Cartoon Cup: ONE

areian13 cdg0228 corkd hibye1217 iktk inwooleeme kaorin ksoosung77
kuhyaku minjunnick nflight11 ohwphil qvixnh22 sksms1375 terrasphere utilforever yyyy7089

by

toycartoon

2025.12.21

문제		의도한 난이도	출제자
A	toycppartoon	**ADVANCED**	toycartoon
B	트리 선물하기	**MASTER**	hibye1217
C	볼록 껍질과 쿼리	**EXPERT**	yyyy7089
D	괄호 조작하기	**ADVANCED**	yyyy7089
E	쿠폰 잘 사용하기?	**BASIC**	sksms1375
F	수열과 인테그랄	**ADVANCED**	toycartoon
G	The sun is a deadly lazer	**EXPERT**	hibye1217
H	1, 2, 3, 많다!	**BASIC**	yyyy7089
I	이진 수열 숫자 야구 게임	**MASTER**	yyyy7089
J	The wheel to the right	**WORLD'S END**	yyyy7089
K	토카는 햄버거를 뿌려라	**EXPERT**	toycartoon, ohwphil
L	검은칠된 격자에 보내는 최단경로	**MASTER**	toycartoon
M	소인수분해 게임	**EXPERT**	yyyy7089

A. toycppartoon

`#implementation`, `#string`

출제진 의도 – **ADVANCED**

- 제출 244번, 정답 58명 (정답률 24.180%)
- 처음 푼 사람: **red_fox**, 5분
- 출제자: **toycartoon**

- 지문에 주어진대로 토카의 새로운 계정 이름을 구하면 됩니다.
- `toycartoon`, `toy`, `artzoo` 등 실수할 수 있는 여지가 많으니 주의해주세요.

toy*cawardtoon*

- ✓ toycppartoon 계정에 링크된 구글 폼에서 가장 빨리 고득점 - **`cubic`**
- ✓ 105점, 58분

토카의 눈 앞에 숨긴 첫 편지

- toycawardtoon의 상품 맞추기 - **익명의 한별이**
- -분 -초
- 출제자: **hibye1217, qvixnh22, yyyy7089**

- 구글 폼에 적힌 언어의 앞 글자를 따면 **ESOLANG CRAFTED BY OUR TEAM**가 나옵니다.
- 이를 토대로 Esolang Wiki에 익명에 인사 잘하는 12월 17일생의 핸들을 넣으면 언어 Matcha가 나옵니다.
- 언어 명세 Step 4의 명령어들 설명의 첫 글자를 보면 **STEAM GAME**이 나옵니다.
- 이를 조합해 Steam Game + Matcha로 정답을 구할 수 있습니다.
- **네, 저희는 이 특별상을 위해 언어를 만들었습니다!!**

B. 트리 선물하기

#ad_hoc, #constructive

출제진 의도 – **MASTER**

- 제출 17번, 정답 9명 (정답률 52.941%)
- 처음 푼 사람: **tony9402**, 36분
- 출제자: **hibye1217**

- 토카가 좋아하는 트리에서 0과 $2N$은 항상 찾을 수 있습니다. 따라서 S에 0이 없으면 답은 -1입니다. 또한 $2N = maxS$여야 합니다.
- 그럼 S에 0이 있으면 항상 토카가 좋아하는 트리를 만들 수 있을까요? 네, 항상 만들 수 있습니다! 출제자보다 더 깔끔하게 푼 검수자의 구성 방식을 소개하겠습니다.

- S의 두 인접한 원소 S_i와 S_{i+1}에 대해 S_{i+1}부터 $S_{i+1}-1$까지의 홀수를 역순으로 채워넣은 수열을 X라고 합시다. X_i가 X_{i+1}를 오른쪽 자식으로 삼는 선형 트리를 만들어준 뒤, 남은 부분에 짝수를 순서대로 채우면 됩니다.
- 예로 $S=\{0,6,8\}$이라고 합시다. 그럼 우리가 원하는 수열은 $X=\langle 5,3,1,7\rangle$이 됩니다. 여기에 짝수를 순서대로 채우면 중위 순회가 $\langle 0,5,2,3,4,1,6,7,8\rangle$이며 $\langle 5,3,1,7\rangle$이 일직선이 트리가 나오게 됩니다.
- 이 방식의 구성을 하면 S_i에 있는 수는 $S_{i+1}-1$을 만나서 제대로 찾게 되지만, 그렇지 않은 수들은 $S_{i+1}-1$을 만나서 찾을 수 없는 수가 됩니다.

