2026 SciCom Qualification Test 해설

Official Solutions

by

kdy40929

2026년 03월 17일

Staff

출제

- 김도윤 kdy40929
- 박시을 phoenixrafael
- 박시유 siyu1121

검수

- pk661
- oh040411
- kuhyaku
- lunarlity
- dinojaemin
- mj1000j
- realpsdoingdamyoo
- kky085437
- yigzo

문제		의도한 난이도	출제자
A	SciComLove (2026)	**Easy**	`phoenixrafael`
B	낙서 제거	**Medium**	`kdy40929`
C	Split the GSHS 5	**Medium**	`kdy40929, siyu1121`
D	두쫀쿠 무료 나눔	**Hard**	`phoenixrafael`
E	매치메이킹	**Hard**	`kdy40929`
F	Split the GSHS 5 (Hard)	**Challenging**	`kdy40929`

A. SciComLove (2026)

`string`, `implementation`

출제진 의도 – **Easy**

- 내부 제출 17번, 정답 10명
- Open Contest 제출 92번, 정답 77명
- 출제자: `phoenixrafael`

A. SciComLove (2026)

Subtask 1 (25점)

- 문자열이 섞이지 않으므로 입력받은 그대로 출력하면 됩니다.

Subtask 2 (100점)

- 문자열이 섞여도 각 알파벳의 개수는 변하지 않습니다.
- "SciCom"에만 속하는 알파벳의 개수와 "Love"에만 속하는 알파벳의 개수를 각각 셉니다.
- 이는 반복문을 이용하거나, python의 s.count() 메소드 등을 이용할 수 있습니다.
- "SciCom"의 개수 x와 "Love"의 개수 y를 구했다면 각각을 x, y번 반복해서 출력하면 됩니다.

B. 낙서 제거

`greedy`
출제진 의도 – **Medium**

- 내부 제출 24 번, 정답 8 명
- Open Contest 제출 75 번, 정답 28 명
- 출제자: `kdy40929`

B. 낙서 제거

Subtask 1 (5점)

- $K = 1$ 이면 한 칸만 색칠 가능하므로 정답은 N 입니다.

Subtask 2 (13점)

- 모든 낙서가 같은 열에 위치하므로 직사각형은 $1 \times K$ 꼴로 만드는 것이 최적입니다.
- 따라서 정답은 $\lfloor \frac{N-1}{K} \rfloor + 1$ 입니다.

Subtask 3 (41 점)

- i 행부터 j 행까지 하나의 색종이가 채우는 경우, 색종이의 넓이의 최솟값은 $(j-i+1) \times (\max(a_i, a_{i+1}, \ldots, a_r) - \min(a_i, a_{i+1}, \ldots, a_r)$ 입니다.
- 이때 $a_i \le a_{i+1} \le \cdots \le a_j$ 이므로 넓이의 최솟값은 $(j-i+1) \times (a_j - a_i)$ 입니다.
- 위 식은 i 가 정해져 있을 때 j 의 값이 커짐에 따라 증가합니다.
- 색종이를 최소한으로 사용하려면 넓이가 K 이하이면서 최대한 많이 덮도록 해야 하므로, $i = 1$ 에서 시작하여 넓이가 K 이하가 되는 최대의 j 를 잡는 것을 반복하는 그리디 알고리즘이 성립합니다.
- 따라서 시간복잡도는 $O(N)$ 입니다.

Subtask 4 (100점)

- Subtask 3과 마찬가지로 i 행부터 j 행까지 하나의 색종이가 채울 때 색종이 넓이의 최솟값은 $(j-i+1) \times (\max(a_i, a_{i+1}, \ldots, a_r) - \min(a_i, a_{i+1}, \ldots, a_r)$ 입니다.
- 이 식 역시 i가 고정될 때 j가 증가하므로 Subtask 3과 같은 그리디 전략이 성립합니다.
- max와 min 값을 따로 관리해주면 SUbtask 3와 동일한 방법으로 해결할 수 있습니다.
- 따라서 시간복잡도는 $O(N)$ 입니다.

C. Split the GSHS 5

dp

출제진 의도 – **Medium**

- 내부 제출 37번, 정답 6명
- Open Contest 미출제
- 출제자: `kdy40929`, `siyu1121`

C. Split the GSHS 5

Subtask 1 (7점)

- $C = 1$ 이면 모든 학생이 순서대로 운행하므로 문제에서 주어진 대로 시뮬레이션하면 됩니다.

