더 넓—게 경쟁하라!

제10회 경인지역 대학 연합
프로그래밍 경시대회

Official
Problemset

2025.01.11 (Sat) **14 - 18**

shake!
shake!
shake!

단국대학교 SWAG

인하대학교 CTP

한양대학교 ERICA 0&1

경희대학교 학생회

아주대학교 ANSI

한국항공대학교 koala

성균관대학교 NPC

주최 아주대학교 AJOU UNIVERSITY | SW중심대학사업 National Center of Excellence in SW · 아주대학교 LINC사업단 AJOU UNIV. Leaders in INdustry-university Cooperation 3.0 · 경희대학교 SW중심대학사업단

후원 HYUNDAI MOBIS · Samsung Software Membership · A. N. S. i 졸업생 일동 · utilForever [옥찬호] · SOLVED.AC · STARTLINK

- 10문제가 정상적으로 인쇄되었는지 확인하십시오.
- 모든 문제의 메모리 제한은 1024MB로 동일합니다.
- 참가자는 대회 중 사전에 승인받은 하드카피를 참고할 수 있습니다.
- 본 대회의 A번 문항은 출제진이 판단한 가장 쉬운 난이도의 문항입니다. 이후 모든 문항의 난이도는 무작위이므로, 모든 문항의 지문을 읽어보는 것을 권장합니다.
- 대회 중 문제에 관한 질문은 대회 페이지 내 질문 기능을 이용해주시기 바랍니다.
- 대회 중 퇴실은 불가합니다.

문항 목록

문제		시간 제한
A	K-정렬	1.5 초
B	또또 수열 문제야	3 초
C	K 국지	1 초
D	나무핑	2 초
E	마슈 반데드와 마법사의 격자판	1 초
F	경인 국가의 행사	2 초
G	원소 합치기	1 초
H	데이터를 추가해 주세요.	1 초
I	꽃바구니	2 초
J	구간이 이븐하지 않아요.	1 초

문제 A. K-정렬

- 시간 제한 1.5초

문제

길이 N의 순열 $A = [A_0, A_1, \ldots, A_{N-1}]$이 주어진다. 길이 N의 순열이란, 0부터 $N-1$까지의 모든 정수가 정확히 한 번씩 등장하는 수열이다.

양의 정수 K가 주어질 때, 다음과 같은 연산을 원하는 만큼 수행할 수 있다.

- 임의의 i $(0 \le i < N)$에 대해 A_i와 $A_{(i+K) \bmod N}$의 값을 바꾼다.

주어진 연산을 통해 순열 A를 오름차순으로 정렬할 수 있는지 확인해 보자.

mod 연산에 대한 설명은 노트를 참고하라.

입력

첫 번째 줄에 순열의 길이 N과 양의 정수 K가 공백으로 구분되어 주어진다. $(1 \le K \le N \le 10^6)$

두 번째 줄에 순열 A의 원소 $A_0, A_1, \ldots, A_{N-1}$이 공백으로 구분되어 주어진다.

출력

주어진 연산을 원하는 만큼 반복하여 순열 A를 오름차순으로 정렬할 수 있다면 YES, 아니면 NO를 출력한다.

예제

standard input	standard output
6 2 2 3 4 1 0 5	YES
4 2 0 1 3 2	NO

노트

mod는 나머지 연산으로, $a \bmod b$는 a를 b로 나눈 나머지를 뜻한다. 예를 들어, $5 \bmod 3 = 2$이다.

문제 B. 또또 수열 문제야

- 시간 제한 3초

문제

길이 N의 양의 정수로 이루어진 수열 A가 주어질 때, 중복 집합(multiset) M을 다음과 같이 정의하자. 중복 집합이란, 중복된 원소를 허용하는 집합을 의미한다.

$$M = \{A_i \times A_j \mid 1 \leq i, j \leq N\}$$

중복 집합 M의 모든 원소가 주어질 때, 원래의 수열 A를 찾아보자.