트리 선물받기

- 문제 제출자 중 대회 날짜 기준 (solved.ac에 등록된) 생일이 가장 빠른 사람 - **kwoncycle**
- 8월 8일(231일 후)

C. 볼록 껍질과 쿼리

#ad_hoc, #convex_hull

출제진 의도 – **EXPERT**

- 제출 26번, 정답 14명 (정답률 53.846%)
- 처음 푼 사람: **sjh1224**, 19분
- 출제자: **yyyy7089**

- 어떤 세 점을 잡더라도 그 세 점의 볼록 껍질은 그 세 점과 같습니다.
- 한 번에 셋보다 많은 점을 쿼리할 경우 볼록 껍질이 그 세 점과 같지 않아 정보에 손실이 있을 수 있습니다.

- 만약 6 개의 점 $1, 2, 3, 4, 5, 6$ 이 있다면 $(1, 2, 3)$, $(3, 4, 5)$, $(2, 4, 6)$ 를 각각 쿼리하는 것으로 $\frac{6}{2}$ 번의 쿼리만에 6 개의 점을 각각 구별할 수 있습니다.
- 같은 방법으로 $5 \le x \le 10$ 을 만족하는 모든 x 에 대해 $\lceil \frac{x}{2} \rceil$ 번의 쿼리만에 x 개의 점을 각각 구별할 수 있습니다.
  - 5: $(1, 2, 3)$, $(2, 3, 4)$, $(3, 4, 5)$
  - 6: $(1, 2, 3)$, $(3, 4, 5)$, $(2, 4, 6)$
  - 7: $(1, 2, 3)$, $(3, 4, 5)$, $(5, 6, 7)$, $(2, 4, 6)$
  - 8: $(1, 2, 3)$, $(4, 5, 6)$, $(1, 4, 7)$, $(2, 5, 8)$
  - 9: $(1, 2, 3)$, $(4, 5, 6)$, $(1, 4, 7)$, $(2, 5, 8)$, $(3, 6, 9)$
  - 10: $(1, 2, 3)$, $(3, 4, 5)$, $(5, 6, 7)$, $(7, 8, 9)$, $(2, 8, 10)$
- N 을 $6k + x$ 으로 쪼개면 $5 \le N$ 를 만족하는 모든 N 에 대해 이 문제를 해결할 수 있습니다.

볼록 껍질과 특별상

- AC를 받은 코드 중 공백이 아닌 모든 문자를 그 중앙의 점으로 대체했을 때 그 점들의 볼록 껍질에 포함되는 점의 개수가 가장 작은 코드를 제출한 사람 - **oolala**
- 제출번호 101237879, 5각형, 68분

D. 괄호 조작하기

#stack, #bruteforcing, #constructive

출제진 의도 – **ADVANCED**

- 제출 47번, 정답 33명 (정답률 70.213%)
- 처음 푼 사람: **Intanz**, 9분
- 출제자: **yyyy7089**

- 주어지는 수의 범위가 충분히 작으므로 각 인덱스를 적절히 바꾸고 스택을 이용해 시뮬레이션하면 총 $\mathcal{O}\left(N|S|^2\right)$ 에 판정 가능합니다.

올바르지 않은 괄호 쌍

- 대회 전체에서, '괄호 쌍이 맞지 않음' 사유로 컴파일 에러를 가장 먼저 받은 사람 - **ishs311317**
- 제출번호 101239114, 122분

E. 쿠폰 잘 사용하기?

#greedy, #string, #constructive

출제진 의도 – **BASIC**

- 제출 106번, 정답 81명 (정답률 65.079%)
- 처음 푼 사람: **ddungsam**, 1분
- 출제자: **sksms1375**

- 먼저 N 과 $|A|$ 를 비교해봅시다.
- $N > |A|$ 라면 얼룩의 유무와 관계없이 항상 가능합니다.
- $N < |A|$ 라면 얼룩의 유무와 관계없이 항상 불가능합니다.
- 따라서 $N = |A|$ 일 때만 확인을 하면 됩니다. 얼룩을 어떻게 바꿔야 할까요? 생각해보면 1 에서 9 까지 수 중에서 9 가 가장 크므로 얼룩을 9 로만 바꾸면 쿠폰의 최댓값이 됩니다.
- A 의 범위가 매우 크므로 문자열로 받거나 큰 수 연산을 지원하는 언어를 사용하여 비교할 수 있습니다.

