아니메컵 2기
공식 에디토리얼

by

아니메컵 제작위원회

지원

1화. DORO

#string

출제진 의도 - Easy

✓ 출제자: Hyperbolic

✓ 제출: 431번, 정답: 348명 (정답률 81.439%), 처음 푼 사람: jjang36524, 0분

1화. DORO

- ✓ 단어들을 순서대로 입력받으면서, 단어를 입력 받을 때 마다 입력 받은 단어를 그대로 출력하고, 추가로 DORO를 출력하면 됩니다.
- ✓ 단어들이 공백으로 구분되기 때문에, 입력을 받을 때 공백 처리에 주의해야 합니다. 문장을 입력받을때 공백이 등장하면 입력 받는 것을 멈추는 입력 방식이 있고, 공백까지 포함하여 전체 한줄을 읽는 입력 방식이 있습니다. C언어의 경우, 전자는 대표적으로 `scanf("%s")`가 존재하며, 후자는 `fgets`가 존재합니다.

2화. 세그먼트 트리보다도 바•로•너•♡

#sorting, #map, #implementation

출제진 의도 - Normal

✓ 출제자: oh040411

✓ 제출: 317번, 정답: 140명 (정답률 44.479%), 처음 푼 사람: dabbler1, 4분

2화. 세그먼트 트리보다도 바•로•너•♡

- ✓ 변수의 크기 제한이 매우 작기 때문에, 시간 초과를 받는 경우는 거의 없습니다. 편한 방법으로 구현하면 됩니다.
- ✓ 출제자가 선호하는 구현 방법은 `silver-chan`이 좋아하는 `sorting`과 `hash_map`을 사용하는 것입니다.
- ✓ 알고리즘의 경우, (이름, 난이도)의 쌍을 배열에 저장해 놓으면 편리합니다.
- ✓ 현재 대답하는 멤버의 이름을 저장하고 있는 문자열 변수 m을 선언합니다.
- ✓ 쿼리 입력이 `nani ga suki?`이면 2번 쿼리를, 그렇지 않으면 1번 쿼리를 실행합니다.
  - ✓ 멤버의 이름으로 `nani`가 입력될 수 있기 때문에 첫 번째 단어로만 판단하지 않도록 주의합니다.

2화. 세그먼트 트리보다도 바•로•너•♡

- ✓ 1번 쿼리의 경우, 멤버의 이름 name을 입력받아서 m에 저장하고, `hai!`를 출력합니다.
- ✓ 2번 쿼리의 경우, 멤버 m의 티어를 t라고 합시다.
  - ✓ 이름을 key, 티어를 value로 하는 map이나 dictionary를 이용하면 t를 쉽게 구할 수 있습니다.
    - ✓ 자료 구조를 사용하지 않고 완전 탐색으로 구할 수도 있습니다.
  - ✓ 알고리즘을 $(|\text{난이도} - t|, \text{이름})$을 기준으로 정렬 후 (2번째 알고리즘의 이름) `yorimo` (1번째 알고리즘의 이름)을 출력합니다.
  - ✓ C++/Python 등 대부분의 언어에서 내장 정렬 함수의 비교 기준을 설정하여 구현할 수 있습니다.
  - ✓ 정렬을 사용하지 않고 최솟값과 두 번째 최솟값을 갱신하는 방법으로 구현해도 됩니다.

3화. 폴카의 수학 공부

#ad_hoc, #math

출제진 의도 - Hard

✓ 출제자: Hyperbolic

✓ 제출: 469번, 정답: 98명 (정답률 21.535%), 처음 푼 사람: dong5995, 4분

3화. 폴카의 수학 공부

✓ 문제에서 주어진 수식을 $a_1 \bigcirc a_2 \bigcirc \ldots \bigcirc a_N \bigcirc a_{N+1}$로 표현하겠습니다. 모든 a_i는 0 이상 9 이하의 숫자이며, $\bigcirc$ 연산자는 $+$ 또는 $-$ 중 하나를 의미합니다.

3화. 폴카의 수학 공부

- ✓ 모든 a_i가 0인 경우를 생각해봅시다. 이 경우는 항상 결과값이 0이므로, 답은 YES입니다.
- ✓ 0이 아닌 수 중 가장 오른쪽에 있는 수를 a_t라고 합시다.
- ✓ a_{t-1}보다 왼쪽에 있는 $-$ 연산자 중, 가장 오른쪽에 있는 $-$ 연산자의 위치를 $a_{s-1} - a_s$라고 합시다.
- ✓ 즉, $a_1 \bigcirc a_2 \bigcirc ... \bigcirc a_{s-1} - a_s + ... + a_{t-1} \bigcirc a_t \bigcirc 0 \bigcirc ... \bigcirc 0$ 꼴을 의미합니다.
- ✓ 이때 $a_{s-1} - (a_s + ... + a_{t-1} \bigcirc a_t)$의 값과 $a_{s-1} - (a_s + ... + a_{t-1}) \bigcirc a_t$의 값이 $2a_t$만큼 다르므로, 답은 NO가 됩니다.
- ✓ 그러한 $-$ 연산자가 없다면, 식은 $a_1 + a_2 + ... + a_{t-1} \bigcirc a_t \bigcirc 0 \bigcirc ... \bigcirc 0$ 꼴입니다. 이 경우 a_{t-1}과 a_t 사이에 어떤 연산자가 오더라도 답이 변하지 않습니다.
- ✓ 결론적으로, a_1과 a_{t-1} 사이에 $-$가 있으면 답은 NO, 아니면 YES입니다.

