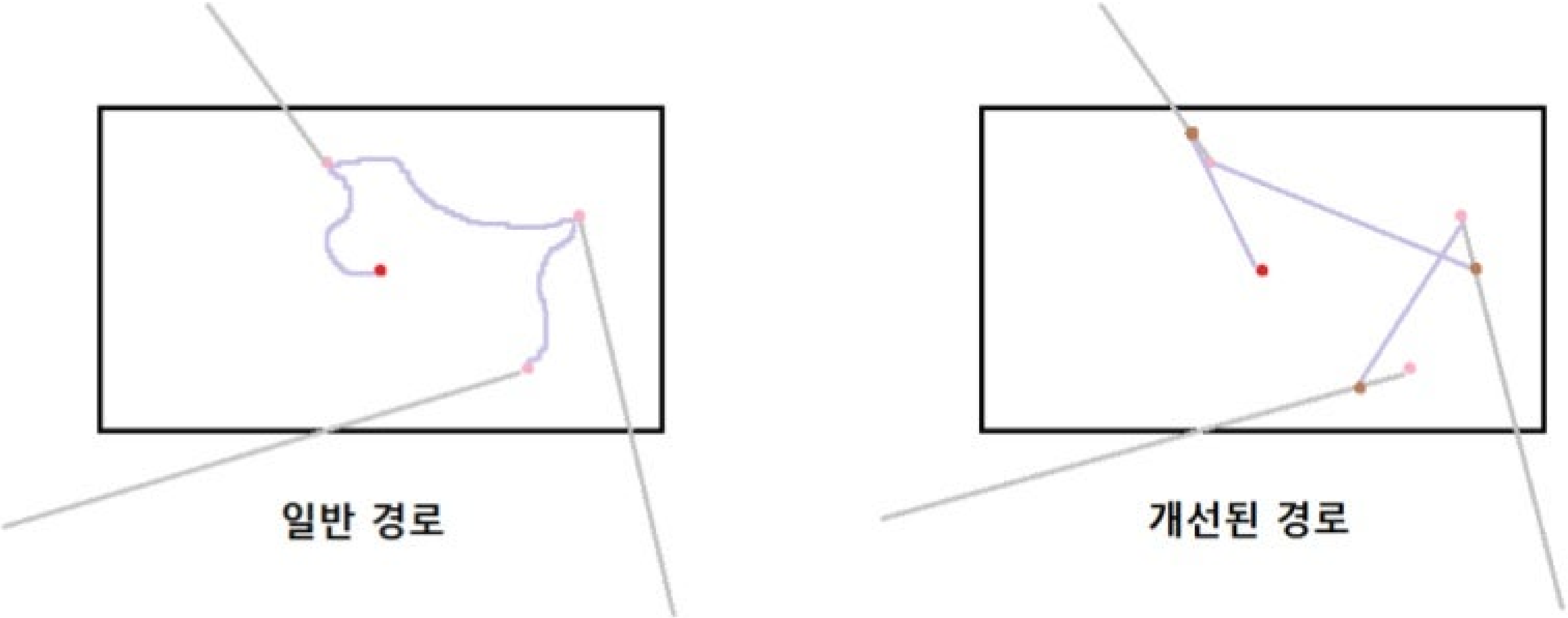
# D. Flowey's Love

정답 팀 : 2

가장 처음 푼 팀 : ACG (최석환, 윤지학, 조승현)

출제자 : functionx (배근우)
해설작성자 : functionx (배근우)
해설자 : functionx (배근우)

# 최적해로 가능한 그림




- 당신과 친절 알갱이의 속력이 1이므로 항상 개선된 경로를 만들 수 있다.

# TSP 문제로 변형 가능

- 방문하는 점의 순서를 지정
- 각 점을 최대한 빨리 방문한다.

- 답을 Brute-Force로 구하면 $O(n!)$ 이므로 시간초과가 난다.

