# 제1회 코더즈 코딩페어 해설

Official Solutions

by

운영진 일동



2025년 12월 25일

| 문제 | | 의도한 난이도 | 출제자 |
|---|---|---|---|
| **A** | 생수병 놓기 | **Easy** | cwrgw |
| **B** | 수미수열 | **Easy** | gmb9817 |
| **C** | 버스 탈주 전쟁 | **Easy** | ymkzz123 |
| **D** | 맞혀보세요 | **Easy** | gmb9817 |
| **E** | 족구왕 와니 | **Medium** | gmb9817 |
| **F** | 대신고와 쿼리 | **Medium** | ymkzz123 |
| **G** | generator | **Medium** | gmb9817 |
| **H** | 오량 | **Hard** | mon_de2738 |
| **I** | 룩 vs 폰 | **Hard?** | js7777 |
| **J** | 균형의 창조자 | **Hard** | nflight11 |
| **K** | 급식 뭐 먹지 | **Hard** | young_out |
| **L** | 해안 경비 | **Hard** | lunarlity |
| **M** | gmb 수열 | **Hard** | choyj416 |
| **N** | 그래프 복원 | **Challenging** | realpsdoingdamyoo |

# A. 생수병 놓기

#implementation

출제진 의도 – **Easy**

- **Onsite**
  - 제출 49번, 정답 16명 (정답률 32.653%)
  - 처음 푼 사람: **오량인8**, 4분
- **Open**
  - 제출 227번, 정답 178명 (정답률 78.414%)
  - 처음 푼 사람: **yyyy7089**, 1분
- 출제자: **cwrgw**

## A. 문제 요약

- 책상 위에 놓인 생수병 배치를 보고,
- 주어진 조건에 맞춰 총 네 가지 경우 중 하나를 출력하는 문제입니다.
- 비교해야 하는 요소는 오직 **생수병의 개수 변화**입니다.

## A. 풀이 아이디어

- 문자열에서 생수병(‘o’)의 개수를 셉니다.
- 이전 상태와 비교하여 다음 세 경우로 분류합니다.
  - 개수가 감소한 경우
  - 개수가 증가한 경우
  - 개수가 그대로인 경우
- 이 중 **개수가 동일한 경우**를 나눠서 한번 더 비교하는 함수를 만드시면 됩니다.

## A. 구현 포인트

- 생수병 개수 판별은 `count` 함수 한 번이면 충분합니다.
- 개수가 동일하다면 아침과 저녁의 물병 배치를 비교하여 출력 조건을 판단합니다.
- 조건에 따른 출력만 명확히 정리하면 구현이 간단히 끝납니다.

# B. 수미수열

#implementation #bruteforce

출제진 의도 – **Easy**

- **Onsite**
  - 제출 51번, 정답 12명 (정답률 23.529%)
  - 처음 푼 사람: **오량인8**, 7분
- **Open**
  - 제출 248번, 정답 162명 (정답률 65.323%)
  - 처음 푼 사람: **random_defense**, 2분
- 출제자: **gmb9817**

## B. 수미수열

- 길이 $N$ 인 수열 $L$ 이 주어집니다.
- **수미수열**의 조건: 길이 $K$ 인 접두사와 접미사가 일치하는 $K \leq \frac{N}{2}$ 가 **존재**하는 것
  - 접두사: $L[1 \cdots K]$
  - 접미사: $L[N-K+1 \cdots N]$
- 예: $[1, 2, 3, 4, 1, 2]$ 는 $K = 2$ 일 때 접두사 $[1, 2]$ 와 접미사 $[1, 2]$ 가 같으므로 수미수열
- 수열이 수미수열인지 판별하여 yes 또는 no를 출력합니다.

## B. 수미수열

- $K = 1$ 부터 $\frac{N}{2}$ 까지 순회하면서 각 $K$ 에 대해 접두사와 접미사를 비교합니다.
- 하나라도 일치하는 $K$ 가 존재하면 yes, 모두 불일치하면 no를 출력합니다.
- 각 $K$ 에 대한 비교는 $\mathcal{O}(K)$ 이고, $K$ 는 최대 $\frac{N}{2}$ 까지 확인하므로

시간복잡도: $\mathcal{O}(N^2)$

- $N \leq 1\,000$ 이므로 충분히 효율적입니다.