4화. 리오와 리쿠의 대난투

`#case_work, #constructive`

출제진 의도 - Hard

✓ 출제자: Hyperbolic

✓ 제출: 181번, 정답: 57명 (정답률 31.492%), 처음 푼 사람: jjang36524, 17분

4화. 리오와 리쿠의 대난투

✓ 먼저, $N = 1, M = 1$인 경우를 생각해봅시다.

✓ 이 경우는 두 사람의 결과가 반대가 되도록 만드는 것은 불가능하므로, 답은 -1입니다.

✓ $N = 1, M = 2$인 경우를 생각해봅시다.

✓ $a = [3], b = [1, 4]$면 충분합니다.

4화. 리오와 리쿠의 대난투

- ✓ $N = 1, M \geq 3$인 경우를 생각해봅시다.
- ✓ $a = [M], b = [1, 2, ..., M-1, 10^9]$로 잡으면 충분합니다.
- ✓ 왜냐하면, $avg_{anime} = M, avg_{normal} = \frac{b_1+...+b_M}{M} > \frac{b_M}{M} > M$이여서 리오는 B집단을 더 잘한다고 생각하며, $cnt_{anime} = M-1, cnt_{normal} = 1$이여서 리쿠는 A집단을 더 잘한다고 생각하기 때문입니다.

4화. 리오와 리쿠의 대난투

- ✓ $N \geq 2, M \geq 1$인 경우를 생각해봅시다.
- ✓ $a = [1, 2, ..., N-1, 10^9], b = [N, N+1, ..., N+M-1]$로 잡으면 충분합니다.
- ✓ 왜냐하면, $avg_{normal} = \frac{b_1+...+b_M}{M} \leq N+M-1, avg_{anime} = \frac{a_1+...+a_N}{N} > \frac{a_N}{N} > N+M-1$이여서 리오는 A집단을 더 잘한다고 생각하며, $cnt_{anime} = M, cnt_{normal} = M(N-1)$이여서 리쿠는 B집단을 더 잘한다고 생각하기 때문입니다.

4화. 리오와 리쿠의 대난투

- ✓ 별해로, 답이 나올때까지 $a_1, ..., a_N, b_1, ..., b_M$값을 랜덤하게 설정해주는 방법이 있습니다.
- ✓ 실험적으로 약 8%의 확률로 리오와 리쿠의 결과가 반대로 나오기 때문에, 기댓값 약 $\frac{100}{8}NM$번의 연산을 통해 답을 구할 수 있습니다.

5화. Rabbit Horse

#math, #combinatorics, #constructive, #greedy

출제진 의도 - Hard

✓ 출제자: ibm2006

✓ 제출: 248번, 정답: 61명 (정답률 25.000%), 처음 푼 사람: ian0704, 22분

5화. Rabbit Horse

✓ 우선, 문자 e는 항상 맨 뒤로 빼주는 것이 그리디하게 이득입니다.

✓ 비슷한 논리로 s도 e보단 앞에 맨 뒤로 빼주는 것이 이득입니다.

✓ 위의 논리를 조금 더 응용해봅시다.

5화. Rabbit Horse

- ✓ a, i, t, h, o, s, e와 같이 $rabbithorse$에서 한번씩만 등장하는 문자들을 생각해봅시다. 이들중 하나를 임의로 x라고 두겠습니다.
- ✓ 최적의 설명문 S에서 어떤 x의 위치는, 가장 많은 $rabbithorse$ 부분수열에 포함됩니다.
- ✓ 이제 S의 모든 x들을 해당 위치로 모아주면 $rabbithorse$의 부분수열 개수는 여전히 최대임을 알 수 있습니다. 이 과정에서 이미 존재하는 어떤 알파벳의 묶음이 갈라지지는 않습니다. 해당 연산을 모든 x에 대해 순차적으로 적용해주면 S의 a, i, t, h, o, s, e들을 전부 한 묶음으로 묶어줄 수 있습니다.
- ✓ 즉, 최적해는 a, i, t, h, o, s, e 문자들이 각각 하나의 묶음으로 인접해있는 형식으로 나타남을 알 수 있습니다. 이 묶음들의 순서는 $rabbithorse$에 등장한 순서와 일치하지 않는다면 부분수열의 개수가 0개가 되므로, 이들의 순서는 $rabbithorse$에 등장한 대로임을 알 수 있습니다.

