폴리매스 제1회 코드 챔피언십 - 후기 및 풀이

이 파일은 폴리매스 코드 챔피언십의 공식 후기 및 풀이입니다.
문제를 풀다가 중간에 막혀서 어려워하셨던 분들은 참고하시면 좋습니다.

총평

이번 PMCC는 제 예상에 비해 상당한 성공을 거두었습니다. 접수자는 총 19명이었고, 이 중 15명이 제출을 한 번 이상 했습니다. 대회 문제들의 난이도가 너무 어렵지 않나 하는 걱정을 했지만, 예상보다 많은 분들이 문제를 잘 풀어 주셔서 기뻤습니다.

A~B번 문제는 0점 방지용 문제였고, 예상대로 10명 가까이 되는 분들이 풀어 주셨습니다. C~D번 문제는 첫 네 문제를 너무 빨리 푸는 사람이 나오지 않기 위해 난이도를 조금 높였지만, 그럼에도 불구하고 parkky님이 대회 시작 7시간 만에 모두 풀어 주셨습니다.

다들 예상보다 문제를 잘 풀어 주셔서 기뻤지만, 한편으로는 다음 네 문제를 어떻게 준비해야 할지 막막했습니다. 그래서 다음 네 문제는 난이도를 올리기로 결심했습니다.

E~H 문제 공개 직전, A~D를 모두 푼 사람은 (수돌이님 포함) 3분이셨습니다. 예상보다 많았고, 한편으로는 다음 네 문제도 잘 풀어주실지 기대되었습니다.

드디어 마지막 네 문제가 공개되었습니다. E가 1시간 안에 풀리지 않을까 기대를 했지만, 그날이 평일이었던 탓이 오후 12시부터 참여하신 분은 많지 않았습니다. E가 처음으로 풀린 것은 공개 3시간 뒤였습니다. E의 난이도가 생각보다 많이 어려웠다는 사실에 당황했습니다. 이후로도 E는 많이 풀리지 않았습니다.

더욱 심각한 것은 제가 F의 난이도를 너무 과소평가했다는 사실입니다. 저는 F를 5명 정도 만점 맞지 않을까 예상했지만, 수돌이님을 제외하고 F를 맞은 사람은 없었습니다. 다행히도 F는 부분 점수 문제였고, 어느 정도 변별력은 갖출 수 있었습니다.

G는 예상치 못한 풀이가 나와서 놀랐습니다. 사실 의도된 풀이는 아래 적혀 있는 대로 필승 전략을 찾는 풀이었는데, 필승 전략을 직접 찾지 않고 DP를 통해 프로그램 내부에서 찾는 풀이가 나온 것입니다. 난이도가 훨씬 쉬워진 것인데, 딱히 막을 방법도 없었기 때문에 그냥 두기로 했습니다.

마지막 문제인 H는 매우 어렵게 출제했고, 어느 정도 사전 지식이 있어야만 풀 수 있는 문제였습니다. 아래 풀이를 보시면 알겠지만, 300점 중 100점을 받는 것도 썩 쉽지많은 않았습니다. 그런데도 100점을 받은 분이 꽤 많이 나와 놀랐습니다.

A. 개구리 1

- 최초 해결: Kimtaei (30분)
- 푼 사람 수: 12

이 문제는 대회에서 가장 쉬운 문제였고, 한 문제도 풀지 못하는 참가자가 생기는 일을 막기 위해 만들어졌습니다. 풀이를 생각하기도 어렵지 않고, 구현도 간단한 문제였습니다. 예상대로 가장 많은 사람이 풀었습니다.

서브태스크 1, 2, 3, 4, 5, 6 (100점)

일반성을 잃지 않고 $a, b \geq 0$이라고 가정해도 좋습니다. (절댓값을 취해주면 됩니다.)

$c < a + b$라면 애초에 (a,b)를 지나는 것조차 불가능합니다. 따라서 이때는 NO를 출력해야 합니다.

$c \geq a + b$라면, c와 $a + b$의 홀짝성을 비교합니다. 개구리가 한 번 움

직일 때마다 x좌표와 y좌표의 합은 홀짝성이 변합니다. 따라서 c와 $a+b$의 홀짝성이 같아야만 개구리가 c번의 이동 후 (a,b)에 위치할 수 있습니다.