입력

첫 번째 줄에 수열 A의 길이인 양의 정수 N이 주어진다. $(1 \leq N \leq 1\,000)$

두 번째 줄에 중복 집합 M의 원소인 $m_1, m_2, \ldots, m_{N^2}$이 공백으로 구분되어 주어진다. $\left(1 \leq m_i \leq 10^{18}\right)$

출력

만약 원래의 수열 A를 구성할 수 있다면, 첫 번째 줄에 YES를 출력하고 두 번째 줄에 수열 A의 원소인 $A_1, A_2, \ldots, A_N$을 공백으로 구분하여 출력한다.

그렇지 않다면 첫 번째 줄에 NO를 출력한다.

가능한 답이 여러 개라면 그중 아무거나 출력한다.

예제

standard input	standard output
3 1 2 2 1 2 1 1 2 4	YES 1 1 2
2 1 3 4 9	NO

문제 C. K 국지

- 시간 제한 1초

문제

당신은 세계적으로 인기 있는 전쟁 시뮬레이션 게임 shake!를 플레이하고 있다. 게임 내에는 1 번부터 N 번까지 번호가 매겨져 있는 N 개의 도시가 있으며 각 도시는 전투력을 가진다. 또한, 서로 다른 두 도시를 양방향으로 잇는 $N-1$ 개의 도로가 있으며 도로를 따라 임의의 도시 간 이동이 가능하다.

당신은 여러 국가를 건국하여 전쟁 시뮬레이션을 진행하려고 한다. 각 국가는 적어도 하나의 도시를 포함해야 하며 각 도시는 정확히 하나의 국가에 포함되어야 한다. 또한, 다른 국가의 도시를 거치지 않으면서 도로를 따라 국가 내 임의의 도시 간 이동이 가능해야 한다.

어떤 국가가 지나치게 강력하여 전쟁에서 쉽게 승리하면 게임의 재미가 떨어지므로 전투력을 균형 있게 분배하는 것이 중요하다. 국가의 전투력은 국가에 포함된 모든 도시의 전투력의 합으로 정의된다. 고민 끝에 당신은 전투력 상한 U 를 정하여 각 국가의 전투력이 U 를 넘지 않고, $k(1 \le k \le N)$ 개의 국가를 건국하여 각 국가의 전투력을 $w_1, w_2, \ldots, w_k$ 라 할 때, $\sum_{i=1}^{k}(U-w_i)^2$ 을 최소로 하는 것이 전투력을 균형 있게 분배하는 방법이라고 결론지었다. 국가의 전투력을 균형 있게 분배하시오.

입력

첫 번째 줄에 도시의 수 N 이 주어진다. $(1 \le N \le 10\,000)$

두 번째 줄부터 N 개의 줄에 걸쳐, i 번 도시의 전투력인 정수 c_i 가 주어진다. $(1 \le c_i \le 100)$

그다음 줄부터 $N-1$ 개의 줄에 걸쳐 도로의 정보가 주어진다. 각 줄에 각 도로가 잇는 두 도시의 번호가 공백으로 구분되어 주어진다.

그다음 줄에 국가의 전투력 상한인 정수 U 가 주어진다. $(\max_{1 \le i \le N} c_i \le U \le 100)$

출력

$\sum_{i=1}^{k}(U-w_i)^2$ 의 최솟값을 출력한다.

예제

standard input	standard output
10 3 2 1 1 6 3 2 1 2 2 2 9 10 4 5 1 3 7 6 9 7 2 9 1 2 10 8 2 10	21
4 2 2 2 2 1 3 3 2 4 3 3	4
4 2 2 2 2 1 3 3 2 4 3 8	0

첫 번째 예제에서, 1번, 6번, 9번 도시를 포함하는 국가와 5번 도시를 포함하는 국가와 나머지 도시를 모두 포함하는 국가를 구성하면 각 국가의 전투력은 $8, 6, 9$이다. $(10-8)^2+(10-6)^2+(10-9)^2=21$이고 이보다 전투력을 균형 있게 분배할 수 없다.