- **주의할 엣지 케이스**
  - $N = 1$인 경우: $K \leq 0.5$이므로 조건을 만족하는 $K$가 존재하지 않습니다.
  - 따라서 $N = 1$일 때는 항상 no를 출력해야 합니다.
- 간단한 구현 문제이지만 조건을 정확히 이해하는 것이 중요합니다.
- 제한이 넉넉하므로 비효율적인 구현도 통과 가능합니다.

# C. 버스 탈주 전쟁

#simulation #time

출제진 의도 – **Easy**

- **Onsite**
  - 제출 66번, 정답 4명 (정답률 6.061%)
  - 처음 푼 사람: **오량인8**, 21분
- **Open**
  - 제출 118번, 정답 64명 (정답률 55.085%)
  - 처음 푼 사람: **leinad2**, 6분
- 출제자: **ymkzz123**

## C. 버스 탈주 전쟁

- 현재 시각 $(h, m)$과 $N$개의 버스 정보가 주어집니다.
- 각 버스는 **첫차 시각**과 **배차 간격**을 가지며, 현재 시각 이후 가장 먼저 도착하는 버스를 찾는 것이 목표입니다.
- 시간을 계산하기 위해 시각을 모두 **분 단위**로 환산합니다.

## C. 버스 탈주 전쟁

- 첫차 시간이 현재보다 늦으면, 그 버스를 바로 탈 수 있습니다.
- 반대로 이미 첫차가 지나갔다면, 경과 시간을 기준으로 **올림 나눗셈**을 이용해 다음 도착 시각을 구합니다.

$$next = first + \left\lceil \frac{cur - first}{interval} \right\rceil \cdot interval$$

- 하루의 총 시간은 $1440$분이며, 이를 넘으면 다음날로 넘어가므로 $1440$을 빼서 조정합니다.
- 모든 버스의 다음 도착 시각 중 가장 빠른 시각을 출력하면 됩니다.
- 시간복잡도 $\mathcal{O}(N)$, 첫차 시각을 잘 구분해주면 어렵지 않게 해결할 수 있는 시뮬레이션 문제입니다.

# D. 맞혀보세요

#implementation #data-structures #string #set-map
출제진 의도 – **Easy**

- **Onsite**
  - 제출 29번, 정답 5명 (정답률 17.241%)
  - 처음 푼 사람: **오량인8**, 24분
- **Open**
  - 제출 90번, 정답 63명 (정답률 72.222%)
  - 처음 푼 사람: **seawon0808**, 8분
- 출제자: **gmb9817**

## D. 맞혀보세요

- 규칙 맞히기 퀴즈: 문자열이 모든 규칙을 만족하면 yes, 아니면 no
- 각 규칙의 형식:
  - 문자 $X$ 의 **바로 뒤**에는 허용된 집합 $\{Y_1, \cdots, Y_M\}$ 에 속하는 문자가 오거나
  - 아무것도 오지 않고 문자열이 끝나야 함
- 규칙이 주어지지 않은 문자 뒤에는 무엇이 오든 상관없습니다.
- $N$ 개의 규칙과 문자열 $S$ 가 주어질 때, 모든 규칙을 만족하는지 판단합니다.

## D. 맞혀보세요

- 각 문자에 대한 **허용 집합**을 `map`이나 `dictionary`에 저장합니다.
  - 예: `rules['a'] = {'b', 'c', 'd'}`
- 문자열 $S$를 순회하며 각 위치 $i$에서:
  - 문자 $S[i]$에 대한 규칙이 존재하는지 확인
  - 규칙이 있다면, $S[i+1]$이 허용 집합에 속하는지 검사
  - $i$가 마지막 위치라면 (문자열이 끝나면) 자동으로 허용됨
  - 규칙이 없는 문자는 검사하지 않음
- 모든 검사를 통과하면 `yes`, 하나라도 실패하면 `no`

## D. 맞혀보세요

- ✓ 시간복잡도: 문자열을 한 번 순회하므로 $\mathcal{O}(|S|)$
- ✓ $|S| \leq 1\,000$ 이므로 매우 효율적입니다.
- ✓ **주의할 점**
  - 규칙이 없는 문자는 무시해야 합니다. (뒤에 무엇이 와도 OK)
  - 문자열의 마지막 문자는 항상 조건을 만족합니다. (뒤에 아무것도 없음)
  - 집합 자료구조를 사용하면 $\mathcal{O}(1)$ 에 원소 포함 여부를 확인할 수 있습니다.