따라서 아래 두 조건이 모두 만족하면 YES, 그렇지 않으면 NO를 출력하면 됩니다.

- $c \geq |a|+|b|$
- $c \equiv a+b \pmod 2$

시간 복잡도는 $O(1)$입니다.

B. 반복

- 최초 해결: Kimtaei (41분)
- 푼 사람 수: 9

이 문제 또한 쉬운 문제로 출제했고, A번에 비해 조금 더 사고력을 요구하는 문제입니다. 풀이를 떠올리는 것이 매우 쉬운 일은 아니지만, 어느 정도 실력을 가진 사람이라면 무난하게 풀 수 있는 문제였습니다.

서브태스크 1, 2, 3, 4, 5 (100점)

첫 번째 글자를 입력하기 위해서는 $K=1$이어야 합니다. 여기서 한 글자씩 더해 가면서 abcdefghijklmnopqrstuvwxyz를 한 번 더 입력해야 하는지 생각해 봅시다.

이번에 입력할 글자가 이전 글자보다 뒤에 있는 글자라면, a..z를 다시 입력할 필요가 없습니다. 이미 존재하기 때문입니다. 반대로 이번에 입력할 글자가 이전 글자보다 앞에 있거나, 이전 글자와 같다면, a..z를 한 번 더 입력해야 합니다.

시간 복잡도는 $O(L)$입니다.

C. 수열 만들기

- 최초 해결: kwoncycle (2시간 23분)
- 푼 사람 수: 6

C번 문제는 앞 두 문제에 비해 난이도가 높습니다. 수학적 사고력과, 풀이를 코드로 짜는 구현 능력을 테스트하는 문제였습니다. 이 문제는 제가 아닌 surface03님이 아이디어를 제공했습니다. 생각보다 푼 사람이 적어서 아쉬웠던 문제입니다.

서브태스크 1, 2 (100점)

이 문제를 그래프 이론으로 바꿀 수 있습니다. 1번, 2번, ..., N번 정점이 있는 그래프에서 임의의 $1 \leq i, j \leq N$에 대해 i번 정점에서 j번 정점으로 가는 유방향 간선이 있다고 할 때, 간선을 최대한 많이 거치는 회로를 구하는 문제입니다.

모든 점에서 들어오는 간선의 개수와 나가는 간선의 개수가 같기 때문에, 오일러 회로가 존재합니다. 자명하게도 오일러 회로보다 더 많은 간선을 포함할 수 있는 회로는 없으므로, 오일러 회로를 찾기만 하면 됩니다.

다양한 풀이가 있겠지만, 출제자가 생각한 수열은 아래와 같습니다.

예시) N=5일 때

1 5 2 5 3 5 4 5 5

1 4 2 4 3 4 4

1 3 2 3 3

1 2 2

1 1

위 다섯 개의 수열을 이어붙이면 오일러 회로가 만들어집니다. 위와 같은 규칙으로 N 값이 뭐든지 오일러 회로를 구성할 수 있습니다.

시간 복잡도: $O(N^2)$

D. 수열의 구간 평균

- 최초 해결: parkky (6시간 58분)
- 푼 사람 수: 4

이 문제는 첫 네 문제 중 가장 어려운 문제로, 뒤 네 문제가 공개되기 전까지 모든 문제를 다 풀어내는 사람을 줄이기 위해 만들어진 문제입니다. 하지만 예상보다는 많은 분들이 풀어 주신 것 같습니다.

서브태스크 1 (8점)

모든 구간을 직접 시행해 보는 $O(N^3)$ 알고리즘을 이용합니다.

서브태스크 2 (39점)

모든 구간을 다 시행해 봅니다. 대신, 각 구간의 수들의 합을 $O(1)$에 계산할 수 있어야 합니다. 이는 누적 합 기법(prefix sum)을 이용하면 됩니다. 초기에 전처리로 $a_1, a_1+a_2, a_1+a_2+a_3, \cdots$을 $O(N)$에 계산하고 나면, 임의의 구간에 있는 수들의 합은 누적 합 배열을 이용해 뺄셈 한 번으로 계산할 수 있습니다. 시간 복잡도는 $O(N^2)$입니다.