문제 D. 나무핑

- 시간 제한 2초

문제

처음에 1번 노드 하나로 이루어진 트리에 Q개의 쿼리를 적용하려 하며 i번째 쿼리는 다음과 같다.

- q_i w_i: 새로운 $i+1$번 노드와 트리의 q_i번 노드 사이를 w_i의 가중치를 가지는 간선으로 잇는다.

각 쿼리마다 노드를 추가한 후의 트리의 지름을 출력하라. 쿼리는 누적된다.

입력

첫 번째 줄에 쿼리의 개수 Q가 주어진다. $(1 \leq Q \leq 200\,000)$

두 번째 줄부터 Q개의 줄에 걸쳐 쿼리 q_i, w_i가 공백으로 구분되어 주어진다. $\left(1 \leq q_i \leq i; 0 \leq w_i \leq 10^9\right)$

출력

Q개의 줄에 걸쳐 각 쿼리마다 노드를 추가한 후의 트리의 지름을 출력한다.

쿼리는 누적됨에 유의하라.

예제

standard input	standard output
6 1 0 2 1000000000 2 999999999 3 0 5 1 4 2	0 1000000000 1999999999 1999999999 2000000000 2000000002

노트

트리에서 가장 먼 두 노드 간의 거리를 트리의 지름이라고 한다.

문제 E. 마슈 반데드와 마법사의 격자판

- 시간 제한 1초

문제

신각자 선발 시험에서 우수한 성과를 이루어낸 마슈 반데드는, 마지막으로 신각자의 자격을 증명하기 위한 단 하나의 시련만을 남겨두고 있다! 마슈가 받은 마지막 시련의 내용은, 다음과 같다.

"더도 말고 덜도 말고 K 만 같아라"

마법계의 관용어와 속담에 대해서 잘 모르는 당신을 위해 시련을 알기 쉽게 번역하면, 다음과 같다:

- K 개의 동전을 $N \times N$ 크기의 격자판에 배치해서, 이 격자판을 마법사의 격자판으로 만들어라. 격자판의 각 칸에 0 개 이상의 동전을 배치해 격자판에서 모든 인접한 두 칸의 동전 개수의 차이가 정확히 1 일 때, 이 격자판을 **마법사의 격자판**이라고 한다.

물론마슈 또한 시련을 이해하지 못한 것은 마찬가지이므로 시련의 감독관이 한눈을 파는 사이 매우 빠르게 당신에게 도움을 청할 것이다! 마슈를 위해 위의 시련에 맞는 마법사의 격자판을 만들 수 있는지 알려주고, 만들 수 있다면 격자판을 어떻게 만들어야 하는지 알려주어라.

입력

첫 번째 줄에 양의 정수 N 과 K 가 공백으로 구분되어 주어진다. $(1 \leq N \leq 1\,000; 1 \leq K \leq 10^9)$

출력

만약 K 개의 동전을 전부 사용해 마법사의 격자판을 만들 수 없다면 -1을 출력한다.

그렇지 않다면 다음 N 개의 줄에 N 개의 정수를 공백으로 구분하여 출력한다. i 번째 줄의 j 번째 수는 격자판의 i 행 j 열 칸에 배치한 동전의 수를 의미한다.

예제

standard input	standard output
3 7	1 2 1 0 1 0 1 0 1
3 1	-1

첫 번째 예제에서 3×3 격자판에 총 $1+2+1+0+1+0+1+0+1 = 7$ 개의 동전을 올렸고, 모든 인접한 칸은 동전의 개수가 1 차이 나므로, 이 격자판은 마법사의 격자판이다.

두 번째 예제에서 어떤 방법을 사용해도 동전이 1 개 올려진 3×3 격자판이 존재하지 않음을 증명할 수 있다.

