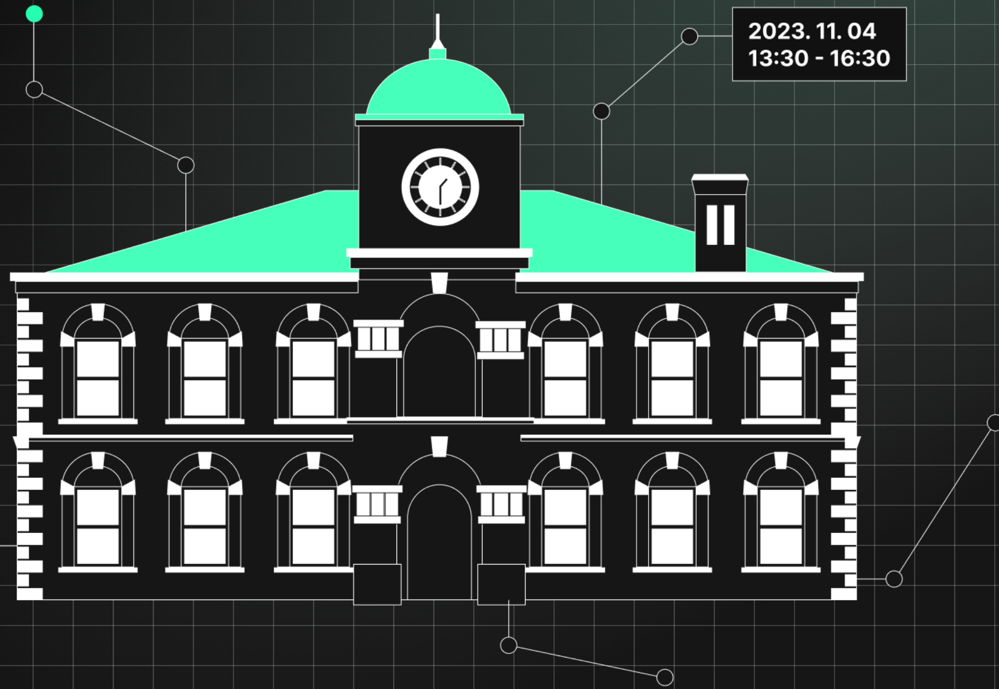
2023
건국대학교
프로그래밍 경진대회
Official Solution



AlKon SW중심대학 STARTLINK SOLVED.AC

🦆	🪿	문제	의도한 난이도	출제자
A		얼룩말을 찾아라!	Beginner	이동훈
B	A	이제는 더 이상 물러날 곳이 없다	Easy	이동훈
C		바닥수	Easy	이동훈, 윤찬규
D	B	단체줄넘기	Easy	이동훈
E		팰린드롬 애너그램	Normal	이동훈
F	C	현수막 걸기	Normal	김수현
G		스위치	Normal	윤찬규
H	D	낚시	Normal	이동훈
I	E	MEXchange	Hard	김명기
	F	K-문자열	Hard	김명기
	G	시간낭비	Hard	이동훈
	H	우정은 BFS처럼, 사랑은 DFS처럼	Challenging	이동훈, 윤찬규
J	I	양손 정렬	Challenging	이동훈, 윤찬규, 김명기
	J	등불 날리기	Challenging	윤찬규

2A. 얼룩말을 찾아라!

implementation, string

출제진 의도 - **Beginner**

🦆 제출 5회, 정답 2명 (정답률 40%)

🥇 이동훈, 5분

🪿 제출 — 회, 정답 — 명 (정답률 — %)

🥇 —

📣 이동훈 aru0504

2A. 얼룩말을 찾아라!

- 주어진 문자열에서 1이 연속으로 등장하는 구간을 알아내는 방법은 무엇일까요?
- 여러 가지 방법이 있지만, 구현이 간단한 방법을 소개합니다.
- 1 이후에 `0` 이 등장한다면, 검은 줄의 구간이 끝나는 것을 의미합니다.
- 주어진 문자열을 훑으면서, `s[i] = 1`이면서 `s[i+1] = 0`인 `i`의 개수를 셉시다.
- 편의를 위해, 주어진 문자열에 `0` 을 추가해도 답은 같습니다.
- 시간복잡도는 $\mathcal{O}(N)$ 입니다.

