2023 가지컵 에디토리얼

kiwiyou

wider93

yup0927

wapas

mingyu331

special thanks to startlink and utilforever

A	Easy	가지 교배	… 3
B	Easy	가지 산사태	… 5
C	Easy	하이퍼 가지 따기	… 7
D	Easy	:danceplant:	… 9
E	Medium	가지 사진 찾기	… 12
F	Medium	가지 이모지	… 14
G	Medium	슬슬 가지를 먹지 않으면 죽는다	… 17
H	Medium	SCP-504	… 19
I	Hard	가지 밭 게임	… 22
J	Hard	가지 자르기	… 24
K	Challenging	가지 볶음	… 26
L	Challenging	가지농장 수확하기	… 30

A. 가지 교배

`implementation`

출제자: **wider93**

First Solve: **riroan** (2분)

A. 가지 교배

- 조수가 하나라도 흰색 가지를 가지고 오면 답은 흰색입니다. 그렇지 않으면 답은 보라색입니다.
- 흰색 가지만 가지고 있는 조수가 있는지 체크하면 됩니다.

B. 가지 산사태

`ad_hoc`

출제자: **yup0927**

First Solve: **riroan** (5분)

B. 가지 산사태

- 비는 1층부터 t_i층까지 r_i만큼 오게 됩니다.
- $1 \leq t_i \leq N$이기 때문에 예보된 모든 비가 1층에는 온다는 것을 알 수 있습니다.

- $r_1, r_2, \cdots, r_k$까지 합쳤을 때 K를 초과하는 제일 작은 k가 무너지는 날이 되고, 무너지는 층으로는 1층을 제시하면 됩니다.
- r_i를 모두 더해도 K를 초과하지 않는다면 어디도 무너지지 않습니다.

- +) 최초로 무너지는 시점에 무너지는 층을 모두 구할 수도 있을까요?

C. 하이퍼 가지 따기

`geometry_hyper, arithmetic, implementation`

출제자: **wider93**

First Solve: **index** (3분)

C. 하이퍼 가지 따기

- 시도하지 않은 마지막 점을 포함해 2^{11}개 꼭짓점의 좌표를 모은 집합을 생각합시다.
- 각 $i(1 \le i \le 11)$에 대해 i번째 차원의 좌표는 항상 두 개의 값 L_i, R_i 중 하나를 갖고, 각각이 등장하는 횟수는 2^{10}개입니다.
- 따라서 민규가 시도하지 않은 마지막 점은 다른 $2^{11} - 1$개 점에서 $2^{10} - 1$번 나온 좌표들을 모아 만든 점입니다.
- 개수를 직접 세서 답을 내도 되고, 좌표별로 xor을 해서 값을 구하는 간편한 방법이 있습니다.

D. :danceplant:

`implementation, simulation`

출제자: **yup0927**

First Solve: **riroan** (15분)

D. :danceplant:

- 정직한 시뮬레이션입니다.

- 양분의 구간 합을 모두 구한 이후 매 확장마다 이용하는 방법이 있습니다.
- 구간 합 전처리에 $\mathcal{O}\left(N^2\right)$, 확장이 일어나는 범위의 값을 구하는 것이 $\mathcal{O}(1)$, 확장이 최대 $2N-4=\mathcal{O}(N)$번 일어납니다.

D. :danceplant:

- 하지만 구간 합을 사용하지 않아도 됩니다.
  확장이 일어나는 범위의 값을 매번 더해서 구해봅시다.

- 맨 처음 확장하는 방향을 결정하는데 8칸의 값을 더합니다.
- 확장이 한 번 일어난 뒤에는 10칸의 값을 더합니다.

$$\vdots$$

- 벽에 닿으면 벽에 닿은 변은 계산을 하지 않아도 되지만 편의상 한다고 가정하면 마지막에는 $4N$칸의 값을 더합니다.
- 총 더하는 횟수는 $8 + 10 + 12 + \cdots + (4N - 2) + 4N = \mathcal{O}\left(N^2\right)$가 됩니다.

E. 가지 사진 찾기

`binary_search`

출제자: **wider93**

First Solve: **hijkl2e** (18분)

E. 가지 사진 찾기

- 가장 중앙의 번호를 받은 사진은 항상 가지 사진임에 주목합니다.
- 쿼리 한 번을 통해 가지 범주의 이름을 알아낸 뒤, 왼쪽과 오른쪽 끝을 각각 이분 탐색으로 알아내면 됩니다.
- 따라서 최대 입력에서 필요한 총 쿼리 횟수는 $1 + 59 + 59$번으로 주어진 120회 이하입니다.

