제4회 디미고 프로그래밍 챌린지 해설

Official Solution

by

대회 운영진 일동

문제		예상 난이도	출제자
A	제4회 디미고 프로그래밍 챌린지	**Bronze IV**	ibasic
B	학습지 가져오기	**Silver IV**	ibasic
C	굴려라 주사위!	**Silver I**	fill2714
D	스택과 팰린드롬	**Silver I**	coding_dana
E	MEXMEX	**Gold V**	ibasic
F	디미고 통제	**Gold III**	jihoojh09
G	회귀의 그래프	**Gold II**	jihoojh09
H	보석 게임	**Gold II**	djangg7

문제		예상 난이도	출제자
I	최단 시간	**Platinum IV**	shinhuichan
J	트리 복제	**Platinum IV**	k_san
K	ibasic Sequences	**Platinum III**	ibasic
L	Factorial Attack	**Platinum III**	k_san
M	최단 거리와 쿼리	**Platinum II**	rumi348
N	아름다운 강산	**Diamond IV**	ibasic

A. 제4회 디미고 프로그래밍 챌린지

`case_work` `implementation`
예상 난이도 – **Bronze IV**

- 본 대회 – 제출 177번, 정답 48명 (정답률 38.418%)
- Open Contest – 제출 236번, 정답 154명 (정답률 65.678%)
- 처음 푼 사람 (본 대회): **frtfrt1**, 58분
- 처음 푼 사람 (Open Contest): **pauljjang410**, 1분
- 출제자: `ibasic`

- 지문 그대로를 구현하시면 됩니다.
- 여담으로, 대회 제출 가능 시간이 변경되면서 지문과 규칙이 일치하지 않게 되었습니다....

B. 학습지 가져오기

implementation
예상 난이도 – **Silver IV**

- 본 대회 – 제출 118번, 정답 36명 (정답률 37.288%)
- Open Contest – 제출 232번, 정답 146명 (정답률 62.931%)
- 처음 푼 사람 (본 대회): **frtfrt1**, 110분
- 처음 푼 사람 (Open Contest): **pkearth**, 2분
- 출제자: `ibasic`

- t 에서 최솟값을 가지는 한 원소를 선택하고, 최솟값을 가지는 배열의 모든 원소를 1 만큼 증가시키고, 그 원소를 t 에서 삭제하는 작업을 N 번 반복하면 됩니다.
- 답은 마지막으로 삭제된 원소의 값이 됩니다.
- 위 과정을 단순히 구현하면 $O(N^2)$ 에 해결할 수 있습니다.
- $O(N \log N)$ 의 시간복잡도를 가지는 정렬 알고리즘을 이용하여 $O(N \log N)$ 에도 해결할 수 있습니다.

C. 굴려라 주사위!

`greedy`
예상 난이도 – **Silver I**

- 본 대회 – 제출 228번, 정답 8명 (정답률 3.509%)
- Open Contest – 제출 165번, 정답 19명 (정답률 12.121%)
- 처음 푼 사람 (본 대회): **ppssbb09**, 503분
- 처음 푼 사람 (Open Contest): **lindelof**, 21분
- 출제자: `fill2714`

- 기본적으로는 앞의 $M-1$ 번에서 더블로 최대한 전진하고, 마지막 M 번째에서 $2N-1$ 칸을 이동하는 것이 최적입니다. 따라서 홀수 번째 무인도는 고려할 필요가 없습니다.
- 그래서 $2N$ 씩 전진하며, 무인도가 없을 때까지 -2 칸씩 줄여 가는 그리디 방식으로 해결할 수 있으며, 시간 복잡도는 $O(M+K)$ 입니다.
- $N=1$ 인 경우에는 항상 더블만 나오므로, $2N(M-1)$ 칸 이내에 무인도가 없다면 멈추지 못하고 M 번 모두 더블을 던지게 됩니다. 이때는 0 을 출력하고, 범위 안에 무인도가 있다면 처음으로 밟게 되는 무인도 번호를 출력하면 됩니다.

