UCPC 2026

전국 대학생 프로그래밍 대회 동아리 연합
여름 대회 2026

Preliminaries

Official Solutions

예선 해설

2026년 6월 27일, 14:00 - 17:00

주최

전국 대학생 프로그래밍 대회 동아리 연합

후원

Jane Street
HRT hudson river trading
HYUNDAI MOBIS
CITADEL | CITADEL Securities
MOLOCO
ASTEROMORPH
NEXON
STARTLINK

김동현(kdh9949) 김영현(kipa00) 나정휘(jhnah917) 정재호(cubic)

예상 난이도

Preliminaries – 예선대회

문제		의도한 난이도	출제자
A	수학 체육	**Easy**	fefe
B	Bakejoon Online Judge	**Medium**	hibye1217
C	Hat Trick	**Easy**	79brue
D	괄호 게임	**Easy**	pk661
E	공항	**Medium**	dadas08
F	괄호와 쿼리	**Challenging**	pyb1031
G	레이저 타워	**Hard**	azberjibiou
H	빈 상자	**Hard**	dadas08
I	논문 공장	**Medium**	leejseo
J	어려운 정렬 문제	**Medium**	_junick
K	상인	**Hard**	lunarlity
L	확률성 정렬	**Hard**	woohyun_jng

문제 A. 체육은 수학과목입니다 3

출제자 신민철 (fefe)

민철이가 오전 9시까지 남은 $60 - M$분 동안 이동할 수 있는 최대 거리는 $V(60 - M)$미터이다.

민철이는 오전 9시 정각에 도착해도 지각이므로, 학교까지의 거리 D가 이 값보다 작아야 한다.

따라서 $D < V(60 - M)$이면 `run`을, 그렇지 않으면 `late`를 출력하면 된다.

문제 B. Bakejoon Online Judge

출제자 이성현 (hibye1217)

시각 X에 모든 빵을 제출할 수 있다면, 시각 $x > X$에도 모든 빵을 제출할 수 있다. 단조성이 성립하므로 답을 기준으로 이진 탐색을 돌려보자.

빵을 제출할 시각 X를 고정하면 문제를 다음과 같이 변환할 수 있다.

- i번째 빵은 시각 $\max(P_i, X - T_i - F_i)$부터 구울 수 있다.
- i번째 빵은 굽는 데 T_i시간이 걸린다.

변환된 문제는 구울 수 있는 빵을 즉시 굽는 그리디를 사용하여 해결할 수 있다. 증명은 빵을 굽기 시작할 수 있는 시각의 오름차순대로 빵을 굽는 것이 최적임을 Exchange Argument로 보이면 된다.

문제 C. Hat Trick

출제자 반딧불 (79brue)

조수는 i번 위치에 i번 관객이 가져간 반대 색의 모자를 넣으면 된다. 관객들이 빨간 모자 r개, 파란 모자 $n-r$개를 가져갔다고 해 보자. 이때 조수에게 필요한 모자의 수는 빨간 모자 $n-r$개, 파란 모자 r개로, 관객이 가져간 모자와 조수에게 필요한 모자를 합하면 각 색의 모자가 정확히 n개씩 쓰임을 알 수 있다. 따라서 이 방식이 가능하다.

실제 프로그램을 구현할 때는 조수와 마술사 모두 입력 문자열에서 `R`과 `B`를 반대로 출력하면 된다.

다음은 Python으로 구현한 정해 코드이다.

```
input()
input()
print(''.join(map(lambda c: 'R' if c == 'B' else 'B', input())))
```

문제 D. 괄호 게임

출제자 손재원 (pk661)

$A_1, A_2, \cdots, A_N$을 내림차순으로 정렬한 것을 $B_1 \geq B_2 \geq \cdots \geq B_N$이라 하자. $N = 2M$이라고 할 때, 정답은

$$X = B_1 - B_2 + B_3 - B_4 + \cdots + B_{2M-1} - B_{2M}$$

이다. 증명은 다음과 같다.

Alice는 X 이상의 점수를 보장할 수 있다.

Alice는 내림차순으로 배열한다고 하자. 올바른 괄호 문자열에서 첫 $2k-1$개의 괄호 중 '('는 최소 k개 있어야 한다. Bob이 올바른 괄호 문자열에서 k번째 '('의 인덱스를 i_k라고 하면, $B_{i_k} \geq B_{2k-1}$이다. 게임의 점수는

$$\sum_{i \in \{i_1, i_2, \cdots, i_M\}} B_i - \sum_{i \notin \{i_1, i_2, \cdots, i_M\}} B_i$$

이고, 이는 $\{i_1, i_2, \cdots, i_M\} = \{1, 3, \cdots, 2M-1\}$일 때 최솟값 X를 가진다. 즉, Alice는 X 이상의 점수를 보장할 수 있다.

Bob은 X 이하의 점수를 보장할 수 있다.

$A_1, A_2, \cdots, A_N$에서 k번째로 큰 값의 위치를 p_k라고 하자. Bob은 다음과 같은 규칙으로 S를 정한다고 하자.

