제1회 블롭컵 해설

출제진/검수진 목록

출제진

dj035, gunwookim

검수진

amsminn, chansol, flappybird, jjang36524, lighton, mjhmjh1104, tlsdydaud1

문제별 해설

A. blobnom



출제: **gunwookim**

최초 해결: **kimjihoon** (2분 37초)

제출 횟수: **853회**

맞은 사람: **247명**

난이도: **Easy**

A. blobnom

- 한 개의 탑을 계속 선택하는 것이 최선임을 보일 수 있습니다.
- i 번째 탑과 (i+1) 번째 탑을 모두 선택할 경우, (i-1) 번째 탑과 (i+2) 번째 탑의 높이가 1만큼 감소하고 i 번째 탑과 (i+1) 번째 탑의 높이는 변하지 않습니다. 처음 상태보다 높아진 탑이 없으므로 최선이 아닙니다.
- 만약 서로 인접하지 않은 두 탑을 선택할 경우, 한쪽 탑을 선택하는 것이 다른 쪽 탑의 높이에 직접적으로 영향을 주지 않습니다. 따라서 이 경우도 고려할 필요가 없습니다.

A. blobnom

- i 번째 탑을 계속 선택할 경우, 최대 $\min(A_{i-1}, A_{i+1})$번 선택할 수 있습니다. 이때 i번째 탑의 높이는 $A_i + \min(A_{i-1}, A_{i+1})$이 됩니다.
- 따라서 2 이상 N-1 이하인 i에 대해 $A_i + \min(A_{i-1}, A_{i+1})$의 최댓값을 구하면 i 번째 탑을 계속 선택했을 때 탑의 높이의 최댓값을 구할 수 있습니다.
- 첫 번째 탑 또는 마지막 탑은 선택할 수 없으므로 이 탑들의 최대 높이는 처음에 주어지는 높이와 같습니다. 따라서 위에서 구한 값과 A_1, A_n 중 최댓값이 만들 수 있는 탑의 최대 높이가 됩니다.

B. blobyum



출제: **dj035**

최초 해결: **woolim** (3분 5초)

제출 횟수: **772회**

맞은 사람: **275명**

난이도: **Easy**

B. blobyum

풀이 1

- x 번째 ~ (x+K-1) 번째 애플파이의 맛의 합과 (x+1) 번째 ~ (x+K) 번째 애플파이의 맛의 합을 식으로 나타내면 다음과 같습니다.

x 번째 ~ (x+K-1) 번째 애플파이의 맛의 합 $S_x = A_x + \mathbf{A_{x+1} + A_{x+2} + \dots + A_{x+K-2} + A_{x+K-1}}$

(x+1) 번째 ~ (x+K) 번째 애플파이의 맛의 합 $S_{x+1} = \mathbf{A_{x+1} + A_{x+2} + \dots + A_{x+K-2} + A_{x+K-1}} + A_{x+K}$

B. blobyum

- 파란색으로 표시된 부분이 서로 같기 때문에, S_x와 S_{x+1}을 관계식으로 나타낼 수 있습니다.

$$S_{x+1} = S_x - A_x + A_{x+K}$$

- 따라서 S_1을 구하면 나머지 S_x의 값도 빠르게 구할 수 있습니다. 이 방법을 슬라이딩 윈도우 기법이라고 부릅니다.

B. blobyum

풀이 2

- 첫 번째 ~ x 번째 애플파이의 맛의 합을 p_x라고 하면 풀이 1에서 정의한 S_x는 다음과 같이 나타낼 수 있습니다.

$$S_x = p_{x+K-1} - S_{x-1}$$

- 따라서 각각의 p_x의 값을 구해 놓으면 S_x를 빠르게 구할 수 있습니다. 이 p_x를 누적 합이라고 부릅니다.