5화. Rabbit Horse

✓ b의 경우에도 비슷한 논리를 적용할 수 있습니다. b의 개수를 m이라고 하고, S에서 b만을 제거한 문자열을 어떤 위치를 기준으로 반으로 쪼갰을 때, 왼쪽에 있는 ra의 부분수열의 개수, 오른쪽에 있는 $ithorse$의 부분수열의 개수의 곱이 최대가 되는 지점을 고르고 나면, 가능한 $rabbithorse$ 부분수열의 개수는 해당 곱에 $\frac{m(m-1)}{2}$를 곱한 값임을 알 수 있고, 이는 실제로 b를 해당 위치에 전부 집어넣음으로써 달성될 수 있습니다.

5화. Rabbit Horse

- ✓ 이제 r을 배치하는 것만이 남았습니다. 앞선 관찰들에 따라 r이 들어가서 답에 기여할 수 있는 위치는 a의 바로 앞, 혹은 o와 s 사이밖에 존재하지 않으므로 r 또한 두 묶음만 존재하게 된다는 것을 어렵지 않게 확인할 수 있습니다.
- ✓ 따라서 최적의 설명문을 고려하기 위해서는, $r...ra...ab...bi...it...th...ho...or...rs...se...e$의 형태만 고려하면 됩니다.
- ✓ b의 개수가 고정되었을 때, $rabbithorse$ 부분수열의 개수를 최대화하는 방법은 인접한 같은 문자들로 구성되는 그룹의 크기를 최대한 균일하게(즉, 가장 큰 그룹의 크기와 가장 작은 그룹의 크기의 차이가 최소화되도록) 집어넣는 것입니다. 크기의 차이가 2 이상인 두 그룹이 있다면, 이들의 차이를 좁히면 이득을 보기 때문입니다. 따라서 가능한 모든 b에 대해서 설명문을 고려하면 최적의 설명문을 얻을 수 있습니다.

5화. Rabbit Horse

✓ Challenge: 원래 문제는 $COCOA$ 부분수열의 개수를 최대화하는 버전이었습니다.

6화. 벨과 와이즈의 가게 홍보

#greedy, #sorting, #permutation_cycle_decomposition

출제진 의도 - Hard

✓ 출제자: Hyperbolic

✓ 제출: 144번, 정답: 49명 (정답률 34.028%), 처음 푼 사람: fermion5, 18분

6화. 벨과 와이즈의 가게 홍보

- ✓ 첫 번째에 위치한 Bangboo와 N번째에 위치한 Bangboo는 항상 웅나해질 수 없습니다. 그러므로, 웅나한 Bangboo의 최댓값은 $N-2$ 이하입니다.
- ✓ 또한, 웅나한 Bangboo의 수가 $N-2$가 되려면 모든 $2 \leq i \leq N-1$에 대하여 $a_{i-1} < a_i < a_{i+1}$ 혹은 $a_{i-1} > a_i > a_{i+1}$을 만족해야합니다.
- ✓ 이 경우는 $a_1, ..., a_N$이 오름차순 혹은 내림차순인 경우 단 2가지 밖에 존재하지 않습니다.
- ✓ 오름차순인 경우 $a_i = i$를 만족하며, 내림차순인 경우 $a_{N+1-i} = i$이므로 수열 a_i를 뒤집은 후 오름차순으로 만드는 것으로 생각할 수 있습니다.
- ✓ 따라서 두 Bangboo의 위치를 맞바꾸는 연산을 최소한으로 하여 $a_i = i$로 만들 수 있으면 충분합니다.

6화. 벨과 와이즈의 가게 홍보

- 다음과 같은 그리디 전략을 생각해봅시다.
  - $i = 1, 2, ..., N$ 순서대로 보면서,
  - $a_i = i$이면 이미 오름차순으로 정렬된 상태이므로 아무 행동도 하지 않고,
  - $a_i \neq i$이면 $a_j = i$인 j를 찾아서 a_i와 a_j를 swap합니다.
- 즉, 선택 정렬(selection sort)을 수행하는 것과 같습니다.
- a_i를 인덱스로, i를 값으로 하는 배열을 관리하면 이 전략을 $\mathcal{O}(N)$에 수행할 수 있습니다.
- 결론적으로 말하면, 이때의 swap 횟수가 최소 연산 횟수가 됩니다.
- 이 전략의 정당성을 증명하려면 순열 사이클 분할에 대해서 알아야 합니다.

6화. 벨과 와이즈의 가게 홍보

- ✓ 정점이 N개인 그래프를 생각해봅시다. 모든 $1 \leq i \leq N$에 대하여, i에서 a_i로 가는 간선을 그어봅시다.
- ✓ a_i는 순열이므로, $1, 2, ..., N$과 일대일 대응이 됩니다. 따라서 각 정점 i로 들어오는 간선 하나와 나가는 간선 하나가 유일하게 존재합니다.
- ✓ 따라서, 이 그래프는 방향을 가지는 사이클(cycle)들로 분할이 됩니다. 이를 순열 사이클 분할이라고 부릅니다.
- ✓ a_i를 오름차순으로 만드는 것은 사이클의 개수를 N까지 늘리는 것과 같습니다.

