SUAPC 2022 Summer 풀이

Official Solutions

신촌지역 대학생 프로그래밍 동아리 연합

2022년 9월 4일

목차

240

문제		의도한 난이도	출제자
A	수 맞히기 게임	**Medium**	신용명 tlsdydaud1
B	내비게이션	**Beginner**	이국렬 lky7674
C	패스	**Medium**	신용명 tlsdydaud1
D	포탈	**Challenging**	신용명 tlsdydaud1
E	1-3 트리	**Medium**	신용명 tlsdydaud1
F	차의 개수	**Easy**	강희원 heeda0528
G	AND, OR, XOR	**Challenging**	장래오 leo020630
H	역삼역	**Challenging**	신정환 shjohw12
I	딸기와 토마토	**Easy**	강희원 heeda0528
J	김밥	**Medium**	신정환 shjohw12
K	줄 세우기	**Hard**	박찬솔 chansol
L	피라미드	**Medium**	박진식 pjshwa
M	My뷰 꾸미기	**Medium**	박진식 pjshwa

240

A. 수 맞히기 게임

`probability`, `dp`

출제진 의도 – **Medium**

- ✓ 제출 40번, 정답 4명 (정답률 10%)
- ✓ 처음 푼 팀: **백준검색창에 lky7674 를검색해보세요** 연세대학교, 194분
- ✓ 출제자: 신용명 `tlsdydaud1`

- ✓ k가 될 수 있는 범위가 a 이상 b 이하인 상태에서 성제의 선택을 분석해 봅시다.
- ✓ $a = b$인 경우 성제의 선택은 한 가지밖에 없고, 질문 한 번으로 게임이 끝납니다.
- ✓ $a < b$이고 $b - a$가 짝수인 경우, 성제가 선택할 수 있는 정수는 $\frac{a+b}{2} - 1, \frac{a+b}{2}, \frac{a+b}{2} + 1$ 이고, 각각의 정수가 선택될 확률은 $\frac{1}{3}$로 동일합니다.
- ✓ $b - a$가 홀수인 경우, 성제가 선택할 수 있는 정수는 $\frac{a+b}{2} - \frac{1}{2}, \frac{a+b}{2} + \frac{1}{2}$ 이고, 각각의 정수가 선택될 확률은 $\frac{1}{2}$로 동일합니다.

- ✓ 이 상태에서 성제가 선택한 정수를 p라고 합시다.
- ✓ $p = k$인 경우 게임이 바로 끝납니다.
- ✓ $p > k$인 경우 k가 될 수 있는 정수의 범위는 a 이상 $p - 1$ 이하로 바뀝니다.
- ✓ $p < k$인 경우 k가 될 수 있는 정수의 범위는 $p + 1$ 이상 b 이하로 바뀝니다.

- 다음으로 질문 횟수의 기댓값을 구해 봅시다.
- 원욱이가 정수 k를 선택했고 현재 k가 될 수 있는 정수의 범위가 a 이상 b 이하일 때 성제의 질문 횟수의 기댓값을 $f(k, a, b)$라고 하면, 앞에서 분석한 결과를 바탕으로 식을 세울 수 있습니다.

$$f(k, a, b) = 1, \text{if } a = b \quad \cdots \quad (1)$$

$$v_1 = \begin{cases} 1, & \text{if } k = \dfrac{a+b}{2} - 1 \\ f(k, a, \dfrac{a+b}{2} - 2) + 1, & \text{if } k < \dfrac{a+b}{2} - 1 \\ f(k, \dfrac{a+b}{2}, b) + 1, & \text{if } k > \dfrac{a+b}{2} - 1 \end{cases} \quad \cdots \quad (2)$$

$$v_2 = \begin{cases} 1, & \text{if } k = \dfrac{a+b}{2} \\ f(k, a, \dfrac{a+b}{2} - 1) + 1, & \text{if } k < \dfrac{a+b}{2} \\ f(k, \dfrac{a+b}{2} + 1, b) + 1, & \text{if } k > \dfrac{a+b}{2} \end{cases} \quad \cdots \quad (3)$$

$$v_3 = \begin{cases} 1, & \text{if } k = \dfrac{a+b}{2} + 1 \\ f(k, a, \dfrac{a+b}{2}) + 1, & \text{if } k < \dfrac{a+b}{2} + 1 \\ f(k, \dfrac{a+b}{2} + 2, b) + 1, & \text{if } k > \dfrac{a+b}{2} + 1 \end{cases} \quad \cdots \quad (4)$$

$$v_4 = \begin{cases} 1, & \text{if } k = \dfrac{a+b}{2} - \dfrac{1}{2} \\ f(k, a, \dfrac{a+b}{2} - \dfrac{3}{2}) + 1, & \text{if } k < \dfrac{a+b}{2} - \dfrac{1}{2} \\ f(k, \dfrac{a+b}{2} + \dfrac{1}{2}, b) + 1, & \text{if } k > \dfrac{a+b}{2} - \dfrac{1}{2} \end{cases} \quad \cdots \quad (5)$$

$$v_5 = \begin{cases} 1, & \text{if } k = \dfrac{a+b}{2} + \dfrac{1}{2} \\ f(k, a, \dfrac{a+b}{2} - \dfrac{1}{2}) + 1, & \text{if } k < \dfrac{a+b}{2} + \dfrac{1}{2} \\ f(k, \dfrac{a+b}{2} + \dfrac{3}{2}, b) + 1, & \text{if } k > \dfrac{a+b}{2} + \dfrac{1}{2} \end{cases} \quad \cdots \quad (6)$$

✓ $(2) \sim (6)$을 일반화하면 다음과 같습니다. 성제가 부른 정수가 p일 경우,

$$v = \begin{cases} 1, & \text{if } k = p \\ f(k, a, p-1) + 1, & \text{if } k < p \\ f(k, p+1, b) + 1, & \text{if } k > p \end{cases} \quad \cdots \quad (7)$$

✓ $f(k, a, b)$는 가능한 모든 v 값의 평균과 같습니다. 즉,

$$f(k, a, b) = \frac{v_1 + v_2 + v_3}{3}, \text{ if } a < b \text{ and } b - a \text{ is even} \quad \cdots \quad (8)$$

$$f(k, a, b) = \frac{v_4 + v_5}{2}, \text{ if } b - a \text{ is odd} \quad \cdots \quad (9)$$

✓ $(1), (8), (9)$ 를 종합하면 다음과 같습니다.

$$f(k,a,b) = \begin{cases} 1, & \text{if } a = b \\ \dfrac{v_1 + v_2 + v_3}{3}, & \text{if } a < b \text{ and } b-a \text{ is even} \\ \dfrac{v_4 + v_5}{2}, & \text{if } b-a \text{ is odd} \end{cases} \quad \cdots \quad (10)$$

✓ (10) 에 나온 각각의 함수값은 DP 로 모두 구할 수 있는 형태가 되었지만, 상태 공간의 크기가 2401^3 으로 너무 큽니다.