1. $\min(p_1, p_2)$번째는 '(', $\max(p_1, p_2)$번째는 ')',
2. $\min(p_3, p_4)$번째는 '(', $\max(p_3, p_4)$번째는 ')',

⋮

M. $\min(p_{2M-1}, p_{2M})$번째는 '(', $\max(p_{2M-1}, p_{2M})$번째는 ')'이다.

이때 게임의 점수는 최대 $(B_1 - B_2) + (B_3 - B_4) + \cdots + (B_{2M-1} - B_{2M}) = X$이다. 임의의 접두사에 대하여 '('의 개수가 ')'의 개수 이상임을 알 수 있다. 그러므로 위 규칙으로 정한 S는 올바른 괄호 문자열이고, Bob은 X 이하의 점수를 보장할 수 있다.

문제 E. 공항

출제자 양승민 (dadas08)

공항을 하나도 사용하지 않는 경우, MST 알고리즘을 이용해 해결 가능하다. 하나 이상 사용하는 경우, 새로운 정점 $N+1$를 만든 다음, 각 i $(1 \le i \le N)$ 에 대해 i번 정점과 $N+1$번 정점을 잇는 비용 A_i의 간선을 추가한 그래프에서의 MST를 구하면 된다.

각 경우 중 최솟값이 답이다. 시간복잡도는 $O((N+M)\log N)$ 이다.

문제 F. 괄호와 쿼리

출제자 박영빈 (pyb1031)

$d_i = b_i - a_i$ 라고 정의하자.

우선 하나의 구간 $[l, r]$을 올바른 괄호 문자열로 채웠을 때 총점의 최댓값을 구하는 방법을 생각해 보자.

모든 위치를 (로 채웠다고 하자. 이때 얻는 점수는 $\sum_{i=l}^{r} a_i$ 이다. 위치 i를)로 바꾸면 점수가 d_i만큼 증가한다.

(을 1,)를 -1로 보았을 때 올바른 괄호 문자열이 되기 위한 조건은 모든 누적합이 0 이상이고 전체 합이 0인 것이다. 점수를 최대화 하기위한 그리디 전략을 생각해보자 d_i가 큰 위치부터)로 바꾸는 것을 시도하고, 바꾼 뒤에도 모든 누적합이 0 이상이면 실제로 바꾼다. d_i가 같은 위치들의 순서는 아무렇게나 정해도 된다.

이 방법이 최적임을 보이자.

)로 바꾼 위치의 집합을 S라 하자. 구간의 왼쪽부터 t개 위치를 보았을 때 누적합이 음이 아닐 조건은

$$|S \cap [l, t]| \le \left\lfloor \frac{t - l + 1}{2} \right\rfloor$$

이고, 전체 합이 0이려면 $|S| = (r - l + 1)/2$여야 한다.

위 조건을 만족하는 집합을 가능하다고 하자. 가능한 두 집합 X, Y에 대해 $|X| < |Y|$이면, 어떤 $y \in Y \setminus X$가 존재하여 $X \cup \{y\}$도 가능하다. 그렇지 않다면 $Y \setminus X$의 각 원소는 X가 제한을 꽉 채운 어떤 prefix 안에 있다. 이 중 가장 큰 prefix를 P라 하면 $Y \setminus X \subseteq P$이고

$$|Y \cap P| \le |X \cap P|,$$

따라서 $|Y \setminus X| \le |X \setminus Y|$가 되어 $|Y| \le |X|$인 모순이다.

이제 d_i가 큰 순서대로 보면서 추가 가능한 위치만 S에 넣는다. 위 성질에 의해 최종적으로 정확히 절반의 위치가 선택된다.

또 지금까지의 그리디 선택을 모두 포함하는 최적해 O가 있다고 하자. 그리디가 새 위치 x를 선택했는데 $x \notin O$라면, 어떤 $y \in O$가 존재하여

$$(O \setminus \{y\}) \cup \{x\}$$

도 가능하다. 이때 y는 아직 처리되지 않은 위치이므로 $d_x \ge d_y$이고, 따라서 점수는 감소하지 않는다. 이를 반복하면 최적해를 그리디 해로 바꿀 수 있으므로 그리디가 최적이다.

이제 쿼리 (l, r, k)에서, 최적해 중 k번째 괄호가 (인 것이 존재하는지 판별하자.

그리디에서 k를 제외한 $d_i \ge d_k$ 인 모든 위치를 먼저 처리하고, 그 뒤에 k를 처리한다고 생각하자. 즉, 같은 d를 가지는 위치들 중에서는 k를 가장 마지막에 처리한다.

이때 k를)로 바꾸는 데 실패하면, 그리디를 끝까지 진행하여 k가 (인 최적해를 얻을 수 있다. 이를 양 방향에 대해 빠르게 판별할 수 있다면 답을 구할 수 있다.