서브태스크 3 (53점)

모든 수에서 K씩 빼고 나면, 이 문제는 구간 합이 0인 구간의 개수를 구하는 것과 같습니다. 이는 누적 합 배열에서 서로 같은 수의 쌍 개수를 구하는 것과 같습니다. 누적 합 배열을 한 번 정렬한 뒤, 어떤 수가 i번 존재하면 $\frac{i(i-1)}{2}$를 배열에 더해주는 것으로 답을 계산할 수 있습니다. 시간 복잡도는 $O(N\log N)$ (정렬의 시간 복잡도)입니다.

E. 개구리 2

- 최초 해결: parkky(1048576) (3일 3시간)
- 푼 사람 수: 4

이 문제는 개구리 1 문제를 어렵게 만든 버전으로, A에 비해 난이도

가 꽤 높아졌습니다. 아이디어가 꽤나 어려웠고, 실제로 대회 시간 중에 이 아이디어를 생각해낸 사람은 많지 않을 거라고 예상했습니다.

서브태스크 1, 2, 3 (100점)

먼저 $a_i+b_i+c_i$의 홀짝성이 모두 같은지 확인하고 넘어갑니다. 만약 홀짝성이 다르다면, 답으로 존재하는 위치가 하나도 없음이 자명하므로 NO를 출력합니다.

만약 홀짝성이 모두 같다면, 더 이상 홀짝성을 신경 쓸 필요는 없습니다. 이제 범위에 주목합시다. 홀짝성 성질을 무시한다면, 각 개구리들이 처음에 있을 수 있는 위치의 범위는 마름모꼴로 주어집니다. 마름모꼴 모양을 연립부등식의 형태로 나타내면, 아래와 같습니다.

$$\begin{cases} a_i+b_i-c_i \le x+y \le a_i+b_i+c_i \\ a_i-b_i-c_i \le x-y \le a_i-b_i+c_i \end{cases}$$

위와 같은 연립부등식 N개를 연립해 풀어 주면 됩니다. 결국 부등식은 $x+y$와 $x-y$의 범위를 알려줄 것입니다. 만약 이 둘 중 존재할 수 없는 것(예를 들어 $3 \le x+y \le 1$과 같은 것)이 있다면 NO를 출력합니다. 아니라면, 가능한 모든 값들 중 x가 최소인 것을 찾으면 됩니다. 결국 $x=\dfrac{(x+y)_{MIN}+(x-y)_{MIN}}{2}$ $y=\dfrac{(x+y)_{MIN}-(x-y)_{MIN}}{2}$이 됩니다.

시간 복잡도는 $O(N)$입니다.

F. 직사각형

- 최초 해결: dillon0108 (3일 9시간 36분) [비공식]
- 푼 사람 수: 1

이 문제는 문제 공개 전날 급하게 만든 문제로, 동점자를 없애기 위해 만든 부분 점수 문제입니다. 5명 정도 만점을 받지 않을까 예상했지만, 실제 만점자는 훨씬 적었습니다. 후일담으로, 대회 8문제 중 제가 난이도를 가장 과소평가한 문제였습니다.

풀이

만점을 받기 위한 제한 조건인 50000은 15^4보다 조금 작습니다. 따라서 15^4를 만들 방법을 찾으면 됩니다. 이를 위해 먼저 30×30 사각형을 가로로 반, 세로로 반, 총 4개의 구역으로 나눕시다.

먼저 우리는 위에서 16번째, 아래에서 16번째 칸 (즉 오른쪽 아래 영역의 맨 왼쪽 위 칸)을 포함하는 사각형에 대해서만 생각할 것입니다. 오른쪽 아래 영역만 생각했을 때, (16, 16)을 포함하는 사각형의 개수는 총 225개인데, 여기 써있는 수들의 합이 0, 1, 2, ..., 224 가 정확히 1개씩 나오게 할 수 있습니다.