- ✓ 임의의 정수 d에 대해 $f(k, a, b)$와 $f(k + d, a + d, b + d)$의 값은 항상 같습니다.
- ✓ 이 사실을 이용하면 k를 하나로 고정할 수 있고, 상태 공간의 크기가 2401^2로 줄어듭니다. 이제 각각의 함수값을 DP로 충분히 계산할 수 있습니다.

B. 내비게이션

`implementation`
출제진 의도 – **Beginner**

- ✓ 제출 156번, 정답 49팀 (정답률 31.41%)
- ✓ 처음 푼 팀: **DSP**서강대학교, 3분
- ✓ 출제자: 이국렬lky7674

B. 내비게이션

240

- ✓ 본 대회에서 가장 쉬운 문제입니다. 지문을 잘 읽고 지시사항을 그대로 구현하시면 됩니다.
- ✓ 지문 요약 : 가장 짧은 경로를 안내하는 내비게이션의 번호를 출력하는 문제입니다.
- ✓ 그냥 각 내비게이션이 안내하는 경로의 길이를 전부 다 구하고, 그 중 가장 짧은 길이를 찾으면 됩니다.
- ✓ 구현할 때 주의할 점은 다음과 같습니다.
  - 답의 범위가 크니 32bits 정수 변수를 사용하는 것이 아닌 64bits 이상의 정수 변수를 사용하셔야 합니다.
  - C++ 이용자는 long long int를, Java 이용자는 long을 사용하시면 됩니다.
  - 경로의 길이가 최대 $101 \times 10^9 - 2$이기에 최솟값을 저장하는 변수를 초기에 잘 설정하셔야 합니다.

240

- 조금 더 쉽게 짜는 방법
  - 절댓값 연산은 C++과 python의 abs 함수, java는 Math.abs 함수를 이용하시면 됩니다.
  - 최솟값이 저장된 배열의 index는 C++의 min_element를 통해서 쉽게 찾을 수 있습니다.

C. 패스

`math`, `constructive`
출제진 의도 – **Medium**

- ✓ 제출 132번, 정답 34팀 (정답률 25.75%)
- ✓ 처음 푼 팀: **정열맨**[홍익대학교], 22분
- ✓ 신용명[tlsdydaud1]

C. 패스

240

- N 이 적힌 카드를 뽑는 사람은 자신에게 공을 패스하게 됩니다.
- 이 사람이 공을 한 번 받는 경우는 맨 처음에 1 번째 사람이 N 이 적힌 카드를 뽑는 경우밖에 없습니다. 따라서 모든 사람이 공을 한 번씩 받았을 경우, 첫 번째로 뽑힌 카드에 적힌 수는 N 입니다.

240

- 이제 $N-1$ 장의 카드가 남았습니다. 이 카드들에 적힌 수의 합은 $\frac{N \cdot (N-1)}{2}$ 입니다.
- N 이 홀수인 경우 이 값은 N 의 배수입니다. 그렇다면 첫 번째로 공을 받은 사람과 마지막으로 공을 받은 사람이 같은 사람이 되므로 모든 사람이 공을 한 번씩 받는 상황은 발생할 수 없습니다.
- 한 가지 예외는 $N = 1$ 로, 이 경우는 따로 처리합니다.

C. 패스

240

- N 이 짝수인 경우 카드를 다음과 같은 순서로 뽑으면 모든 사람이 공을 한 번씩 받게 됩니다.

$$N, 1, N-2, 3, N-4, 5, \ldots, N-5, 6, N-3, 4, N-1, 2$$

- 그 밖에도 다른 풀이가 존재할 수 있습니다.

240

D. 포탈

`lca`, `case work`

출제진 의도 – **Challenging**

- 제출 5번, 정답 0팀 (정답률 0%)
- 출제자: 신용명 `tlsdydaud1`

- 미로에서 현준이와 진우가 하는 게임은 술래잡기와 같습니다. 현준이의 목표는 진우를 잡는 것이고, 진우의 목표는 잡히지 않는 것입니다.
- 게임에서 발생할 수 있는 상태의 종류는 현준이가 있는 방의 위치, 진우가 있는 방의 위치, 이번에 차례를 진행하는 사람에 따라 결정됩니다. 따라서 상태의 종류는 $2n^2$을 넘지 않습니다.
- 두 사람이 모두 최선의 전략으로 움직이기 때문에, 게임을 진행하는 도중 이전에 발생한 상태가 또 발생했을 경우 앞으로 그 상태가 무한히 발생할 수 있음을 의미합니다. 이것이 진우가 승리할 수 있는 유일한 경우입니다.

- 미로에 포탈을 추가하면 방들과 통로로 이루어진 사이클 하나가 생깁니다. 이 사이클에 속하는 방들 중 p_x 번 방과 가장 가까운 방의 번호를 v_x, p_y 번 방과 가장 가까운 방의 번호를 v_y 라고 합시다.
- 현준이가 v_y 번 방에 있고 그 방이 사이클에 속하는 방들 중 진우가 있는 방과 가장 가까운 방일 경우, 현준이가 단순히 진우가 있는 방 쪽으로 계속 움직이면 진우가 도망갈 수 있는 방이 점점 줄어들고, 결국 막다른 곳에 몰려서 현준이에게 잡히게 됩니다.
- 따라서 진우가 잡히지 않고 계속 도망가려면 일단 사이클에 속하는 방으로 들어가야 합니다. 그중 진우가 가장 빨리 갈 수 있는 방은 v_y 번 방입니다.

240

- 진우가 v_y 번 방에 도착하기 전에, 또는 도착한 직후에 현준이에게 잡히는 경우 현준이가 승리하며, (현준이가 v_y번 방까지 움직이는 데 필요한 최소 턴의 수) $\leq$ (진우가 v_y번 방까지 움직이는 데 필요한 최소 턴의 수) $+ 1$ 인 경우가 이에 해당합니다.
- a 번 방과 b번 방 사이의 최단 거리를 $dist(a, b)$ 라고 하면, 진우가 v_y 번 방까지 움직이는 데 필요한 최소 턴의 수는 $dist(y, v_y)$ 입니다.
- 현준이가 v_y 번 방까지 움직이는 데 필요한 최소 턴의 수를 구하는 과정은 조금 복잡합니다. 다음 슬라이드에서 살펴봅시다.