문제 F. 경인 국가의 행사

- 시간 제한 2초

문제

2024년, shake! 나라가 건국되었다. 건국을 맞아 shake! 나라는 초대 대통령을 선출하고자 하며, X 후보와 Y 후보, 총 2명의 후보가 당선을 놓고 경쟁하게 된다. shake! 나라는 총 N개의 도시로 이루어져 있으며 i번 도시에는 $A_i (1 \le i \le N)$명의 국민들이 거주하고 있다. 민주주의 국가인 shake! 나라는 대통령을 뽑기 위한 선거 방식을 고민하던 중, 수연이와 현빈이가 각자 아이디어를 내게 되었다.

1. 수연이의 방식
    - shake! 나라의 국민들이 투표를 진행하고, 각 국민들이 거주하는 도시와 관계 없이 표를 합산한다.
    - 최종적으로 더 많은 표를 얻은 후보가 당선된다.
2. 현빈이의 방식
    - shake! 나라의 국민들이 투표를 진행하고, 각 국민들의 표는 국민들이 거주하는 도시의 표에 합계된다.
    - 각 도시마다 더 많은 표를 얻은 후보가 해당 도시에서 승리한다.
    - 최종적으로 더 많은 도시에서 승리한 후보가 당선된다.

예를 들어, 3개의 도시에 각각 801명, 556명, 500명이 살고 있고, 투표 결과가 아래와 같다고 해보자.

후보	1번 도시	2번 도시	3번 도시
X	650	100	180
Y	151	456	320

만약 선거를 수연이의 방식으로 진행하게 된다면 X 후보는 $650 + 100 + 180 = 930$표, Y 후보는 $151 + 456 + 320 = 927$표를 받아 X 후보가 당선된다. 반면 현빈이의 방식으로 선거를 진행하게 되면, X 후보는 1번 도시에서만 승리하고, Y 후보는 $2, 3$번 도시에서 승리하여 Y 후보가 당선된다.

모든 국민이 투표에 참여하며 하나의 후보에게만 투표를 할 수 있을 때 선거 진행을 수연이의 방식으로 하면 X 후보가, 현빈이의 방식으로 하면 Y 후보가 승리하는 경우를 찾아보자. 단, 투표가 동률이라면 두 후보 모두 패배한 것으로 생각한다.

입력

첫 번째 줄에 테스트 케이스의 개수 T가 주어진다.$(1 \le T \le 100\,000)$

다음 줄부터 각 테스트 케이스의 정보가 주어진다. 하나의 테스트 케이스는 두 개의 줄로 이루어져 있으며, 첫 번째 줄에 shake! 나라의 도시 수 N이 주어진다. $(1 \le N \le 10^6)$

두 번째 줄에 각 도시에 사는 국민의 수를 나타내는 정수 $A_1, A_2, \ldots, A_N$이 공백으로 구분되어 주어진다. $(1 \le A_i \le 10^9)$

모든 테스트 케이스의 N의 합은 10^6을 넘지 않는다.

출력

각 테스트 케이스마다 shake! 나라의 선거 진행을 수연이의 방식으로하면 X 후보가, 현빈이의 방식으로하면 Y 후보가 승리하는 경우가 존재한다면, 첫 번째 줄에 YES를 출력하고 두 번째 줄에 1번부터 N번 도시까지 X 후보가 얻은 표의 수를, 세 번째 줄에 Y 후보가 얻은 표의 수를 공백으로 구분하여 출력한다.

그렇지 않다면 첫 번째 줄에 NO를 출력한다.

가능한 경우가 여러 개라면, 그중 아무거나 출력한다.

예제

standard input	standard output
2 3 801 556 500 3 1 1 2	YES 650 100 180 151 456 320 NO

문제 G. 원소 합치기

- 시간 제한 1초

문제

음이 아닌 정수 N 개로 이루어진 배열 $A = [A_1, \ldots, A_N]$ 이 주어진다. 배열 A 에 다음 연산을 K 번 진행한다.