D. 스택과 펠린드롬

`ad_hoc` `string` `stack`
예상 난이도 – **Silver I**

- 본 대회 – 제출 52번, 정답 14명 (정답률 30.769%)
- Open Contest – 제출 71번, 정답 35명 (정답률 49.296%)
- 처음 푼 사람 (본 대회): **frtfrt1**, 481분
- 처음 푼 사람 (Open Contest): **namnamseo**, 7분
- 출제자: `coding_dana`

- 매번 가장 긴 팰린드롬을 직접 찾으면 비효율적입니다.
- 이전 단계가 끝난 뒤의 R에는 길이 2 이상의 팰린드롬이 존재하지 않습니다.
- 따라서 새 문자를 하나 추가했을 때 새로 생기는 팰린드롬은 반드시 마지막 문자를 포함해야 합니다.
- 그런데 길이 4 이상의 팰린드롬은 내부에 반드시 길이 2 이상인 팰린드롬을 포함합니다.

- 그 내부 팰린드롬은 마지막 문자를 포함하지 않으므로, 추가 전의 R에도 이미 있었어야 합니다.
- 이는 모순이므로, 새로 생길 수 있는 팰린드롬의 길이는 2 또는 3뿐입니다.
- 즉 마지막 2글자의 aa 형태, 마지막 3글자의 aba 형태만 확인하며 제거하면 됩니다.
- 각 문자는 한 번 들어오고 최대 한 번 삭제되므로 전체 복잡도는 $O(N)$입니다.

E. MEXMEX

`ad_hoc constructive case_work`

예상 난이도 – **Gold V**

- 본 대회 – 제출 91번, 정답 13명 (정답률 15.385%)
- Open Contest – 제출 94번, 정답 29명 (정답률 32.979%)
- 처음 푼 사람 (본 대회): **ppssbb09**, 519분
- 처음 푼 사람 (Open Contest): **mj1000j**, 15분
- 출제자: `ibasic`

- $K = 0$ 일 때는 연속 부분수열의 MEX 값이 0 이면 안되므로 모든 A_i 는 0 이어야 합니다.
- $K = 1$ 일 때는 연속 부분수열의 MEX 값이 1 이면 안되므로 모든 A_i 는 0 이 아니어야 합니다.
- $K = 2$ 일 때는 연속 부분수열의 MEX 값이 0 인 구간과 1 인 구간이 필요하므로 $N = 1$ 일 때는 구성할 방법이 없습니다. $N \geq 2$ 일 때는 다양한 풀이가 존재할 수 있으나, 2 이상의 원소와 0 이 모두 존재하고 1 이 존재하지 않는 수열을 구성할 경우 조건을 만족합니다.
- $K \geq 3$ 일 때는 다양한 풀이가 존재할 수 있으나, $A = [0, 1, \ldots, K-2, 0, \ldots, 0]$ 와 같이 구성할 경우 조건을 만족합니다.

F. 디미고 통제

`ad_hoc` `graphs` `graph_traversal` `dfs`
예상 난이도 – **Gold III**

- 본 대회 – 제출 39번, 정답 8명 (정답률 20.513%)
- Open Contest – 제출 30번, 정답 16명 (정답률 53.333%)
- 처음 푼 사람 (본 대회): **ecode**, 427분
- 처음 푼 사람 (Open Contest): **lsdas**, 17분
- 출제자: `jihoojh09`

F. 디미고 통제

- 한 정점에서는 최대 한 번의 통제 작업을 진행할 수 있고, 한 컴포넌트에서는 최대 (컴포넌트 크기) - 1 번의 통제 작업을 진행할 수 있습니다.
- 한 컴포넌트에서 dfs를 돌려봅시다. 탐색 과정에서 정점 A에서 정점 B로 이동하였다고 합시다. 이는 A와 B 사이에 간선이 존재한다는 의미이고, A에서 통제 작업이 진행되지 않았다면 어떠한 경우에도 B에서 통제 작업을 진행 할 수 있음을 의미합니다.

  즉, dfs 후위 순회 순서로 통제 작업을 진행하면 dfs를 시작한 정점을 제외하고 모든 정점에서 통제 작업이 가능합니다. B를 통제할 때 A가 아직 통제되지 않았기 때문입니다.