B. blobyum

- 애플파이가 원형으로 배치되어 있기 때문에 N번째 애플파이 다음에 첫 번째 애플파이가 배치되어 있고, N번째 애플파이와 첫 번째 애플파이를 모두 선택할 수 있는 경우 그런 경우도 다 체크해야 합니다.
- 배열을 두 번 반복해서 $A_{N+i} = A_i$인 2N 크기의 배열을 만들면 이런 경우를 쉽게 체크할 수 있습니다.

C. blobblush



출제: **dj035**

최초 해결: **jh05013** (1분 50초)

제출 횟수: **396회**

맞은 사람: **154명**

난이도: **Easy**

C. blobblush

- 뽑은 카드들에 적힌 수를 모두 XOR했을 때 나올 수 있는 최댓값이 얼마인지 알아봅시다.
- N보다 큰 2의 거듭제곱 중 최솟값을 p라고 하면, p 이상의 비트의 값이 1인 카드가 없기 때문에 뽑힌 카드들에 적힌 수를 모두 XOR했을 때 p 이상의 값이 나오게 할 수는 없습니다.
- 만약 p보다 작은 2의 거듭제곱이 적힌 카드를 모두 뽑는다면, 카드들에 적힌 수를 모두 XOR한 값은 (p-1)이 됩니다. 앞에서 카드들에 적힌 수의 XOR 값이 p 이상이 될 수 없음을 보였으므로, 카드들에 적힌 수를 모두 XOR했을 때 나올 수 있는 최댓값이 (p-1)임을 알 수 있습니다.

C. blobblush

- 다음으로 카드의 개수를 최소화하는 방법을 생각해 봅시다. 만약 $N = p-1$인 경우 N이 적힌 카드 한 장만 뽑으면 됩니다.
- $N < p-1$인 경우, $p/2 \le N$이므로 $N \oplus (p-1) < N$이 성립합니다. 따라서 N과 $N \oplus (p-1)$이 적힌 카드를 뽑으면 카드에 적힌 수의 XOR이 $(p-1)$이 됩니다. 이때 뽑은 카드는 두 장입니다.

C. blobblush

- 이제 두 장의 카드를 뽑는 방법 중 사전순으로 가장 앞서는 방법을 찾아야 합니다.
- (p-1)은 p보다 작은 모든 비트의 값이 1이므로, 0 이상 (p-1) 이하의 정수 x에 대해 (p-1) ⊕ x = (p-1) - x가 성립합니다. 따라서 두 장의 카드를 뽑았을 경우 두 장의 카드에 적힌 수의 합은 (p-1)입니다.
- 앞쪽 카드에 적힌 수가 작을수록 사전순으로 앞서므로, 두 장의 카드에 적힌 수의 합이 일정할 경우 뒤쪽 카드에 적힌 수를 최대한 크게 하면 됩니다. 카드에 적힌 수는 최대 N이므로 (p-1-N)이 적힌 카드와 N이 적힌 카드를 뽑는 것이 사전순으로 가장 앞서는 방법입니다.

D. blobaww



출제: **dj035**

최초 해결: **hyperbolic** (12분 46초)

제출 횟수: **311회**

맞은 사람: **93명**

난이도: **Medium**

D. blobaww

풀이 1

- S의 위치 (x_2, y_2)를 고정하면, 세 칸을 차례로 누르는 경우의 수는 $(x_1 \le x_2, y_1 \le y_2, ESM_{x1, y1}$ = 'E'인 쌍 (x_1, y_1)의 수) × $(x_3 \ge x_2, y_3 \ge y_2, ESM_{x3, y3}$ = 'M'인 쌍 (x_3, y_3)의 개수)와 같습니다.
- E의 개수와 M의 개수를 2차원 누적 합으로 구해 놓으면 조건을 만족하는 쌍의 개수를 빠르게 구할 수 있습니다. 이제 각각의 S의 위치마다 가능한 경우의 수를 구해서 모두 더하면 답이 나옵니다.