쿠폰 잘 사용하기 :smiling_face_with_3_hearts:

- ✓ 제출 창에 적힌 정보 (제출 번호, 시간, 메모리, 코드 길이)에 9가 가장 많이 적힌 사람 - `red_fox`
- ✓ 제출번호 101236994, 4개, 37분

F. 수열과 인테그랄

#calculus, #geometry, #constructive

출제진 의도 – **ADVANCED**

- 제출 78번, 정답 29명 (정답률 53.333%)
- 처음 푼 사람: **jeoffrey0522**, 17분
- 출제자: **toycartoon**

- 넓이는 작은 범위에서 직접 계산해보면 $N - 1 + N^2$ 이라는 식을 구할 수 있습니다.
- A 는 1, (양수들), 0, (음수들), -1 꼴 또는 -1, (음수들), 0, (양수들), 1 꼴로 구성하면 됩니다.
- 이게 왜 될까요? 엄밀한 증명해주신 ohwphil 님께 감사를 표합니다.

- A에 대한 가정을 세워봅시다.
- **가정.** A는 $N \geq 2$에서는 1, (양수들), 0, (음수들), -1 꼴 또는 -1, (음수들), 0, (양수들), 1 꼴밖에 존재하지 않는다.
- 두 경우로 이 **가정**을 증명해 봅시다.
- $f(x)$와 x축 사이 넓이는, $N(N+1)$에서 일정한 값을 뺀 것입니다. 이 일정한 값은 $\frac{|A_1| + |A_{2N+1}|}{2}$+(양수에서 음수로, 음수에서 양수로 교차하면서 생긴 넓이 손실)과 같습니다. 부호가 바뀌면서 생긴 넓이 손실은 $\frac{xy}{x+y} = \frac{1}{\frac{1}{x} + \frac{1}{y}}$와 같습니다. (편의상 양수, 음수에 0이 포함된다고 가정합니다.)

- **Case 1.** 0이 양 끝 중 하나에 있지 않은 경우
- 우선 $\frac{|A_1| + |A_{2N+1}|}{2} \geq \frac{1}{2}$ 이고, 등호는 $A_{2N+1} = 1$ 일 때 성립합니다.
- 그리고 (양 $\rightarrow$ 음, 음 $\rightarrow$ 양 넓이 손실) ≥ 0 이고, 등호는 모든 transition 에서 한 끝 점이 0일 때만 성립합니다.
- 따라서 가능한 최소 넓이 손실은 총 1입니다.
- 이 경우 최적해를 만족하는 해는 1, (양수들), 0, (음수들), -1 꼴 또는 -1, (음수들), 0, (양수들), 1 꼴 밖에 없습니다.

- **Case 2.** 0이 양 끝 중 하나에라도 있는 경우
- 편의상 $A_1 = 0$으로 둡시다. 우선 $\frac{|A_1| + |A_{2N+1}|}{2} \geq \frac{1}{2}$이고, 등호는 $|A_{2N+1}| = 1$일 때 성립합니다.
- 이제, (양 $\to$ 음, 음 $\to$ 양 넓이 손실)을 봅시다. 일반성을 잃지 않고 $A_{2N+1} = 1$로 둡시다. 맨 처음의 0을 제외하면, 최소한 한 번의 부호 전환이 있습니다.
- 그런데 부호 전환에서 발생하는 넓이 손실의 합은 항상 $\frac{1}{2}$ 초과입니다.

- $\frac{1}{\frac{1}{x}+\frac{1}{y}}$ 의 최솟값은 항상 $\frac{1}{2}$ 이므로 한 번의 전환이 있으면 반드시 손실이 $\frac{1}{2}$ 이상입니다. (등호 조건 $x=y=1$)
- 따라서 정확히 한 번의 부호 전환이 있고, 그 전환의 넓이 손실이 $\frac{1}{2}$ 인 경우가 없다는 것을 보이면 됩니다.
- 만약 그런 경우가 존재한다면, $0 \to$ 음수 $\to$ 양수 $\to 1$ 의 전환이 이루어져야 하고, 음수와 양수의 경계가 각각 -1과 1이어야 하는데, 1은 이미 A_{2N+1} 에서 쓰였으므로 이러한 경우는 $N=1$ 을 제외하고는 불가능합니다.
- 따라서 최적해는 **Case 1**에서만 나오며, A 의 개형은 1, (양수들), 0, (음수들), -1 꼴 또는 -1, (음수들), 0, (양수들), 1 꼴밖에 없습니다.