이를 빠르게 판별하는 방법을 생각하자. 다음 수열의 누적합을 S라고 하자.

$$c_i = \begin{cases} -1 & (d_i \ge d_k), \\ 1 & (d_i < d_k), \end{cases} \qquad S_0 = 0, \qquad S_j = \sum_{i=1}^{j} c_i.$$

여기서는 $i = k$인 경우도 $c_k = -1$로 둔다. 즉, S는 k를 포함하여 $d_i \ge d_k$인 모든 위치를)로 바꾼 상태를 나타낸다.

먼저 $[k+1, r]$에 있는 위치를 생각하자. 이 구간의 어떤 위치를)로 바꾸는 데 실패했다면, 그보다 왼쪽의 k까지)로 바꾸었을 때에는 누적합이 더 작아지므로 k를 바꾸는 것도 불가능하다. 따라서 k를 바꿀 수 있으려면 오른쪽의 모든 해당 위치를 바꿀 수 있어야 한다.

이제 $[l, k-1]$만 보자. 이 구간에서는 시도하는 순서와 상관없이 최대한 많은 (가)로 바뀐다. 가중치가 1과 0밖에 없는 상황에서 위의 그리디 전략을 사용한다고 보면 된다.

$m_1 = S_{l-1} - \min_{l-1 \le t \le k-1} S_t$ 라고 하자. 하나의 -1을 1로 바꾸면 그 위치 이후의 누적합이 모두 2 증가하므로, 시도의 실패횟수는 $q = \lceil \frac{m_1}{2} \rceil$ 이다.

$$m_2 = S_{l-1} - \min_{k-1 \le t \le r} S_t$$

라고 하자. 왼쪽에서 실패한 q개의 위치는 모두 k보다 왼쪽에 있으므로, $k-1$ 이후의 모든 누적합을 $2q$만큼 증가시킨다. 따라서 k와 오른쪽의 모든 해당 위치까지)로 바꾼 상태가 가능할 필요충분조건은 $-m_2 + 2q \ge 0$ 이다.

따라서 최적해 중 k번째 괄호가 (인 것이 존재할 필요충분조건은 $m_2 > 2\lceil \frac{m_1}{2} \rceil$ 이다.

즉 오프라인 쿼리와 레이지 세그먼트 트리로 누적합을 관리해 각 쿼리를 빠르게 처리할 수 있다.

문제 G. 레이저 타워

출제자 장근영 (azberjibiou)

모든 2^N 개의 방향 배치에 대한 2^S 의 합을 먼저 구하고, 마지막에 2^N 으로 나누자. 높이 k 의 가로 띠를 보면, 왼쪽으로 쏘는 타워들이 덮는 부분은 항상 어떤 prefix이고, 오른쪽으로 쏘는 타워들이 덮는 부분은 항상 어떤 suffix이다. 따라서 덮이지 않은 부분이 존재한다면 항상 하나의 구간이다. 즉, 위에서 아래로 내려오며 생각하면 아직 덮이지 않은 부분은 여러 조각으로 갈라지지 않고, 하나의 빈 구간으로만 남는다.

재귀 함수 $f(s,e,\ell,a)$ 를 다음과 같이 정의한다. 현재 남아 있는 빈 구간은 $(s-1,e+1)$ 이고, 높이 ℓ 이상은 이미 처리되었으며, 지금까지 확정적으로 덮인 넓이가 a 라고 하자. 이 상태에서 가능한 나머지 방향 배치들이 만드는 2^S 의 합을 $f(s,e,\ell,a)$ 로 계산한다. 초기 상태는 $f(1,N,N+1,0)$ 이다.

현재 $[s,e]$ 안에서 가장 높은 타워의 높이를 M, 위치를 p 라고 하자. 높이 배열은 순열이므로 이는 유일하다. 그러면 높이 $M+1,M+2,\dots,\ell-1$ 인 타워들은 모두 현재 빈 구간의 바깥에 있다. 이 타워들 중 하나라도 빈 구간 쪽으로 쏘면, 그 순간 현재 빈 구간 전체가 해당 높이 이하에서 전부 덮인다. 따라서 이 경우들은 재귀로 내려갈 필요 없이 한 번에 더할 수 있다.

구체적으로 높이 k 인 타워가, 높이 $M+1,\dots,\ell-1$ 중 처음으로 빈 구간 쪽으로 쏘는 타워라고 하자. 더 높은 타워들의 방향은 모두 빈 구간의 반대쪽으로 고정되고, 높이 k 의 타워는 빈 구간 쪽으로 고정된다. 그보다 낮은 $k-1$ 개의 타워들의 방향은 더 이상 넓이에 영향을 주지 않으므로 자유롭게 정할 수 있다. 현재 빈 구간의 폭은 $e-s+2$ 이므로, 이때의 기여는

$$2^{k-1}\cdot 2^{a+(e-s+2)k}=2^{a-1+(e-s+3)k}$$

이다. 이를 $k=M+1$ 부터 $\ell-1$ 까지 더하면 다음 등비수열이 된다.

$$2^{a-1+(e-s+3)(M+1)}\cdot\frac{2^{(e-s+3)(\ell-M-1)}-1}{2^{e-s+3}-1}$$

따라서 위 값을 현재 상태의 답에 더한다.