1 $v_y = p_x$ 또는 $v_y = p_y$ 인 경우, 현준이가 x 번 방에서 v_y 번 방으로 직접 갈 수도 있고, v_y 번 방과 포탈로 연결된 방으로 가서 포탈을 따라 v_y 번 방으로 올 수도 있습니다. 둘 중 더 빠른 경로를 선택하면 됩니다.

$$D = \min(dist(x, p_x), dist(x, p_y)), \text{if } v_y = p_x \text{ or } v_y = p_y$$

1에 해당하지 않는다면 다음으로 넘어갑니다.

2 $v_x = p_x$ 또는 $v_x = p_y$ 인 경우, 현준이가 v_x 번 방에 도착하면 포탈을 따라 이동해야 합니다. 여기서 v_y 번 방으로 이동할 수도 있고, 옆 방으로 이동했다가 다시 돌아오면 포탈을 따라 다시 이동하게 되어 v_x 번 방으로 돌아오는데 여기서 v_y 번 방으로 이동할 수도 있습니다. 둘 중 더 빠른 경로를 선택하면 됩니다.

$$D = \begin{cases} \min(dist(x, p_x) + dist(p_y, v_y), dist(x, v_y) + 2), & \text{if } v_x = p_x \\ \min(dist(x, p_y) + dist(p_x, v_y), dist(x, v_y) + 2), & \text{if } v_x = p_y \end{cases}$$

2에 해당하지 않는다면 다음으로 넘어갑니다.

3 이제 남은 경우는 v_x, v_y, p_x, p_y 가 모두 다른 경우입니다. 이때는 세 가지 선택이 가능합니다. 우선 x 번 방에서 v_x 번 방으로 나가는 것까지는 동일하며, 여기서 포탈을 타지 않고 바로 v_y 번 방으로 이동하거나 포탈과 연결된 방으로 이동해서 포탈을 탄 다음 거기서 v_y 번 방으로 이동할 수 있습니다. 셋 중 가장 빠른 경로를 선택하면 됩니다.

$$D = \min(dist(x, v_y), dist(x, p_x) + dist(p_y, v_y), dist(x, p_y) + dist(p_x, v_y))$$

이 경우에 포탈을 두 번 이상 타지 않아도 된다는 사실은 미로의 구조를 생각해 보면 알 수 있습니다.

- ✓ 앞에서 1을 설명할 때 현준이가 v_y 번 방으로 이동해서 포탈을 타고 이동하는 턴까지만 고려하고 다시 v_y 번 방으로 돌아오는 턴은 고려하지 않았습니다. 이것을 고려하지 않아도 되는 이유를 살펴봅시다.
- ✓ 진우는 v_y 번 방에 도착하면 포탈을 따라 이동해야 합니다. 현준이가 이미 포탈과 연결된 방에 있거나, 그 다음 차례에 현준이가 포탈로 연결된 두 방 중 하나로 이동하면 현준이가 이깁니다.
- ✓ 따라서 현준이는 포탈로 연결된 두 방이나, 그 방들과 인접한 방 사이를 왔다갔다하면 됩니다. 만약 포탈을 따라 이동해서 v_y 번 방에 도착했다면, 그 다음 차례부터는 진우가 있는 방 쪽으로 계속 이동하면 이깁니다.
- ✓ 게임의 승자가 현준이로 동일하기 때문에 1은 올바른 설명입니다.

240

- (현준이가 v_y번 방까지 움직이는 데 필요한 최소 턴의 수) > (진우가 v_y번 방까지 움직이는 데 필요한 최소 턴의 수) $+ 1$ 인 경우, 진우가 현준이에게 잡히지 않고 v_y 번 방으로 이동할 수 있습니다.
- 일단 사이클에 들어갔다면, 사이클에 포함된 방 사이를 따라 계속 도는 것이 진우에게는 최선의 전략입니다.
- 진우가 중간에 사이클을 벗어나더라도, 현준이에게 잡히지 않기 위해서는 사이클에 포함된 방으로 언젠가는 되돌아와야 하며, 사이클을 벗어났다가 다시 돌아오는 것은 현준이가 더 가까이 올 수 있게 만드는 것과 같아서 이득이 없습니다. 같은 이유로 현준이 역시 사이클에 들어갔다면 사이클에 포함된 방 사이를 따라 계속 도는 것이 최선의 전략입니다.

D. 포탈

240

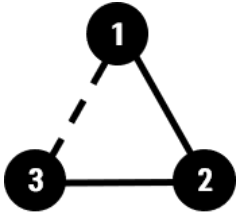
- 진우가 현준이에게 잡히지 않고 사이클에 포함된 방으로 이동했을 경우 게임에서 누가 이기는지 분석해 봅시다. 앞에서 두 사람 모두 한번 사이클에 들어가면 사이클을 벗어나지 않는 것이 최선의 전략임을 밝혔으므로, 두 사람 모두 사이클에 포함된 방을 따라서만 움직인다고 가정합니다.
- 현준이가 한쪽 방향으로만 진우를 쫓아간다면 진우가 계속 도망가면서 사이클을 돌기 때문에 진우를 잡을 수 없습니다.
- 현준이가 진우의 움직임에 따라서 방향을 바꾼다면 어떻게 될까요?

D. 포탈

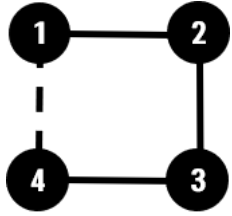
240

✓ 사이클의 크기가 p일 때 사이클에 포함된 방이 순서대로 $1, 2, \ldots, p$번 방이고 1번 방과 p번 방 사이에 포탈이 있다고 가정합니다. 이때 현준이가 1번 방, 진우가 $2 \sim (p-1)$번 방 중 한 곳에 있고 현준이의 차례가 된 경우를 생각해 봅시다.

1 $p = 3$인 경우, 진우가 있는 방으로 가능한 곳은 2번 방밖에 없습니다. 그런데 2번 방에 진우가 있는 경우는 현준이가 1번 방에서 2번 방으로 이동하면 항상 현준이가 승리합니다. 이것은 p가 더 큰 값일 때도 마찬가지입니다.



2 $p = 4$인 경우, 진우가 있는 방은 2번 방과 3번 방이 될 수 있습니다. 진우가 2번 방에 있는 경우 현준이가 승리함은 앞에서 보였고, 진우가 3번 방에 있는 경우 현준이가 2번 방으로 이동하면 진우는 4번 방으로 이동해야 잡히지 않는데, 이때 포탈을 따라 1번 방으로 이동하게 되므로 그 다음 차례에 현준이가 2번 방에서 다시 1번 방으로 이동하면 현준이가 승리합니다.