# E. 족구왕 와니

#implementation #bruteforce #geometry #pythagorean
출제진 의도 – **Medium**

- **Onsite**
  - 제출 15번, 정답 2명 (정답률 20%)
  - 처음 푼 사람: **오량인8**, 31분
- **Open**
  - 제출 86번, 정답 48명 (정답률 55.814%)
  - 처음 푼 사람: **seawon0808**, 13분
- 출제자: **gmb9817**

## E. 족구왕 와니

- $N$ 명의 팀원은 연습 내내 고정된 위치에 서있습니다.
- 로봇은 총 $M$ 개의 공을 미리 정해진 목표 좌표로 순서대로 발사합니다.
- 공은 목표 좌표에 정확히 도달하며, 이동 중에는 찰 수 없습니다.
- 매 공마다 **독립적으로 처리 여부를 판단(중요!!)**합니다.

## E. 족구왕 와니

- 어떤 공의 목표 좌표를 $(T_x, T_y)$ 라 할 때, 각 팀원 $(X_i, Y_i)$ 에 대해 거리를 측정합니다.

$$\sqrt{(T_x - X_i)^2 + (T_y - Y_i)^2} \leq R_i$$

- 위 조건을 만족하는 팀원만이 해당 공을 찰 수 있습니다.

- 공을 찰 수 있는 팀원이 **아무도 없다면** 해당 공을 놓칩니다.
- 이 경우 재정비를 위해 **다음 차례의 공은 차지 않습니다**.
- 이를 구현하기 위해 연속 상태를 관리하는 기준점, 플래그를 하나 사용합니다.

## E. 족구왕 와니

”공을 연속해서 2번 놓친 경우에도 그 다음에 재정비를 하나요?”

- ✓ **공을 안 찬 경우**와 **공을 못 찬 경우**를 다르게 판정하셔야 합니다.
  - **공을 못 찬 경우**: 해당 공의 목표 좌표에 도달했을 때, 사정거리 내에 있는 팀원이 **아무도 없는 경우**를 의미합니다.
  - **공을 안 찬 경우**: 이전 공을 못 찬 경우로 인해 규칙에 따라 **의도적으로 해당 공을 건너뛰는 경우**입니다.
- ✓ 재정비는 **공을 못 찼을 때 단 한 번만** 발생합니다.
- ✓ 즉, 이전 공을 못 찬 경우 다음 공은 자동으로 무시되며, 그 공은 ”못 찬 것”이 아니라 ”안 찬 것”으로 처리됩니다.

## E. 족구왕 와니

- 공을 찰 수 있는 팀원이 1명 이상 있다면,
- 그중 **실력 수치 $S_i$ 가 가장 높은 팀원**이 공을 찹니다.
- 실력 수치는 모두 서로 다르므로 추가적인 동점 처리는 필요 없습니다.

## E. 족구왕 와니

- 첫 번째 팀원은 항상 와니입니다.
- 공을 찬 사람이 와니일 경우 정답 카운트를 1 증가시킵니다.
- 공을 찬 이후에도 모든 팀원은 다시 제자리로 돌아갑니다.
- 따라서 공을 독립적으로 처리하기 때문에, 팀원의 위치는 변하지 않습니다.

- 각 공마다 모든 팀원을 검사하므로 시간복잡도는

$$\mathcal{O}(NM)$$

- $N \leq 100, M \leq 1000$ 이므로 제한 시간내에 충분히 풀 수 있습니다!
- 아까 언급했던 **안 차는 경우와 못 차는 경우를 구분**해주시는게 키 포인트입니다.

# F. 대신고와 쿼리

#simulation #2D-array #query
출제진 의도 – **Medium**

- **Onsite**
  - 제출 8번, 정답 3명 (정답률 37.500%)
  - 처음 푼 사람: **오량인8**, 40분
- **Open**
  - 제출 73번, 정답 49명 (정답률 67.123%)
  - 처음 푼 사람: **yyyy7089**, 21분
- 출제자: **ymkzz123**

- 대신고는 **5층**, 각 층은 **1반** ~ **N반**으로 구성됩니다.
- 쿼리는 두 종류입니다.
  - 1 a b : $a$ 층 $b$ 반 + 상하좌우 반에 소음 +1
  - 2 a : $a$ 층에서 **가장 시끄러운 반** 출력
- 모든 소리는 누적되며, 마지막에는 전 층 중 **최대 소음 반을 출력합니다**.

