2023 KSA Automata Winter Contest

Official Solutions

by

Automata

문제		의도한 난이도	출제자
A	소수가 아닌 수	어려워요	flakepowders
B	그래서 대회 이름 뭐로 하죠	어려워요	eric00513
C	수학 퀴즈	어려워요	eric00513
D	2배 또는 0.5배	어려워요	flakepowders
E	퀸 움직이기	어려워요	eric00513
F	멋진 부분집합	어려워요	flakepowders
G	시그마 시그마 시그마 시그마	어려워요	eric00513
H	깃발 꽂기	어려워요	flakepowders
I	이상한 판 뒤집기 게임	어려워요	parkky
J	팬케이크 탑	어려워요	flakepowders

Thanks to

- 운영진
  - ai4youej
  - eric00513
  - flakepowders
  - parkky
- 후원
  - BaaaaaaaaaarkingDog
  - leejseo
  - utilforever
- 검수진
  - BaaaaaaaaaarkingDog
  - eaststar
  - heeda0528
  - hjroh0315
  - jk410
  - leejseo
  - kiwiyou
  - martinok1103
  - ssamt
  - tony9402
  - vector
  - whqkrkt04
  - wizardrabbit

A. 소수가 아닌 수

`number_theory`
출제진 의도 – **어려워요**

- 제출 1034번, 정답 348명 (정답률 36.577%)
- 처음 푼 사람: **gs18115**, 0분
- 출제자: `flakepowders`

A. 소수가 아닌 수

- N이 최대 10^9이고, 가능한 정답도 최대 10^9이므로 10^9은 N과 무관하게 항상 정답입니다.
- 따라서 10^9를 출력하면 됩니다.
- 별해로, N이 짝수이면 N, 홀수이면 $N+1$을 출력해도 됩니다.
- 이 경우 $N=1$과 $N=2$에 대해서는 예외 처리가 필요합니다.

B. 그래서 대회 이름 뭐로 하죠

`string, greedy`
출제진 의도 – **어려워요**

- 제출 773번, 정답 181명 (정답률 24.062%)
- 처음 푼 사람: **xiaowuc1**, 3분
- 출제자: `eric00513`

B. 그래서 대회 이름 뭐로 하죠

- 운영진은 실제로 몇 주 동안 대회 이름을 정하지 못했었습니다.
- **서브태스크 2**에서는 주어지는 S가 곧 대회 이름입니다.
- 맨 뒷 글자가 자음이고 그 앞에는 두 개의 A가 있는지 확인하면 됩니다.
- **서브태스크 3**에서는 그리디하게 글자를 선택하면 됩니다.
- 가장 뒤에 있는 자음을 꺼내고 뒤에 있는 글자들은 버립니다.
- 마찬가지로 남아 있는 글자들에서 가장 뒤에 있는 A를 꺼내고 뒤에 있는 글자들은 버립니다.
  A가 두 개 필요하므로 같은 작업을 한 번 더 합니다.
- 이제 남아 있는 글자들 중에서 $m-3$개의 글자를 붙여서 출력하면 됩니다.
- 만약 $m-3$개 미만의 글자가 남아있다면 답은 NO입니다.

C. 수학 퀴즈

`math`

출제진 의도 – **어려워요**

- 제출 374번, 정답 185명 (정답률 54.545%)
- 처음 푼 사람: **ychangseok**, 3분
- 출제자: `eric00513`

- **서브태스크 2**는 $\omega^2+\omega+1=0$에서 $\omega^2=-\omega-1$임을 이용하면 $A_1, A_2, \cdots, A_N$ 중 $0, 1, 2$의 개수를 세어 답을 간단히 계산할 수 있습니다.
- **서브태스크 4**에서 각 항의 차수를 2 이하로 줄이면 **서브태스크 2**와 같은 문제가 됩니다.
- $\omega^2+\omega+1=0$의 양변에 $\omega-1$을 곱하면 $\omega^3=1$이 성립함을 알 수 있습니다.
- 따라서 $\omega^c=\omega^{c \bmod 3}$입니다.
- p와 q는 항상 정수임이 보장되므로 실수 오차는 신경쓰지 않아도 됩니다.

- 별해로, $\omega^3 = 1$ 성질을 파악하지 않고 ω에 관한 두 일차식을 곱하는 과정을 반복해서 답을 구할 수도 있습니다.
- 앞 페이지의 $\omega^2 = -\omega - 1$을 이용하면 $(a\omega + b)(c\omega + d) = (-ac + ad + bc)\omega + (-ac + bd)$입니다.
- 위 식을 활용하여 ω^k을 구하면 됩니다.
- $\omega^0, \omega^1, \cdots, \omega^{10^6}$을 전처리한 후 더하면 **서브태스크 3**을 해결할 수 있습니다.
- 분할 정복을 이용한 거듭제곱을 구현하면 $\mathcal{O}\left(N \log 10^9\right)$의 시간복잡도로 **서브태스크 4**를 맞힐 수 있습니다.