2B/1A. 이제는 더 이상 물러날 곳이 없다

`game_theory, ad_hoc`

출제진 의도 - **Easy**

🦆 제출 5회, 정답 2명 (정답률 40%)

🥇 이동훈, 5분

🪿 제출 30회, 정답 12명 (정답률 40%)

🥇 김명기, 20분

📣 이동훈 `aru0504`

2B/1A. 이제는 더 이상 물러날 곳이 없다

- 건덕이와 건구스는 자신의 차례에 **반드시** 움직여야 합니다.
- 건덕이와 건구스는 다음과 같은 네 가지 종류로 움직입니다: RR, RL, LL, LR
- 어떻게 움직이더라도 두 플레이어 사이 간격의 **홀짝성**$_{\texttt{parity}}$은 변하지 않습니다.

2B/1A. 이제는 더 이상 물러날 곳이 없다

- 전장의 길이가 주어지는 순간 승자는 결정되며, 승자는 상대를 향해 전진합니다.
- 패자의 경우 앞으로 가면 패배로 더 빠르게 도달하므로, 가능하다면 뒤로 후퇴합니다.
- 전장을 벗어나도록 이동할 수 없으므로, **두 플레이어간 간격은 단조감소**합니다.

2B/1A. 이제는 더 이상 물러날 곳이 없다

- ✓ 간격을 짝수로 만드는 사람은 결국 두 플레이어 사이의 거리를 0으로 만듭니다.
- ✓ 플레이어 사이의 거리가 0이 되는 순간, 다음 차례의 플레이어는 공격할 수 있습니다.
- ✓ 주어진 전장의 길이가 짝수라면, 선공이 먼저 둘 사이의 간격을 홀수로 만듭니다.
- ✓ 홀수라면, 선공이 둘 사이의 간격을 짝수로 만듭니다.
- ✓ 따라서 전장의 길이가 짝수인 경우, 건덕이가 승리합니다.
- ✓ 주어진 전장의 길이의 홀짝성을 판단하므로 시간복잡도는 $\mathcal{O}(1)$입니다.

💪 $N \times N$ 전장에서 각 플레이어가 $(1, 1)$, (N, N)에서 시작하면 누가 승리할까요?

2C. 바닥수

`constructive, math`

출제진 의도 - **Easy**

🦆 제출 5회, 정답 2명 (정답률 40%)

🥇 이동훈, 5분

🦢 제출 — 회, 정답 — 명 (정답률 — %)

🥇 —

📣 이동훈 `aru0504`, 윤찬규 `dldyou`

2C. 바닥수

- 바닥수 N이 주어졌을 때, 바닥수가 N인 길이 L의 원래 수를 구해야 합니다.
- 길이 L의 모든 수에 대해서 바닥수가 되는지 확인하는 것은 오래 걸립니다.
- 곱셈의 성질을 활용해서 쉽게 문제를 해결해 봅시다.

2C. 바닥수

- 곱셈의 항등원을 이용합시다. 1은 여러 번 곱해도 1입니다.
- 1을 $L-1$번 적은 뒤, N를 뒤에 덧붙인 수의 바닥수는 N입니다.
- 원래 수는 0으로 시작하지 않음에 주의합시다.

2D/1B. 단체줄넘기

`implementation`

출제진 의도 - **Easy**

🦆 제출 5회, 정답 2명 (정답률 40%)

🥇 이동훈, 5분

🪿 제출 30회, 정답 12명 (정답률 40%)

🥇 김명기, 20분

📣 이동훈 `aru0504`

2D/1B. 단체줄넘기

- 학생의 키가 같은 쌍이 존재하지 않는다면, 모든 학생이 참여할 수 있습니다.
- 모두가 한 쪽을 바라보고, 바라보는 방향으로 키가 작아지도록 줄세우면 됩니다.
- 학생의 키가 같은 쌍이 존재하는 경우에는 몇 명까지 참여할 수 있을까요?