240

3 $p = 5$인 경우, 진우가 있는 방은 $2 \sim 4$번 방 중 하나인데, 2번 방에 있는 경우를 제외하면 3번 방과 4번 방이 남습니다.

✓ 여기서부터는 진행되는 차례와 가능한 경우의 수가 많아지기 때문에 약어를 사용합니다.
현준이가 a번 방으로 이동하는 것을 Ha, 진우가 b번 방으로 이동하는 것을 Jb라고 합시다.

✓ 진우가 3번 방에 있다면 게임은 다음과 같이 진행되고 현준이가 승리합니다.

$$H2 \to J4 \to H3 \to J5(J1) \to H2 \to J2$$

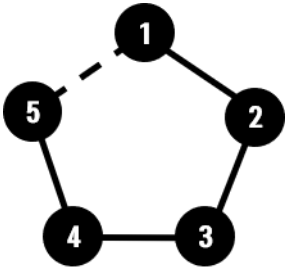
D. 포탈

240

- 진우가 4번 방에 있다면 게임의 진행은 아래의 두 가지가 가능하며 두 경우 모두 현준이가 승리합니다.

$$H2 \rightarrow J3 \rightarrow H3$$

$$H2 \rightarrow J5(J1) \rightarrow H1$$



4 $p = 6$인 경우, 진우가 있는 방은 $2 \sim 5$번 방 중 하나입니다. 진우가 2번 방에 있는 경우는 현준이가 승리함을 앞에서 보였습니다.

✓ 진우가 3번 방에 있다면 게임은 다음과 같이 진행되며 현준이가 승리합니다.

$$H2 \to J4 \to H3 \to J5 \to H4 \to J6(J1) \to H3 \to J2 \to H2$$

✓ 진우가 4번 방에 있다면 게임은 다음과 같이 진행되며 현준이가 승리합니다.

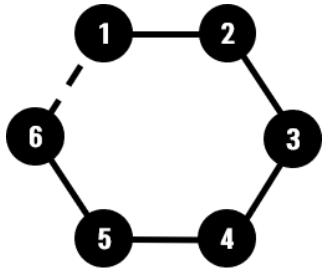
$$H2 \to J5 \to H3 \to J6(J1) \to H2 \to J2$$

D. 포탈

240

- 진우가 5 번 방에 있다면 게임은 다음과 같이 진행되며 현준이가 승리합니다.

$$H2 \to J4 \to H3 \to J5 \to H4 \to J6(J1) \to H3 \to J2 \to H2$$



- 지금까지 살펴본 경우들은 전부 현준이가 승리했습니다.

240

5 $p = 7$인 경우, 진우가 있는 방은 $2 \sim 6$번 방 중 하나입니다. 진우가 2번 방에 있는 경우는 현준이가 승리함을 앞에서 보였습니다.

✓ 진우가 3번 방이나 5번 방에 있다면 처음에는 다음과 같이 차례를 진행합니다.

$$H2 \rightarrow J4 \rightarrow H3 \rightarrow J5$$

여기서 다음과 같은 진행이 반복되며, 진우가 승리합니다.

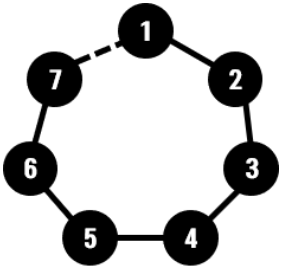
$$\rightarrow H4 \rightarrow J6 \rightarrow H5 \rightarrow J7(J1) \rightarrow H4 \rightarrow J2 \rightarrow H3 \rightarrow J1(J7)$$

D. 포탈

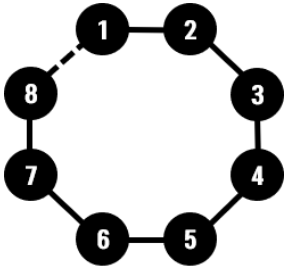
240

- 진우가 4번 방이나 6번 방에 있다면 게임은 다음과 같이 진행되며 현준이가 승리합니다.

$$H2 \to J5 \to H3 \to J6 \to H4 \to J7(J1) \to H3 \to J2 \to H2$$

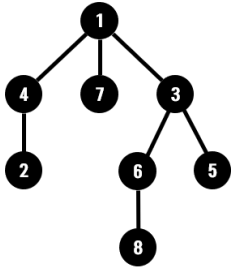


6 $p \geq 8$인 경우, 진우가 현준이가 있는 방 쪽으로 계속 이동하다가 현준이와의 거리가 2 이하가 됐을 때 다시 반대 방향으로 도망가는 과정을 반복하면 현준이가 진우를 잡을 수 없어 진우가 승리합니다.



- ✓ 위의 모든 상황을 정리하면 다음과 같습니다.
  - 현준이가 v_y 번 방에 진우보다 먼저 도착하거나 진우가 도착하고 바로 다음 차례에 도착할 수 있는 경우 현준이가 승리
  - 그렇지 않다면,
    - $p \leq 6$인 경우 현준이가 승리
    - $p = 7$이고 $dist(x, y)$가 홀수인 경우 현준이가 승리
    - $p = 7$이고 $dist(x, y)$가 짝수인 경우 진우가 승리
    - $p \geq 8$인 경우 진우가 승리

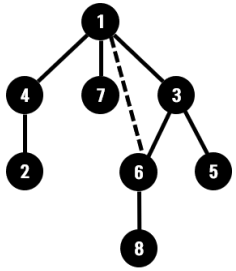
- 이제 v_x 와 v_y 를 구하는 방법을 알아봅시다. 두 정점은 구하는 방법이 동일하기 때문에 v_x 를 구하는 방법만 설명합니다. 먼저 이렇게 생긴 미로가 있다고 가정해 봅시다.



- 미로는 포탈을 제외하면 트리 구조이므로 방 하나를 트리의 루트로 간주할 수 있습니다. 여기서는 1 번 방을 루트로 간주합니다. 이제 포탈이 연결하는 두 방 p_x, p_y 는 트리에서 조상-자손 관계일 수도 있고 그렇지 않을 수도 있습니다.