Subtask 3 (40점)

- $dp[i]$ 를 i 번째 학생까지 객실 층수에 도착하고 엘리베이터가 다시 1 층에 도착한 시각의 최솟값으로 정의합시다.
- 마지막 운행에서 j 번째 학생까지 태웠었다면 i 번째 학생까지 태워서 운행한 후 엘리베이터가 돌아오는 데까지 걸린 시각은 $\max(dp[j], t_i) + 2\max(f_{j+1}, f_{j+2}, \ldots, f_i) - 2$ 입니다.
- 엘리베이터에 한 번에 탈 수 있는 사람 수는 C 명으로 제한되므로 $dp[i] = \min(\max(dp[j], t_i) + 2\max(f_{j+1}, f_{j+2}, \ldots, f_i) - 2)\ (i - C \le j \le i - 1)$ 입니다. 이를 그대로 구현하면 시간복잡도는 $O(NC^2)$ 이 됩니다.

Subtask 4 (100점)

- Subtask 3의 풀이에서 시간복잡도가 $O(NC^2)$ 이 되는 이유는 $\max(f_j, f_{j+1}, \ldots, f_i)$ 를 구하는 과정 때문입니다.
- $j = i - 1$ 부터 1씩 줄여가며 $max(f_j, f_{j+1}, \ldots, f_i)$ 의 값을 저장해 놓으면 $O(NC)$ 에 문제를 해결할 수 있습니다.

D. 두쫀쿠 무료 나눔

`graphs`, `shortest_path`, `dijkstra`

출제진 의도 – **Hard**

- 내부 제출 46 번, 정답 0 명
- Open Contest 제출 40 번, 정답 11 명
- 출제자: `phoenixrafael`

D. 두쫀쿠 무료 나눔

- 각 학생이 받은 두쫀쿠 개수의 홀짝성만 고려하면 됩니다.
- 따라서 r_i 가 짝수인 영업 사원은 아무런 영향을 주지 못하므로 선택할 필요가 없습니다.

Subtask 2 (7점)

- 모든 영업사원이 학생 한 명에게만 나누어줍니다.
- 따라서 모든 학생에 대해 홀수 개의 두쫀쿠를 주는 영업사원 중 주는 두쫀쿠 개수가 최소인 것으로 골라 모두 더하여 출력하면 됩니다.

Subtask 3 (8점)

- 문제 조건에 의해 모든 학생에게 두쫀쿠를 적어도 하나 나누어주려면 모든 영업사원을 고용해야 합니다.
- $p_{i+1} = q_i + 1, p_1 = 1, q_M = N$을 만족하고, 각 영업사원이 나눠주는 두쫀쿠 수가 모두 홀수인지 확인하면 됩니다.

Subtask 4 (43점)

- r_i가 홀수이면 i번째 영업 사원이 $[p_i, q_i]$ 구간에 1을 더한다고 생각합시다.
- 이 상태를 차분 배열의 홀짝성으로 생각하면, p_i 위치와 $q_i + 1$ 위치의 홀짝성만 바꾸게 됩니다.

D. 두쫀쿠 무료 나눔

- 즉, 이는 각 영업 사원을 p_i와 $q_i + 1$를 잇는 무향 간선 하나로 생각할 수 있게 됩니다.
- 그리고 이 간선의 가중치를 해당 영업사원이 나눠주는 두쫀쿠 수로 잡으면 $(q_i - p_i + 1) \times r_i$ 입니다.
- 모든 학생이 홀수 개의 두쫀쿠를 받으면 차분 배열에서 1번과 $N + 1$번만 1이고 나머지는 모두 0이어야 합니다.
- 즉, 선택된 간선을 모으면 1번 정점과 $N + 1$번 정점만 홀수 번, 나머지 정점은 짝수 번 등장해야 하므로 이 간선들은 결국 1에서 $N + 1$로 가는 경로를 이룹니다.
- 따라서 정답은 이 그래프에서 1번 정점에서 출발해 $N + 1$번 정점으로 가는 최단 경로의 길이를 구하여 풀 수 있습니다.

D. 두쫀쿠 무료 나눔

- 문제 조건에 의해 간선을 항상 번호가 증가하는 방향으로만 따라 이동해야 합니다.
- $dp[i]$를 i번 정점까지 오는 최단 경로의 길이로 잡으면, $dp[i] = dp[j] + w(j, i)$ $(1 \leq j < i, j$⊠i 간선 존재)로 식을 세울 수 있습니다.
- 이렇게 DAG dp와 비슷한 방식으로 문제를 해결할 수 있습니다.
- 따라서 시간복잡도는 $O(N + M)$입니다.

Subtask 5 (100점)

- 다익스트라 알고리즘을 이용하면 $O(MlogN)$에 문제를 해결할 수 있습니다.