인테그랄 양의 "작은" 선물

- ✓ "수열과 인테그랄"을 가장 작은 바이트 수로 해결 - **jiwon0630**
- ✓ 제출번호 101241306, 31B, 237분

G. The sun is a deadly laser

`#prefix_sum`

출제진 의도 – **EXPERT**

- 제출 58번, 정답 7명 (정답률 25.000%)
- 처음 푼 사람: **tony9402**, 31분
- 출제자: **hibye1217**

- 1차원에서 이 문제를 풀어봅시다. 마력을 x 만큼 사용해야 하는 방 $[s, e]$ 가 쿼리 l, r 에 "포함된다"는 것은 어떻게 표현할 수 있을까요?
- 만약 방의 크기가 1 이었다면 그냥 평범한 누적 합 문제가 됩니다. 그럼 이 문제도 누적 합으로 풀 수 있지 않을까요?

- 결론부터 말하자면 가능합니다. 다만 조금의 응용이 들어갑니다. 수열 L_s 를 각 방 $[s, e]$ 에 대해 s 번째 위치에 x 를 저장한 수열이라고 합시다. 비슷하게, R_e 는 e 번째 위치에 x 를 저장한 수열이라고 합시다.
- L 과 R 의 누적 합인 L' 과 R' 을 만들면, 문제의 답은 $L'_r - R'_{l-1}$ 이 됩니다. 직관적으로 설명하자면 $[1, r]$ 과 한 칸이라도 겹치는 방들 중 $[1, l)$ 에 완전히 포함되는 방을 제외한다는 것이죠.
- 이를 2차원에 그대로 확장해서 적용할 수 있습니다. 방이 $[i_1, i_2] \times [j_1, j_2]$ 에 위치해있다면 누적 합을 4개 만들어서 각각 P_{i_1,j_1}, Q_{i_1,j_2}, R_{i_2,j_1}, S_{i_2,j_2} 에 값을 저장한 뒤 쿼리 $[r_1, r_2] \times [c_1, c_2]$ 에 대해 $P_{r_2,c_2} - Q_{r_2,c_1-1} - R_{r_1-1,c_2} + S_{r_1-1,c_1-1}$ 을 출력하면 됩니다.

The sun is a (Deadly) flower

- AC를 받은 사람 중, 스코어보드에서 G번 AC와 인접한 빨간 연결 요소의 크기의 합이 가장 큰 사람 - **익명의 한별이**
- 제출번호 -, -개, -분 -초

H. 1, 2, 3, 많다!

#math, #arithmetic, #implementation

출제진 의도 – **BASIC**

- 제출 152번, 정답 101명 (정답률 80.105%)
- 처음 푼 사람: **dong5995**, 1분
- 출제자: **yyyy7089**

- 입력으로 들어오는 ! 을 10 으로 변경하고 더한 뒤 출력해야 하는 값이 10 이상이라면 ! 으로 변경하여 출력하면 쉽게 구현할 수 있습니다.

많다!

- ✓ 페널티 총합이 가장 큰 사람 - **playsworld16**
- ✓ 2041분

I. 이진 수열 숫자 야구 게임

#ad_hoc,#randomization,#differential_cryptanalysis,#divide_and_conquer

출제진 의도 – **MASTER**

- 제출 78번, 정답 10명 (정답률 26.316%)
- 처음 푼 사람: **golazcc83**, 44분
- 출제자: **yyyy7089**

- 맨 처음 임의의 수열을 정하고, 정한 수열을 조금씩 변경하여 쿼리를 합니다.
- 쿼리의 결과로 반환되는 스트라이크의 증감을 바탕으로 S의 항을 추론할 수 있습니다.

- 결정론적인 방법으로는 한 번에 하나 이상의 항을 알아내는 것이 불가능합니다.
- 인터랙터가 적응적이지 않으면 무엇을 할 수 있을까요?

- 항을 한 번에 두 개씩 묶어서 비트를 뒤집어 쿼리해 봅시다.
- 이때 스트라이크의 증감이 $+2$ 또는 -2 라면 해당하는 S 의 두 항을 모두 알 수 있습니다.
- 아니라면 둘 중 하나만 다시 뒤집어서 쿼리했을 때 나머지 하나의 증감이 확정되어 해당하는 S 의 두 항을 모두 알 수 있습니다.
- 따라서 이 방법은 한 번에 하나의 항을 알아내는 쿼리 방법보다 무조건 이득입니다.