D. 2배 또는 0.5배

`constructive`

출제진 의도 – **어려워요**

- 제출 227번, 정답 85명 (정답률 39.207%)
- 처음 푼 사람: **jjang36524**, 14분
- 출제자: `flakepowders`

- 인접한 수들의 차가 $1, 2$ 이 번갈아 나오게끔 하면 됩니다.
- 먼저 N 이 짝수일 때만 고려합니다.
- $[1, 2], [3, 4], [5, 6] \cdots$ 와 같이 1 부터 N 까지의 수들을 순서대로 나열한 후, 연속된 수들을 둘씩 묶습니다.
- 그 다음, $[2, 1], [3, 4], [6, 5] \cdots$ 처럼 홀수번째 묶음을 뒤집어주면 됩니다.
- N 이 홀수일 때는, 모든 수에 1 을 더하고 맨 앞에 1 을 추가해주면 됩니다.

E. 퀸 움직이기

`dp, prefix_sum`

출제진 의도 – **어려워요**

- 제출 191번, 정답 20명 (정답률 10.471%)
- 처음 푼 사람: **qwerasdfzxcl**, 35분
- 출제자: `eric00513`

- **서브태스크 1**은 시작 위치에서 도착 위치까지 퀸이 장애물을 지나지 않고 갈 수 있는지 확인하면 됩니다.
- 시간복잡도는 $O(N+M)$ 입니다.
- **서브태스크 2**는 완전 탐색으로 해결할 수 있습니다.
- 어떠한 칸에서 퀸이 이동할 수 있는 새로운 칸은 $4 \times \max(N, M)$ 을 넘지 않으므로 시간복잡도는 대략 $\mathcal{O}\left((4(N+M))^K\right)$ 입니다.

- **서브태스크 3**부터는 다이나믹 프로그래밍을 이용하면 됩니다.
- $dp[k][i][j][d]$를 "k번째 턴에 d방향으로 이동해서 (i, j) 칸에 도착하는 경우의 수"라고 합시다.
- 여기서 d는 0 이상 7 이하의 정수입니다.
- 예를 들어 $d = 0$(위로 이동)에 대해서 점화식을 세우면,
  $dp[k][i][j][0] = \sum_{d' \neq 0} (dp[k-1][i+1][j][d'] + dp[k-1][i+2][j][d'] + \cdots)$입니다.
- 시간복잡도는 $\mathcal{O}\left(KNM(N+M) \times 8^2\right)$입니다. 상수를 8^2에서 8로 줄일 수 있으나,
  서브태스크 4를 풀기 위해서는 더 빠른 방법이 필요합니다.

- **서브태스크 4**에서는 $\mathcal{O}\left(KNM \times 8^2\right)$ 시간 내에 풀어야 합니다.
- 앞서 세웠던 점화식에서 $dp[k][i+1][j][0]$에 포함되는 항들은 전부 $dp[k][i][j][0]$에 포함됨을 알 수 있습니다.
- 따라서 아래에서 위 방향으로 누적 합을 만들어서 계산하면 됩니다.
- 8가지의 방향에 대해 각각 누적 합을 만들면 문제를 $O(KNM \times 8^2)$에 풀 수 있습니다. 물론 상수를 더 줄일수도 있습니다.

F. 멋진 부분집합

`randomization`

출제진 의도 – **어려워요**

- 제출 212번, 정답 22명 (정답률 10.377%)
- 처음 푼 사람: **gs18115**, 21분
- 출제자: `flakepowders`

- **서브태스크 1**은 가능한 $\binom{N}{\lceil \frac{N}{2} \rceil}$ 개의 부분집합에 대해 최대공약수를 계산해보면 풀 수 있습니다.
- **서브태스크 2**는 집합 속 모든 수의 소인수들 중 절반 이상의 수의 약수인 것이 있는지 확인하면 됩니다.
- **서브태스크 3**에서는 모든 수를 소인수분해해서 각 소수들로 나누어떨어지는지를 확인하기에는 N 의 범위가 너무 큽니다.