이제 높이 $M+1,\dots,\ell-1$ 인 타워들이 모두 빈 구간의 반대쪽으로 쏜 경우만 남는다. 이 경우 빈 구간은 높이 M 까지 살아남고, 다음으로 영향을 주는 타워는 위치 p 의 타워이다. 이 타워가 오른쪽으로 쏘면 빈 구간의 오른쪽 부분이 덮이고, 남은 빈 구간은 $(s-1,p)$ 가 된다. 반대로 왼쪽으로 쏘면 빈 구간의 왼쪽 부분이 덮이고, 남은 빈 구간은 $(p,e+1)$ 가 된다. 따라서 다음 두 재귀 호출을 하면 된다.

$$f(s,p-1,M,a+(e-p+1)M),\qquad f(p+1,e,M,a+(p-s+1)M)$$

기저 사례는 $s=e+1$ 일 때이다. 이때 빈 구간의 폭은 1이고, 그 안에는 타워가 없다. 남은 높이 $1,\dots,\ell-1$ 의 타워들 중 처음으로 빈 구간 쪽으로 쏘는 타워의 높이를 k 라고 하면, 기여는

$$2^{k-1}\cdot 2^{a+k}=2^{a+2k-1}$$

이다. 이를 모두 더하면

$$2^{a+1}\cdot\frac{2^{2\ell-2}-1}{3}$$

이고, 끝까지 아무도 빈 구간 쪽으로 쏘지 않는 경우의 기여 2^a 도 더한다.

이 재귀는 각 방향 배치를 정확히 한 번씩 센다. 현재 빈 구간 바깥의 높은 타워가 빈 구간을 없애는 경우는 그 높이에 따라 등비수열로 처리하고, 그런 타워가 없다면 현재 빈 구간 안의 최고 타워가 빈 구간을 왼쪽 또는 오른쪽으로 줄인다. 각 타워는 한 번만 현재 구간의 최댓값으로 선택되므로 재귀 호출 수는 $O(N)$ 이다. 구간 최댓값은 세그먼트 트리로 $O(\log N)$ 에 찾을 수 있으므로 전체 시간 복잡도는 $O(N\log N)$ 이다. 마지막에 계산된 합을 2^N 으로 나누면 원하는 기댓값을 얻는다.

문제 H. 빈 상자

출제자 양승민 (dadas08)

1 풀이

원소 i가 상자 안에서 가지는 위치를 다음과 같이 정의하자.

$$p_i = \begin{cases} -i & (B_i = 0), \\ i & (B_i = 1). \end{cases}$$

그러면 어떤 시점에서 상자 안 원소들의 앞뒤 순서는 p_i가 작은 순서와 같다.

값이 제거될 때에는 가장 작은 값이 제거되고, 같은 값이 여러 개라면 가장 앞에 있는 값이 제거된다. 따라서 원소의 우선순위는 (A_i, p_i)를 사전순으로 크게 비교한 순서이다. 즉 값이 클수록, 값이 같다면 더 뒤에 있을수록 오래 남는다.

시점 t에서 용량이 X라면 상자에는 지금까지 들어온 원소 중 우선순위가 가장 높은 X개가 남는다. 관찰되는 값은 그 원소들 중 p_i가 가장 작은 원소의 값이다.

시점 t에서 용량 X일 때 관찰되는 값을 $f_t(X)$라 하자. 용량이 하나 증가하면 우선순위가 더 낮은 원소 하나만 추가되므로

$$f_t(1) \geq f_t(2) \geq f_t(3) \geq \cdots$$

가 항상 성립한다.

따라서 어떤 두 용량이 구별되지 않는다면 그 사이의 모든 용량도 구별되지 않는다. 즉 처음으로 구별되지 않는 경우는 항상 인접한 두 용량 $X, X+1$에서 발생한다. 그러므로 모든 X에 대해 용량 X와 $X+1$이 구별 가능한지만 확인하면 충분하다.

전체 원소를 (A_i, p_i)가 큰 순서대로 정렬하고, 그 순위를 q_i라고 하자.

어떤 시점에서 삽입된 원소들을 q_i 순으로 보면 용량 X일 때 남는 원소들은 앞의 X개이다. 이 중 실제로 앞에서 보이는 원소는 p_i가 가장 작은 원소이다.

따라서 q_i 순으로 보면서 p_i의 새로운 최솟값을 만드는 원소들만 중요하다. 이러한 원소들을 후보열이라고 하자. 후보열에서는 q_i가 증가할수록 p_i가 엄격히 감소한다.

후보열에서 인접한 두 원소를 u, v라 하자. 용량이 증가하여 v가 처음 포함되는 순간 맨 앞 원소는 u에서 v로 바뀐다.