2D/1B. 단체줄넘기

- ✓ 두 명의 키가 서로 같다면, 서로 반대 방향을 보고 서면 됩니다.
- ✓ 세 명의 키가 서로 같다면, 한 명은 어느 쪽을 보든 점프할 타이밍을 놓칠 수 있습니다.
- ✓ 따라서 키가 같은 사람은 **최대 두 명**까지만 참여할 수 있습니다.
- ✓ 줄을 세울 수 있는지의 여부를 물어보았으므로, 각 사람들의 키에 대해서 참여할 수 있는 사람의 수를 세어 줍시다.
- ✓ 시간복잡도는 $\mathcal{O}(N)$입니다.

💪 올바르게 줄 세우는 방법 중 하나를 구할 수 있을까요?

2E. 팰린드롬 애너그램

ad_hoc

출제진 의도 - **Normal**

🦆 제출 5회, 정답 2명 (정답률 40%)

🥇 이동훈, 5분

🪿 제출 – 회, 정답 – 명 (정답률 – %)

🥇 –

📣 이동훈 `aru0504`

2E. 팰린드롬 애너그램

- ✓ 왼쪽 절반의 문자와 오른쪽 절반의 문자의 교환을 통해 팰린드롬을 만들어 봅시다.
- ✓ 연산을 통해 임의의 위치에 존재하는 두 문자를 교환할 수 있다면,
  모든 문자를 원하는 곳에 배치할 수 있습니다. 과연 가능할까요?
- ✓ 편의상 배열의 길이가 짝수라고 가정합시다. 교환하는 경우의 수는 두 가지입니다.
  - * 두 문자가 서로 다른 절반에 속하는 경우
  - * 두 문자가 같은 절반에 속하는 경우

2E. 팰린드롬 애너그램

- 두 문자가 서로 다른 절반에 속하는 경우, 두 원소를 직접 교환할 수 있습니다.
- 두 문자가 서로 같은 절반에 속하는 경우에는 두 원소를 직접 교환할 수 없습니다.
- 이때, 반대쪽 절반에 속하는 원소 한 개를 임시로 사용해 교환할 수 있습니다.

1	2	3	4
4	2	3	1
3	2	4	1
1	2	4	3

2E. 팰린드롬 애너그램

- 문자열의 길이가 홀수인 경우에는 가운데 글자를 교환할 수 없음에 유의합시다.
- 해당 문자를 세지 않고, 짝수 길이의 문자열이라고 생각하면 됩니다.
- 팰린드롬이 되기 위해서는 양 쪽에 같은 수의 문자가 존재해야 합니다.
- 알파벳이 문자열에 짝수번 등장한다면 양 쪽에 골고루 분배할 수 있습니다.
- 시간복잡도는 $\mathcal{O}(N)$입니다.

2F/1C. 현수막 걸기

`binary_search`

출제진 의도 - **Normal**

🦆 제출 5회, 정답 2명 (정답률 40%)

🥇 이동훈, 5분

🪿 제출 30회, 정답 12명 (정답률 40%)

🥇 김명기, 20분

📣 김수현 creampuffshu

2F/1C. 현수막 걸기

- 두 개의 말뚝과 깃대 하나를 골라 만들 수 있는 삼각형 넓이 중 최댓값을 구합시다.
- 단, 넓이가 R 이하여야 합니다.
- 간단한 방법은 모든 말뚝 쌍에 대해서, 모든 깃대를 탐색하는 방법입니다.
- 해당 풀이는 $\mathcal{O}(N^2M)$으로, 제한 시간 안에 해결할 수 없습니다.

2F/1C. 현수막 걸기

- 만약 밑변을 고정한다면, 주어진 깃대를 정렬한 뒤 이분탐색을 사용할 수 있습니다.
- 가능한 모든 밑변을 구하는 데에는 $\mathcal{O}(N^2)$입니다.
- 하나의 밑변 길이에 대해서, R 이하인 최대 넓이를 구하는 데 $\mathcal{O}(\log M)$이 듭니다.
- $\mathcal{O}(N^2 \log M)$ 제한에는 충분히 통과할 수 있습니다.