F. 멋진 부분집합

- 만약 $\left\lceil \frac{N}{2} \right\rceil$ 개의 수들로 이루어진 멋진 부분집합이 존재한다면, 어떤 수를 골랐을 때 그 수가 멋진 부분집합에 속해 있을 확률은 $\frac{1}{2}$ 이상입니다.
- 따라서, 어떤 수를 임의로 골라 소인수분해한 뒤, 각 소인수들로 나누어떨어지는 수의 개수를 세주는 시행을 반복하면 충분히 높은 확률로 답을 찾을 수 있습니다.
- 시행을 k 번 반복했을 때, 멋진 부분집합이 있음에도 찾지 못했을 확률은 $1 - \left(\frac{1}{2}\right)^k$ 입니다. 시간 제한에 걸리지 않게 k 를 20 에서 30 정도로 조정해주면 됩니다.

G. 시그마 시그마 시그마 시그마

segtree

출제진 의도 – **어려워요**

- 제출 99번, 정답 22명 (정답률 23.232%)
- 처음 푼 사람: **flappybird**, 50분
- 출제자: eric00513

G. 시그마 시그마 시그마 시그마

- 입력을 받을 때 $A_i := A_i \bmod 998\,244\,353$를 하지 않는 것이 좋습니다. 나머지 연산은 두 수의 크기 관계를 유지시켜주지 않습니다.
- **서브태스크 1**은 문제에 주어진 식을 그대로 구현하면 $\mathcal{O}\left(N^4\right)$의 시간복잡도로 해결할 수 있습니다.
- **서브태스크 2**는 A_i가 몇 번 더해지는지 알아내면 됩니다.
- $\max(A_j, A_i)$는 $j(N-i+1)$번 더해집니다. $(j < i)$ 따라서 A_i는 $\frac{i(i-1)}{2} \times (N-i+1)$번 더해집니다.
- 답은 $\sum_{i=1}^{N} \frac{i(i-1)(N-i+1)}{2} \times A_i$입니다.

- **서브태스크 3**은 앞 페이지의 풀이와 거의 비슷합니다.
- $\max(A_i, A_j)$는 $i(N-j+1)$번 더해집니다. $(1 \le i < j \le N)$
- 따라서 답은 $\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} i(N-j+1) \times \max(A_i, A_j)$이며, $\mathcal{O}\left(N^2\right)$의 시간복잡도로 계산할 수 있습니다.

- **서브태스크 4**는 **서브태스크 3**에서 사용한 식을 빠르게 구할 수 있어야 합니다.
- 크기 N의 두 배열 X와 Y를 만들고 $A_1, A_2, \cdots, A_N$ 중 작은 수부터 차례로 살펴봅니다.
- 현재의 수를 A_i라고 할 때 답에
  $((N-i+1) \times sum(X[1..i-1]) + i \times sum(Y[i+1..n])) \times A_i$를 더합니다.
- 여기서 $sum(x)$는 합, $x[l..r]$은 부분배열을 의미합니다.
- 답을 업데이트한 후 $X[i]$를 i로, $Y[i]$를 $n-i+1$로 설정합니다.
- 위 과정을 펜윅/세그먼트 트리 등으로 처리하면 $\mathcal{O}(N \log N)$의 시간복잡도에 풀 수 있습니다.

H. 깃발 꽂기

`2_sat, graph_traversal`
출제진 의도 – **어려워요**

- 제출 57번, 정답 8명 (정답률 14.035%)
- 처음 푼 사람: **gs18115**, 63분
- 출제자: `flakepowders`

- i번 건물의 깃발 유무를 F_i라고 정의합시다. 즉 i번 건물에 깃발을 꽂으면 True, 꽂지 않으면 False입니다.
- 지상 통로의 경우, 간선이 연결하는 두 정점 중 적어도 하나에는 깃발을 꽂아야만 합니다.
- 즉, 지상 통로가 정점 i와 정점 j를 연결했을 때, $F_i \lor F_j =$True를 만족해야 합니다.
- 여기서 이 문제가 2-SAT 문제임을 추측할 수 있습니다.

- 구름다리의 경우, 구름다리만 고려하여 분리한 각 연결 그래프에서 최대 1개의 깃발이 있어야 합니다. 이러한 상황을 At-Most-One Constraint라고 합니다.
- 즉, 어떤 연결 그래프에 $b_1, b_2, \cdots, b_m$ 번 정점들이 포함되어 있다면, $1 \le i < j \le m$인 모든 순서쌍 (i, j)에 대해 $\neg F_{b_i} \lor \neg F_{b_j}$ =True를 만족하면 됩니다.
- 이를 그대로 2-SAT로 만들면 최악의 경우 clause의 개수는 $\frac{N(N-1)}{2} + A$가 되므로, **서브태스크 2**를 풀 수 있습니다.