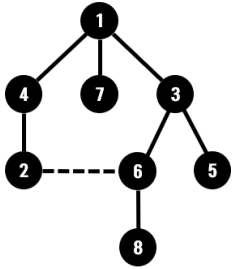
240

- ✓ p_x 번 방이 p_y 번 방의 조상인 경우, $LCA(p_x, p_y) = p_x$ 입니다.
- ✓ 여기서는 $k = LCA(p_y, x)$ 를 구합니다. v_x 는 k 번 방과 p_x 번 방 중 깊이가 더 깊은 방(1번 방으로부터 더 먼 방)입니다.
- ✓ p_y 번 방이 p_x 번 방의 조상인 경우도 같은 방식으로 v_x 를 구할 수 있습니다.



240

- 두 방이 조상-자손 관계가 아닌 경우, v_x 는 $LCA(p_x, x), LCA(p_y, x), LCA(p_x, p_y)$ 중 가장 깊은 방과 같습니다.



240

- ✓ Sparse Table을 전처리하는 데 $O(N \lg N)$, 쿼리 하나를 처리하는 데 $O(\lg N)$ 시간이 걸리므로 총 시간복잡도는 $O((N+Q) \lg N)$ 입니다.

E. 1-3 트리

`tree`

출제진 의도 – **Medium**

- ✓ 제출 110번, 정답 5팀 (정답률 4.545%)
- ✓ 처음 푼 팀: **삼대진미**연세대학교, 172분
- ✓ 출제자: 신용명`tlsdydaud1`

- ✓ 트리에 루트가 있다고 가정합시다. 루트는 N이 적힌 노드 중 하나를 임의로 선택합니다.
- ✓ 루트를 정했을 경우, 루트가 아닌 모든 노드는 리프 노드이거나 자식 노드가 2개입니다.
- ✓ 루트가 아닌 어떤 노드가 자식 노드 2개를 가지고 그 노드에 적힌 수가 x일 경우, 자식 노드 중 하나에는 $x-1$이 적혀 있어야 하며 다른 하나에는 $x-1$ 이하의 수가 적혀 있어야 합니다.
- ✓ $N=1$일 경우 루트 노드는 자식 노드가 1개입니다. 그렇지 않을 경우 루트 노드는 자식 노드가 3개입니다.
- ✓ 루트 노드가 3개의 자식 노드를 가질 경우, 그 중 두 개 이상에는 $N-1$ 이상의 수가 적혀 있어야 하며 N은 최대 한 개의 자식 노드에만 적혀 있어야 합니다.

240

- ✓ 이제 노드들에 자식을 연결해 봅시다. 1이 적힌 노드는 전부 리프 노드이며, 다른 노드의 자식 노드가 될 수 있습니다. 현재 다른 노드의 자식 노드가 될 수 있는 노드의 수를 s라고 하면, 처음에 $s = C_1$ 입니다.
- ✓ 2가 적힌 노드는 1이 적힌 노드 하나와 1 이하의 수가 적힌 노드 하나를 자식으로 가져야 합니다. 즉, 자식 노드를 전부 연결하기 위해서는 1이 적힌 노드가 C_2개 이상, 1 이하의 수가 적힌 노드가 $2C_2$개 이상 존재해야 합니다.
- ✓ 충분한 개수의 자식 노드가 존재한다면 자식 노드를 연결합니다. 자식 노드가 될 수 있는 s개의 노드 중 $2C_2$개의 부모 노드가 정해졌고, 새로 자식 노드가 될 수 있는 C_2개의 노드가 생겨났으므로 s는 C_2만큼 감소합니다.

E. 1-3 트리

240

- 다음으로 3이 적힌 노드는 2가 적힌 노드 하나와 2 이하의 수가 적힌 노드 하나를 자식으로 가져야 합니다. 자식 노드를 전부 연결할 수 있다면, s는 C_3만큼 감소합니다.
- 같은 방식으로 i가 적힌 노드의 자식 노드들을 연결할 경우 s는 C_i만큼 감소합니다. 중간에 $C_{i-1} > C_i$이거나 $s < 2C_i$인 경우가 한 번이라도 생기면 답은 NO입니다.
- N이 적힌 노드까지 모두 처리했을 경우, 부모 노드가 정해지지 않은 s개의 노드는 모두 2개의 자식 노드를 갖거나 자식 노드가 없습니다. 이제 s개의 노드를 서로 연결하는 수밖에 없는데, 그 노드들을 서로 연결해서 조건을 만족하는 트리가 나오려면 $s = 2$여야 합니다. 그렇지 않은 경우 답은 NO입니다.
- 또한 앞에서 언급한 루트 노드의 조건에 의해서 $C_N > 2$이거나 $C_{N-1} = 1$인 경우 답은 NO입니다.
- 그 외의 경우 답은 전부 YES입니다.

240

F. 차의 개수

`math`, `constructive`
출제진 의도 – **Easy**

- 제출 123번, 정답 47팀 (정답률 38.211%)
- 처음 푼 팀: **기러기토마토**서강대학교, 6분
- 출제자: 강희원`heeda0528`

240

최댓값부터 풀어봅시다.

- ✓ 편의상 서로 다른 차의 개수를 X 라고 하겠습니다.
- ✓ X 는 모든 쌍의 개수인 $N(N-1)/2$ 보다 작거나 같습니다.
- ✓ X 를 $N(N-1)/2$ 와 같게 만들 수 있을까요?

240

- ✓ $2^{29} < 10^9 < 2^{30}$ 이라는 점에서 2의 거듭제곱을 떠올려 봅시다.
- ✓ **claim**: $2^0, 2^1, 2^2, \cdots, 2^{N-1}$ 으로 얻은 X 는 $N(N-1)/2$ 입니다.
- ✓ **proof**: 이는 귀류법으로 증명할 수 있습니다.
  - WLOG $2^a - 2^b = 2^c - 2^d$ $(a > b > d,\ c > d)$ 라고 가정합시다.
  - 양변을 2^d 로 나누면 $2^{a-d} - 2^{b-d} = 2^{c-d} - 1$ 입니다.
  - 좌변은 짝수, 우변은 홀수이므로 모순입니다.
  - 따라서 차가 같은 서로 다른 두 쌍은 존재하지 않습니다.

F. 차의 개수

240

- $O(N)$에 최댓값과 2의 거듭제곱 N개를 출력할 수 있습니다.
- 사실, 범위 내에서 수 N개를 랜덤하게 뽑아도 매우 높은 확률로 정답을 받을 수 있습니다.
- 그 밖에도 다양한 풀이가 존재합니다.

240

최솟값을 풀어봅시다.