6화. 벨과 와이즈의 가게 홍보

- ✓ i번 정점을 포함하는 사이클의 길이를 L이라고 가정합시다. 이 경우 적절한 정점들 $v_1, ..., v_L$에 대하여 사이클을 $v_1 \to v_2 \to ... \to v_L \to v_1$ 형태로 표현할 수 있습니다. 일반성을 잃지 않고 $v_1 = i$라고 가정합니다.
- ✓ 만약 $L = 1$이라면, 이 사이클은 $i \to i$ 형태입니다. 이는 $a_i = i$를 의미하는 것이며, 이미 조건을 만족하므로 어떠한 연산을 해줄 필요가 없습니다.
- ✓ 그 외의 경우, v_L은 i와 다르며, 이 값을 j라고 정의합시다. 다시 말해, 이 사이클이 $i \to a_i \to ... \to j \to i$라고 가정합니다.
- ✓ i번 Bangboo와 j번 Bangboo를 바꾸는 연산을 하는 경우, $i \to a_i$는 $i \to i$로, $j \to i$는 $j \to a_i$로 바뀝니다. 따라서 이 사이클은 $i \to i$와 $a_i \to ... \to j \to a_i$라는 사이클 2개로 나누어지게 됩니다. 즉, 사이클의 개수가 1 증가합니다.
- ✓ 한편, 어떠한 swap 연산도 사이클의 개수를 2 이상 증가시킬 수 없음을 증명할 수 있습니다. 따라서 선택 정렬은 사이클의 개수를 가장 빠르게 증가시키는 전략입니다.
- ✓ 이때의 swap 횟수는 $N - (\text{초기 사이클의 개수})$와 같습니다.

7화. 좋아하는 다이아몬드가 안경을 깜빡했다

#bfs, #shortest_path, #traceback

출제진 의도 - Hard

✓ 출제자: oh040411

✓ 제출: 85번, 정답: 44명 (정답률 51.765%), 처음 푼 사람: coxld, 26분

7화. 좋아하는 다이아몬드가 안경을 깜빡했다

- ✓ 1번 장소와 N번 장소를 잇는 최단 경로에 포함되는 정점만 답의 후보가 될 수 있습니다.
- ✓ 이러한 정점들의 집합을 구하는 다양한 방법이 있습니다. 그 중 역추적을 사용하지 않는 한 가지 방법을 소개합니다.
  - ✓ 두 정점 u, v를 잇는 최단 경로의 길이를 $\mathrm{dist}(u, v)$라고 정의합시다.
  - ✓ 어떤 정점 u가 최단 경로에 포함되는 것과, $\mathrm{dist}(1, u) + \mathrm{dist}(u, N) = \mathrm{dist}(1, N)$이 성립하는 것은 동치입니다.
  - ✓ 두 번의 BFS로 $\mathrm{dist}(1, u)$와 $\mathrm{dist}(N, u)$를 전처리하면 위 식을 만족하는 모든 u를 $\mathcal{O}(M)$에 구할 수 있습니다.
- ✓ 1번 정점에서 BFS를 수행한 후 N번 정점에서 역추적을 해도 됩니다.

7화. 좋아하는 다이아몬드가 안경을 깜빡했다

- ✓ 이제 최단 경로에 포함되는 정점만 고려하겠습니다.
- ✓ 학교에서 어떤 정점까지의 최단 거리 (이하 "거리") $d_u := \mathrm{dist}(1, u)$를 정의합시다.
- ✓ 모든 최단 경로는 거리가 $0, 1, 2, ..., d_N - 1, d_N$인 정점으로 구성됩니다.
- ✓ 어떤 정점 u가 이 모든 최단 경로에 포함된다는 것은, 거리가 d_u인 정점이 u로 유일하다는 것과 동치입니다.
  - ✓ $d_u = d$인 정점 u가 유일하다면, 모든 최단 경로는 거리가 d인 정점을 지나므로 u를 포함해야 합니다.
  - ✓ $d_u = d_v$인 다른 정점 v가 존재한다면, u를 지나지 않고 v를 지나는 최단 경로가 존재하는 것입니다.
- ✓ 따라서 최단 경로에 포함되는 정점 중 거리가 $1, 2, ..., d_N - 1$인 정점의 개수를 세는 배열을 만들고, 특정 거리에 있는 정점의 개수가 1이라면 그것을 출력하면 됩니다.

7화. 좋아하는 다이아몬드가 안경을 깜빡했다

- √ Challenge: 간선의 가중치가 음이 아닌 정수로 주어졌을 때도 이 문제를 해결할 수 있습니다.
  - √ 다음 문제에 도전해 보세요: https://www.acmicpc.net/problem/11209

8화. 흰수염과 해적들

#greedy, #priority_queue

출제진 의도 - Expert

✓ 출제자: gmb9817, Hyperbolic

✓ 제출: 118번, 정답: 26명 (정답률 22.034%), 처음 푼 사람: octane, 50분

8화. 흰수염과 해적들

- ✓ 입력으로 주어지는 모든 해적들은 초기에 같은 위치에 존재하는 경우가 없습니다. 그런데, 임의의 시간이 흐른 후에도 이 성질은 만족합니다. 다시 말해, 임의의 기절하지 않은 두 해적은 그 어떤 시점에도 같은 위치에 존재할 수가 없습니다. 이는 해적들의 좌표를 극좌표로 생각해보면 쉽게 알 수 있습니다.
- ✓ 따라서, 흰수염이 능력을 사용할 때, 거리가 L이하인 해적 단 한명만 기절시키는 것으로 생각해도 문제가 바뀌지 않습니다. 이제부터, 흰수염이 능력을 사용할때 특정 위치에 존재하는 해적 단 한명만 기절시키는 것으로 생각하겠습니다.