- 이를 최적화하는 대표적인 방법은 $\mathcal{O}(N)$의 Ladder Encoding이나 $\mathcal{O}(N \log N)$의 Binary Encoding 등이 있습니다. 다음 슬라이드에서는 이 중 Ladder Encoding을 설명합니다.
- clause의 수를 줄이기 위해, 새로운 변수 $X_1, X_2, \cdots, X_N$을 도입합니다.
- 위 슬라이드처럼 $F_{b_1}, F_{b_2}, \cdots, F_{b_m}$ 중 최대 하나만 True가 되게 하려는 상황을 생각해 봅시다.
- $F_{b_1} \to X_{b_1}, F_{b_1} \vee F_{b_2} \to X_{b_2}, \cdots F_{b_1} \vee F_{b_2} \vee \cdots F_{b_m} \to X_{b_m}$으로 정의합시다.
- 이를 위해서는 $F_{b_i} \to X_{b_i}$와 $X_{b_i} \to X_{b_{i+1}}$을 clause에 추가하면 됩니다.
- 마지막으로 $X_{b_i} \to \neg F_{b_{i+1}}$를 추가해 주면, $F_{b_1}, F_{b_2}, \cdots, F_{b_i}$ 중 하나가 True일 때 $F_{b_{i+1}}$이 True가 될 수 없게끔 할 수 있습니다.

I. 이상한 판 뒤집기 게임

ad_hoc, game_theory
출제진 의도 – **어려워요**

- 제출 26번, 정답 0명 (정답률 0.000%)
- 처음 푼 사람: **없음**
- 출제자: parkky

- 게임이 시작될 때 판 i가 파란색이라면 $B_i = 1$, 빨간색이라면 $B_i = 0$이라고 둡시다.
- 버튼 i가 눌러질 때 뒤집기를 수행한다면 $A_i = 1$, 그렇지 않는다면 $A_i = 0$이라고 둡시다.
- 게임이 종료될 때 판 i가 파란색이라면 $D_i = 1$, 빨간색이라면 $D_i = 0$이라고 둡시다.
- 편의를 위해 임의의 i에 대해 $A_i = A_{i+N}, B_i = B_{i+N}, D_i = D_{i+N}$가 성립하도록 둡시다.
- $\oplus$는 xor 연산입니다. $(0 \oplus 0 = 1 \oplus 1 = 0, 0 \oplus 1 = 1 \oplus 0 = 1)$
- $D_i = B_i \oplus A_i \oplus A_{i-1}$가 성립합니다.

- ✓ $\bigoplus_{i=0}^{N-1} B_i = 1$일 때: $T =$ here를 고릅니다.
  - 게임 규칙에 맞게 아무렇게나 두는 전략을 사용합니다.
  - 증명은 아래와 같습니다.
    - ▶ $\bigoplus_{i=0}^{N-1} D_i = \bigoplus_{i=0}^{N-1} B_i = 1 \neq \frac{N}{2} \mod 2$
    - ▶ $\sum_{i=0}^{N-1} D_i \neq \frac{N}{2}$

- ✓ $\bigoplus_{i=0}^{N-1} B_i = 0$, $M =$ here일 때: $T =$ everyone을 고릅니다.
  - $A_i \oplus A_{i+N/2} = (i \bmod 2) \oplus \bigoplus_{j=i+1}^{i+N/2} B_j$가 성립하도록 하는 전략을 사용합니다.
  - 증명은 아래와 같습니다.
    - $D_i \oplus D_{i+N/2} = B_i \oplus B_{i+N/2} \oplus A_i \oplus A_{i+N/2} \oplus A_{i-1} \oplus A_{i+N/2-1} = 1$
    - $D_i + D_{i+N/2} = 1$
    - $\sum_{j=0}^{N-1} D_j = \frac{N}{2}$

- ✓ $\bigoplus_{i=0}^{N-1} B_i = 0, M =$ everyone 일 때: $T =$ here 를 고릅니다.
  - $A_i \oplus A_{i+N/2} = \left(\left(\frac{N}{4}+1\right) \bmod 2\right) \oplus \bigoplus_{j=i+1}^{i+N/2} B_j$ 가 성립하도록 하는 전략을 사용합니다.
  - 증명은 아래와 같습니다.
    - ▶ $D_i \oplus D_{i+N/2} = B_i \oplus B_{i+N/2} \oplus A_i \oplus A_{i+N/2} \oplus A_{i-1} \oplus A_{i+N/2-1} = 0,\ D_i = D_{i+N/2}$
    - ▶ $\bigoplus_{j=i+1}^{i+N/2} D_j = A_i \oplus A_{i+N/2} \oplus \bigoplus_{j=i+1}^{i+N/2} B_j = \left(\frac{N}{4}+1\right) \bmod 2 \neq \frac{N}{4} \bmod 2$
    - ▶ $\sum_{j=i+1}^{i+N/2} D_j \neq \frac{N}{4}, \sum_{j=0}^{N-1} D_j = 2 \cdot \sum_{j=i+1}^{i+N/2} D_j \neq \frac{N}{2}$