D. blobaww

풀이 2

- 칸 (i, j)에 대해 $1 \le x \le i$, $1 \le y \le j$를 만족하는 칸 (x, y) 중에서 E를 선택하는 경우의 수를 $E_{i,j}$, E와 S를 선택하는 경우의 수를 $S_{i,j}$, E, S, M을 모두 선택하는 경우의 수를 $M_{i,j}$라고 합시다.
- 누적 합과 비슷하게 $E_{i,j} = E_{i-1,j} + E_{i,j-1} - E_{i-1,j-1}$이고, $S_{i,j}$와 $M_{i,j}$도 같은 방법으로 구할 수 있습니다.
- 만약 $ESM_{i,j}$ = 'S'인 경우 $S_{i,j}$에 $E_{i,j}$를 더하고, $ESM_{i,j}$ = 'M'인 경우 $M_{i,j}$에 $S_{i,j}$를 더해 줍니다.
- $M_{N,M}$에 답이 저장됩니다.

E. blobsad



출제: **dj035**

최초 해결: **cross_ratio** (20분 14초)

제출 횟수: **159회**

맞은 사람: **46명**

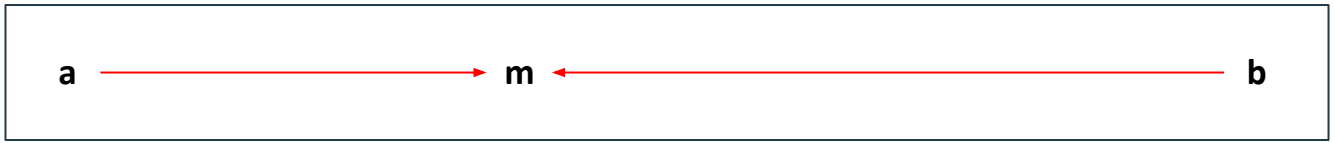
난이도: **Medium**

E. blobsad

- 모든 A_i의 합이 K의 배수가 아니면 일을 마칠 수 없습니다.
- 모든 A_i의 합이 K의 배수일 경우, 앞에서부터 블롭을 K 마리씩 묶은 다음 그중 $\lceil K/2 \rceil$ 번째 블롭이 있는 위치에 모두 모이게 하는 것이 최적의 방법입니다.
- 이를 그대로 구현하면 문제를 풀 수 있습니다.

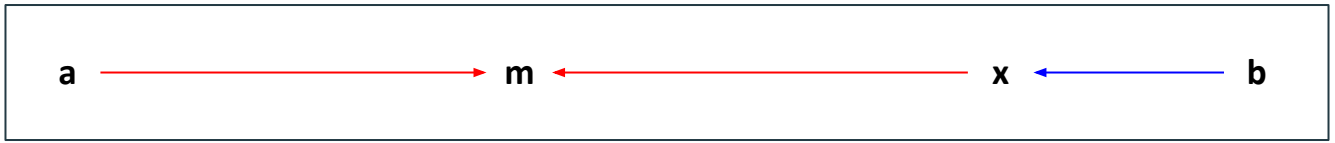
E. blobsad

- 구현을 더 간단하게 하는 방법을 알아봅시다.
- K 마리의 블롭이 a 번째 ~ b 번째 칸에 존재하고, ⌈K/2⌉ 번째 블롭이 m 번째 칸에 존재한다고 가정하면, 이 블롭들을 m 번째 칸에 모으는 것이 시간이 가장 적게 걸리는 방법입니다.



E. blobsad

- 이제 x 번째 칸에 블롭 K 마리를 더 넣었다고 가정합시다. ($m \le x \le b$) 블롭의 수는 2K가 되고, 블롭을 앞쪽 K 마리와 뒤쪽 K 마리로 나누면 블롭이 모이는 칸도 두 군데가 됩니다.
- 이때 K 번째 블롭의 위치는 x 번째 칸이 되며, 블롭이 모이는 칸은 m 번째 칸과 x 번째 칸이 됩니다. 결과적으로 블롭이 모이는 데 걸리는 최소 시간은 변하지 않습니다.