- 이제 S가 완전히 무작위인 길이 $4\,000$의 이진 수열이라고 가정하고 이 방법의 쿼리 횟수의 기댓값을 계산해 봅시다.
- 1번의 쿼리에 2개의 항을 알아낼 확률은 $\frac{2}{4} = \frac{1}{2}$입니다.
- 1번의 쿼리에 알아내지 못한다면 항상 2번째 쿼리를 통해 2번만에 알아낼 수 있습니다.
- 따라서 2번의 쿼리에 2개의 항을 알아낼 확률이 $1 - \frac{1}{2} = \frac{1}{2}$입니다.
- 따라서 이 방법의 쿼리 횟수의 기댓값은 $1.5 \times 2\,000 = 3\,000$입니다.
- 또한 이 분포는 표준정규분포를 따르므로, $3\,000$회 초과 쿼리해야 할 확률이 약 $\frac{1}{2}$입니다.
- 테스트케이스의 개수는 충분히 많을 것이므로 이 방법은 AC를 받기에 불충분합니다.

- 항을 한 번에 두 개씩 묶었던 과정을 재귀적으로 반복해 봅시다.
- 항을 2^k 개씩 쿼리했을 때 스트라이크의 증감이 $+2^k$ 또는 -2^k 라면 해당하는 S 의 2^k 개의 항을 모두 알 수 있습니다.
- 만약 증감이 $+2^k$ 또는 -2^k 가 아니라면 2^{k-1} 개씩 묶어서 재쿼리했을 때 나머지 하나의 쿼리 결과도 알 수 있으므로 재귀적으로 반복하는 것이 무조건 이득입니다.

- 재귀적으로 반복했을 때의 효율은 매우 빠르게 감소합니다.
- 항을 2^k 개씩 쿼리했을 때 스트라이크의 증감이 $+2^k$ 또는 -2^k 일 확률은 $\frac{2}{2^{2^k}}$ 지만 해당 조건을 만족해도 쿼리 횟수가 겨우 1 회 감소하기 때문입니다.
- 또한 각 시도의 쿼리의 길이가 2^k 이므로 k 가 1 증가할 때마다 시도의 횟수도 절반으로 줄어듭니다.
- 따라서 재귀적으로 상술한 과정을 k 번 반복하였을 때의 쿼리 횟수의 기댓값은 $(1-\sum_{x=1}^{k}\frac{2}{2^x 2^{2^x}})N$ 이 됩니다.

- $k = 1$번 반복하였을 때의 쿼리 횟수의 기댓값은 $4\,000 \times (1 - \frac{1}{4}) = 3\,000$회입니다.
- $k = 2$번 반복하였을 때의 쿼리 횟수의 기댓값은 $4\,000 \times (1 - \frac{1}{4} - \frac{1}{32}) = 2\,875$회입니다.
- $k = 3$번 반복하였을 때의 쿼리 횟수의 기댓값은 $4\,000 \times (1 - \frac{1}{4} - \frac{1}{32} - \frac{1}{1\,024}) \approx 2\,871$회입니다.
- $k = 4$번 반복하였을 때의 쿼리 횟수의 기댓값은 $4\,000 \times (1 - \frac{1}{4} - \frac{1}{32} - \frac{1}{1\,024} - \frac{1}{2^{19}}) \approx 2\,871$회입니다. 사실상 이 시점부터는 유의미한 효과를 기대하기 힘듭니다.

- 어떤 k 값이 실제로 이 문제를 해결하기에 충분한지 검증하는 것은 $\mathcal{O}\left(2^k N^2\right)$ dp를 통해 할 수 있습니다.
- $N = 4\,000$을 가정하고 프로그램을 만들어 검증해 보면 다음과 같습니다.
  - $k = 1$의 경우 쿼리 횟수가 $3\,000$회 이하일 확률이 약 50.89195% 입니다.
  - $k = 2$의 경우 쿼리 횟수가 $3\,000$회 이하일 확률이 약 99.99932% 입니다.
  - $k = 3$의 경우 쿼리 횟수가 $3\,000$회 이하일 확률이 약 99.99950% 입니다.
- 테스트케이스의 개수를 100개로 가정하면 $k = 2$의 경우 약 99.932%, $k = 3$의 경우 약 99.950%의 확률로 AC를 받을 수 있습니다.

- 하지만 모든 입력 데이터가 무작위적이라는 보장이 없습니다.
- 추정하는 이진 수열과 S를 xor 연산한 결과만 무작위적이면 되므로, 추정하는 이진 수열을 처음에 무작위화하면 됩니다.

IX

- 출제자 맞추기 폼에서 고득점 - **angello10**
- 9 스트라이크 2 볼, 257분

J. The wheel to the right

#ad_hoc, #constructive

출제진 의도 – **WORLD'S END**

- 제출 33번, 정답 0명 (정답률 0.000%)
- 처음 푼 사람: **익명의 한별이**, 0초
- 출제자: **yyyy7089**

- 이 문제를 푸는 방법은 여러 가지가 있을 수 있지만, 출제/검수진이 찾은 구성 방법은 한 가지입니다.
- 다음 시트부터는 해당 방법을 서술합니다.