- 배열에 남아 있는 원소 중에서 인접한 두 원소를 선택해, 해당 원소들에 대해 bitwise OR 연산을 적용하고 그 결괏값으로 두 원소를 대체한다. 다시 말해 현재 배열의 길이를 L 이라 하면, 정수 $i(1 \le i < L)$ 를 선택해 $[A_1, A_2, \ldots, A_i, A_{i+1}, \ldots, A_L]$ 을 $[A_1, A_2, \ldots, A_i | A_{i+1}, \ldots, A_L]$ 로 변경한다. 이 연산을 진행한 뒤 배열의 길이는 1 만큼 줄어든다. K 번의 연산을 진행한 이후 배열에 남은 $N - K$ 개의 원소를 모두 bitwise AND 연산한 값의 최댓값을 구해보자.

bitwise AND와 bitwise OR 연산에 대한 설명은 노트를 참고하라.

입력

첫 번째 줄에 N 과 K 가 공백으로 구분되어 주어진다. $(2 \le N \le 500\,000; 1 \le K \le N - 1)$

두 번째 줄에 배열 A 의 원소 $A_1, A_2, \ldots, A_N$ 이 공백으로 구분되어 주어진다. $\left(0 \le A_i \le 10^9\right)$

출력

총 K 번의 연산을 진행한 후 배열에 남은 $N - K$ 개의 원소를 모두 bitwise AND 연산한 값의 최댓값을 출력한다.

예제

standard input	standard output
5 3 1 4 2 5 1	5

위 예제의 최댓값을 구하기 위한 총 3 번의 연산 순서의 예시는 다음과 같다.

- $[1, 4, 2, 5, 1] \Rightarrow [1, 4|2, 5, 1] = [1, 6, 5, 1]$
- $[1, 6, 5, 1] \Rightarrow [1|6, 5, 1] = [7, 5, 1]$
- $[7, 5, 1] \Rightarrow [7, 5|1] = [7, 5]$

노트

bitwise 연산자들은 비트 단위로 연산을 시행한다.

- bitwise AND 연산(&)은 두 수의 각 비트마다 다음과 같은 연산을 진행한다.
  - 같은 자릿수의 비트를 비교해 두 비트 다 1 일 때만 1, 나머지의 경우는 0 이다.
  - 다음은 예시이다.
$$\begin{array}{rcl} 0101_2 & = & 5 \\ \& \, 0011_2 & = & 3 \\ \hline 0001_2 & = & 1 \end{array}$$
- bitwise OR 연산(|)은 두 수의 각 비트마다 다음과 같은 연산을 진행한다.
  - 같은 자릿수의 비트를 비교해 두 비트 다 0 일 때만 0, 나머지의 경우는 1 이다.
  - 다음은 예시이다.
$$\begin{array}{rcl} 0110_2 & = & 6 \\ | \, 1100_2 & = & 12 \\ \hline 1110_2 & = & 14 \end{array}$$

문제 H. 데이터를 추가해 주세요.

- 시간 제한 1초

문제

현빈이는 다이나믹 프로그래밍의 고수가 되고 싶어서 매일매일 백준에서 다이나믹 프로그래밍 문제를 푼다. 오늘도 어김없이 다이나믹 프로그래밍 문제를 찾던 현빈이는 다음과 같은 문제를 발견했다.