E. blobsad

- a ≤ x ≤ m인 경우도 같은 논리가 성립하므로 임의의 칸에 블롭 K 마리를 추가해도 일을 마치기까지 걸리는 최소 시간은 변하지 않습니다.
- 그렇다면 첫 번째 칸부터 확인하면서 각각의 칸에 있는 블롭의 수를 가장 가까운 K의 배수로 맞춰 주면 됩니다.

F. blobfearful



출제: **gunwookim**

최초 해결: **jh05013** (16분 55초)

제출 횟수: **277회**

맞은 사람: **21명**

난이도: **Medium**

F. blobfearful

- 문제를 단순화하면 $A_q \cdot x!$이 K의 배수가 되게 하는 양의 정수 x의 최소값을 찾는 것으로 해석할 수 있습니다. 먼저 $Q = 1$인 경우를 해결해 봅시다.

F. blobfearful

- $A_q \cdot x$에서 중요한 것은 K의 소인수가 곱해지는 횟수이고, 소인수가 곱해지는 횟수가 서로 영향을 주지 않으므로 각각의 소인수를 따로 체크할 수 있습니다.
- K의 소인수 p에 대해, A_q와 K를 소인수분해했을 때 p의 지수 p_a와 p_k를 계산합니다. $p_a \geq p_k$인 경우 이 소인수는 체크하지 않아도 됩니다.
- $p_a < p_k$인 경우, p의 배수를 작은 것부터 하나씩 확인하면서 소인수 p가 몇 번 더 곱해지는지 확인합니다. p가 곱해진 횟수가 처음으로 p_k 이상이 되었을 때 확인한 p의 배수를 P라고 하면, x는 모든 P 중 최댓값과 같습니다. 만약 체크한 소인수가 없다면 $x = 1$입니다.

F. blobfearful

- 한 개의 쿼리를 처리할 경우 K를 소인수분해하는 데 $O(\sqrt{K})$, A_q에 소인수 p가 곱해진 횟수를 세는 데 $O(\lg K)$, p의 배수를 하나씩 확인하는 데 $O(\lg K)$ 시간이 걸립니다.
- K를 소인수분해한 결과를 전처리해 놓으면 쿼리의 수가 늘어나도 A_q와 소인수 p를 처리하는 데 걸리는 시간만 Q배로 늘어나며, $O(\sqrt{K} + Q \lg K)$ 시간에 전체 문제를 해결할 수 있습니다.

G. blobcry



출제: **gunwookim**

최초 해결: **hyperbolic** (77분 46초)

제출 횟수: **29회**

맞은 사람: **7명**

난이도: **Hard**

G. blobcry

- 섬을 정점으로, 다리를 간선으로 간주하면 그래프를 만들 수 있습니다.
- 어떤 컴포넌트에 홀수 개의 간선이 존재한다면 모든 간선을 제거할 수는 없습니다.
- 마지막에 간선이 하나만 남았을 경우, 처음에 그 간선을 포함한 컴포넌트에는 홀수 개의 간선이 존재합니다.
- 주어지는 그래프에 존재하는 홀수 개의 간선을 포함한 컴포넌트가 하나가 아닌 경우 간선을 하나만 남길 수가 없으므로 마지막에 남을 수 있는 간선의 수는 0입니다.

G. blobcry

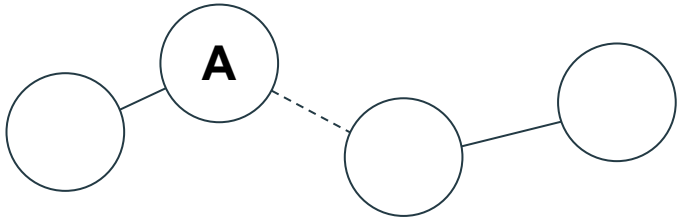
- 간선은 한 번에 두 개씩 제거되므로, 짝수 개의 간선을 가진 컴포넌트에 존재하는 간선들은 마지막 간선이 될 수 없습니다.
- 홀수 개의 간선을 가진 컴포넌트에서 어떤 간선이 마지막 간선이 되려면 나머지 간선들을 모두 제거해야 합니다.
- 만약 그 간선이 단절선이고 그 간선을 제거했을 때 생기는 두 개의 컴포넌트가 간선을 홀수 개씩 포함할 경우 그 간선은 마지막 간선이 될 수 없습니다.