8화. 흰수염과 해적들

✓ i번째 해적들에 대해서, deadline[i] = i번째 해적이 흰수염의 능력 범위를 벗어날 때까지 걸리는 시간으로 정의하겠습니다. 엄밀하게, deadline[i]는 다음과 같이 정의가 됩니다.

- 만약 $\sqrt{x_i^2 + y_i^2} > L$이라면 deadline[i] $= 0$ 입니다.

- 그 이외의 경우, 만약 적절한 정수 t에 대하여 $L - t < \sqrt{x_i^2 + y_i^2} \leq L - t + 1$이라면, deadline[$i$] $= t$입니다.

✓ 각 i마다 deadline은 이분탐색을 통하여 $\mathcal{O}(\log L)$의 시간에 계산할 수 있습니다. 따라서, 모든 해적들의 deadlline의 계산에 $\mathcal{O}(N \log L)$이 필요합니다.

8화. 흰수염과 해적들

기절 시킨 모든 해적들의 집합을 S라고 정의합시다. S는 다음을 만족시켜야 합니다.

- ✓ 음이 아닌 정수 t에 대하여, deadline이 t이하인 해적은 S라는 집합에 최대 t명만 존재해야 합니다. 그 이외의 경우, 적어도 한 명 이상의 해적을 기절시킬 수 없다는 것을 증명할 수 있습니다.
- ✓ 기절시킨 해적들의 현상금의 합을 최대화 하고 싶기 때문에, S는 위 조건을 만족하는 것 중에서 해적들의 현상금의 합이 최대여야 합니다.

이러한 S는 다음과 같은 방법으로 계산할 수 있습니다.

- ✓ 초기에 $S = \{\}$, 즉 공집합으로 설정합니다.
- ✓ 양의 정수 $t = 1, 2, 3, ..., L$에 대하여, S에 deadline이 t인 모든 해적들을 insert합니다. insert한 이후, S의 크기가 t이하가 될 때 까지 가장 현상금이 적은 해적들을 pop합니다.
- ✓ 위 과정을 거친 후, S에 속한 모든 해적들의 현상금의 합이 이 문제의 정답이 됩니다.

8화. 흰수염과 해적들

- ✓ $L \leq N$이라면 문제가 없지만, $L > N$인 경우 L이 10^9까지 가능하므로, 위 과정을 그대로 거치면 시간초과를 받게 됩니다.
- ✓ 하지만, $t \geq N$이라면 S의 크기가 t이하여야 한다는 조건은 무시할 수 있기 때문에 pop 연산을 하지 않는다고 생각해도 무방합니다. 그러므로, $t = 1, 2, .., N-1$일 때만 앞에서 언급한 방법을 사용하고, 마지막에 deadline이 N이상인 모든 해적들을 S에 insert 해주어도 충분합니다.
- ✓ S는 우선순위 큐 등으로 관리가 가능하며, 이 경우 insert와 pop에 $\mathcal{O}(\log N)$의 시간이 필요합니다. 임의의 해적은 최대 1번 insert되며 최대 1번 pop되기 때문에, 모든 insert와 pop의 수를 합치면 $2N$이하가 됩니다. 따라서, S의 관리에 총합 $\mathcal{O}(N \log N)$의 시간이 필요합니다.

시간복잡도는 $\mathcal{O}(N \log L + N \log N)$입니다.

9화. 일천광년

#math, #invariant

출제진 의도 - Append

✓ 출제자: jalapen0_p1ckle

✓ 제출: 41번, 정답: 11명 (정답률 29.268%), 처음 푼 사람: lukanel, 59분

9화. 일천광년

✓ 운영진도 난이도를 예측할 수 없는 문제입니다(?)
✓ 개인차가 상당히 커서 Append 난이도로 표기했습니다.

✓ $r - l > 1$이면 사랑을 옮긴 후에 $r - l$이 작아집니다.

✓ $r - l$의 값이 1 또는 0일 때 시행을 멈추므로 최종 상태에서 사랑이 존재하는 위치는 서로 인접한 두 곳이거나 한 곳임을 알 수 있습니다.

✓ 시행을 멈췄을 때 존재하는 가장 앞쪽 위치가 모래의 행성에서 $a\ \mathrm{km}$ 떨어진 곳이라고 합시다.

✓ $L_a > 0$, $L_{a+1} \geq 0$이고 그 외의 위치에서 $L_i = 0$입니다.