이때

$$A_v < A_u$$

라면 관찰되는 값도 달라지므로 해당 용량 X와 $X+1$은 구별 가능하다.

후보열은 ordered set으로 관리한다.

새 원소 x를 삽입할 때 q_x보다 작은 마지막 후보를 pre라고 하자.

만약

$$p_{pre} < p_x$$

라면 x보다 우선순위가 높은 원소 중 이미 더 앞에 있는 원소가 있으므로 x는 후보가 될 수 없다.

그렇지 않다면 x를 후보열에 넣고, x보다 뒤에 있으면서 $p_i > p_x$인 후보들은 모두 제거한다.

각 원소는 후보열에 최대 한 번 삽입되고 최대 한 번 제거되므로 전체 변화 횟수는 $O(N)$이다.

후보열에서 인접한 쌍 (u, v)가 유지되는 동안을 생각하자.

시점 t에서 v가 현재까지 삽입된 원소 중 몇 번째 우선순위인지는

$$\mathrm{rank}_t(v) = \#\{\, i \le t \mid q_i \le q_v \,\}$$

이다.

따라서 이 쌍이 담당하는 용량은

$$X = \mathrm{rank}_t(v) - 1$$

이다.

쌍이 유지되는 동안 $\mathrm{rank}_t(v)$는 단조 증가하므로 표시되는 용량들은 하나의 연속 구간이 된다.

rank는 펜윅 트리로 관리한다. 시간 순서대로 원소를 삽입하면서 위치 q_i에 1을 더하면

$$\mathrm{rank}_t(v) = \sum_{j=1}^{q_v} \mathrm{BIT}[j]$$

로 계산할 수 있다.

후보열의 인접한 쌍이 새로 생기면 현재 rank를 시작점으로 저장하고, 쌍이 사라질 때 다시 rank를 계산하여 끝점을 얻는다. $A_v < A_u$인 경우에만 해당 구간을 차분 배열에 표시한다.

마지막으로 차분 배열을 누적하면 각 X에 대해 용량 X와 $X+1$이 구별 가능한지 알 수 있다. 처음으로 표시되지 않은 X가 답이며, 모두 표시되었다면 답은 N이다.

시간복잡도는 정렬, ordered set, 펜윅 트리 연산을 합쳐

$$O(N \log N)$$

이다.

문제 I. 논문 공장

출제자 이종서 (leejseo)

목표로 하는 각 키워드의 최종 사용 횟수를 X라고 하고, 키워드 i와 $i+1$을 함께 사용한 횟수를 x_i라고 하자. 그러면 $A_1 + x_1 = X$, $2 \le i \le N-1$에 대해 $A_i + x_{i-1} + x_i = X$, 그리고 $A_N + x_{N-1} = X$를 만족해야 한다.

$x_1, x_2, \cdots, x_{N-1}$의 값을 차례로 결정해보자. 먼저 $x_1 = X - A_1$으로 정해진다. 그다음 $x_2 = X - A_2 - x_1 = A_1 - A_2$로 정해지고, $x_3 = X - A_3 - x_2 = X - A_1 + A_2 - A_3$으로 정해진다.

즉, $P_i = A_1 - A_2 + A_3 - \cdots + (-1)^{i+1} A_i$라 하면,

$$x_i = \begin{cases} X - P_i & i \equiv 1 \pmod 2, \\ P_i & i \equiv 0 \pmod 2 \end{cases}$$

이다. 따라서 핵심은 X를 정하는 것과, 모든 x_i가 0 이상인지 확인하는 것이다.

N이 홀수인 경우: 마지막 조건에서 $X = P_N$으로 정해진다. 따라서 $X = P_N$으로 두고 모든 $x_i \ge 0$인지 확인하면 된다. 모두 만족하면 가능하고, 하나라도 만족하지 않으면 불가능하다.

N이 짝수인 경우: 마지막 조건을 살펴보자. 위 식에 의해 $x_{N-1} = X - P_{N-1}$이고, 마지막 조건은 $A_N + x_{N-1} = X$이다. 이를 대입하면 $A_N + X - P_{N-1} = X$이므로 $P_{N-1} = A_N$이어야 한다. 이는 곧 $P_N = P_{N-1} - A_N = 0$과 같다.

따라서 $P_N \neq 0$이면 불가능하다.

이제 $P_N = 0$이라고 하자. 이때 X는 하나로 정해지지 않는다. 홀수 i에 대해서는 $x_i = X - P_i$이므로, X를 충분히 크게 잡으면 항상 $x_i \ge 0$이 되게 할 수 있다. 반면 짝수 i에 대해서는 $x_i = P_i$로 X와 무관하게 정해진다.

따라서 모든 짝수 i $(1 \le i < N)$에 대해 $P_i \ge 0$이면 가능하고, 하나라도 그렇지 않으면 불가능하다.

이를 구현하면 각 테스트 케이스를 $O(N)$ 시간에 해결할 수 있다.