이제, 사각형을 위와 왼쪽으로 확장해 봅시다. 왼쪽으로 한 번 확장할 때마다 225를 더하고, 위로 한 번 확장할 때마다 225×15를 더합니다. 이렇게 되면 결국 15^4 미만의 모든 수들을 나타낼 수 있게 됩니다.

아래는 예시 답안입니다. 아래 예시는 57599까지 나타낼 수 있는 답안입니다. 위쪽으로 확장할 때마다 225×16씩 늘어나게 바꾸면 됩니다. 아래 답안을 조금만 수정하면 57600까지 나타내는 것도 가능합니다.

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3600	0	0	0	0	0	0	0	0	0	0	0	0	0	0
225	225	225	225	225	225	225	225	225	225	225	225	225	225	225	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0

G. 돌멩이 게임

- 최초 해결: dillon0108 (3일 10시간 27분) [비공식]
  parkky (3일 12시간 57분)
- 푼 사람 수: 4

이 문제는 muse와 참가자가 님 게임을 해서 이기는 문제로, 다른 문제와는 다르게 서로 게임을 해서 이기는 구조이기 때문에 준비하기 힘들었습니다. 또 필승 조건이 상식을 완전히 벗어나기 때문에 쉽지 않은 문제라고 생각했습니다.

서브태스크 1 (4점)

NO를 출력하고, 버퍼를 비웁니다.

서브태스크 2 (96점)

아래에서는 필승 전략을 소개합니다. 증명은 어렵지 않습니다.

1. **필승 조건**: $N \equiv 1,4 \pmod 5$
2. **아이디어**: 상대에게 $N \equiv 0 \pmod 5$인 상황을 넘겨주면 이긴다!
3. **승리 방법**: N을 5로 나눈 나머지가 1 또는 4인 경우에는, 한 번 또는 두 번의 턴을 이용해 muse에게 돌의 개수가 5의 배수인 상황을 넘겨줄 수 있습니다.

   muse가 가진 돌의 개수는 5의 배수이고, muse는 최대 3개의 돌을 가져갈 수 있게 됩니다. (최대 2개일 수도 있습니다.)
   이때 muse가 1개를 가져가면 1-1-1-2 or 1-1-2-1 패턴으로, muse가 2개를 가져가면 2-3 패턴으로, muse가 3개를 가져가면 3-2 패턴으로 응수하며 muse가 계속 돌의 개수가 5의 배수인 상황을 맞이하게 할 수 있습니다.

   이 과정을 반복하면 플레이어가 이깁니다.

H. 삼각 분할

- 최초 해결: 없음
- 푼 사람 수: 0

이 문제는 대회에서 가장 어려운 문제로, 아무도 대회 시간 안에 풀지 못할 문제를 만들겠다는 생각으로 낸 문제입니다.

서브태스크 1, 2 (100점)

정N각형을 삼각 분할하는 가짓수는 카탈란 수와 같다는 사실은 이미 널리 알려져 있습니다. 하지만 여기에서는 빨간색과 파란색 삼각형으로 나누어 색칠한 뒤, 한 변이 빨간색 삼각형인 삼각 분할만 빨간색 삼각형의 개수를 더해야 하기 때문에 문제의 난이도가 증가합니다.

하지만 잘 생각해 보면, 이것도 결국 카탈란 수와 비슷하게 삼각 분할과 관련되어 있기 때문에 카탈란 수와 비슷한 형태로 점화식이 나오지 않을까 추측해볼 수 있습니다. 실제로 삼각 분할 색칠의 개수는, 단순히 삼각 분할 개수와 같으므로, 앞으로 N번째 카탈란 수를 $C(N)$으로 표기하겠습니다. 이 표현은 문제에 많은 도움이 됩니다.

N+2각형에서 변 P_1P_2가 포함된 삼각형은 단 하나입니다. 변 P_1P_2가 빨간색으로 칠해진 $C(N)$개 경우에 대해 빨간색 영역의 개수를 ${}_Na_1, {}_Na_2, \cdots, {}_Na_{C(N)}$이라고 합시다. 또, 변 P_1P_2가 파란색으로 칠해진 경우에 빨간색 영역의 개수를 ${}_Nb_1, {}_Nb_2, \cdots, {}_Nb_{C(N)}$이라고 합시다. $a(N) = \sum_{i=1}^{C(N)} {}_Na_i$로 정의하고, 비슷한 방식으로 $b(N)$도 정의합니다.