F. 가지 이모지

`ad_hoc, hash_set, number_theory`

출제자: **mingyu331**

First Solve: **riroan** (24분)

F. 가지 이모지

- 최대공약수를 k로 고정했을 때 적절히 반응을 할 수 있는지 판단하는 문제를 생각해 봅시다.
- 가능한 반응 방법은 존재한다면 유일합니다.
- 또한, 어떤 p에 대해서 최대공약수로 k와 pk가 모두 가능하다면 반응 위치가 같습니다.
- A_1과 A_1+1을 소인수분해한 후, 각 소인수 p에 대해서 모든 A_i가 $p^x \mid A_i + a_i$ $(a_i \in \{0,1\})$ 를 만족하도록 하는 x의 최댓값을 구하고, 이때의 a_i가 같은 p^x끼리의 곱을 구합니다.
- 곱 중의 최댓값이 답입니다.

F. 가지 이모지

- 별해로 모든 반응 방법을 백트래킹하되, 최대공약수가 2 이상인 경우만 시도하는 $\mathcal{O}(12N \log(A))$의 풀이가 존재합니다.
- 별해로 빠른 나눗셈을 사용하는 $\mathcal{O}(N\tau(A))$ 풀이가 존재합니다.

G. 슬슬 가지를 먹지 않으면 죽는다

`greedy, disjoint_set`

출제자: **kiwiyou**

First Solve: **hijkl2e** (22분)

G. 슬슬 가지를 먹지 않으면 죽는다

- 구하는 것은 그래프의 신장 트리 중 가중치의 MEX가 최대인 것입니다.
  (단, 0부터 세지 않음에 주의합시다.)
- 모든 가중치가 서로 다르다는 조건에 의해, 가중치가 1인 간선, 2인 간선, ⋯, N인 간선을 순서대로 고르다가, 사이클이 발생하면 멈추는 전략이 유효합니다.
- 크루스칼 알고리즘과 비슷한 방식으로 분리 집합을 이용해 $\mathcal{O}(M\alpha(M))$에 해결 가능합니다.
- 별해로, 프림 알고리즘을 이용해 최소 신장 트리를 구한 뒤 MEX를 구하는 방법이 있습니다.

H. SCP-504

`string, hash_map`

출제자: **kiwiyou**

First Solve: **ruykun** (82분)

H. SCP-504

- 문장에 단어 $a = \texttt{abac}$가 있다고 생각해 봅시다.
- 사전의 단어 b가 a의 웃기는 정도를 계산할 때 포함된다는 것과, b가 아래 정규식 중 하나를 만족함은 동치입니다.

$$\texttt{aa*}, \texttt{ab[ab]*}, \texttt{ab[ab]c[abc]*}$$

- 이 정규식열은 a의 모든 접두사에 다음 규칙을 차례로 적용해 만들어집니다.
  1. 단어에서 어떤 위치의 문자가 그 앞에서 이미 쓰였으면 `[앞쪽 모든 문자]`로 바꿉니다.
  2. 앞에서 쓰이지 않았다면 그 위치까지의 문자열 뒤에 `[앞쪽 모든 문자]+`를 붙이고 정규식열에 추가합니다.
- 정규식을 그대로 사용하면 $\mathcal{O}(NM|a|)$로 시간 초과를 받습니다.

H. SCP-504

- 위 정규식들은 포괄하는 범위가 서로 겹치지 않으므로, 사전에 있는 단어를 전처리한 후 맵에서 모든 접두사를 검색하는 방법을 생각할 수 있습니다.
- 이때, `[앞쪽 모든 문자]`를 나타내는 문자는 입력으로 주어지지 않는 문자를 사용하면 됩니다.
- 전처리 과정에서 `[ab]`와 `[abc]` 등이 같은 문자로 변환되더라도 괜찮다는 것을 증명할 수 있습니다.
- 시간복잡도는 해시맵을 사용하여 $\mathcal{O}\left(N|b| + M|a|^2\right)$입니다.
- 저장되는 문자열이 많으므로 트리맵은 시간 초과를 받을 수 있습니다.
- 트라이를 이용하면 $\mathcal{O}(N|b| + M|a|)$에 해결할 수 있지만 시간 및 메모리 제한에 걸릴 위험이 있습니다.