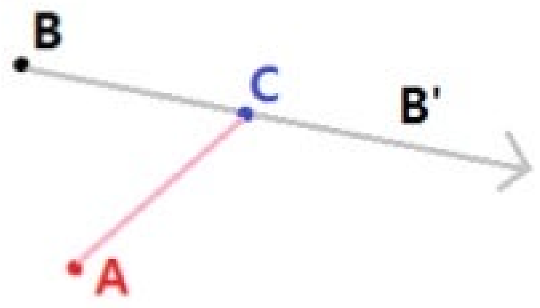
# 비트 DP 식 정의

- DP[1100100] [4] : 0, 1, 4번 점을 방문하며, 마지막으로 4번 점을 방문할 때 걸리는 최소 시간 (불가능하면 -1으로 처리)

- DP[bit] [a] 에서 DP[bit+2b] [b] 로 뿌려주는 방식으로 답을 계산
- DP[bit] [a] 초 후 상태에서 a번 점에서 b번 점으로 가는 시간 계산

- 시간복잡도는 $O(n^2 \cdot 2^n)$ 이다.

# 점들 간의 도달시간 계산




- 점 A와 반직선 BB’가 있으면 반직선 위에 AC=BC 인 점 C를 잡아야 함
- BC의 길이를 x로 놓고 벡터 방정식을 세워서 풀면 x를 구할 수 있음
- 만약 답이 되는 지점이 직사각형 영역 밖에 있으면 점이 직사각형 안으로 들어올 때까지 기다려야 함

# E. 닉네임에 갓 붙이기

정답 팀 : 69

가장 처음 푼 팀 : Never give up (이승재, 이창수, 장홍준)

출제자 : functionx (배근우)
해설작성자 : functionx (배근우)
해설자 : functionx (배근우)

# E. 닉네임에 갓 붙이기

- gets, fgets, getline 등을 이용하여 줄 단위로 입력합니다.
- strlen 함수를 이용하여 문자열의 길이를 구한 후, for문과 if문을 이용해서 첫 공백을 찾습니다.

- 우선 god을 출력해준 다음, 첫 공백 이후의 문자들을 출력해줍니다. 다만, 공백은 출력하지 말아야 합니다.

# F. 행복 유치원

정답 팀 : 62

가장 처음 푼 팀 : ACG (최석환, 윤지학, 조승현)

출제자 : myungwoo (전명우)
해설작성자 : kriii (김경근)
해설자 : kriii (김경근)

# F. 행복 유치원

처음 모든 학생이 한 조에 들어가면 키 차이는 $x_N - x_1$입니다.

$$[x_1, \dots, x_i, \qquad x_{i+1}, \dots, x_N]$$

# F. 행복 유치원

$x_i, x_{i+1}$를 기준으로 조를 나누면 키 차이가 $x_{i+1} - x_i$줄어듭니다.

$$[x_1, \ldots, x_i] \qquad [x_{i+1}, \ldots, x_N]$$

# F. 행복 유치원

즉, 조를 하나 늘리려면 $x_{i+1} - x_i$중 하나를 선택해 줄이면 됩니다.

그러므로 $x_{i+1} - x_i$를 내림차순으로 정렬하여 앞의 $K - 1$개를 선택 하여 $x_N - x_1$에서 빼면 됩니다.

# G. 이것도 해결해 보시지

정답 팀 : 3

가장 처음 푼 팀 : hYEAHyea (고지훈, 강한필, 이종원)

출제자 : xhark (김재홍)
해설작성자 : xhark (김재홍)
해설자 : kriii (김경근)

# Check about

$$AB = C$$

모두 $N \times N$행렬이기 때문에, $O(N^3)$의 시간이 걸려야 체크할 수 있습니다. 빠른 행렬 곱셈 방법이 있기는 하지만, 이런 식으로는 아무리 빨라도 $O(N^2)$은 불가능합니다.

# Freivalds' algorithm

$$ABv = Cv$$

$v$가 $N \times 1$행렬이라고 할 때 $A(Bv)$순서로 계산하면 $O(N^2)$의 시간에 위의 식이 참인지 체크 가능하며, $v$가 0과 1로 이루어진 행렬이라고 하더라도 ½이상의 성공 확률을 보장합니다.

# 출제자의 변

$v$의 원소로 가능한 범위가 더 커지면 확률이 어떻게 변할까?

$ABx = Cx$의 근은 어떤 형태일까?

별해는 FFT

# H. 범죄 파티

정답 팀 : 23

가장 처음 푼 팀 : Anti-ACG (박범수, 박서홍, 박성관)

출제자 : Acka (김현정)
해설작성자 : Acka (김현정)
해설자 : Acka (김현정)

# 2-SAT 접근

- i의 입장에서: (iA || iB)
- i, j에게 변호를 요청받은 A에 대해서: (!iA || !jA)

- !iA => iB, !iB => iA, iA => !jA, jA => iA

- 모든 (x1 || y1) 절을 만족하는 문제.