이때 우리는 변 S가 포함된 삼각형의 나머지 한 꼭짓점이 어디냐를 이용해 아래와 같은 점화식을 세울 수 있습니다.

${}_Na_i : 1 + {}_xb_j + {}_{N+1-x}b_k$꼴로 표현

${}_Nb_i = {}_ya_j + {}_{N+1-y}a_k$꼴로 표현

$$a(N) = C(N) + 2\sum_{i=0}^{N-1} b(i)C(N-1-i)$$
$$b(N) = 2\sum_{i=0}^{N-1} a(i)C(N-1-i)$$

위 식을 이용하면 $O(N^2)$ 시간에 두 함수 a,b를 계산할 수 있습니다. 문제의 정의에 의해, 출력해야 하는 답은 $a(N-2)$입니다. 따라서 서브태스크 1, 2를 풀 수 있습니다.

서브태스크 3 (200점)

이 문제를 만점 받기 위해서는 위 점화식을 더 빠른 시간 안에 계산해야 합니다. 이것이 어떻게 가능할까요?

$b(n)$의 점화식을 $a(n)$에 대입하면, $a(n)$은 아래와 같은 형태로 정리할 수 있습니다.

$$a(n) = c(n) + \sum_{i=0}^{n-1} a(i)d(n-1-i)$$

참고로 직접 계산을 해보면, $d(n) = 4c(n-2)$입니다. 이것은 카탈란 수열의 점화식을 생각하면 당연합니다.

이러한 형태의 점화식을 어떻게 $O(N^2)$보다 빠른 시간 안에 구할 수 있는지 생각해 봅시다. 먼저 우리는 구해야 할 $a(i)$ 값들을 k개씩 나눠서, 한 번에 k개씩 N/k번 구하는 방법을 생각해볼 수 있습니다. 하지만 이 방법을 사용해서 이득이 되는 게 대체 뭐가 있을까요?

첫 k개의 값을 구하고 나면, 그 다음부터 우리는 남은 $N-k$개의 값에 첫 k개의 값이 미칠 영향을 계산할 수 있습니다. 일정 가중치를 두고 곱해지는 것인데요, 이 과정이 다항식의 곱셈과 상당히 유사하다는 것을 생각할 수 있습니다. 예를 들어,

$$(a_1x + a_2x^2 + a_3x^3 + \cdots + a_kx^k)\{d(n)x + d(n-1)x^2 + \cdots + d(1)x^n\}$$

에서 각 동류항의 계수와 같은 형태로 점화식이 계산될 것입니다.

두 다항식의 곱은 FFT(Fast Fourier Transform)을 이용해 빠르게 계산할 수 있습니다. 이 알고리즘의 자세한 설명은 생략합니다. FFT로 길이 N의 두 다항식을 곱할 때 걸리는 시간은 $O(N\log N)$입니다.

이제 시간 복잡도를 계산해 봅시다. FFT의 시간 복잡도가 전체 알고리즘의 시간 복잡도를 좌우하게 됩니다. FFT의 횟수는 N/k, FFT에 걸리는 총 시간은 $O(N\log N)$입니다. 반면 k개 항 내에서는 이중 for문으로 계산을 해야 하고, 이때 걸리는 시간은 Nk입니다. 따라서 시간 복잡도는 $O(N^2/k\log N+Nk)$이 됩니다. 이 값을 최소로 하는 k를 찾으려면, 미분을 해야 합니다. 하지만 곱이 일정하므로, 산술-기하 평균 부등식을 쓸 수 있겠네요. $N^2/k\log N+Nk \geq 2\sqrt{N^3\log N}$입니다. 따라서 $a(n),b(n)$을 구하는 최소 시간 복잡도는 $O(N\sqrt{N}\sqrt{\log N})$이 됩니다. 또 이때의 k 값을 구해보면 $\sqrt{N\log N}$이 됩니다.

실제로는 k 값을 약 3000으로 잡으면 시간이 4초 정도로 최소가 됩니다.