- ✓ 가장 작은 수를 고정했을 때 나머지 $N-1$ 개의 수와의 차는 모두 다릅니다.
- ✓ 따라서 X 는 $N-1$ 보다 크거나 같습니다.
- ✓ **claim**: $1, 2, 3, \cdots, N$ 으로 얻은 X 는 $N-1$ 입니다.
- ✓ **proof**: 가능한 차의 최소는 1, 최대는 $N-1$ 이므로 X 는 $N-1$ 을 넘지 않습니다.
- ✓ $O(N)$ 에 최솟값과 $1, \cdots, N$ 을 출력할 수 있습니다.
- ✓ 그 밖에도 아무 등차수열을 출력하면 정답을 받습니다.

G. AND, OR, XOR

`dp_sum_over_subsets`

출제진 의도 – **Challenging**

- ✓ 제출 46번, 정답 1팀 (정답률 2.174%)
- ✓ 처음 푼 팀: **백준검색창에 lky7674 를검색해보세요** 연세대학교, 263분
- ✓ 출제자: 장래오 `leo020630`

1. XOR

- ✓ 비교적 쉬운 XOR의 경우를 먼저 해결합시다.
- ✓ XOR의 성질에 따라 각 A_i에 대해 $A_i \oplus A_j = K$인 A_j의 값은 유일하게 결정되며 이 값은 $A_i \oplus K$입니다.
- ✓ 따라서 x의 등장 횟수를 $cnt[x]$라 했을 때, $cnt[x] \times cnt[x \oplus K]$를 모두 더한 값이 조건을 만족하는 (i, j) 쌍의 개수가 됩니다.
- ✓ 구해야 하는 값은 $i < j$인 쌍의 개수입니다.
- ✓ 따라서 $K = 0$인 경우에는 합에서 N을 빼준 후 2로 나눈 것, 그 외에는 합을 2로 나눈 것이 답이 됩니다.

2. AND

- ✓ 우선, 몇 가지 관찰을 통해 문제를 단순화할 수 있습니다.
- ✓ $A_i \,\&\, A_j = K$ 인 A_i, A_j 가 만족해야 할 조건은 $A_i \,\&\, K = K, A_j \,\&\, K = K$ 입니다. 이를 만족하지 않는 수는 고려하지 않아도 됩니다.
- ✓ 이제 모든 A_i 에 대해 $A_i \,\&\, K = K$ 를 만족하니 모든 A_i 에서 K 를 빼줄 수 있습니다.
- ✓ 이와 같은 과정을 거치면 이 문제는 $A_i \,\&\, A_j = 0, i < j$ 를 만족하는 쌍을 세는 문제가 됩니다.
- ✓ 즉, $K = 0$ 인 문제를 해결하면 전체 문제를 해결할 수 있습니다.

2. AND

- ✓ XOR의 경우와 마찬가지로 A_i를 고정한 후 비트 별로 생각해봅시다.
- ✓ 모든 비트에 대해 $A_i \,\&\, A_j = 0$이려면 다음을 만족해야 합니다.
  - A_i에서 켜져 있는 비트는 A_j에서 반드시 꺼져 있어야 합니다.
  - A_i에서 꺼져 있는 비트는 A_j에서 어떤 상태여도 괜찮습니다.
- ✓ 즉, A_j는 $\neg A_i$의 부분 마스크여야 합니다.
- ✓ A가 B의 부분 마스크라는 것은 $A \,\&\, B = A$임과 동치입니다.

2. AND

- 이제 모든 수 x에 대해 x의 부분 마스크인 수들의 등장 횟수의 합을 구해야 합니다. 이를 $dp[x]$라 하겠습니다.
- 이러한 형태의 문제는 SOS DP를 통해 $O(N2^N)$에 해결할 수 있음이 알려져 있습니다.
- 이 때 N은 비트의 개수로, 이 문제에서는 $2^{19} < 10^6 < 2^{20}$이므로 $N = 20$입니다.
- SOS DP를 통해 $dp[x]$를 모두 계산했다면, XOR의 경우와 비슷하게 $cnt[x] \times dp[\neg x]$의 합을 구하면 모든 (i, j) 쌍의 개수를 구할 수 있습니다.
- $A_i = K$인 i들에 대해 $i = j$인 경우가 합에 포함되므로, 구한 합에서 $cnt[K]$를 빼준 후 2로 나눈 것이 답이 됩니다.

3. OR

- AND와 동일한 방법으로 접근해봅시다.
- $A_i \mid A_j = K$인 A_i, A_j가 만족해야 할 조건은 $A_i \mid K = K, A_j \mid K = K$입니다. 즉, K에서 꺼져 있는 비트들은 없는 비트로 생각합시다.
- $A_i \mid A_j = K$은 $\neg(A_i \mid A_j) = \neg K$와 동치입니다.
- 이를 다시 정리하면 $\neg A_i \,\&\, \neg A_j = 0$이 됩니다.
- 즉, $A_i \mid K = K$를 만족하는 모든 A_i들을 $A_i \oplus K$로 바꾸어 주면 AND의 경우와 동일한 상황이 됩니다.
- 따라서 같은 방법으로 문제를 해결할 수 있습니다.

H. 역삼역

`string`, `manacher's`, `suffix array and lcp array`
출제진 의도 – **Challenging**

- ✓ 제출 32번, 정답 3팀 (정답률 9.375%)
- ✓ 처음 푼 팀: **djs212222**[서강대학교], 173분
- ✓ 출제자: 신정환[shjohw12]

240

- 죄송합니다. 사실 재미없는 문제입니다. :(
- 우선 서로 다른 substring의 개수는 Longest Common Prefix를 이용하여 빠르게 구할 수 있음이 잘 알려져 있습니다. 전체 substring의 개수에서 각 접미사의 LCP 값을 빼면 됩니다.
- 이 문제 역시 비슷한 방법으로 해결할 수 있습니다.

- ✓ 모든 substring은 어떤 접미사의 접두사입니다. 따라서 각 접미사에 대하여 조건을 만족하는 접두사의 개수를 구하는 것으로 생각할 수 있습니다.
- ✓ 예를 들어, 접미사가 "banana"이고 $K = 3$인 경우 "ana"를 포함하면 조건을 만족하므로 길이 4이상인 접두사는 모두 조건을 만족합니다.
- ✓ 즉, 각 접미사가 포함하는 길이 K 이상인 팰린드롬 중 끝 인덱스의 최솟값을 구할 수 있다면 접미사에서 조건을 만족하는 접두사의 개수를 구할 수 있습니다.