# 2-SAT 접근

- 파티 비용을 정하고 => Parametric Search
- 모든 조건들을 만족할 수 있는지를 조사한다.

- 각 친구들에게 용의자 친구는 두명까지만 존재하므로 한 번의 2-sat에 소요되는 시간은 O(E) => O(N)
- 시간복잡도 O(NlogN)

# 이분매칭 접근

- 용의자 i에 대해서: i -〉 Ai, i - 〉Bi
- 전체 N명의 용의자에 대해 최대 매칭이 N이면 가능하다.

- 역시 파티 비용을 먼저 잡고 =〉 Parametric Search

- Hopcroft-Karp: worst Nsqrt(N) 짱짱 이분매칭 알고리즘

- 사실 이런 거 필요없고,
  용의자와 친구 간의 관계가 최대 2:2를 보장하므로 그리디도 가능합니다^^..

# I. 포스터

정답 팀 : 2

가장 처음 푼 팀 : ACG (최석환, 윤지학, 조승현)

출제자 : myungwoo (전명우)
해설작성자 : functionx (배근우)
해설자 : functionx (배근우)

# 2D에서의 접근

- Plane-Sweeping을 이용하여 각 y-축에 대해 포스터가 보이는 길이를 구한 후, y-좌표의 차이를 곱하여 더함.

- 따라서 우리는 1D에서의 포스터 문제를 풀면 됨.

# 1D에서의 접근

- 좌표 정렬은 이미 한 상태라고 생각.
- Segment Tree 등을 이용하여 1번 포스터부터 덮으면 $O(N^2 logN)$ 으로 TLE가 날 수도 있음.
- Union-Find를 이용하여 N번 포스터부터 덮는 방법을 사용한다. 이 경우 시간복잡도는 $O(N^2\alpha(N))$ 이다.

# Union-Find 알고리즘

- 맨 처음에 next[i]를 i+1로 초기화

| index | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| next | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| post | - | - | - | - | - | - | - | - | - |

- S~E 범위를 포스터로 채우면 됨

| index | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| next | 2 | 3 | **5** | **5** | 6 | 7 | 8 | 9 | 10 |
| post | - | - | **N** | **N** | - | - | - | - | - |

# Union-Find 알고리즘

- S, next[S], next[next[S]], …를 채우므로 이미 채워진 곳은 안 채워짐

| index | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| next | 2 | **6** | **6** | **6** | **6** | 7 | 8 | 9 | 10 |
| post | - | **N-1** | N | N | **N-1** | - | - | - | - |

- next 배열의 값을 일일이 바꾸면 $O(n^2)$ 이다. 하지만 Union-Find 알고리즘을 이용하여 이를 $O(n)$으로 줄일 수 있다.
- Cover(3, 4, N)에서는 union(3, 4)를 한다. 이때 next의 값은 3이 있는 컴포넌트의 루트 노드에서만 바꾸면 된다.
- 비슷하게, Cover(2, 5, N-1)은 union(2, 3), union(3, 5)를 한다.

# J. 내일로 여행

정답 팀 : 50

가장 처음 푼 팀 : bubble_bath_with_ntopia (김인섭, 박성원, 한수환)

출제자 : functionx (배근우)
해설작성자 : functionx (배근우)
해설자 : functionx (배근우)

# J. 내일로 여행

- 내일로 티켓을 샀을 때의 그래프와 사지 않았을 때의 그래프 두 개에 대해서 처리한다.

- M−1개의 경로에 대해서 최단경로를 구한다.

- 플로이드-워셜 알고리즘을 이용하면 $O(n^3 + m + k)$ 에 풀 수 있다.
- 다익스트라 알고리즘을 이용하면 $O(km\ logn)$ 에 풀 수 있다.

# K. Xor of sums

정답 팀 : 5

가장 처음 푼 팀 : hYEAHyea (고지훈, 강한필, 이종원)

출제자 : kriii (김경근)
해설작성자 : kriii (김경근)
해설자 : kriii (김경근)

# Meet in the middle

$n \le 30$이라는 조건은 노골적인 힌트가 될 수 있습니다.

$n$이 $O(2^n)$정도에 풀기는 어렵고, $O(2^{n/2})$정도의 시간에는 해결할 수 있을 것 같다면 한번쯤 생각해 봐야 합니다.