- 같은 원리로 dfs 전위 순회 역순과 bfs 순회 역순으로도 해결 가능합니다.

G. 회귀의 그래프

`graphs` `shortest_path` `dijkstra`
예상 난이도 – **Gold II**

- 본 대회 – 제출 39번, 정답 5명 (정답률 12.821%)
- Open Contest – 제출 22번, 정답 11명 (정답률 50.000%)
- 처음 푼 사람 (본 대회): **magic_spirit**, 711분
- 처음 푼 사람 (Open Contest): **hyperbolic**, 34분
- 출제자: `jihoojh09`

- 회귀 행동을 사용하지 않는 경우는 다익스트라 알고리즘을 이용하여 1번 정점과 N번 정점의 거리를 구하면 됩니다.

- 회귀 행동을 사용하는 경우는 K번 정점에서 회귀 행동을 하는 것이 최적임이 자명합니다. 따라서, K번 정점과 거리가 T 이하인 정점 i의 대하여, 1 -> K -> i(회귀) -> N 의 최소 시간을 구하면 됩니다.

- K번 정점에서 회귀 동작으로 정확히 정점 i에 도착하기 위해서는 K번 정점에서 T - dist(i, K) 시간만큼 기다려야 합니다. 그러면 총 시간은 dist(1, i) + dist(i, K) + (T - dist(i, K)) + dist(i, N)이 됩니다. 정리하면 T + dist(1, i) + dist(i, N)가 됩니다. 이는 1번 정점, K번 정점, N번 정점에서 다익을 돌려 전처리 하여 구할 수 있습니다.

H. 보석 게임

`game_theory` `dp`
예상 난이도 – **Gold II**

- 본 대회 – 제출 42번, 정답 4명 (정답률 9.524%)
- Open Contest – 제출 7번, 정답 6명 (정답률 85.714%)
- 처음 푼 사람 (본 대회): **magic_spirit**, 762분
- 처음 푼 사람 (Open Contest): **hyperbolic**, 46분
- 출제자: djangg7

H. 보석 게임

- dp 상태를 dp[i][j] = (현재 지하 i층에 있고, 지하 i+1층에서 선택된 열은 j열이다) 로 정의하면, 게임은 턴마다 최적 선택을 하는 dp 문제로 볼 수 있습니다.

- 홀수 층은 값을 최대화하고, 짝수 층은 값을 최소화합니다. 같은 열을 선택하면 다음 층에서 해당 열이 0이 되는 점을 고려해야 합니다.

- 각 층마다 모든 열을 직접 비교하면 $O(K^2)$ 이지만, 현재 층에서의 값 + 다음 dp에 대해 최댓값 2개 / 최솟값 2개를 미리 구하면 $O(K)$ 에 처리할 수 있습니다.

- 이전 열과 현재 열이 다른 경우에는 전체 최적값을 사용하고, 이전 열과 현재 열이 같은 경우에는 두 번째 최적값을 사용합니다.

- 즉, 점화식은 다음과 같습니다.

$$dp[i][j] = \begin{cases} \max\left(dp[i+1][j],\ \max\limits_{k \neq j}\left(s_{i,k} + dp[i+1][k]\right)\right), & (i \text{ is odd}) \\ \min\left(dp[i+1][j],\ \min\limits_{k \neq j}\left(-s_{i,k} + dp[i+1][k]\right)\right), & (i \text{ is even}) \end{cases}$$

- 이를 모든 층에 대해 역순으로 수행하면 전체 시간복잡도는 $O(RK)$ 입니다.
- 또한 $O(R \cdot KlogK)$ 도 가능합니다.