2G. 스위치

`dynamic_programming`

출제진 의도 - **Normal**

🦆 제출 5회, 정답 2명 (정답률 40%)

🥇 이동훈, 5분

🦢 제출 — 회, 정답 — 명 (정답률 — %)

🥇 —

📣 윤찬규 dldyou

2G. 스위치

- 스위치를 누르면 3초동안 두 배의 점수를 얻을 수 있습니다.
- 음수의 점수도 두 배로 계산됨에 유의해야 합니다.
- 그리디하게 스위치를 누르는 전략은 오답을 받습니다.
- 스위치를 누른 뒤, i초 시점에 스위치의 효과가 j초 남았다는 것을 활용합시다.

2G. 스위치

- i초에 $j=0$이라면, $i-1$초에 $j=0$이거나 $j=1$인 경우 중 최댓값을 가져옵시다.
- $j=1$이라면 아직 스위치의 효과가 남아있으므로 $i-1$초의 $j=2$를 가져옵니다.
- $j=2$인 경우도 마찬가지로 $i-1$초의 $j=3$을 가져옵니다.
- $j=3$은 지금 스위치를 누른 상황으로, $i-1$초의 $j=0$이거나 $j=1$인 경우에서 가져올 수 있습니다.
- 스위치의 효과가 적용되는 경우에는 배열의 값의 두배를 더해줍시다.
- 시간복잡도는 $\mathcal{O}(N)$입니다.

2H/1D. 낚시

`prefix_sum`

출제진 의도 - **Normal**

🦆 제출 5회, 정답 2명 (정답률 40%)

🥇 이동훈, 5분

🪿 제출 30회, 정답 12명 (정답률 40%)

🥇 김명기, 20분

📣 이동훈 aru0504

2H/1D. 낚시

1	2	3	4	5
5	2	1	4	6
0	2	4	2	1
0	0	2	1	7

- 일감호입니다. 🎣
- 무게추 3을 달아 3의 힘으로 낚싯대를 휘두르면 회색 칸에 찌가 도달합니다.
- 빨간 색 테두리에 해당하는 칸에 존재하는 8마리의 물고기가 사로잡힙니다.

2H/1D. 낚시

1	2	3	4	5
5	2	1	4	6
0	2	4	2	1
0	0	2	1	7

- ✓ 낚싯줄을 한 바퀴 감아올리면 찌의 위치가 바뀝니다.
- ✓ 바뀐 칸에서 물고기 4마리가 사로잡힙니다.
- ✓ 이를 미끼가 일감호를 벗어날 때까지 반복합니다.