9화. 일천광년

- ✓ 사랑을 옮기는 시행에서 두 가지 불변량을 찾을 수 있습니다.
- ✓ 모든 시각에서 사랑의 총량 $\sum_{i=0}^{N} L_i$은 일정합니다.
  - ✓ $L_a + L_{a+1} = x + y$ 를 만족합니다.
- ✓ l의 사랑 M만큼이 $l+1$로 이동할 때 r의 사랑 M만큼이 $r-1$로 이동하므로 $\sum_{i=0}^{N} iL_i$는 일정합니다.
  - ✓ 이를 직관적으로 사랑의 "평균 위치" $\frac{\sum iL_i}{\sum L_i}$가 일정하다는 의미로 생각할 수도 있습니다.
  - ✓ $aL_a + (a+1)L_{a+1} = Ny$ 를 만족합니다.

9화. 일천광년

- ✓ 첫 식의 양변에 a를 곱해 두 식을 연립하면, 하나의 식으로 나타낼 수 있습니다:
  - ✓ $a(x+y)+L_{a+1}=Ny$
- ✓ $L_a+L_{a+1}=x+y$이고 $L_a>0$, $L_{a+1}\geq 0$이므로 $0\leq L_{a+1}<x+y$입니다.
- ✓ a는 Ny를 $x+y$로 나눈 몫, L_{a+1}은 Ny를 $x+y$로 나눈 나머지가 됩니다.
  - ✓ 사랑의 "평균 위치" $\dfrac{\sum iL_i}{\sum L_i}=\dfrac{Ny}{x+y}$가 불변량이므로, 모든 사랑은 그 위치 근처로 모일 것입니다.
  - ✓ 따라서 사랑이 모이는 위치는 $\left\lfloor \dfrac{Ny}{x+y} \right\rfloor$, $\left\lceil \dfrac{Ny}{x+y} \right\rceil$인 것으로 볼 수 있습니다.
- ✓ L_a는 $L_a+L_{a+1}=x+y$에 L_{a+1}를 대입해서 구할 수 있습니다.

10화. Lv2부터 치트였던 전직 아이돌 한별이의 알록달록 트리 라이프

#dp_tree, #combinatorics

출제진 의도 - Expert

✓ 출제자: kdy40929

✓ 제출: 21번, 정답: 17명 (정답률 80.952%), 처음 푼 사람: 79brue, 65분

10화. Lv2부터 치트였던 전직 아이돌 한별이의 알록달록 트리 라이프

- ✓ 트리의 레벨이 2 이상인 경우의 수는 전체 경우의 수에서 트리의 레벨이 0 또는 1인 경우의 수를 빼서 구할 수 있습니다.
- ✓ 트리의 레벨이 0 또는 1임은 각 정점에 대해 인접한 정점 중 다른 색인 정점의 수가 최대 1개라는 것을 의미합니다.
- ✓ 트리의 레벨이 0 또는 1이 되도록 트리를 색칠하는 방법의 수는 tree dp를 이용해서 셀 수 있습니다.

10화. Lv2부터 치트였던 전직 아이돌 한별이의 알록달록 트리 라이프

- ✓ 1번 정점을 루트로 잡겠습니다.
- ✓ 각 정점 v에 대해서 $A[v]$를 v의 색상이 정해진 상태에서 v의 모든 자식 노드를 v와 같은 색이 되도록 v가 루트인 서브트리를 칠하는 방법의 수로 정합니다.
- ✓ 각 정점 v에 대해서 $B[v]$를 v의 색상이 정해진 상태에서 v의 자식 노드 중 오직 하나를 v와 다른 색이 되도록 v가 루트인 서브트리를 칠하는 방법의 수로 정합니다.
- ✓ 리프 노드에 대해서는 $A[v] = 1, B[v] = 0$으로 잡을 수 있습니다.

10화. Lv2부터 치트였던 전직 아이돌 한별이의 알록달록 트리 라이프

- ✓ S를 v의 자식 노드들의 집합으로 정의합니다.
- ✓ 정점 v를 v의 모든 자식과 같은 색으로 칠하는 경우 각 자식 c를 루트로 하는 서브트리를 칠하는 경우의 수는 $A[c]+B[c]$입니다.
- ✓ 따라서 $A[v]=\prod_{c\in S}(A[c]+B[c])$가 성립합니다.
- ✓ 정점 v를 v의 자식 중 하나와 다른 색으로 칠할 때는 다른 색으로 칠하려는 자식 u를 루트로 하는 서브트리를 칠하는 경우의 수가 $A[u]$입니다.
- ✓ 따라서 $B[v]=\sum_{u\in S}\left(A[u]\times\prod_{c\in S,c\neq u}(A[c]+B[c])\right)$ 가 성립합니다.
- ✓ 하지만 위 식을 그대로 구현하면 $\Omega(N^2)$이 소요됩니다.

10화. Lv2부터 치트였던 전직 아이돌 한별이의 알록달록 트리 라이프

- ✓ $A[c] + B[c]$의 곱이 연산 과정에서 반복되므로 이를 누적 곱의 형태로 미리 계산하여 시간복잡도를 $\mathcal{O}(N)$으로 줄일 수 있습니다.
- ✓ 구체적으로, v의 자식을 $c_1, c_2, ..., c_j$라고 할 때 $p(k) = \prod_{i=1}^{k} (A[c_i] + B[c_i])$와 $s(k) = \prod_{i=k}^{j} (A[c_i] + B[c_i])$를 계산합니다. (단, $p(0) = 1, s(j+1) = 1$)
- ✓ 그러면 $B[v] = \sum_{k=1}^{j} (p(k-1) \cdot A[k] \cdot s(k+1))$가 성립합니다.
- ✓ 한편, 각 정점을 코랄색과 개나리색 중 하나의 색으로 칠하는 경우의 수는 2^N입니다.
- ✓ 1번 노드에 대해 색상을 2개 중 하나로 정할 수 있으므로 $2^N - 2(A[1] + B[1])$을 M으로 나눈 나머지가 정답입니다.