I. 최단 시간

`ad_hoc` `bruteforcing`
예상 난이도 – **Platinum IV**

- 본 대회 – 제출 29번, 정답 3명 (정답률 17.241%)
- Open Contest – 제출 14번, 정답 4명 (정답률 28.571%)
- 처음 푼 사람 (본 대회): **frtfrt1**, 555분
- 처음 푼 사람 (Open Contest): **hyperbolic**, 62분
- 출제자: `shinhuichan`

- 현재 갈 수 있는 구간을 관리한다. 구간이 겹치면 합쳐준다.
- 구간의 개수는 $O(\sqrt{min(n,t)})$ 개이다. 그래서 나이브하게 코드를 짜면 된다.
- 구간을 한개씩 늘려간다고 생각해보자. 1분이 지날때마다 구간은 최대 한개만 늘어난다. 총 k 개의 구간을 만들고 싶다면 적어도 k 만큼의 시간이 걸린다. 하지만 구간의 크기는 매분 2씩 늘어난다. 매분 늘어나는 구간의 크기를 다 더하면 k^2 을 넘는다.따라서 최악의 상황에도 루트개보다 더 적은 개수의 구간이 존재하게 된다.

J. 트리 복제

ad_hoc trees
예상 난이도 – **Platinum IV**

- 본 대회 – 제출 1번, 정답 0명 (정답률 0.000%)
- Open Contest – 제출 9번, 정답 3명 (정답률 33.333%)
- 처음 푼 사람 (본 대회): **N/A**
- 처음 푼 사람 (Open Contest): **hyperbolic**, 83분
- 출제자: k_san

- 모든 리프를 동시에 제거하면 각 A_i 에서 동일한 변화가 일어난다.
- 이 과정을 반복하면 트리는 결국 일자(path) 형태만 남는다.
- 남은 정점들은 각 A_i 의 루트에 대응된다.
- 남은 정점의 개수가 $k \geq 3$ 이면 $B = k$ 이다.
- 남은 정점의 개수가 2 보다 큰 짝수이면 $B = 2$ 일때와 구분할 수 없는데 이 경우는 $B = 2$ 인 해가 항상 존재한다.
- 위 과정을 통해 B 를 결정하고, 구조를 역추적하여 A 를 복원할 수 있다.

K. ibasic Sequences

`ad_hoc` `constructive`

예상 난이도 – **Platinum III**

- 본 대회 – 제출 8번, 정답 1명 (정답률 12.500%)
- Open Contest – 제출 14번, 정답 3명 (정답률 21.429%)
- 처음 푼 사람 (본 대회): **ppssbb09**, 581분
- 처음 푼 사람 (Open Contest): **fermion5**, 103분
- 출제자: `ibasic`

- 이 문제에는 다양한 풀이가 존재합니다.

- 모든 길이 3의 연속 부분수열의 중앙값이 $N+1$ 이상 $2N$ 이하인 경우를 생각해봅시다.
- 길이 3의 연속 부분수열은 N 이하의 원소와 $2N+1$ 이상의 원소가 각각 최대 1개만 존재합니다. 이에 따라 길이 N 이하의 원소, $2N+1$ 이상의 원소는 각각 연속으로 등장할 수 없습니다.
- 길이가 m인 연속 부분수열은 최대 $\lceil \frac{m}{3} \rceil$개의 N 이하 / $2N+1$ 이상 원소를 가지므로, 길이 5 이상의 연속 부분수열은 중앙값이 $N+1$ 이상 $2N$ 이하입니다.
- 그러므로, 모든 길이 4 이하의 연속 부분수열의 중앙값이 $N+1$ 이상 $2N$ 이하라면 길이 5 이상의 모든 연속 부분수열의 중앙값도 $N+1$ 이상 $2N$ 이하입니다.

- $N=1$일 때에는 조건을 만족하는 수열이 존재하지 않습니다. 2는 1 또는 3과 항상 인접하나, 그 두 원소가 포함되는 길이 2의 연속 부분수열의 중앙값은 2가 아니기 때문입니다.
- $N \geq 2$일 때는
  $A_1 = 1, A_2 = 2N+1, A_3 = 2N-1, A_{3N-2} = 2N, A_{3N-1} = 2, A_{3N} = 3N$이고,
  $2 \leq i \leq N-1$에서 $A_{3i-2} = 1+i, A_{3i-1} = 2N+i, A_{3i} = 2N-i$인 수열 A가 조건을 만족합니다.

- 여담으로, 체커는 문제 제작 당시에 길이 5인 연속 부분수열을 검사하지 않아도 될 것이라는 확신이 없었던 관계로 길이 5 이하의 연속 부분수열을 모두 검사하는 방식으로 구현되었습니다.