2H/1D. 낚시

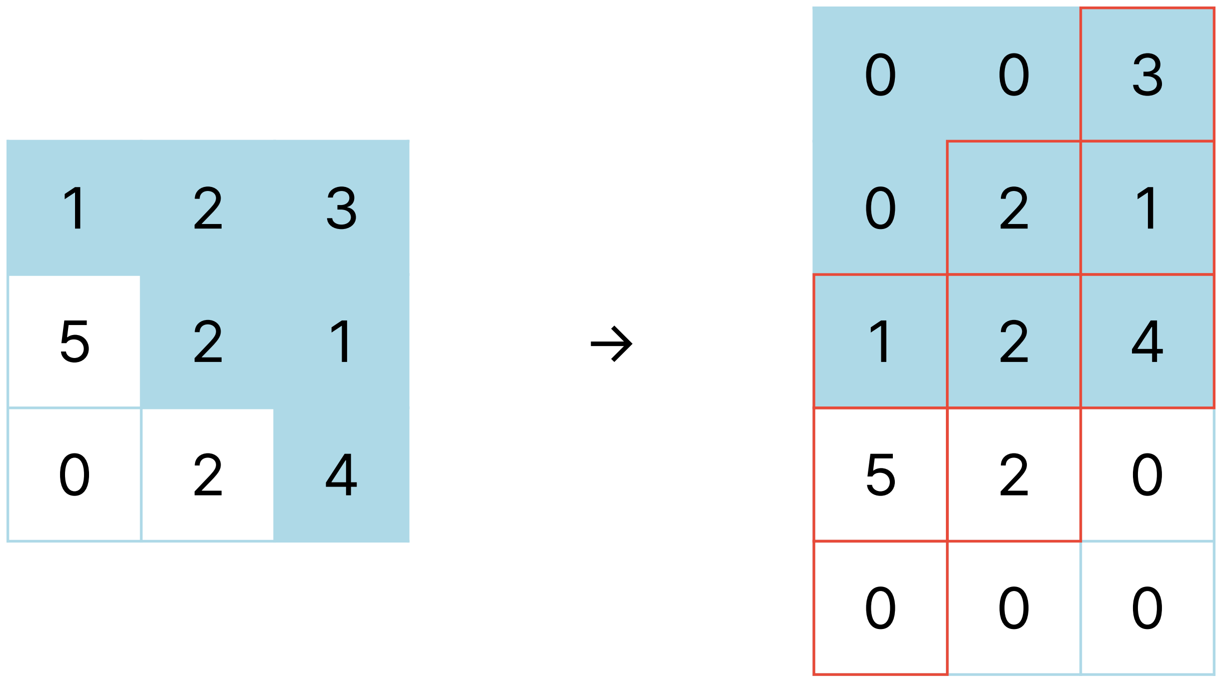
1	2	3	4	5
5	2	1	4	6
0	2	4	2	1
0	0	2	1	7

- 쿼리가 주어질 때마다 다음과 같은 계단 모양에 칠해진 수의 합을 구하는 문제입니다.
- 간단하게 생각하면, 매 쿼리마다 계산하는 $\mathcal{O}(NMQ)$ 풀이가 있습니다 🤯.
- 이는 시간 초과입니다. 어떻게 빠르게 해결할 수 있을까요?

2H/1D. 낚시

- 여러 방법 중 하나를 소개합니다.
- 2차원 배열에서 직사각형 모양의 부분합은 $\mathcal{O}(NM + Q)$에 구할 수 있습니다.
- 쿼리가 직사각형의 부분합을 구하게끔 일감호를 재배치할 수 있을까요?

2H/1D. 낚시



- 가능합니다.
- 시간복잡도는 2차원 부분합과 같은 $\mathcal{O}(NM + Q)$입니다.

2I/1E. MEXchange

`ad_hoc, constructive`

출제진 의도 - **Hard**

🦆 제출 5회, 정답 2명 (정답률 40%)

🥇 이동훈, 5분

🪿 제출 30회, 정답 12명 (정답률 40%)

🥇 김명기, 20분

📣 김명기 `riroan`

2I/1E. MEXchange

- 우선 언제 NO 를 출력해야할 지 생각해봅니다.
- MEX의 정의에 의해, 길이가 N인 수열의 MEX는 $N+1$보다 클 수 없습니다.
- A는 순열이기 때문에 $B_N = N+1$입니다.
- $S_i = \{A_1, A_2, \ldots, A_i\}$라고 할 때, $S_i \subset S_{i+1}$이므로
  $B_i > B_{i+1} (1 \le i \le N-1)$일 수 없습니다. 즉 B는 **단조증가**합니다.

2I/1E. MEXchange

- 몇 가지 관찰을 해 봅시다.
- 만약 $A_i = K$라면, $i \leq j$일 때 $B_j = K$일 수 없습니다.
- K를 포함하는 집합의 MEX는 정의상 K일 수 없기 때문입니다.
- 결국 $B_j = K$일 때, $A_i = K (j < i)$입니다.

2I/1E. MEXchange

- $B_i \neq B_{i+1} (1 \leq i \leq N-1)$인 경우에 주목합니다.
- MEX값이 B_{i+1}이기 위해서는 해당 집합이 B_i를 가지고 있어야 합니다.
- 이전 관찰과 종합하면 $B_i \neq B_{i+1}$일 때, $A_{i+1} = B_i$가 됨을 알 수 있습니다.

2I/1E. MEXchange

- 이제 남은 값을 배치하는 방법은 여러 가지가 있습니다.
- 그 중 하나는 작은 수부터 빈 곳에 채워 넣는 것입니다.
- 왜 이 방법이 가능할까요?