11화. 결계 배치하기

#combinatorics, #dp

출제진 의도 - Expert

✓ 출제자: flakepowders

✓ 제출: 21번, 정답: 11명 (정답률 52.381%), 처음 푼 사람: 79brue, 73분

11화. 결계 배치하기

- ✓ 에너지원들을 M개의 연속된 그룹으로 묶어서 생각합시다. 각 그룹당 결계를 하나씩 할당해야 합니다.
- ✓ 각 그룹의 양쪽 끝 에너지원이 같은 결계와 충돌하게끔 하기만 하면, 그 그룹 안의 모든 에너지원들도 같은 결계에 충돌할 것임을 알 수 있습니다.
- ✓ 편의상 그룹의 크기 $L = \frac{M}{N}$에 대해, j번째 그룹의 첫 에너지원을 $s_j = a_{(j-1)L+1}$, 마지막 에너지원을 $e_j = a_{jL}$로 잡습니다.
- ✓ 두 결계 사이의 중간점을 기준으로, 에너지원이 그보다 왼쪽에 있는지 오른쪽에 있는지 여부에 따라 어느 결계와 충돌할지 결정됩니다.
- ✓ 정확히는, $m_j = \frac{b_j + b_{j+1}}{2}$으로 두면, $e_j \leq m_j < s_{j+1}$을 만족해야 합니다.
- ✓ 식을 정리하면, 모든 j에 대해 $2e_j \leq b_j + b_{j+1} < 2s_{j+1}$이 되어야 합니다.

11화. 결계 배치하기

- ✓ 이를 이용해 점화식을 세워봅시다.
- ✓ $dp[j][x]$를 처음 j개까지의 결계를 선택했으며 $b_j = x$인 경우의 수로 정의합니다.
- ✓ 직전에 선택한 에너지원 위치, 즉 $b_{j-1} = y$에 대해, $b_j = x$를 선택하려면,
- ✓ $y < x \wedge 2e_{j-1} < y + x < 2s_j$를 만족해야 합니다.
- ✓ 이 조건을 만족하는 전 상태들의 합을 계산하면 되는데, 누적 합을 통해 빠르게 구하면 됩니다.
- ✓ 그런데 사실 누적 합을 쓰지 않아도 충분히 빠릅니다. 위의 합칠 범위를 잘 보면, 각 j에 대해 범위들을 모두 이어붙여보면 1부터 K까지의 범위가 됩니다.
- ✓ 따라서 순서대로 나이브하게 구간 합을 계산해도 합칠 값의 개수는 K로 bound됩니다.
- ✓ $\sum_{x=1}^{K} dp[M][x] \bmod 998244353$을 최종 답으로 출력하면 됩니다.
- ✓ Challenge: $\mathcal{O}(K)$ 등 더 낮은 시간복잡도에도 해결할 수 있는 문제입니다.

12화. D메일

#bitmasking, #constructive

출제진 의도 - Master

✓ 출제자: pyb1031

✓ 제출: 32번, 정답: 6명 (정답률 25.000%), 처음 푼 사람: flappybird, 67분

12화. D메일

- ✓ 문제의 상황에서 예상할 수 있듯이 굉장히 다양한 답이 존재합니다.
- ✓ 그 중 출제자가 의도한 풀이를 소개합니다.

12화. D메일

- ✓ 쿼리로 날리는 값을 조금씩 바꿔가며 그 변화를 관찰하면 좋을것 같습니다.
- ✓ f를 1비트의 개수를 중심으로 설계한다면 숨겨진 값과 상관없이 변화를 알기 쉬울것 같습니다.
- ✓ 어떤수 x를 이진수로 나타냈을 때의 1의 개수를 $\mathrm{BitCount}(x)$라고 정의합시다.
- ✓ N의 값이 3으로 나눠지지 않는 경우, $\mathrm{BitCount}(x)\%3 = 0$이라면 $f(x) = 1$, 그렇지 않다면 $f(x) = 0$으로 값을 정합니다.
- ✓ N의 값이 3으로 나눠지는 경우에는 $\mathrm{BitCount}(x)\%3 = 1$이라면 $f(x) = 1$, 그렇지 않다면 $f(x) = 0$으로 값을 정합니다.
- ✓ 구해야 하는 값, 즉 초기 세계선의 번호를 k라고 합시다.
- ✓ f의 값을 모두 정한 후, $f(k \oplus x) = 0$가 되는 x를 아무거나 하나 찾고 이를 base라고 합시다. 어떤 x에 대해 $f(k \oplus x) = 1$이라면 $f(k \oplus (x \oplus 1)) = 0$이 되기 때문에 이런 x를 한번에 찾을 수 있습니다.