목표 시간 복잡도는 $O\left(n \cdot 2^{n/2} \cdot lg \sum_{i=1}^{n} a_i\right)$입니다.

# $N = 4$

입력으로 네 개의 수 $a, b, c, d$가 주어졌다고 하면 다음의 수를 구해야 합니다. 편의를 위해 0도 같이 적습니다.

$$
\begin{aligned}
&0 \oplus a \oplus b \oplus (a+b) \oplus \\
&c \oplus (a+c) \oplus (b+c) \oplus (a+b+c) \oplus \\
&d \oplus (a+d) \oplus (b+d) \oplus (a+b+d) \oplus \\
&(c+d) \oplus (a+c+d) \oplus (b+c+d) \oplus (a+b+c+d)
\end{aligned}
$$

# $N = 4$

수들을 두 그룹 $[a, b]$와 $[c, d]$로 나눈 뒤 항을 생각해 봅니다.

$$
\begin{aligned}
&0 \oplus a \oplus b \oplus (a+b) \oplus \\
&c \oplus (a+c) \oplus (b+c) \oplus (a+b+c) \oplus \\
&d \oplus (a+d) \oplus (b+d) \oplus (a+b+d) \oplus \\
&(c+d) \oplus (a+c+d) \oplus (b+c+d) \oplus (a+b+c+d)
\end{aligned}
$$

# $N = 4$

수들을 두 그룹 $[a, b]$와 $[c, d]$로 나눈 뒤 항을 생각해 봅니다.

$$(0 + 0) \oplus (a + 0 + 0) \oplus (b + 0 + 0) \oplus (a + b + 0 + 0) \oplus$$
$$(c + 0) \oplus (a + c + 0) \oplus (b + c + 0) \oplus (a + b + c + 0) \oplus$$
$$(0 + d) \oplus (a + 0 + d) \oplus (b + 0 + d) \oplus (a + b + 0 + d) \oplus$$
$$(c + d) \oplus (a + c + d) \oplus (b + c + d) \oplus (a + b + c + d)$$

# $N = 4$

즉, 첫 번째 그룹으로 다음 식의 값을 빨리 구할 수 있도록 준비하고,

$$x \oplus (a + x) \oplus (b + x) \oplus (a + b + x)$$

두 번째 그룹으로 만들 수 있는 모든 합을 위에 있는 식에 넣어 값을 모두 구하면 됩니다.

# 값 구하기

어떤 수들을 xor한 결과는 각 비트에 대해 독립적으로 작용하므로 $x$가 특정할 수일 때 각 비트의 값을 적어봅니다.

| $x$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $x$의 $2^0$ 비트 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| $x$의 $2^1$ 비트 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| $x$의 $2^2$ 비트 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| $x$의 $2^3$ 비트 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

$2^k$비트는 $x$에 대해 $2^{k+1}$의 주기를 가지고 반복됩니다.

# 값 구하기

$(x + a)$의 비트는 $x$의 비트가 $a$칸씩 쉬프트 되어 나옵니다.

| $x$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $x$의 $2^2$ 비트 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| $(x+1)$의 $2^2$ 비트 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| $(x+2)$의 $2^2$ 비트 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| $(x+3)$의 $2^2$ 비트 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| $(x+4)$의 $2^2$ 비트 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |

# 값 구하기

$$x \oplus (a + x) \oplus (b + x) \oplus (a + b + x)$$

같은 식에서, $a, b, a + b$를 $2^{k+1}$로 나눈 나머지를 가지고, 어떤 지점에서 비트가 몇 번 바뀌는지를 이분검색을 이용해 찾을 수 있습니다.

# L. 떨어진 수정

정답 팀 : 53

가장 처음 푼 팀 : Anti-ACG (박범수, 박서홍, 박성관)

출제자 : xhark (김재홍)
해설작성자 : xhark (김재홍)
해설자 : kriii (김경근)

# 답

$$\left\lceil \frac{P}{W} \right\rceil$$

# 설명

최악의 경우는 $K$번째 보다 강도가 약한 수정은 모두 강도가 0에 가깝고, 강한 수정은 모두 강도가 $P$에 가까운 경우 입니다.

이 때, $K$번째 수정을 제외하고는 아무리 수정을 내리쳐도 정보를 뽑아낼 수 없습니다.

그러므로, $W, 2W, 3W, \ldots, P$로 내리치는 것이 최선의 방법입니다.