240

- 길이 $N(N > 2)$인 팰린드롬을 포함하면 길이 $N - 2$인 팰린드롬도 포함하므로 여기서는 길이가 K 또는 $K + 1$인 팰린드롬만 확인하면 됩니다.
- 팰린드롬을 구하는 것은 Manacher's algorithm을 이용합니다. Manacher's algorithm은 문자열에서 각 문자를 중심으로 하는 가장 긴 팰린드롬의 길이를 선형 시간에 구하는 알고리즘입니다.
- 어떤 문자를 중심으로 하는 가장 긴 팰린드롬의 길이가 K 이상이라면 길이가 K(또는 $K + 1$)인 팰린드롬 구간을 저장합니다.
- 구간을 모두 구했다면 sweeping을 통해 각 접미사가 포함하는 팰린드롬 중 끝 인덱스의 최솟값을 구할 수 있습니다.

- ✓ 마지막으로 LCP를 이용하여 중복을 처리합니다.
- ✓ Suffix Array 상에서 인접한 두 접미사에 대하여, 조건을 만족하는 접두사 중에서 길이가 LCP 이하인 것은 한 번만 세면 됩니다.
- ✓ 총 시간 복잡도는 $O(N \log N)$ 입니다. Suffix Array를 $O(N(\log N)^2)$ 에 구하여도 무리 없이 AC를 받을 수 있습니다.

I. 딸기와 토마토

case work

출제진 의도 – **Easy**

- ✓ 제출 244번, 정답 38팀 (정답률 15.984%)
- ✓ 처음 푼 팀: **백준검색창에 lky7674를 검색해보세요**연세대학교, 21분
- ✓ 출제자: 강희원heeda0528

240

- ✓ 먼저, 씨앗이 존재하는 칸의 수를 C라고 하겠습니다.
- ✓ $C = 2K$라면 딸기와 토마토가 같이 자라는 칸이 없습니다.
- ✓ $C < 2K$라면 다음과 같이 두 경우로 나눌 수 있습니다.
  - 딸기를 심은 줄과 토마토를 심은 줄이 수직으로 만난다.
  - 딸기를 심은 줄과 토마토를 심은 줄이 수평으로 만난다.

- 딸기를 심은 줄과 토마토를 심은 줄이 수직으로 만나는 경우, 딸기와 토마토가 같이 자라는 칸은 정확히 1 개입니다.
- 씨앗이 심어진 모든 칸을 보면서 두 줄이 수직으로 만나는 칸이 있는지 확인하면 됩니다.
- 딸기를 심은 줄과 토마토를 심은 줄이 수평하게 만나는 경우, 씨앗이 심어진 모든 칸은 한 행 또는 열에 있으며 딸기와 토마토가 같이 자라는 칸은 정확히 $2K - C$ 개입니다.
- 씨앗이 심어진 줄을 찾아 양 끝에서 각각 $C - K$ 개를 제외하고 출력하면 됩니다.

J. 김밥

`sorting`, `sweeping`, `binary search`, `prefix sum`
출제진 의도 – **Medium**

- ✓ 제출 82번, 정답 10팀 (정답률 12.195%)
- ✓ 처음 푼 팀: **백준검색창에 lky7674를 검색해보세요**연세대학교, 62분
- ✓ 출제자: 신정환`shjohw12`

240

- 문제의 조건을 만족하는 어떤 식재료의 집합을 생각해봅시다. 집합 내 나머지 식재료를 모두 포함하는 어떤 식재료가 존재할 것입니다. 그 식재료를 A라 하겠습니다.
- 만약 A가 집합에 속하지 않는 식재료 B에 포함된다면 A가 포함하는 식재료를 B도 모두 포함하므로 B를 집합에 추가해도 조건을 만족하고 맛이 증가합니다.
- 따라서 문제의 답이 되는 집합에서 집합 내 나머지 식재료를 모두 포함하는 식재료는 다른 어떤 식재료에도 포함되지 않습니다. 그러한 식재료를 편의상 "김"이라 하겠습니다.

240

- 김을 어떻게 찾을 수 있는지 알아보겠습니다.
- 주어진 식재료를 왼쪽 끝 값 순으로 정렬하여 순회합니다. 이때 오른쪽 끝 값을 관리하여 현재 식재료의 오른쪽 끝 값이 전에 나온 오른쪽 끝 값의 최댓값보다 크다면 현재 식재료는 다른 어떤 식재료에도 포함되지 않는 식재료이므로 김입니다.
- 한 가지 주의할 점은 정렬할 때 왼쪽 끝 값이 같다면 오른쪽 끝 값의 역순으로 정렬해야 한다는 것입니다.

- 이제 문제에서 요구한 최댓값을 구해봅시다.
- 김이 아닌 식재료의 경우 여러 김에 포함될 수 있습니다. 김이 아닌 식재료에 대하여, 자신을 포함하는 김의 맛에 자신의 맛을 더해서 마지막에 모든 김의 맛을 비교하면 최댓값을 구할 수 있습니다.
- 이때 맛을 더하는 것을 naive하게 구현 한다면 연산 횟수가 $O(N^2)$ 이 되어 시간 초과를 받습니다.

240

- ✓ 맛을 여러 김에 더하는 것을 빠르게 처리해야 합니다.
- ✓ 김이 아닌 식재료에 대하여, 자신을 포함하는 김은 현재까지 나온 김 중에 자신보다 오른쪽 값이 큰 것들입니다. 김을 순서대로 보면 오른쪽 끝 값이 증가합니다. 따라서 이분탐색을 이용하여 자신을 포함하는 김의 구간을 빠르게 찾을 수 있습니다.
- ✓ 구간에 더하는 것은 imos 법이라는 잘 알려진 테크닉으로 $O(1)$ 에 처리할 수 있습니다. 정해는 아니지만 segment tree 등을 사용하여도 AC를 받을 수 있습니다.
- ✓ 총 시간 복잡도는 $O(N \log N)$ 입니다.

K. 줄 세우기

`prefix_sum`, `smaller_to_larger`, `deque`, `linked_list`, `disjoint_set`, `offline_queries`

출제진 의도 – **Hard**

- 제출 41번, 정답 4팀 (정답률 9.756%)
- 처음 푼 팀: **DSP**서강대학교, 173분
- 출제자: 박찬솔chansol

- Union-Find를 사용하면 다음 작업을 쉽게 할 수 있습니다.
  - 어떤 사람이 어느 줄에 속하는지
  - 두 사람이 같은 줄에 속하는지
  - 서로 다른 두 줄을 합치는 연산

- ✓ 실제로 구성된 줄의 순서를 알기 위해서는 Union-Find 외의 데이터를 더 관리해야 합니다.
- ✓ deque을 사용하면 컨테이너의 앞뒤로 쉽게 원소를 넣고 뺄 수 있습니다.
  - 두 줄을 합칠 때 항상 사람의 수가 적은 줄에 있는 사람들을 많은 줄의 앞 또는 뒤로 모두 옮겨줍니다.
  - Smaller to Larger에 의해 1 번 쿼리를 모두 처리하는 데 시간 복잡도가 $O(N \log N)$ 으로 보장됩니다.
- ✓ linked list는 다른 컨테이너로 현재 컨테이너의 모든 값을 $O(1)$ 시간에 옮기는 것을 지원합니다.
- ✓ Smaller to Larger + deque 또는 linked list를 사용하면 1 번 쿼리를 처리할 수 있습니다.