J. 팬케이크 탑

`combinatorics, probability`

출제진 의도 – **어려워요**

- 제출 39번, 정답 3명 (정답률 10.256%)
- 처음 푼 사람: **serin**, 64분
- 출제자: `flakepowders`

- **서브태스크 2**는 $dp[i][j][k]$를 "위에서부터 i번째 팬케이크까지 고려할 때, j개가 상했고 i번째 팬케이크의 상태가 k일 확률"이라고 정의하면 됩니다.
- $\mathcal{O}\left(N^2\right)$의 시간 복잡도에 계산할 수 있습니다.
- **서브태스크 3**을 풀기 위해서는 다르게 접근해봅시다.
- 팬케이크를 위에서부터 하나씩 뽑는다고 생각해봅시다.
- N개의 팬케이크 중 상한 것과 상하지 않은 것 중 어느 것이 더 많은지 판별할 때, 한 종류를 $(N+1)/2$개 뽑는 순간 결과는 어느 쪽이 더 많은지 결과는 확정됩니다.
- 따라서 한 종류가 $(N+1)/2$개 나온 이후는 반드시 다른 종류가 나온다고 가정해도, 어느 한 종류가 더 많을 확률에는 영향을 미치지 않습니다.

- 상한 것이 더 많은 경우를 고려해봅시다. 위 가정에 따라 상한 것이 정확하게 $(N+1)/2$개, 상하지 않은 것이 정확히 $(N-1)/2$개라고 생각해도 됩니다.
- 상한 것이 정확히 $(N+1)/2$개이므로, N개의 팬케이크 중 “상할 확률이 p/q인 것”은 상한 팬케이크(마지막 것 제외) 바로 뒤 팬케이크 $(N-1)/2$개와 맨 위의 팬케이크 1개를 합쳐 정확히 $(N+1)/2$개입니다.
- 핵심은 상한 것이 더 많은 경우, 상한 것이 p/q 확률로 등장하는 것은 정확히 $(N+1)/2$개라고 고정해도 상관없다는 사실입니다.

- 즉, 주어진 조건에서 상한 것이 더 많을 확률을 구하는 문제는 다음을 구하는 것과 동치입니다.
  - N개의 팬케이크가 있고 그 중 "상할 확률이 p/q인 것"이 $(N+1)/2$개, "상할 확률이 $1-p/q$인 것"이 $(N-1)/2$개일 때 상한 것이 더 많을 확률
- 이제 바꾼 문제를 풀어 봅시다.
- "상할 확률이 p/q인 것" 하나는 뽑지 않고, "상할 확률이 p/q인 것"과 "상할 확률이 $1-p/q$인 것"을 각각 $(N-1)/2$개씩 뽑는다고 생각해 봅시다.
- 위 문제에서, 상한 것과 상하지 않은 것은 모든 조건이 같습니다.
- 따라서 $N-1$개 중 상한 것이 더 많을 확률과 상하지 않은 것이 더 많을 확률은 같습니다.
- 신경써야 할 부분은 상한 것과 상하지 않은 것의 개수가 같은 경우인데, 이 확률을 k라고 합시다.

- 식을 짧게 줄이기 위해 $n = (N-1)/2, x = p/q$라고 둡니다.
- "상할 확률이 p/q인 것" 중 실제로 i개가 상한 것이었을 때, 그러한 상황이 발생할 확률은 $\binom{n}{i}^2 x^{2i}(1-x)^{2(n-i)}$ 입니다.
- 따라서 $k = \sum_{i=0}^{n} \binom{n}{i}^2 x^{2i}(1-x)^{2(n-i)}$ 입니다.
- 결론적으로 상한 것이 더 많을 확률은 $N-1$개를 뽑으면서 이미 결과가 결정된 경우의 확률 $(1-k)/2$와 $N-1$개를 뽑고도 결과가 결정되지 않은 경우의 확률 kx를 더해서 $(1-k)/2 + kx$ 입니다.
- 이것을 빠른 시간 내에 구하면 **서브태스크 3**을 풀 수 있습니다.