문제 J. 어려운 정렬 문제

출제자 김민준 (_junick)

왼쪽에서 p번째 구간의 초기 최솟값과 최댓값을 각각 원소에 대응시켜주자. 각 구간에 대한 제약 $[m_p, M_p]$를 각 원소에다 적용시킬 수 있다. i번째 위치에 대하여 값의 범위를 $[m_i, M_i]$라고 한다.

이제 최종 수열이 비내림차순이어야 하므로 앞쪽 위치의 값이 뒤쪽 위치의 값보다 커지면 안 된다. 이를 위해 왼쪽에서 오른쪽으로 보면서 각 위치의 하한을 전파하고, 오른쪽에서 왼쪽으로 보면서 각 위치의 상한을 전파한다. 구체적으로 다음과 같다.

- i번째 값은 앞 값보다 작을 수 없으므로 하한을 올린다.

$$m_i = \max(m_{i-1}, m)$$

- i번째 값은 뒷 값보다 클 수 없으므로 상한을 줄인다.

$$M_i = \min(M_{i+1}, M_i)$$

이때 $m_i > M_i$인 위치가 있다면 불가능하므로 `-1`을 출력하면 된다.

이제 문제는 각 위치 i에 대해 구간 $[m_i, M_i]$ 안의 값을 하나씩 배정하되, 초기 수열 A에 존재하던 값들을 최대한 많이 그대로 사용하는 문제로 바뀐다. 이는 최대 이분 매칭 문제임이 자명하다.

값들을 작은 순서대로 처리하면서 그리디하게 매칭할 수 있다. 정렬된 수열 A'을 기반으로 현재 값 x를 넣을 수 있는 위치 후보들을 우선순위 큐로 관리하는 식으로 풀면 각 구간이 많아야 한 번 들어가고 한 번 나오기 때문에 시간복잡도는 $\mathcal{O}(N \log N)$이 된다.

문제 K. 상인

출제자 송주헌 (lunarlity)

2 풀이

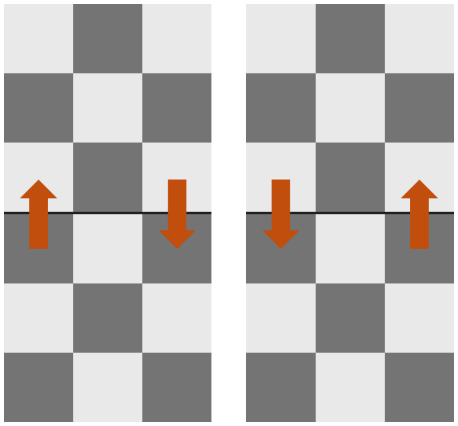
문제를 간단히 요약하면, 3×3 정사각형 블록 N개를 변끼리 맞닿게 이어 붙여 만든 격자 그래프에서 해밀턴 사이클의 존재 여부를 판정하고, 가능하다면 이를 구성하는 것이다. 이때 블록들의 인접 관계로 이루어지는 그래프를 T라 하자. 문제의 조건에 의해 T는 트리이다.

먼저 해밀턴 사이클이 존재하기 위한 필요조건부터 살펴보자. 전체 그래프의 정점 수는 각 블록이 포함하는 9개의 칸을 모두 합쳐 $9N$개이다. 격자 그래프는 체커보드 색칠이 가능한 이분 그래프이므로, 모든 사이클의 길이는 짝수이다. 따라서 해밀턴 사이클이 존재하려면 전체 정점 수 $9N$ 역시 짝수여야 하며, 이는 곧 N이 짝수여야 함을 뜻한다.

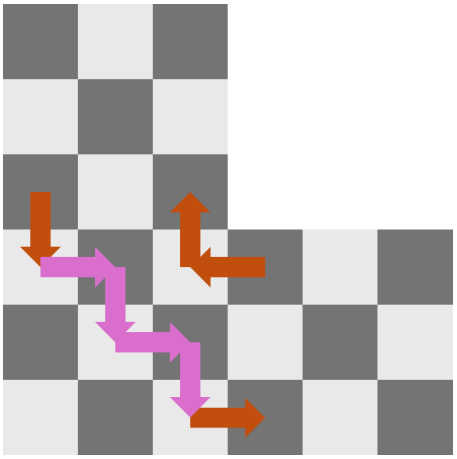
이제 블록 하나 B를 고정하고, T에서 B를 제거하여 생기는 연결 성분들을 생각하자. 이를 $T_1, T_2, \ldots, T_d$라 하자. 각 T_i는 B에 인접한 한 방향에 대응하는 서브트리이다.

이 중 어떤 성분 T_i의 크기 $|T_i|$가 홀수라고 하자. 그러면 T_i에 포함된 작은 칸의 개수는 $9|T_i|$개이므로 역시 홀수이다. 한편 전체 해밀턴 사이클이 B와 T_i 사이의 경계를 지나는 횟수는 컷을 지나는 간선 수이므로 반드시 짝수이다. 또한 T_i의 칸들도 모두 방문해야 하므로, 그 횟수는 0일 수 없다. 그런데 B와 T_i 사이의 경계에는 가능한 간선이 정확히 3개뿐이므로, 실제로 사용되는 간선 수는 정확히 2개이다.