I. 가지 밭 게임

`parametric_search, convex_hull, polygon_area`

출제자: **wapas**

First Solve: **hijkl2e** (69분)

I. 가지 밭 게임

- 나무 기둥을 최적의 방법으로 선택하면 항상 다각형은 설치된 나무 기둥의 블록 껍질이 되어야 합니다.
- 다각형이 항상 블록 껍질이므로 다각형의 넓이는 플레이어 차례의 횟수가 증가함에 따라 단조 증가합니다.
- 다각형의 넓이를 이분 탐색하여 어느 플레이어가 최초로 이겼었는지 알아낼 수 있습니다.
- M번째 차례 때 블록 껍질을 $\mathcal{O}(M \log M)$안에 구할 수 있습니다.
- 꼭짓점이 M개인 다각형의 넓이는 가우스 면적 공식(사선 공식)으로 $\mathcal{O}(M)$안에 구할 수 있습니다.
- 이분 탐색의 결정 함수의 시간 복잡도는 $\mathcal{O}(M \log M)$입니다.
- 따라서 $\mathcal{O}\left(N \log^2 N\right)$에 해결할 수 있습니다.

J. 가지 자르기

`math, polygon_area, linear_algebra`

출제자: **mingyu331**

First Solve: **dabbler1** (104분)

J. 가지 자르기

- 가지를 한번 자른 후의 다각형 각 점의 좌표는, 이전 다각형 점 좌표의 선형 결합으로 표현됩니다.
- 이때, 선형 결합 행렬의 특성을 이용해, 두 행렬의 거듭제곱을 다항식 곱셈을 이용해 $\mathcal{O}\left(N^2\right)$에 곱할 수 있습니다.
- $\mathcal{O}\left(N^2 \log K\right)$에 행렬의 K제곱을 구한 후, 부호를 주의하여 사선 공식을 사용하면 $\mathcal{O}\left(N^2 \log K\right)$에 문제를 풀 수 있게 됩니다.
- 별해로 고속 푸리에 변환을 사용하는 $\mathcal{O}(N \log N \log K)$ 풀이가 존재합니다.

K. 가지 볶음

`number_theory, inclusion_and_exclusion, physics`

출제자: **wapas, wider93**

First Solve: **없음** (N/A)

K. 가지 볶음

- 두 재료가 충돌하는 경우 속도가 교환되는데 이를 굳이 생각하지 않아도 됩니다.
- 그냥 서로 통과하는 것으로 간주하고 같은 시점에 같은 지점을 통과하는 횟수를 세 주는 것과 같습니다.
- x, y좌표는 서로 개입하지 않습니다. 따라서 왠만한 계산을 1차원에서 분리해서 할 수 있습니다.
- x좌표의 초기 위치 x와 속도 dx를 갖는 가지의 t초 시점의 위치를 계산해 봅시다. 편의상 좌표에 N을 더해서 구간 $[0, 2N)$ 위로 올려서 생각합니다.
- 다음과 같은 함수들을 생각합시다.
  1. $f(t) := x + tdx$ (벽이 없는 공간에서 등속운동할 때의 위치)
  2. $g(x) := x - \left\lfloor \frac{x}{4N} \right\rfloor 4N$ (벽과 짝수 번 충돌한 경우의 실제 위치)
  3. $h(x) := 4N - x$ `if` $x > 2N$ `else` x (벽과 최대 한 번 더 충돌한 경우의 실제 위치)

K. 가지 볶음

- 그러면 $h(g(f(t)))$ 는 시점 t에서 실제 x좌표가 됩니다. 편의상 $dx \geq 0$인 것처럼 적었지만 $dx < 0$일 때도 성립합니다.
- 이제 임의의 두 재료가 시점 t에서 x좌표가 일치할 필요충분조건을 구했습니다. $h_0(g_0(f_0(t))) \pm h_1(g_1(f_1(t)))$ 중 $4N$의 배수인 것이 있으면 됩니다.
- 이 때 위 두 조건이 동시에 일어날 수 있음에 주의해야 합니다.
- 위 두 조건은 합동식으로

$$(x_0 \pm x_1) + t(dx_0 \pm dx_1) \equiv 0 (\mathrm{mod}\, 4N)$$

  꼴로 쓸 수 있습니다. 즉 $a + mt \equiv 0 (\mathrm{mod}\, 4N)$ 꼴입니다.
- x좌표와 y좌표가 동시에 충돌해야 하므로, $a + mt \equiv 0 (\mathrm{mod}\, 4N)$ 꼴의 합동식 여러 개를 연립하여 풀어주는 함수가 있다면 문제를 풀 수 있습니다.