2I/1E. MEXchange

- 채워야하는 곳은 $B_{i-1} = B_i$인 A_i입니다.
- 해당 방법으로 빈 곳을 배치하면 $B_i < A_i$입니다.
- MEX값이 B_i인 배열에 B_i보다 큰 값을 배치하더라도 MEX값은 변하지 않습니다.
- 그리고 $B_{k-1} \neq B_k$인 k에 대해 총 k개의 공간에 B_k보다 작은 $B_k - 1(\leq k)$개의 수를 배치하므로 작은 수부터 배치하면 원하는 결과를 얻을 수 있습니다.

2I/1E. MEXchange

- ✓ $B_i \neq B_{i+1}$일 때, $A_{i+1} = B_i$

1	2	2	4	6

B

?	1	?	2	4

A

2I/1E. MEXchange

- 남은 값을 작은 수부터 빈 곳에 채워 넣습니다.

1	2	2	4	6

B

3	1	5	2	4

A

- 시간복잡도는 $\mathcal{O}(N)$입니다.

1F. K-문자열

`string, bitmask, combinatorics`

출제진 의도 - **Hard**

🦆 제출 — 회, 정답 — 명 (정답률 — %)

🥇 —

🪿 제출 30회, 정답 12명 (정답률 40%)

🥇 김명기, 20분

📣 김명기 `riroan`

1F. K-문자열

- 주어진 문자열을 하나의 집합으로 생각합니다.
- 예를 들어 문자열이 `0011223344`라면 대응되는 집합은 $\{0, 1, 2, 3, 4\}$입니다.
- 숫자는 10개 이므로 만들어질 수 있는 집합의 개수는 $2^{10} = 1024$개입니다.
- 이제 집합 두 개를 합쳐 크기가 K인 집합의 개수를 세면 됩니다.

1F. K-문자열

- 그럼 크기가 K인 집합의 개수는 어떻게 셀까요?
- 만들어질 수 있는 1024개의 집합 중 $A \cup B$의 원소의 개수가 K개인 두 집합 A, B를 순서를 고려하지 않고 뽑습니다.
- $A = B$일 때는 정답에 $\frac{\mathrm{cnt}(A) \times (\mathrm{cnt}(A) - 1)}{2}$를 더합니다.
- $A \neq B$일 때는 정답에 $\mathrm{cnt}(A) \times \mathrm{cnt}(B)$를 더합니다.
- 시간복잡도는 $\mathcal{O}\left(N + 1024^2 \times 10\right)$입니다.
- 비트집합을 사용하면 $\mathcal{O}\left(N + 1024^2\right)$으로 줄일 수 있지만, 사용하지 않아도 됩니다.

1G. 시간낭비

`dp, graphs, topological_sorting`

출제진 의도 - **Hard**

🦆 제출 — 회, 정답 — 명 (정답률 — %)

🥇 —

🦢 제출 30회, 정답 12명 (정답률 40%)