> $N \times N$ 크기의 미로가 있다. 미로의 가장 왼쪽 위는 $(1, 1)$ 칸이고 가장 오른쪽 아래는 (N, N) 칸이며 각 칸은 빈칸 또는 벽으로 구성된다. 시작점인 $(1, 1)$ 칸에서 출발해 도착점인 (N, N) 칸에 도달하려 하는데 다음과 같은 규칙에 따라 이동하려 한다.
>
> - 현재 위치가 (r, c) 칸 일 때, $(r, c + 1)$ 이 빈칸이라면, $(r, c + 1)$ 칸으로 이동할 수 있다.
> - 현재 위치가 (r, c) 칸 일 때, $(r + 1, c)$ 이 빈칸이라면, $(r + 1, c)$ 칸으로 이동할 수 있다.
> - 현재 위치가 (r, c) 칸 일 때, $(r + 1, c)$ 칸과 $(r + 2, c)$ 칸이 빈칸이라면, $(r + 2, c)$ 칸으로 한 번에 이동할 수 있다.
>
> 미로의 시작점과 도착점은 항상 빈칸임이 보장될 때 시작점에서 출발해 도착점에 도달하는 이동 방법의 개수를 M 으로 나눈 나머지를 출력한다. 단, 시작점에서 출발해 도착점에 도달할 수 없다면 -1 을 출력한다.

문제를 보자마자 너무 간단한 다이나믹 프로그래밍 문제라고 생각한 현빈이는 신나서 1분 만에 코드를 작성했다. 현빈이가 작성한 코드의 의사코드는 다음과 같다.

```
function solve(N, M, Map):
    DP = [N + 1][N + 1] # 0 으로 초기화된 (N + 1) * (N + 1) 크기의 DP 배열 선언
    DP[1][1] = 1 % M # 시작점 설정

    # 빈칸은 '.' 벽은 '#'
    for r from 1 to N:
        for c from 1 to N:
            if Map[r][c] is '#':
                continue

            if r + 1 <= N and Map[r + 1][c] is '.':
                DP[r + 1][c] = (DP[r + 1][c] + DP[r][c]) % M # 이동 방법 1

            if c + 1 <= N and Map[r][c + 1] is '.':
                DP[r][c + 1] = (DP[r][c + 1] + DP[r][c]) % M # 이동 방법 2

            if r + 2 <= N and (Map[r + 1][c] and Map[r + 2][c]) is '.':
                DP[r + 2][c] = (DP[r + 2][c] + DP[r][c]) % M # 이동 방법 3

    if DP[N][N] is 0:
        DP[N][N] = -1

    return DP[N][N]
```

현빈이는 위와 같은 코드를 제출하고 **맞았습니다!!** 를 받았지만, 이를 옆에서 지켜보던 민우는 도착점에 도달할 수 있으며 이동 방법의 개수가 M 의 배수라면 0 을 출력해야 하는데 현빈이의 코드는 -1 을 출력한다며 틀린 코드라고 지적했다. 현빈이의 코드를 틀리게 하는 데이터를 추가해 보자.

입력

첫 번째 줄에 양의 정수 M이 주어진다. ($1 \leq M \leq 314\,159\,265$)

출력

첫 번째 줄에 미로의 크기를 나타내는 정수 N을 출력한다. ($2 \leq N \leq 60$)

다음 N개의 줄에 걸쳐 .과 #로만 이루어진 미로를 나타내는 문자열을 출력한다. i번째 줄의 j번째 문자는 (i, j)칸의 상태를 나타내며, .이라면 빈칸을, #이라면 벽을 의미하고, 시작점과 도착점은 항상 빈칸이어야 한다.

예제

standard input	standard output
1	2 .. #.
2	3 .## .## ...

문제 I. 꽃바구니

- 시간 제한 2초

문제

N 개의 꽃과 M 개의 바구니로 꽃바구니를 만들려 한다. 꽃바구니는 하나의 바구니와 하나 이상의 꽃으로 만들어진다. 꽃꽂이는 심오해서 꽃의 궁합을 잘 생각해야 한다. **궁합**은 꽃 i와 꽃 j에 대해 정수 C_{ij}로 정의된다. 꽃바구니의 **아름다움**은 꽃바구니에 포함된 꽃 간 궁합의 합이다. 다시 말해, 어떤 꽃바구니 S의 아름다움은 다음과 같다.

$$\sum_{i,j \in S;\ i \le j} C_{ij}$$

예를 들어 꽃 $1, 2$로 꽃바구니를 만든다면 그 아름다움은 $C_{11} + C_{12} + C_{22}$이다.