G. blobcry

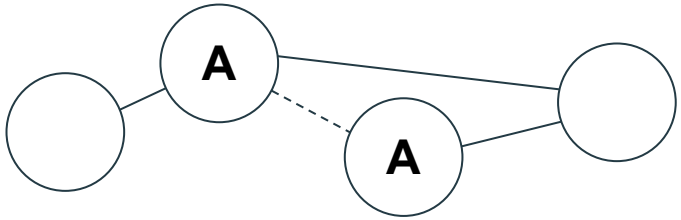
- 이제 어떤 컴포넌트에 짝수 개의 간선이 있으면 그 컴포넌트의 간선들을 조건에 맞게 모두 제거할 수 있음을 확인합시다.
- 컴포넌트에 2개의 간선이 있을 경우 그 간선들을 한번에 제거하면 모든 간선이 제거됩니다.

G. blobcry

- 컴포넌트에 4개 이상의 간선이 있을 경우, 먼저 임의의 간선 하나를 제거합시다. 어떤 간선을 제거하던간에, 홀수 개의 간선을 포함하는 컴포넌트 하나가 생깁니다. 제거한 간선이 연결하던 정점 중 이 컴포넌트에 속하는 정점 하나를 골라서 A라고 합시다.



Case 1: 제거한 간선이 단절선인 경우



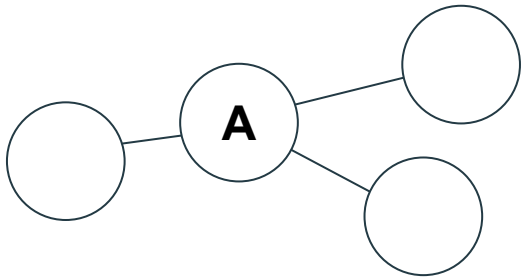
Case 2: 제거한 간선이 단절선이 아닌 경우
(두 정점 모두 A가 될 수 있음)

G. blobcry

- A와 연결된 나머지 간선들 중 단절선이 아닌 간선이 있다면 그 간선을 제거합니다. 이제 그래프의 컴포넌트들은 모두 짝수 개의 간선을 포함하며, 각각의 컴포넌트에 대해서 재귀적으로 간선을 제거해 나가면 모든 간선을 제거할 수 있습니다.

G. blobcry

- A와 연결된 나머지 간선들이 모두 단절선이라면, 정점 A를 제거했을 때 생기는 컴포넌트를 기준으로 간선들을 그룹으로 묶습니다. 컴포넌트 전체에 존재하는 간선의 개수가 홀수이므로, 분류된 그룹들 중 홀수 개의 간선이 포함된 그룹이 하나 이상 존재합니다.
- 그 그룹에서 A와 연결된 간선을 제거하면 컴포넌트는 짝수 개의 간선을 포함한 작은 컴포넌트 둘로 분리되며, 이 컴포넌트들에 대해서 재귀적으로 간선을 제거해 나가면 모든 간선을 제거할 수 있습니다.



G. blobcry

- 따라서, 주어지는 그래프에 홀수 개의 간선을 포함한 컴포넌트가 한 개 존재할 때, 그 컴포넌트의 간선들 중 단절선이 아닌 모든 간선과 제거했을 때 짝수 개의 간선을 포함하는 컴포넌트 두 개가 나오는 모든 단절선이 마지막 간선이 될 수 있습니다.