🥇 김명기, 20분

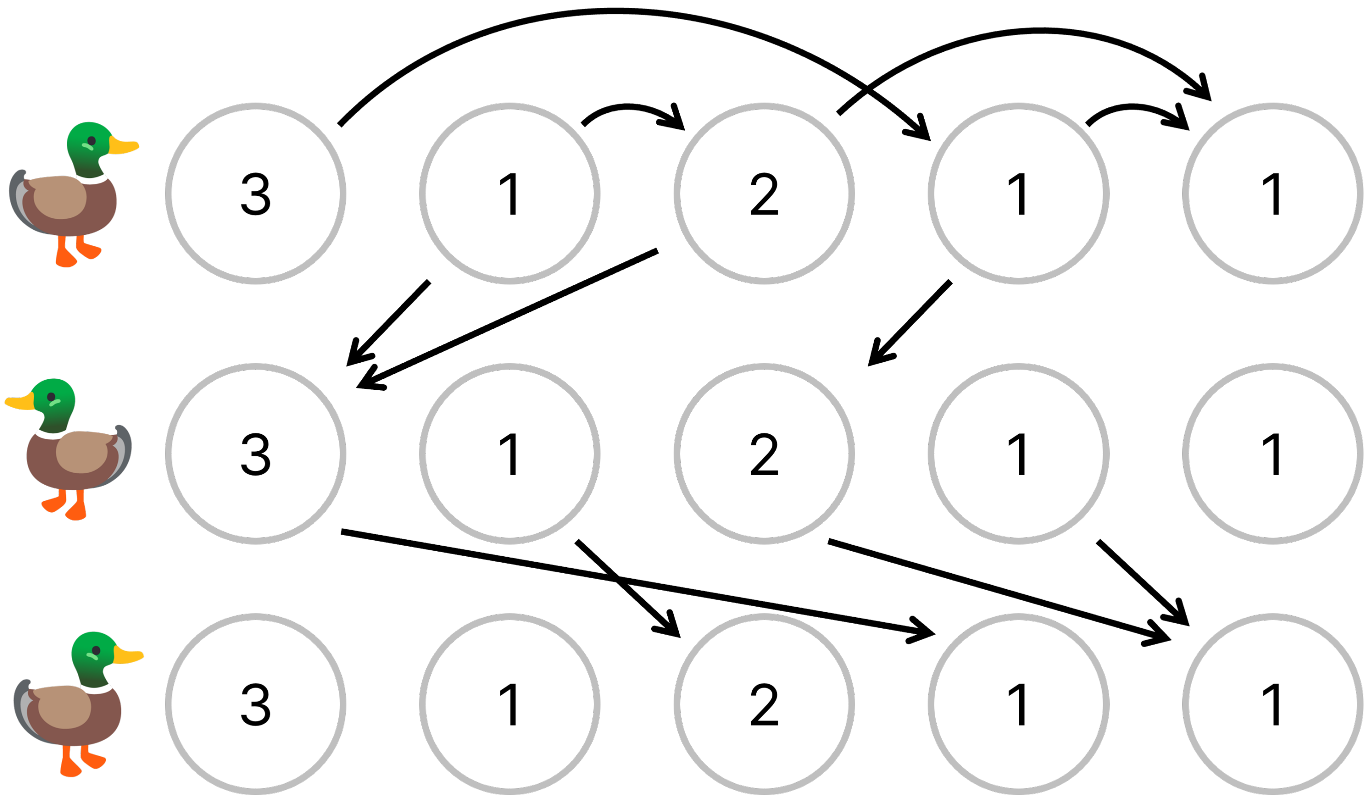
📣 이동훈 aru0504

1G. 시간낭비

- 주어진 등굣길을 그래프로 모델링해 봅시다.
- 각 칸에 쓰인 수만큼 양쪽 방향으로 떨어진 칸으로 향하는 간선을 만들 수 있습니다.
- 최대 두 번 방향을 반전할 수 있으므로, 같은 칸이라도 0번 반전했을 때와 2번 반전했을 때는 다르게 생각해야 합니다.

1G. 시간낭비

✓ 정점을 세 개로 나누어 아래와 같은 그래프 모형을 생각할 수 있습니다.



1G. 시간낭비

- 문제의 정답은 1번 정점에서 학교에 해당하는 정점들까지의 최장거리와 같습니다.
- 구축한 그래프는 사이클 없는 방향 그래프입니다.
- 1번 정점에서 위상정렬을 시행한 뒤, 정렬한 순서대로 정답을 갱신해나가면 됩니다.
- 학교에 처음 도착한 시간을 늦추는 것에 유의합시다. 학교에서 되돌아가는 간선은 존재할 수 없습니다.
- 시간복잡도는 $\mathcal{O}(N)$입니다.

1G. 시간낭비

- 더 쉬운 풀이로, 단순하게 오른쪽으로 한 번, 왼쪽으로 한 번, 다시 오른쪽으로 한 번 DP를 훑어도 해결할 수 있습니다.
- 아직 나에게 오는 경로가 없거나, 이미 학교에 도착한 경우를 잘 처리해야 합니다.
- 시간복잡도는 위와 같은 $\mathcal{O}(N)$이지만, 상수가 작아 더 빠르게 통과합니다.