E. 매치메이킹

`implementation`, `simulation`, `data_structures`, `priority_queue`, `tree_set`

출제진 의도 – **Hard**

- 내부 제출 9번, 정답 0명
- Open Contest 제출 49번, 정답 8명
- 출제자: `kdy40929`

Subtask 1 (6점)

- 2번 쿼리가 주어질 때마다 현재까지 추가된 모든 사람 쌍을 탐색하며 레이팅이 최소인 쌍을 찾아 출력하면 됩니다.
- 이때 시간복잡도는 $O(Q^3)$ 입니다.

Subtask 2 (17점)

- 레이팅 차가 최소인 쌍은 레이팅 순으로 정렬했을 때 인접한 쌍에서만 발생합니다.
- 따라서 2번 쿼리가 주어질 때마다 현재까지 들어온 사람들을 모두 레이팅 기준으로 정렬하고 인접한 쌍만 탐색할 수 있습니다.
- 이렇게 구현할 경우 시간복잡도는 $O(Q^2 logQ)$ 입니다.

Subtask 4 (100점)

- 레이팅 순으로 정렬된 채로 사람들을 관리하기 위해 tree set을 활용합니다.
- 레이팅 차가 최소인 쌍을 찾기 위해 우선순위 큐를 이용합니다.
- 1번 쿼리가 주어진 경우 tree set에 사람을 추가합니다.
- 한 명의 사람을 추가하면 레이팅 기준 정렬에서 새로운 인접쌍이 2개 생기므로 이를 각각 우선순위 큐에 추가합니다.

E. 매치메이킹

- 2번 쿼리가 주어진 경우 우선순위 큐에서 레이팅 차가 최소인 쌍을 꺼내면서 두 사람 모두 아직 매치메이킹 시스템에 존재하는지 확인합니다.
- 두 사람 모두 매치메이킹 시스템에 존재하는 레이팅 최소 쌍을 얻었다면, 이를 출력하고 tree set에서 두 사람을 삭제합니다.
- 두 사람을 삭제함으로 인해서 원래 인접하지 않았던 인접쌍이 새로 생기므로 이를 다시 우선순위 큐에 추가합니다.
- 우선순위 큐에 들어가는 원소의 수는 최대 $4Q$개, tree set은 Q개이므로 시간복잡도는 $O(QlogQ)$가 됩니다.

F. Split the GSHS 5 (Hard)

`dp`, `segtree`, `lazyprop`
출제진 의도 – **Challenging**

- 내부 미출제
- Open Contest 제출 18 번, 정답 3 명
- 출제자: `kdy40929`

Subtask 5 (18점)

- $M = 2$이면 $dp[i] = \min(\max(dp[j], t_i) + 2\ (i - C \le j \le i - 1)$로 단순화됩니다.
- $dp[i]$는 단조성을 가지므로 $dp[i] = \max(dp[\max(i - C, 0)], t_i) + 2$입니다.
- 따라서 시간복잡도 $O(N)$에 문제를 해결할 수 있습니다.

Subtask 6 (100점)

- 세그먼트 트리를 이용하여 dp를 최적화합니다.
- 위 dp 식에서, $dp[i]$는 단조 증가하므로 $dp[p] \leq t_i < dp[p+1]$인 p를 잡을 수 있습니다.
- $j \leq p$에서 전이하는 경우 식은 $t_i + 2\max(f_j, f_{j+1}, \ldots, f_i) - 2$가 됩니다.
- 이 중 최솟값은 항상 $j = p$일 때 생기므로 $j = p$만 보면 됩니다.
- $\max(f_p, f_{p+1}, \ldots, f_i)$는 세그먼트 트리로 $O(logN)$에 구할 수 있습니다.

F. Split the GSHS 5 (Hard)

- $j > p$에서 전이하는 경우를 위해 lazy segment tree와 모노톤 스택을 구성합니다.
- $dp[i]$를 계산하려고 할 때 세그먼트 트리의 j번 노드는 $dp[j] + 2\max(dp[j+1], dp[j+2], \ldots, dp[i]) - 2$를 저장합니다.
- 그러면 $j > p$일 때 $dp[i]$의 값의 후보는 구간 최대 쿼리 한 번으로 구할 수 있습니다.
- i가 증가함에 따라 세그먼트 트리의 각 값을 업데이트해야 합니다.
- 이때 자신보다 앞에 위치하고 객실 층수가 높은 학생들을 모아둔 모노톤 스택을 만들면, 이 스택에 push하거나 pop할 때마다 구간 add 연산으로 세그먼트 트리 노드의 값을 갱신할 수 있습니다.
- 따라서 전체 시간복잡도 $O(NlogN)$에 문제를 해결할 수 있습니다.