H. blobhyperthink



출제: **gunwookim**

최초 해결: **tncks0121** (48분 41초)

제출 횟수: **91회**

맞은 사람: **36명**

난이도: **Hard**

H. blobhyperthink

- 구해야 하는 (i, j, k, l, m, o, p, q, r, s, t) 쌍의 개수는 길이가 11인 증가 수열의 개수와 같습니다.
- $a_{x, y}$를 '길이가 x이고 마지막 원소가 y인 증가 수열의 개수'로 정의하면, $a_{1, y}$는 수열에 y가 등장하는 횟수와 같습니다.
- $x \geq 2$일 때, $a_{x, y}$는 y의 왼쪽에 있는 원소들 중 y보다 작은 원소로 끝나는 길이가 (x-1)인 증가 수열의 개수와 같습니다.
- 수열의 원소를 왼쪽부터 하나씩 처리할 경우, $a_{x, y} = a_{x-1, 1} + a_{x-1, 2} + \ldots + a_{x-1, y-2} + a_{x-1, y-1}$로 나타낼 수 있게 됩니다.

H. blobhyperthink

- 이 형태의 식은 세그먼트 트리를 사용하면 빠르게 계산할 수 있습니다.
- 개수를 구하는 증가 수열의 길이가 11이므로 세그먼트 트리도 열한 개를 사용합니다.

I. blobpopcorn



출제: **gunwookim**

제출 횟수: **7회**

맞은 사람: **0명**

난이도: **Challenging**

I. blobpopcorn

- x 번째 블롭에서 시작해서 왼쪽으로 이동할 때 x 번째 블롭보다 키가 큰 블롭이 처음 나오는 위치를 l_x, 오른쪽으로 이동할 때 x 번째 블롭보다 키가 큰 블롭이 처음 나오는 위치를 r_x라고 합시다.
- l_x 번째 블롭과 x 번째 블롭은 사이의 블롭들을 구경할 수 있고, x 번째 블롭과 r_x 번째 블롭은 사이의 블롭들을 구경할 수 있습니다.
- x 번째 블롭과 y 번째 블롭이 사이의 블롭들을 구경할 수 있고 $A_x < A_y$일 경우, $y = l_x$이거나 $y = r_x$이어야 합니다.

I. blobpopcorn

- 조건에 맞는 쌍을 (키가 작은 블롭, 키가 큰 블롭)의 순서로 생각하면 각각의 블롭이 키가 작은 블롭이 되는 쌍이 최대 두 개이므로 답은 2N을 넘지 않습니다.
- x 번째 블롭이 키가 작은 블롭이 되는 쌍이 두 개보다 작은 경우는 l_x가 존재하지 않거나 r_x가 존재하지 않는 경우입니다.
- l_x와 r_x 중 존재하지 않는 값의 개수만큼 답도 줄어들기 때문에, 1 이상 N 이하의 x에 대하여 l_x가 존재하지 않는 x의 개수 w_l과 r_x가 존재하지 않는 x의 개수 w_r를 세면 2번 쿼리를 처리할 수 있습니다. 이때 답은 $2N - w_l - w_r$가 됩니다.

I. blobpopcorn

- I_x가 존재하지 않는 x의 개수를 세 봅시다. 먼저 I_1은 존재하지 않습니다.
- 다음으로 A_x를 차례대로 확인합니다. A_1 ~ A_x의 최댓값이 A_x인 경우 I_x는 존재하지 않습니다.
- I_x가 존재하지 않는 x에 대해 A_x를 나열하면 증가 수열이 됩니다. 이 증가 수열의 길이가 w_l과 같습니다.

I. blobpopcorn

- 다음으로 두 개의 증가 수열을 합쳤을 때 증가 수열의 변화를 확인합시다.
- 왼쪽 증가 수열의 길이가 s이고 왼쪽 증가 수열의 마지막 원소보다 큰 원소가 오른쪽 증가 수열에 k개 있을 때, 합친 증가 수열의 길이는 (s+k)입니다.