1H. 우정은 BFS처럼, 사랑은 DFS처럼

`constructive, greedy, graph_traversal`

출제진 의도 - **Challenging**

🦆 제출 — 회, 정답 — 명 (정답률 — %)

🥇 —

🪿 제출 30회, 정답 12명 (정답률 40%)

🥇 김명기, 20분

📣 이동훈 `aru0504`, 윤찬규 `dldyou`

1H. 우정은 BFS처럼, 사랑은 DFS처럼

2J/1I. 양손 정렬

`permutation_cycle_decomposition`

출제진 의도 - **Challenging**

🦆 제출 5회, 정답 2명 (정답률 40%)

🥇 이동훈, 5분

🪿 제출 30회, 정답 12명 (정답률 40%)

🥇 김명기, 20분

📣 이동훈 `aru0504`, 윤찬규 `dldyou`, 김명기 `riroan`

2J/1I. 양손 정렬

- 팰린드롬 애너그램에서 보였듯, N이 짝수인 경우 항상 가능합니다.
- 홀수인 경우에는 한가운데의 원소가 자신의 자리에 있을 때에만 가능합니다.
- 어떻게 최소한의 연산으로 원하는 배열을 만들 수 있을까요?

1J. 등불 날리기

`segment_tree, sliding_window`

출제진 의도 - **Challenging**

🦆 제출 — 회, 정답 — 명 (정답률 — %)

🥇 —

🪿 제출 30회, 정답 12명 (정답률 40%)

🥇 김명기, 20분

📣 윤찬규 dldyou

1J. 등불 날리기

- 하나의 등불이 다른 등불을 앞지르는 횟수가 곧 소원을 비는 횟수입니다.
- 상대적으로 오른쪽의 등불이 상승하는 정도가 더 크면,
  왼쪽의 등불을 앞지르는 순간이 존재합니다.
- $i \leq j$ 에서 $A_i < A_j$인 쌍의 개수를 길이가 S인 구간에서 세어줘야 합니다.

1J. 등불 날리기

- 간단하게 생각하면 구간 내 원소에 대해서 모든 쌍을 찾는 방법을 떠올릴 수 있습니다.
- 구간 하나에 $\mathcal{O}(N^2)$이므로, 모든 구간에서 계산한다면 $\mathcal{O}(N^3)$으로 시간초과입니다.
- 앞지르는 쌍의 개수를 더 빠르게 구할 수 있을까요?

1J. 등불 날리기

- Merge Sort를 응용하면 문제를 조금 더 효율적으로 풀 수 있습니다.
- 두 배열을 합치는 과정에서, 양쪽 배열은 정렬된 상태입니다.
- 왼쪽 절반 배열의 원소가 더 작다면, 해당 원소보다 뒤에 존재하는 원소 모두가 오른쪽에서 고른 원소보다 작습니다.
- 하나의 구간에서 모든 쌍의 개수를 $\mathcal{O}(N \log N)$에 구할 수 있습니다.
- 여전히 모든 구간을 확인하는 데 $\mathcal{O}(N^2 \log N)$으로, 더 빠른 방법이 필요합니다.

1J. 등불 날리기

- 문제의 특성을 파악해서 빠르게 풀어 봅시다.
- k번 등불부터 S개 고른 구간을 T_k라고 합시다.
- T_k 구간과 T_{k+1} 구간의 공통된 구간은 $[k+1, k+S-1]$입니다.
- 공통된 구간에서 발생하는 앞지르는 쌍의 개수는 k번째 등불과 $k+S$번째 등불에 영향받지 않습니다.
- 해당 구간의 연산을 하지 않으면 효율적으로 풀 수 있습니다.

1J. 등불 날리기

- ✓ T_k구간에서 다음 구간으로 넘어갈 때 아래의 값들을 빼고 더해줘야 합니다.
  - * k 번째 등불을 제거했을 때 감소하는 앞지르는 쌍의 개수
  - * $k+S$ 번째 등불을 추가할 때 증가하는 앞지르는 쌍의 개수

- ✓ 세그먼트 트리를 활용하면 앞지르는 쌍의 개수를 빠르게 구할 수 있습니다.
  - * k번째 등불을 제거할 때, A_k보다 상승하는 정도가 큰 등불의 개수
  - * $k+S$번째 등불을 추가할 때, A_{k+S}보다 상승하는 정도가 작은 등불의 개수

1J. 등불 날리기

- 주어진 등불 상승 속도의 범위가 최대 10^9이므로 좌표압축을 먼저 진행해야 합니다.
- 시간복잡도는 $\mathcal{O}(N \log N)$입니다.