즉 해밀턴 사이클을 T_i 쪽에 제한하면, T_i 전체를 지나는 하나의 해밀턴 경로가 된다. T_i의 정점 수는 홀수이므로, 이 경로의 두 끝점은 같은 색이어야 한다. 공유한 한 변 위의 세 칸은 색이 번갈아 나타나므로, 같은 색인 두 칸은 양 끝 칸뿐이다. 따라서 $|T_i|$가 홀수라면, 해밀턴 사이클은 반드시 그림과 같이 그 변의 양 끝 칸을 통해 T_i를 출입해야 한다.



이제 B에 대해 홀수 크기 성분이 두 개 이상 존재한다고 가정하자. 만약 그 둘이 서로 이웃한 방향에 있다면, 방금 얻은 출입 방식이 동시에 강제되고, 그림에서 보듯이 B 내부에서 작은 사이클이 생겨 전체가 하나의 해밀턴 사이클이 되는 것이 불가능하다. 핑크색은 해밀턴 사이클을 만들기 위해선 자동결정되는 방향들이다.



그렇다면 두 홀수 성분이 서로 마주보는 경우를 생각해 보자. N은 짝수이므로,

$$|T_1| + |T_2| + \cdots + |T_d| = N - 1$$

은 홀수이다. 따라서 홀수 크기 성분의 개수는 홀수 개여야 한다. 그런데 서로 마주보는 두 개만 홀수일 수는 없으므로, 적어도 하나의 다른 성분도 홀수이다. 그러면 네 방향 중 홀수 성분이 세 개 이상 존재하게 되는데, 네 방향에 세 개 이상이 있으면 서로 이웃한 두 방향이 반드시 존재한다. 이는 앞의 경우와 모순이다.

결론적으로, 임의의 블록 B에 대해 $T \setminus \{B\}$의 연결 성분들 중 홀수 크기인 것은 정확히 하나이다. 정리하면 다음 필요조건을 얻는다.

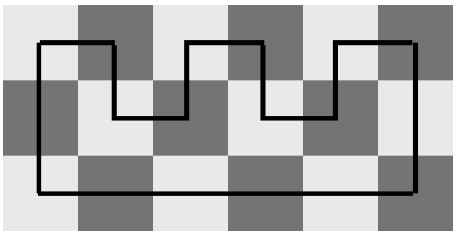
> 모든 블록 B에 대하여, B를 제거했을 때 생기는 연결 성분들 중 홀수 크기인 것은 정확히 하나이다.

이제 이 조건이 사실상 블록 트리 T에 완전 매칭, 즉 Domino Covering이 존재한다는 조건과 동치임을 보이자. 먼저 T에 완전 매칭이 존재한다고 하자. 임의의 정점 B를 보자. B는 매칭에서 정확히 하나의 이웃 B'와 짝지어져 있다. $T \setminus \{B\}$에서 B'가 속한 연결 성분을 C라 하자. 매칭 간선 BB'를 제거하면 $C \setminus \{B'\}$는 완전 매칭을 가지므로 $|C| - 1$은 짝수이고, 따라서 $|C|$는 홀수이다. 반면 다른 연결 성분들은 모두 내부에서 완전 매칭을 가져야 하므로 크기가 짝수이다. 따라서 각 정점에서 홀수 크기 성분은 정확히 하나이다.

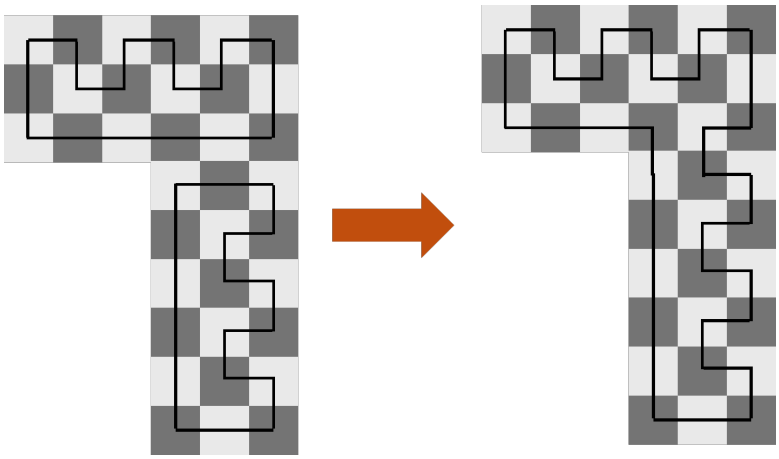
반대로, 모든 정점에서 홀수 크기 성분이 정확히 하나라고 하자. 이때 T에 완전 매칭이 존재함을 귀납적으로 보일 수 있다. T의 임의의 리프를 x, 그 유일한 이웃을 y라 하자. $T \setminus \{y\}$에서 $\{x\}$는 크기 1인 홀수 성분이므로, 가정에 의해 나머지 성분들은 모두 짝수 크기이다. 따라서 간선 xy를 매칭에 포함시켜도 다른 성분들의 내부에서 서로 독립적으로 완전 매칭을 찾을 수 있다. 각 성분에 대해 같은 논리를 재귀적으로 적용하면, 결국 T 전체의 완전 매칭을 얻는다.