I. blobpopcorn

- 세그먼트 트리를 사용하여 A_i가 변할 때 증가 수열의 길이를 빠르게 구해 봅시다.
- 각각의 구간에 대해 그 구간에서의 증가 수열의 길이 a_i와 증가 수열의 최댓값 v_i를 저장합니다.
- A_i가 변할 경우, 노드(lg N) 개의 정보가 변경됩니다. 각각의 노드에서 왼쪽 증가 수열의 마지막 원소보다 큰 원소가 오른쪽 증가 수열에 몇 개 존재하는지는 자식 노드들의 a_i와 v_i 값을 재귀적으로 확인하면 되므로 O(lg N) 시간에 구할 수 있습니다. 따라서 A_i가 변했을 때 세그먼트 트리를 갱신하는 데 걸리는 시간은 $O(\lg^2 N)$입니다.

I. blobpopcorn

- 수열 전체의 증가 수열의 길이는 루트 노드의 a_i 값과 같으므로 $O(1)$ 시간에 구할 수 있습니다.
- r_x가 존재하지 않는 x의 개수를 세는 것 역시 같은 방법으로 할 수 있습니다. 이때는 감소 수열의 길이를 구하게 됩니다.
- 전처리를 $O(N \lg^2 N)$ 시간에 할 수 있고, 1번 쿼리를 $O(\lg^2 N)$ 시간에, 2번 쿼리를 $O(1)$ 시간에 처리할 수 있으므로 $O((N+Q)\lg^2 N)$ 시간에 전체 문제를 해결할 수 있습니다.

J. blobfacepalm



출제: **gunwookim**

최초 해결: **tncks0121** (32분 29초)

제출 횟수: **25회**

맞은 사람: **6명**

난이도: **Challenging**

J. blobfacepalm

풀이 1

- 길이가 2N인 수열에는 홀수 인덱스와 짝수 인덱스가 N개씩 존재합니다.
- 수열에 존재하는 짝수 k에 대해, k의 두 인덱스 중 하나는 홀수, 하나는 짝수입니다.
- 수열에 존재하는 홀수 k에 대해, k의 두 인덱스는 홀수, 홀수이거나 짝수, 짝수입니다.
- 따라서 0부터 (N-1)까지의 정수 중 홀수가 홀수 개 존재하면 2N개의 인덱스에 값을 다 매기지 못합니다. N을 4로 나눈 나머지가 2나 3일 때 이렇게 되며, 이때의 답은 No입니다.

J. blobfacepalm

- N = 4k 꼴로 나타내어질 때, 아래와 같은 해를 찾을 수 있습니다.

(4k-1) (4k-3) … 3 1 (4k-2) 1 3 … (4k-3) (4k-1) (2k-2) (4k-4) (4k-6) … (2k) (2k-4) … 2 (4k-2) (2k-2) 2 4 … (2k-4) 0 0 (2k) … (4k-4)

- 이 외에도 다른 해가 있을 수 있습니다.

J. blobfacepalm

- N = 4k+1 꼴로 나타내어질 때, 아래와 같은 해를 찾을 수 있습니다.

(4k) (4k-2) … 4 2 (2k+1) (4k-1) 2 4 … (4k-2) (4k) (2k+1) (4k-3) (4k-5) … (2k+3) (2k-1) … 3 1 (4k-1) 1 3 … (2k-3) (2k-1) 0 0 (2k+3) … (4k-3)

- 이 외에도 다른 해가 있을 수 있습니다.

J. blobfacepalm

- N = 1, 4, 5인 경우 예외가 생길 수 있으므로 필요한 경우 따로 처리합니다.

  N = 1일 때의 해: 0 0
  N = 4일 때의 해: 2 3 1 2 1 3 0 0
  N = 5일 때의 해: 0 0 2 3 4 2 1 3 1 4

J. blobfacepalm

풀이 2

- 구해야 하는 수열에서 0을 제거하면 랭퍼드 수열과 같습니다.
- 따라서 1부터 (N-1)까지의 정수가 포함된 랭퍼드 수열을 찾은 다음, 끝에 0 0을 붙이면 됩니다.