K. 가지 볶음

- 합동식 $a_0 + m_0 t \equiv 0$과 $a_1 + m_1 t \equiv 0$이 성립하면, 정수 q에 대해 $(a_0 + m_0 t) + q(a_1 + m_1 t) \equiv 0$이 성립합니다.
- 나아가 정수 q가 0이 아니면, 다음 둘은 동치입니다.
  1. $a_0 + m_0 t \equiv 0$ and $a_1 + m_1 t \equiv 0$
  2. $a_1 + m_1 t \equiv 0$ and $(a_0 + m_0 t) + q(a_1 + m_1 t) \equiv 0$
- m에 유클리드 호제법을 적용하여 줄여 나가면, 유지되는 두 식 중 하나가 $a + 0t \equiv 0$ 꼴이 됩니다. 따라서 $a \equiv 0$이 성립하지 않으면 원래 두 합동식을 동시에 만족하는 실수 t는 없는 것입니다. 그렇지 않으면, 나머지 식을 만족하는 t가 원래 두 합동식을 동시에 만족하는 t가 됩니다.
- 위 과정을 통해 합동식 둘을 합동식 하나로 바꿀 수 있으면, 모든 $1 \le i < j \le M$에 대해 포함 배제의 원리를 통해 충돌 횟수를 셀 수 있습니다.
- 총 시간 복잡도는 $O\left(M^2 \log N^4\right)$입니다.

L. 가지농장 수확하기

`dp_tree`

출제자: **mingyu331**

First Solve: **kcm1700** (117분)

L. 가지농장 수확하기

- 노드의 집합 A에 대해, 창고에서 시작해서 A의 각 원소에 위치한 가지를 수확하고 다시 창고로 오는 것을 "A를 수확한다" 라고 정의합시다.
- $\mathrm{d}(u)$를 루트 노드에서 u까지 도달하는 데 거쳐야 하는 간선의 수로 정의합니다.
- $\{u, v\}$를 수확할 때 필요한 거리는 $\mathrm{d}(u) + \mathrm{d}(v) - \mathrm{d}(\mathrm{lca}(u, v))$가 됩니다.
- $\{u, v, w\}$를 수확할 때 필요한 거리는 $\mathrm{d}(u) + \mathrm{d}(v) + \mathrm{d}(w) - \mathrm{d}(\mathrm{lca}(u, v))) - \mathrm{d}(\mathrm{lca}(v, w))$가 됩니다.
- 위 식에서, $\mathrm{d}(u)$, $\mathrm{d}(v)$, $\mathrm{d}(w)$ 항들의 합들은 일정하기 때문에, $\mathrm{d}(\mathrm{lca}(u, v))$, $\mathrm{d}(\mathrm{lca}(v, w))$ 항들의 최대를 구하면 됩니다.
- 노드 u 밑의 서브트리를 수확할 때, a번 1개의 슈퍼 가지를 수확하고, b번 2개의 슈퍼 가지를 수확하고 나머지 슈퍼 가지는 3개씩 한번에 수확할 때 구하는 항들의 최댓값을 $\mathrm{dp}_{u,a,b}$로 정의합시다.

L. 가지농장 수확하기

- 이때, 1개의 슈퍼 가지를 2번 수확하는 대신 2개의 슈퍼 가지를 한 번에 수확하거나, 1개의 슈퍼 가지 수확과 2개의 슈퍼 가지 수확 대신 3개의 슈퍼 가지를 한 번에 수확할 수 있습니다.
- 이 dp식을 이용하면 $\mathcal{O}\left(N^4\right)$에 문제를 해결할 수 있습니다.
- 이때, 깊이 우선 탐색의 순서를 따르면 d 값은 dp식이 합쳐질수록 줄어듭니다.
- 어떤 두 수확이 dp_u를 계산하는 과정에서 하나의 수확으로 합쳐진다면, 이보다 d가 낮은 노드에서는 두 수확이 합쳐져도 최적이 되지 못합니다.
- 즉, $\mathrm{dp}_{u,a,b}$에서 a 또는 b가 적당한 상수 M에 대해서 $a < M$ 또는 $b < M$일 때의 $\mathrm{dp}_{u,a,b}$만 저장해도 정확한 답을 구할 수 있습니다.
- 크기 u와 크기 v의 dp값을 $\mathcal{O}(uv)$에 합치게 된다면 $\mathcal{O}\left(N^2\right)$에 문제를 해결 할 수 있습니다.