- 어떤 경로가 최단 경로라고 할 때, 그 경로를 왕복할 수 있다면 꽤나 높은 효율을 보일 것이라고 추정할 수 있습니다.
- 왕복할 수 있지만 중간에 방향을 180도 돌릴 수 없는 경로가 있을까요?

- 다음 모양을 P-Turn으로 정의합니다.

```
#####
#..##
#...<
##.##
##^##
```

- P-Turn은 한쪽 방향에서 들어왔으면 반대쪽 방향으로만 나갈 수 있는 특징을 지녔습니다.
- P-Turn은 돌리거나 뒤집어도 이 특징이 유지됩니다.

- P-Turn 두 개를 잘 붙이면 다음과 같은 모양을 만들 수 있습니다. 해당 모양을 Intersecting P-Turn으로 정의합니다.

```
##A####
##v####
A>..###
##...##
###..<B
####^##
####B##
```

- Intersecting P-Turn은 A에서 들어왔으면 반대쪽 A로, B에서 들어왔으면 반대쪽 B로만 나갈 수 있는 특징을 지녔습니다.
- 이제 이 Intersecting P-Turn을 서로 겹치지 않게 최대한 많이 넣어 주면 됩니다.

- 실제로 구성해 보면 다음과 같은 형태가 됩니다.

```
..#.#...#.#...#.#
.....#.....#.....
#.#...#.#...#.#..
...#.....#.....#.
#...#.#...#.#...#
.#.....#.....#...
..#.#...#.#...#.#
.....#.....#.....
#.#...#.#...#.#..
...#.....#.....#.
#...#.#...#.#...#
.#.....#.....#...
```

- 위 모양은 규칙적입니다.
- 따라서 이 모양의 효율을 구하려면 이 모양의 반복되는 구간에 대해 왕복하는 데 걸리는 총 시간을 구하여 반복되는 구간의 크기로 나누면 됩니다.
- 효율을 구하면 $16/12 = 4/3 \approx 1.333 > 1.3$ 이므로, 문제의 조건에 해당하는 효율인 1.3 이상을 어렵지 않게 만들 수 있습니다.
- 격자의 가장자리 근처에서 서로 다른 경로 구간들끼리 이어주어야 합니다. 남는 공간이 충분히 많으므로 P-Turn을 이용해도 좋고 다른 방법을 사용해도 좋습니다.
- 이 모양은 유턴을 해서 돌아올 때만 충분한 효율이 나오므로 출발지의 뒤쪽에 도착지를 놓고 경로의 끝 부분에 다음 U-Turn 모양과 같은 유턴을 넣어야 합니다.

```
#####
#...#
#...#
##^##
```

- 서로 다른 경로 구간들끼리 잇는 다른 방법으로는, U-Turn 모양과 다음과 같은 K-Ladder 모양을 잘 활용하는 방법이 있습니다. 이 경우 각 경로 구간에서 이미 유턴을 한 번 하였으므로 유턴해 전체를 돌아올 필요가 없습니다. (다음은 K-Ladder 모양의 한 형태인 3-Ladder 입니다. K-Ladder 모양은 들어온 출입구의 왼쪽에 있는 출입구로만 나갈 수 있다는 특징이 있습니다.)

  ```
  ##v##v##v##v##v##v
  ##....##....##....
  ...##....##....##.
  ```

- 핵심적인 구성법인 Intersecting P-Turn이 동일하므로 같은 방법으로 취급합니다.

理論値

- 대회 시간 중 해당 문제에서 가장 높은 점수 (= 출발지에서 도착지까지 걸리는 시간) 획득 - **js7777**
- 제출번호 101242616, $1\,189\,445$ 점, 296 분

K. 토카는 햄버거를 뿌려라

#dp, #dp_bitfleid

출제진 의도 – **EXPERT**

- 제출 74번, 정답 21명 (정답률 38.182%)
- 처음 푼 사람: **jeoffrey0522**, 6분
- 출제자: **toycartoon, ohwphil**

- 햄버거를 지금까지 받은 사람들의 집합이 정해져 있을 때, 햄버거를 받은 순서와 관계없이, 특정 사람이 다음 햄버거를 받을 확률이 정해집니다.
- 따라서 햄버거를 지금까지 받은 사람들의 집합을 기준으로 DP(다이나믹 프로그래밍)를 진행할 수 있습니다.
- 햄버거를 지금까지 받은 사람들을 N 자리 이진수로 나타낼 수 있습니다. $1 \leq i \leq N$ 에 대하여, i 번 사람이 햄버거를 받았다면 비트마스크의 $i-1$ 번째 비트(2^{i-1} 의 자리)가 켜져 있는 이진수로 나타내는 방법을 생각할 수 있습니다.