다음 규칙을 지키면서 꽃바구니의 아름다움의 합을 최대화해 보자.

- 각 꽃은 하나의 바구니에만 포함되거나 어떤 바구니에도 포함되지 않는다.
- 각 꽃바구니에 포함된 꽃의 크기 A_i의 합이 바구니의 크기 B_j를 넘을 수 없다.
- 각 꽃바구니에 포함된 꽃의 가치 V_i의 합이 판매할 가치 K를 넘으면 안 된다. 즉, 손해 보며 팔 수는 없다.

입력

s 첫 번째 줄에 꽃의 개수 N, 바구니의 개수 M, 판매할 가치 K가 공백으로 구분되어 주어진다. $(1 \le N \le 15; 1 \le M \le 200; 1 \le K \le 10^7)$

두 번째 줄에 꽃의 크기를 나타내는 정수 $A_1, A_2, \ldots, A_N$이 공백으로 구분되어 주어진다. $(1 \le A_i \le 10^7)$

세 번째 줄에 바구니의 크기를 나타내는 정수 $B_1, B_2, \ldots, B_M$이 공백으로 구분되어 주어진다. $(1 \le B_i \le 10^7)$

네 번째 줄에 꽃의 가치를 나타내는 정수 $V_1, V_2, \ldots, V_N$이 공백으로 구분되어 주어진다. $(1 \le V_i \le 10^7)$

다섯 번째 줄부터 N개의 줄에 걸쳐 각 줄마다 N개의 정수가 공백으로 구분되어 주어진다. 그중 i번째 줄의 j번째 수는 꽃 i와 j의 궁합 C_{ij}이다. $(|C_{ij}| \le 10^7; C_{ij} = C_{ji})$

출력

완성한 꽃바구니의 아름다움의 최대 합을 출력한다.

예제

standard input	standard output
1 1 1 2 9 1 6	6
5 2 10 2 1 2 8 3 9 2 17 1 7 3 9 29 -6 43 81 73 -6 97 -4 57 79 43 -4 39 90 -3 81 57 90 97 66 73 79 -3 66 29	290

문제 J. 구간이 이븐하지 않아요.

- 시간 제한 1초

문제

길이가 같은 두 수열 s와 t가 다음의 연산을 원하는 만큼 반복해 서로 같아질 수 있다면, 두 수열은 서로 **이븐하다**.

- $1 \le i < j \le |s|$를 만족하는 두 양의 정수 i와 j를 골라 s_i와 s_j의 값을 바꾼다.

길이 N의 수열 A가 주어진다. $A[l, r] = [A_l, A_{l+1}, \ldots, A_r]$을 A의 l번째 원소부터 r번째 원소까지만을 포함하는 연속된 부분 수열이라고 할 때, 다음 조건을 만족시키는 $A[l, r]$의 최대 길이를 구해보자.

$m = \left\lfloor \frac{l+r}{2} \right\rfloor$에 대하여, $A[l, m]$과 $A[m+1, r]$이 서로 이븐하다.

입력

첫 번째 줄에 수열 A의 길이인 정수 N이 주어진다. $(1 \le N \le 5\,000)$

두 번째 줄에 수열 A의 원소를 나타내는 N개의 정수 $A_1, A_2, \ldots, A_N$이 공백으로 구분되어 주어진다. $\left(1 \le A_i \le 10^9\right)$

출력

연속된 부분 수열 $A[l, r]$을 절반으로 나눠 얻은 두 수열이 서로 이븐할 때, $A[l, r]$의 최대 길이를 출력한다.

만약 그러한 연속된 부분 수열이 존재하지 않는다면, 0을 출력한다.

예제

standard input	standard output
3 1 1 1	2
5 1 2 3 2 3	4

마지막 페이지입니다.