즉, 블록 트리 T에 완전 매칭이 존재할 필요충분조건은, 모든 정점에서 홀수 크기 연결 성분이 정확히 하나인 것이다.

이제 충분조건을 보이자. T에 완전 매칭이 존재한다고 하자. 매칭된 두 블록을 하나의 도미노로 묶으면 항상 3×6 직사각형이 되며, 이 안에는 그림과 같은 고정된 해밀턴 사이클을 그릴 수 있다.



서로 인접한 두 도미노의 경우, 그림과 같이 각 도미노의 해밀턴 사이클에서 경계 근처의 간선 두 개를 끊고 교차로 다시 이어 붙이면, 두 개의 분리된 사이클을 하나의 큰 사이클로 합칠 수 있다.



이 연산은 항상 가능하며, 합쳐진 뒤에도 각 칸의 차수는 여전히 2로 유지된다. 트리의 간선을 따라 이 과정을 반복하면, 마지막에는 모든 칸을 정확히 한 번씩 지나는 하나의 해밀턴 사이클을 얻는다.

결론적으로, 원래 문제는 블록 트리 T에 Domino Covering, 즉 완전 매칭이 존재하는지 판정하는 문제와 동치이며, 완전 매칭이 존재하면 실제 해밀턴 사이클도 항상 구성할 수 있다. 시간복잡도는 $O(N)$이다.

참고) 검수진 코드 중 가장 짧은 길이는 2502B이다.

문제 L. 확률성 정렬

출제자 정우현 (woohyun_jng)

수열 A의 최종 길이의 기댓값을 $f(A)$라 정의하자.

또한 수열 A에서 길이가 k인 증가 부분 수열의 개수를 $g(A,k)$라 정의하자.

$$f(A)=\sum_{i=1}^{|A|}\frac{g(A,i)}{\binom{|A|}{i}}$$

이다.

귀납적으로 보이자.

만약 A가 이미 정렬돼있다면 $g(A,i)=\binom{|A|}{i}$이므로 $f(A)=|A|$로 성립한다.

정렬돼있지 않은 수열 A에 대하여 A의 모든 부분 수열에 대하여 위 Claim이 성립한다고 가정하자.

$$\begin{aligned}f(A)&=\frac{1}{|A|}\sum_{x\in A}f(A\setminus\{x\})\\&=\frac{1}{|A|}\sum_{x\in A}\left(\sum_{i=1}^{|A|-1}\frac{g(A\setminus\{x\},i)}{\binom{|A|-1}{i}}\right)\\&=\frac{1}{|A|}\sum_{i=1}^{|A|-1}\frac{1}{\binom{|A|-1}{i}}\sum_{x\in A}g(A\setminus\{x\},i)\end{aligned}$$

$\sum_{x\in A}g(A\setminus\{x\},i)$의 의미를 보자. A의 길이가 i인 어떤 증가 부분 수열은 A에 속한 원소를 제거하는 $|A|$가지 경우 중 $|A|-i$회만 살아남는다. 따라서

$$\sum_{x\in A}g(A\setminus\{x\},i)=(|A|-i)g(A,i)$$

이다.

$$\frac{1}{A}\times\frac{1}{\binom{|A|-1}{i}}\times(|A|-i)=\frac{1}{\binom{|A|}{i}}$$

이므로 결과적으로

$$f(A)=\sum_{i=1}^{|A|-1}\frac{g(A,i)}{\binom{|A|}{i}}$$

이고 A는 정렬되지 않은 수열이므로 $g(A,|A|)=0$, 결국 i를 $|A|$까지 적용시켜도 식이 성립한다. 결국 해당 Claim이 참이 된다.

$\binom{|A|}{i}$는 팩토리얼과 팩토리얼의 역수를 전처리하면 $O(1)$에 계산 가능하다.

$g(A,i)$는 DP를 통해 구해줄 수 있다.

$dp[x][y]$를 x번째 수가 마지막이고 길이가 y인 증가 부분 수열의 개수라고 정의하자.

$$dp[x][y]=\sum_{i<x-1,A[i]<A[x]}dp[i][y-1]$$

이다. 따라서 상태가 $O(N^2)$개, 각 상태를 전이하는데 $O(N)$이 걸리므로 $O(N^3)$의 시간복잡도로 DP 테이블을 모두 채워줄 수 있다.

전이 과정을 세그먼트 트리를 이용해서 $O(\log N)$에 처리 가능하다. 이 경우 문제를 $O(N^2 \log N)$의 시간복잡도로 해결 가능하며 이것이 의도한 풀이이다.