- X 에서 Y 개의 비트가 켜져 있는 비트마스크$(0 \leq Y \leq N)$를 생각해 봅시다.
- $dp[X]$를 처음 Y 명을 뽑았을 때 뽑힌 사람의 비트마스크가 X 일 확률이라고 해 봅시다.
- 그럼 X 에 포함되지 않은 $i(1 \leq i \leq N)$에 대하여, X 의 집합에 포함되어 있지 않은 i 번째 사람이 다음에 뽑히는 경우를 생각할 수 있습니다.
- 이 경우 새로운 비트마스크는 $X|2^{i-1}$ 가 됩니다.

- Y 명을 추첨하였을 때, 바로 다음에 뽑히는 사람이 i 번째 사람이 될 확률이 p 라고 해 봅시다.
- 비트마스크 $X|2^{i-1}$ 에 대하여, i 번째 사람이 $Y+1$ 번째로 뽑히는 확률은 $dp[X] \times p$ 가 됩니다.
- 따라서 Y (비트의 개수)가 낮은 비트마스크 X 부터 시작하여, $dp[X|2^{i-1}] += dp[X] \times p$ 를 통해 갱신해 주면 모든 dp 값을 계산할 수 있습니다.
- 최종 답은 1 번인 한별이가 포함되어 있으면서, 즉 0 번 비트가 켜져 있으면서, M 개의 비트가 켜져 있는 비트마스크를 더해 주면 답을 얻을 수 있습니다.
- cf) 단순히 X 가 산술적으로 커지는 방향 $(0, 1, \cdots, 2^N - 1)$ 로 갱신해 주어도 됩니다. 왜 그럴까요?

- p를 계산하였다고 치고, 나머지 p를 계산해 주는 방법에 따라서 시간복잡도가 달라집니다.
- dp 값의 갱신은 $\mathcal{O}(N2^N)$ 번의 곱셈 또는 나머지 연산을 통해 할 수 있습니다.
- p 값의 계산은 일반적으로 $\mathcal{O}(N2^N + 2^N \log P)(P = 10^9 + 7)$ 번의 곱셈 또는 나머지 연산이 소요됩니다. 그러나 구현 방식에 따라서 시간복잡도의 상수가 달라집니다.

- 우선 모든 비트마스크에 대하여 확률 p의 공통분모를 계산할 수 있는데, 단순히 각 비트마스크에 대하여 비트를 순회하는 방식으로 구현하면 $\mathcal{O}(N2^N)$ 번의 곱셈 또는 나머지 연산이 소요됩니다. 하지만, 각 비트마스크의 최하위 비트(또는 최상위 비트)를 제거한 다음에, 이 작은 비트마스크의 분모를 참고하는 방식으로 계산하면 $\mathcal{O}(2^N)$ 번의 연산으로 분모를 계산할 수 있고, 구현 방식과 언어에 따라서 2 4배 정도의 실행 시간 이득을 볼 수 있습니다.

드..드리겠습니다..

- ”토카는 햄버거를 뿌려라” 정답자 중 10명을 문제와 같이 선정 - `angello10`, `jeoffrey0522`, `jjang36524`, `kwoncycle`, `lindelof`, `lntanz`, `oh040411`, `red_fox`, `sjh1224`, `ychangseok`

L. 검은칠된 격자에 보내는 최단경로

#binary_search,#parametric_search,#dijkstra,#constructive

출제진 의도 – **MASTER**

- 제출 65번, 정답 7명 (정답률 46.667%)
- 처음 푼 사람: **tony9402**, 82분
- 출제자: **toycartoon**

- 격자를 모두 변경해야하니 최솟값과 최댓값부터 결정해봅시다. 최솟값은 채울 수 있는 모든 격자를 0으로 채웠을 때고, 최댓값은 채울 수 있는 모든 격자를 9로 채웠을 때입니다. 편의를 위해 각각의 경우를 L과 H이라고 정의합시다.
- 당연하게도, D가 L보다 작거나 H보다 크다면 -1을 출력합니다.
- 격자 L에서 한 칸씩 값을 1 늘려본다면, 최단 경로가 0 또는 1 증가하게 됩니다.
- 이를 통해 최단 경로가 단조 증가한다는 사실을 관찰할 수 있으며, 매개변수 탐색을 통해 사잇값 D를 찾을 수 있습니다. 그렇다면 어떻게 매개변수 탐색을 해야할까요?