L. Factorial Attack

`case_work` `bruteforcing` `bitmask`
예상 난이도 – **Platinum III**

- 본 대회 – 제출 15번, 정답 0명 (정답률 0.000%)
- Open Contest – 제출 9번, 정답 3명 (정답률 33.333%)
- 처음 푼 사람 (본 대회): **N/A**
- 처음 푼 사람 (Open Contest): **ncy09**, 126분
- 출제자: k_san

- $N \geq 12$ 인 경우, 정답은 항상 1 또는 2 이다.
- $6 < N < 12$ 인 경우, 먼저 $(7!)$ 번의 공격만으로 가능한 최소 횟수 k 의 상한을 구한다.
- 이후 $\binom{\binom{N+1}{2}}{k-1}$ 의 모든 경우를 브루트포스로 탐색한다.
- $N \leq 6$ 인 경우, 가능한 구간은 최대 15 개이므로 2^{15} 가지 모든 경우를 브루트포스로 탐색한다.
- 이때 크기 2 이상의 같은 구간에 대한 공격을 2 번 이상 수행하는 것은 비효율적이므로 고려하지 않는다.
- 나머지 공격은 크기 1 구간에 대한 공격으로 채운다.

M. 최단 거리와 쿼리

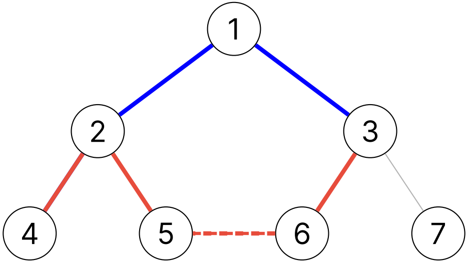
`lca shortest_path tree graphs graph_travel dijkstra`

예상 난이도 – **Platinum II**

- 본 대회 – 제출 10번, 정답 2명 (정답률 20.000%)
- Open Contest – 제출 15번, 정답 2명 (정답률 20.000%)
- 처음 푼 사람 (본 대회): **seojin3123**, 2847분
- 처음 푼 사람 (Open Contest): **alreadysolved**, 29분
- 출제자: rumi348

- 먼저, M이 N - 1인 경우를 생각해봅시다.
- 이러한 경우는 LCA 알고리즘을 이용해서 logN만에 쉽게 최단 거리를 구할 수 있습니다.

✓ 트리와 추가 간선 예시



파란색: 3-4의 LCA 경로 빨간색: 추가 간선 (5,6)을 이용한 경로

- 이제 M이 N인 경우, 즉 스패닝 트리에 간선이 하나 더 추가된 경우를 관찰해 봅시다.
- 이 경우는 두 가지로 나눌 수 있습니다.
- 1. 추가된 간선을 이용하지 않고 이동하는 경우
- 2. 추가된 간선을 이용하는 경우
- 1의 경우는 LCA를 이용해 쉽게 구할 수 있고, 2의 경우에는 추가된 간선이 u, v를 연결할 때 이를 경유하는 경로의 길이를 계산해 구할 수 있습니다.
- 이제 M이 N보다 큰 일반적인 경우에는, 추가된 간선의 양 끝점들에 대해 다익스트라를 전처리한 뒤 LCA로 구한 값과 비교하여 정답을 구하면 됩니다.
- 추가된 간선의 양 끝점대신, 한쪽 점들에 대해서만 다익스트라를 전처리해도 됩니다.

N. 아름다운 강산

`greedy`
예상 난이도 – **Diamond IV**

- 본 대회 – 제출 3번, 정답 0명 (정답률 0.000%)
- Open Contest – 제출 8번, 정답 3명 (정답률 37.500%)
- 처음 푼 사람 (본 대회): **N/A**
- 처음 푼 사람 (Open Contest): **alreadysolved**, 112분
- 출제자: `ibasic`

- N 이 홀수일 때는 아름다운 강산 조건을 만족할 수 없고, 짝수일 때는 만족할 수 있음은 자명합니다.