- ✓ 한번 같은 줄이 되면, 줄 내에서의 순서는 더 이상 변하지 않습니다.
- ✓ 모든 1 번 쿼리를 적용한 후의 결과를 알고 있다면, 누적합으로 전처리 해둘 수 있습니다.
- ✓ 2 번 쿼리가 들어왔을 때, 현재 두 사람이 같은 줄에 속한다면 미리 전처리해둔 누적합을 통해 답을 $O(1)$ 에 구해줄 수 있습니다.

L. 피라미드

`ad_hoc`, `greedy`
출제진 의도 – **Medium**

- ✓ 제출 48번, 정답 10팀 (정답률 20.833%)
- ✓ 처음 푼 팀: **정열맨**[홍익대학교], 67분
- ✓ 출제자: 박진식[pjshwa]

240

- ✓ 맞닿아 있는 블록의 색상이 모두 달라야 한다는 조건을 활용해봅시다. 이 조건에 따르면, $(1, 1)$과 $(2, 1)$의 블록 색상이 결정되는 순간 나머지 블록들의 색상도 모두 고정됩니다.
- ✓ 따라서 $(1, 1)$, $(2, 1)$, $(2, 2)$의 색상이 모두 다르지 않다면 목표를 이루는 것이 무조건 불가능하고, 모두 다르다면 가능한 피라미드의 종류는 현재 2행 상태를 그대로 유지했을 때 구성되는 피라미드 / 2행의 두 블록을 교환했을 때 구성되는 피라미드 2가지 밖에 없습니다.

240

- ✓ 목표로 하는 피라미드를 미리 구성해 두고, 3 행부터 시뮬레이션하며 교환이 가능한지, 가능하다면 최소 연산 횟수는 몇 번인지를 구해볼 수 있습니다.
  - 현재 행과 목표 행의 3 가지 블록 색상의 수가 각각 같지 않다면 불가능합니다.

240

- 현재 행과 목표 행의 블록 색상의 수가 같다면, 블록별로 아래와 같은 9 가지 상태 중 하나를 가진다고 생각할 수 있습니다.
  - 현재 색상: 0, 바뀌어야 할 색상: 0
  - 현재 색상: 0, 바뀌어야 할 색상: 1
  - ...
  - 현재 색상: 2, 바뀌어야 할 색상: 2
- 각각 $(0,0), \cdots, (2,2)$ 라고 표현하고, 각각이 등장하는 횟수를 세어 봅시다.

- ✓ 블록의 교환은 아래의 5가지 연산 중 하나로 표현할 수 있습니다.
  - $(0, 1), (1, 0)$ pair 를 없애는 데 1회
  - $(0, 2), (2, 0)$ pair 를 없애는 데 1회
  - $(1, 2), (2, 1)$ pair 를 없애는 데 1회
  - $(0, 1), (1, 2), (2, 0)$ triple 을 없애는 데 2회
  - $(0, 2), (2, 1), (1, 0)$ triple 을 없애는 데 2회
- ✓ 이 중 $4, 5$번 연산은 동시에 사용할 수 없습니다. 왜냐하면 $4, 5$번 연산을 1회씩 수행할 경우 교환을 4회 해야 하는데, $1, 2, 3$번 연산을 1회씩 수행하면 3회 교환으로 같은 효과를 낼 수 있으므로 후자가 더 좋은 방법이기 때문입니다.
- ✓ 따라서 $1, 2, 3$번 연산을 쓸 수 있을 때까지 최대한 쓰고, 잔여분에 대해 4번 또는 5번을 쓰는 greedy 전략이 최적입니다.

- 위 작업을 각 행에 대해 수행한 뒤 연산 횟수를 모두 더합니다. 이 값을 2가지 가능한 피라미드 구성에 대해서 각각 구한 뒤, 둘 중 더 작은 값을 출력하면 됩니다.

240

M. My뷰 꾸미기

`combinatorics`
출제진 의도 – **Medium**

- 제출 143번, 정답 15팀 (정답률 10.490%)
- 처음 푼 팀: **우승하러 왔습니다**서강대학교, 48분
- 출제자: 박진식pjshwa

- ✓ 서로 다른 관심 분야를 큐레이팅 하는 사건들은 독립적이므로, 1 부터 N 까지의 i 에 대해 i 번째 관심 분야를 큐레이팅하는 방법의 수를 구해서 모두 곱하면 됩니다.
- ✓ i 번째 관심 분야를 큐레이팅하는 방법의 수는 $\sum_{k=1}^{min(A_i,B_i)} \binom{A_i}{k}\binom{B_i}{k}$ 입니다.

- ✓ 하지만 N 과 A_i, B_i 가 모두 최대 30 만이므로 Naive한 계산으로는 시간 안에 들어올 수 없습니다.
- ✓ 다음을 생각해 봅시다. $\binom{A_i}{k} = \binom{A_i}{A_i - k}$ 이고, 따라서 $\sum_{k=1}^{min(A_i,B_i)} \binom{A_i}{k}\binom{B_i}{k} = \left(\sum_{k=0}^{min(A_i,B_i)} \binom{A_i}{A_i - k}\binom{B_i}{k}\right) - 1$ 입니다.

- $\left(\sum_{k=0}^{min(A_i,B_i)} \binom{A_i}{A_i-k}\binom{B_i}{k}\right)$ 의 식을 조합론적으로 생각해 봅시다. A_i 개의 글에서 $A_i - k$ 개의 글을, B_i 개의 글에서 k 개의 글을 고르는 경우의 수를 모든 가능한 k 에 대해서 더해주고 있으므로, 결국 $A_i + B_i$ 개의 글에서 A_i 개의 글을 고르는 경우의 수와 같음을 알 수 있습니다. 이는 Vandermonde's identity라는 이름으로도 불립니다.
- 따라서 i 번째 관심 분야를 큐레이팅하는 방법의 수는 $\binom{A_i+B_i}{A_i} - 1$와 같고, 이를 1 부터 N 까지의 모든 i 에 대해 구한 뒤 곱한 결과를 출력하면 됩니다.