- L과 R의 중간 상태인 격자를 하나 잡아줍시다. 편의를 위해 이 격자를 X라고 합시다.
- 중간 상태인 격자를 구하는 방법은 다양하게 적용 가능합니다. 정해는 왼쪽 위에서부터 . 에 값을 1씩 채우는 방식을 채택했습니다.
- 이 격자에서 최단 경로를 구해 매개변수 탐색을 시행하면 됩니다. 최단 경로를 구하는 방법도 다양하게 적용 가능합니다. 이 경우는 다익스트라 알고리즘이나 다이얼을 사용했습니다.

페널티 많아도 AC가 받고 싶어!

- L번 문제에서 가장 많은 페널티(CE 제외) 후 맞은 사람 - **jeoffrey0522**
- 제출번호 101239892, 2회, 162분

M. 소인수분해 게임

#number_theory, #sieve,
#prime_factorization, #game_theory

출제진 의도 – **EXPERT**

- 제출 79번, 정답 19명 (정답률 26.389%)
- 처음 푼 사람: **sadtreap**, 13분
- 출제자: **yyyy7089**

- N을 소인수분해한 결과를 생각합시다.
- 그러면 소인수분해 게임은 소인수분해한 결과에서 소인수를 하나씩 골라 삭제하는 게임으로 치환됩니다.
- 각 소인수의 개수는 의미가 있지만 각 소인수의 값 자체는 의미가 없습니다.
- 따라서 N은 각 소인수의 개수를 들고 있는 배열 하나로 나타낼 수 있습니다.
  - 예를 들어, $N = 120$이라면 $120 = 2^3 \times 3^1 \times 5^1$이므로 120은 $[3, 1, 1]$과 같이 나타낼 수 있습니다.
- 이 배열을 A라 합시다. 이제 소인수분해 게임은 A에서 임의의 양수를 하나 골라 1만큼 줄이되 직전에 상대가 선택한 인덱스에 해당하는 수를 고를 수 없는 게임으로 치환됩니다.

- $2 \times \max(A) > \text{sum}(A)$ 면, 선공은 후공이 무엇을 고르든 그 때 A에서 선택할 수 있는 가장 큰 양수를 골라 1을 줄이면 이깁니다.
- 아니고 $\text{sum}(A)\%2 = 0$ 이라면, 후공은 선공이 무엇을 고르든 그 때 A에서 선택할 수 있는 가장 큰 양수를 골라 1을 줄이면 이깁니다.
- 아니라면, 선공은 후공이 무엇을 고르든 그 때 A에서 선택할 수 있는 가장 큰 양수를 골라 1을 줄이면 이깁니다.
- 따라서 $2 \times \max(A) > \text{sum}(A)$ 거나 $\text{sum}(A)\%2 = 1$ 이면 선공이 이기고, 아니면 후공이 이깁니다.

- 소인수분해를 $\mathcal{O}\left(\sqrt{N}\right)$ 에 하면 충분히 빠르게 동작하지 않습니다.
- N 까지 소수의 밀도는 $\mathcal{O}\left(\frac{1}{\log N}\right)$ 이므로, 미리 에라토스테네스의 체를 사용하여 소수를 모두 전처리하면 소인수분해를 $\mathcal{O}\left(\frac{\sqrt{N}}{\log N}\right)$ 에 할 수 있습니다.
- 이때 총 소요 시간은 $\mathcal{O}\left(\sqrt{N}\log N + T\frac{\sqrt{N}}{\log N}\right)$ 입니다.
- 충분히 빠른 다른 소인수분해 방법을 사용해도 좋습니다.

- 이와 다른 방법으로는 DP를 이용해서도 문제를 해결할 수 있습니다.
- 소인수분해한 결과 얻은 배열 A가 같으면 같은 게임입니다.
- 주어진 범위에서 얻을 수 있는 서로 다른 소인수 개수의 배열은 약 1900개입니다.
- 모든 가능한 소인수 개수의 배열의 모임 전체를 순회하는 DP를 구현하면 빠른 시간에 풀 수 있습니다.
- 마찬가지로 소인수분해를 $\mathcal{O}\left(\frac{\sqrt{N}}{\log N}\right)$ 또는 이에 준하는 충분히 빠른 소인수분해를 사용해야 합니다.

Highly Composite Number

- M번 문제에서 AC를 받은 소스 코드 중, 제출 번호를 소인수분해했을 때 소인수가 가장 많은 소스 코드를 제출한 사람 - **pichulia**
- 제출번호 101236512, 8개, 20분