- $d_i = h_{i+1} - h_i$ 으로 합시다.
- 아름다운 강산 조건은 $|d_i| = 1$ 이고 모든 $1 \le j \le N$ 인 정수 j 에 대해 $\sum_{i=1}^{j} d_i \ge 0$ 을 만족하고, $\sum_{i=1}^{N} d_i = 0$ 을 만족해야 한다는 조건으로 표현할 수 있습니다.
- 괄호 문자열 조건과 동일하지 않나요?

- 어떤 $1 \leq i \leq N$ 인 i 를 선택해 d_i 에 1 을 더하거나 빼는 연산을 고려해 봅시다.
- 이 연산을 사용해 아름다운 강산으로 만든다면, $\sum_{i=1}^{N} d_i = 0$ 이어야 하므로 1 을 더하는 연산과 빼는 연산은 동일한 횟수로 사용됩니다.
- 괄호 문자열과 동일한 구조라는 점에서 착안했을 때, 모든 아름다운 강산 조건을 만족하는 수열 d 는 초기에 모든 $d_i = -1$ 로 두고 $k = 1, 3, \ldots, N-1$ 에 대해 $1 \leq i \leq k$ 이고 $d_i = -1$ 인 i 를 골라 1 로 변경하는 작업을 통해 만들 수 있고 다음 방법을 통해 만든 수열 d 는 항상 아름다운 강산 조건을 만족합니다.
- 따라서 위 문제는 그리디 알고리즘을 통해 해결할 수 있습니다. d_i 를 -1 로 만들 때와 1 로 만들 때의 연산 횟수는 $-2, 0, 2$ 만큼만 차이날 수 있으므로 우선순위 큐를 사용하지 않고도 $O(N)$ 에 구현할 수 있습니다.

- 문제에서 주어진 수정 연산은 $1 \le i, j \le N; i \ne j$ 인 i, j 를 선택해 $d_i \leftarrow d_i + 1, d_j \leftarrow d_j - 1$ 으로 바꾸는 것입니다.
- 이 연산은 앞페이지에서 설명한 d_i 증감 연산을 두 번 사용하여 똑같이 할 수 있습니다.
- 그러므로, 최소 수정 횟수는 d_i 증감 연산을 통해 아름다운 강산으로 만드는 최소 횟수의 절반 이상입니다.

- d_i 증감 연산을 통해 아름다운 강산으로 만드는 최소 횟수의 절반 횟수만큼만 수정연산을 사용하기 위해서는, $h_i \geq 0$ 조건을 만족하도록 값이 증가해야 하는 위치들과 감소해야 하는 위치들을 $1:1$로 짝지어야 합니다.
- 증가와 감소 연산을 아무렇게나 매칭해봅시다. 증가 연산 시행 횟수와 감소 연산 시행 횟수가 동일하므로 모든 원소를 매칭할 수 있습니다.
- 짝지어진 두 위치 중 증가해야 하는 위치가 더 앞선다면 그 사이 구간의 높이에 1을 더하는 연산이 되고, 감소해야 하는 위치가 더 앞선다면 구간의 높이에 1을 빼는 연산이 됩니다. 같은 위치에 증가 연산과 감소 연산을 둘 다 사용하는 경우는 없으므로 모든 연산은 유효한 수정 연산입니다.

- 구성한 연산들 중, 구간의 높이에 1 을 더하는 연산들을 모두 우선적으로 수행하고, 그 뒤에 1 을 빼는 연산들을 수행하도록 전체 순서를 재배치해 봅시다.
- 1 을 더하는 연산이 진행되는 동안에는 산의 높이가 초기 상태에서 계속 증가하기만 하므로, 모든 중간 과정에서 $h_i \geq 0$ 조건이 자명하게 유지됩니다.
- 이후 1 을 빼는 연산들이 진행되는 동안에는 산의 높이가 감소하나 목표 상태가 $h_i \geq 0$ 조건을 만족하므로 연산 중에도 $h_i \geq 0$ 을 만족합니다.
- 그러므로, d_i 증감 연산을 통해 아름다운 강산으로 만드는 최소 횟수의 절반 횟수가 답이 됩니다.