12화. D메일

- ✓ $f(k \oplus \text{base} \oplus 2^i)$와 $f(k \oplus \text{base} \oplus 2^j)$의 값을 비교해봅시다.
- ✓ 잘 생각해보면 $k \oplus \text{base}$의 i번째 비트와 j번째 비트가 다를 때 $f(k \oplus \text{base} \oplus 2^i) \neq f(k \oplus \text{base} \oplus 2^j)$가 됨을 알 수 있습니다.
- ✓ 따라서 $0 \leq i < N$인 모든 i에 대해 $f(k \oplus \text{base} \oplus 2^i)$의 값을 구한다면 서로 다른 두 비트가 같은지 다른지 알 수 있고 답의 후보는 2개로 줄어듭니다. 이 때 답의 후보를 a, b라고 한다면 $a = \sim b$가 됨을 알 수 있습니다.
- ✓ 이 때 $f(k \oplus x) = 1$인 x를 하나라도 알고 있다고 가정합시다. N이 3의 배수가 아닌 경우에는 $\text{BitCount}(k \oplus x)\%3 = 0$가 될것입니다. 이 때 $\text{BitCount}(\sim k \oplus x)\%3 \neq 0$가 되므로 $\text{BitCount}(a \oplus x)$와 $\text{BitCount}(b \oplus x)$의 값을 계산해 답을 특정할 수 있습니다.
- ✓ N이 3의 배수인 경우에도 비슷하게 할 수 있습니다.
- ✓ 만약 쿼리 중 $f(k \oplus x) = 1$인 x가 하나도 나오지 않았다면 $k \oplus \text{base}$의 모든 비트가 같다는 것이고 $f(k \oplus \text{base}) = f(k \oplus \text{base} \oplus 1) = 0$인 경우이므로 $k \oplus \text{base} = 2^N - 1$인 경우밖에 없습니다.

13화. 일반 쿼리가 구간 쿼리에 온라인 쿼리인 수열과 쿼리는 어떠신가요?

#segtree, #lazyprop

출제진 의도 - Master

✓ 출제자: ibm2006

✓ 제출: 19번, 정답: 4명 (정답률 21.053%), 처음 푼 사람: potato167, 176분

13화. 일반 쿼리가 구간 쿼리에 온라인 쿼리인 수열과 쿼리는 어떠신가요?

- ✓ 관찰: 주어진 구간에 c 이상 $2c-1$ 이하인 원소가 존재하거나, c 미만인 원소를 전부 더했을 때의 값이 c 이상이라면 답이 1이고, 아니라면 답이 0입니다.
- ✓ 이때 위 조건은 $2c-1$ 이하인 원소를 전부 더했을 때의 값이 c 이상인 것과 동치입니다.
- ✓ 마찬가지로 위 조건은 $2^k \le c < 2^{k+1} \le 2c$인 정수 k에 대해서 2^{k+1} 이상 $2c-1$ 이하인 원소가 존재하거나, 2^{k+1} 미만인 원소를 전부 더했을 때의 값이 c 이상인 것과 동치입니다. 2^{k+1} 이상 $2c-1$ 이하인 원소는 c 이상 $2c-1$ 이하이며, 이러한 원소가 없을 경우 $2c-1$ 이하의 모든 수의 합은 2^{k+1} 미만인 모든 수들의 합이 되기 때문입니다.

13화. 일반 쿼리가 구간 쿼리에 온라인 쿼리인 수열과 쿼리는 어떠신가요?

- ✓ $m = \log 1024 + 2$와 같이 두고, m개의 레이지 세그먼트 트리 $seg_0, seg_1, ..., seg_{m-1}$을 생각합시다. 각 세그먼트 트리들은 수열의 특정한 원소들의 합, 최솟값을 관리합니다.
- ✓ $1 \leq i \leq N$에 대해서 a_i가 $2^k \leq a_i < 2^{k+1}$이었다고 합시다. 그렇다면 seg_k으로 해당 원소를 다뤄줍니다.

13화. 일반 쿼리가 구간 쿼리에 온라인 쿼리인 수열과 쿼리는 어떠신가요?

- ✓ 주어진 쿼리가 갱신 쿼리라면, 앞서 말한 m개의 세그먼트 트리들 각각에 대해 구간 갱신 연산을 처리해주면, $\mathcal{O}(\log N \log 1024)$의 시간복잡도에 쿼리를 처리할 수 있습니다.
- ✓ 주어진 쿼리가 답을 구하는 쿼리인 케이스를 생각해봅시다. $2^k \leq c$인 최대의 k를 잡고, 주어진 구간의 $seg_0, ..., seg_k$까지의 합이 c 이상인지 체크합니다.
- ✓ 또한, seg_{k+1}의 최솟값이 $2c-1$ 이하인지 체크합니다.
- ✓ 해당 판정 중 하나라도 성립할 경우 쿼리의 답이 1이고, 그렇지 않다면 쿼리의 답이 0이 됩니다.

모두 수고하셨습니다!

아니메컵 제작위원회