- 1 a b 처리는 다음과 같습니다:

$$(a, b) \to (a, b), (a \pm 1, b), (a, b \pm 1)$$

  존재하는 좌표만 증가시키므로 인덱스 범위를 체크해야합니다.
- 2 a 쿼리 실행 시 → $a$ 층의 **최댓값 위치**(가장 작은 반 번호을 우선으로)를 출력합니다.
- 모든 쿼리 후 전체 5층에서 동일 방식으로 최대 반을 한번 더 출력해줍니다.
- 나이브한 시뮬레이션 기반으로 구현 가능하며 전체 시간복잡도는 $\mathcal{O}(Q + 5N)$ 입니다.

- **모든 쿼리가 2 a 형식만 존재하는 경우**
  - 소음이 단 한 번도 발생하지 않음 → 모든 칸이 0이므로,
  - 2 a는 최댓값 0 중 **가장 반 번호가 작은 곳 = 1반**이 됩니다.
  - 전체 결과도 **1층 1반** 출력으로 동일합니다.

최댓값 동일 시 층 오름차순 → 반 오름차순

- 즉, 발생 이벤트가 없더라도 결과는 결정가능하며 코드 작성시 이를 잡아줘야 하는게 함정 포인트였습니다.

# G. generator

#implementation #bruteforce #constructive

출제진 의도 – **Medium**

- **Onsite**
  - 제출 16번, 정답 2명 (정답률 12.500%)
  - 처음 푼 사람: **오량인8**, 66분
- **Open**
  - 제출 97번, 정답 37명 (정답률 38.144%)
  - 처음 푼 사람: **wlaud**, 22분
- 출제자: **gmb9817**

## G. generator

- 수미수열 문제의 손상된 데이터를 복구하는 문제입니다.
- 손상된 항은 $L_i = 0$으로 표시되며, 이를 $1$ 이상 $2\,000$ 이하의 정수로 채워야 합니다.
- 복구된 수열을 수미수열 판정기에 넣었을 때 의도한 출력 $S$ (yes 또는 no)가 나오도록 해야 합니다.
- 가능한 답이 여러 개라면 아무거나 출력해도 됩니다.

- **경우 1:** $S$ **= no**
  - 접두사와 접미사가 **절대 일치하지 않도록** 만들어야 합니다.
  - 가장 간단한 방법: **모든 값을 서로 다르게** 만들기
  - 이미 사용된 값들을 체크하고, 손상된 항을 만날 때마다 **사용되지 않은 최소값**으로 채웁니다.
  - 모든 값이 서로 다르면 어떤 $K$에 대해서도 접두사와 접미사가 일치할 수 없습니다.

- **경우 2:** $S$ **= yes**
  - 어떤 $K \leq \frac{N}{2}$ 에 대해 길이 $K$ 인 접두사와 접미사가 일치하도록 만들어야 합니다.
  - $K = 1$ 부터 순회하면서 가능한 $K$ 를 찾습니다:
    - 위치 $i$ 와 대칭 위치 $N - K + i$ 를 확인
    - 둘 다 손상되지 않았는데 값이 다르면 이 $K$ 는 불가능
    - 충돌하지 않는 $K$ 를 찾으면 → 손상되지 않은 값을 대칭 위치에 복사
    - 남은 손상된 값들은 임의의 값(예: 1)으로 채움

- 시간복잡도: $\mathcal{O}(N^2)$
  - no인 경우: $\mathcal{O}(N)$
  - yes인 경우: $K$ 마다 $\mathcal{O}(K)$ 검사 → 총 $\mathcal{O}(N^2)$
- **핵심 아이디어**
  - no: 모든 값을 달라지게 만들기
  - yes: 손상되지 않은 값끼리 충돌하지 않는 $K$ 찾아서 대칭 복사
- 문제 조건에서 답이 항상 존재함을 보장하므로, 위 전략은 항상 작동합니다.