AtCoder Beginner Contest 457 题目讲解

云浅

2026-5-11

Outline

A. Array

A. Array

略

B. Arrays

B. Arrays

略

C. Long Sequence

C. Long Sequence

略

D. Raise Minimum

D. Raise Minimum

给定一个长度为 N 的序列 $A=(A_1, A_2, \dots, A_N)$ 和一个整数 K。

你可以执行以下操作 **0 到 K 次（包含端点）**：

- 选择一个满足 $1 \le i \le N$ 的整数 i，并将 i 加到 A_i 上。

求操作后序列的 **最小值** $\min_{1\le i\le N} A_i$ 的最大可能值。

数据范围：

- $1 \le N \le 2 \times 10^5$
- $1 \le A_i \le 10^{18}$
- $1 \le K \le 10^{18}$

D. Raise Minimum

可以考虑二分答案 v，那么想要让 $A_i \ge v$，需要至少加 $\left\lceil \frac{\max(v-A_i,0)}{i} \right\rceil$ 次。

于是 v 合法当且仅当 $\sum_{i=1}^{N} \left\lceil \frac{\max(v-A_i,0)}{i} \right\rceil \le K$，那么可以 $O(N)$ 判定。

二分的上界可以设为 $A_1 + K$，那么总复杂度就是 $O(N \log(A + K))$。

E. Crossing Table Cloth

E. Crossing Table Cloth

有 N 个单元格排成一行。从左数第 i 个单元格（$1 \le i \le N$）称为单元格 i。

有 M 块布。铺设第 i 块布（$1 \le i \le M$）会覆盖单元格 L_i 到 R_i。

回答 Q 个查询。对于第 q 个查询（$1 \le q \le Q$），给定整数 S_q 和 T_q，请回答以下问题：

判断能否从 M 块布中**恰好选择两块布**并铺设，使得满足以下条件：

- 单元格 S_q 到 T_q 被至少一块布覆盖，
- 且没有其他单元格被任何布覆盖。

数据范围：

- $1 \le N, Q \le 2 \times 10^5, 2 \le M \le 2 \times 10^5$
- $1 \le L_i \le R_i \le N$
- $1 \le S_q \le T_q \le N$

E. Crossing Table Cloth

首先我们求出 $[S,T]$ 这个子区间内是否至少能放 2 块布，即是否至少有两个 $[L_i,R_i] \subseteq [S,T]$。这个可以简单做到 $O(N)$：扫描线扫 r，维护 l 最大的两个元素即可。

现在只需要判断能否从中取出至多 2 个覆盖 $[S,T]$，那么一定是选取 $L_i = S, R_i \leq T$ 且 R_i 最大的，以及 $R_i = T, L_i \geq S$ 且 L_i 最小的一个。

这个可以直接开 N 个 vector 然后每次二分，实际上可以做到线性：把询问全部离线下来，然后按顺序双指针维护就好了。

这样本题可以做到 $O(N+M+Q)$

F. Second Gap

F. Second Gap

给定一个整数 N 和一个长度为 $N-1$ 的整数序列 $D=(D_1, D_2, \ldots, D_{N-1})$。

求满足以下条件的 $(1, 2, \ldots, N)$ 的排列 $P=(P_1, P_2, \ldots, P_N)$ 的个数，结果对 998244353 取模。

对于每个 $1 \le i \le N-1$，设 P_a 和 P_b 分别是 $(P_i, P_{i+1}, \ldots, P_N)$ 中最大值和次大值所在位置的元素，则 $|a-b| = D_i$。

数据范围：

- $2 \le N \le 2 \times 10^5$
- $1 \le D_i \le N-i$

F. Second Gap

考虑从后往前 DP，维护元素的相对顺序。

设 $\mathrm{dp}[i][j]$ 表示把 1 到 $n-i+1$ 这些数填入 $[i,n]$ 中，满足所有要求且最大值位置为 j 的方案数。

加入现在考虑确定 $\mathrm{dp}[i+1]$ 转移到 $\mathrm{dp}[i]$，有 3 种情况:

- p_i 是新的最大值，原本的最大值位置是 $i+D_i$，有 $\mathrm{dp}[i][i] \leftarrow \mathrm{dp}[i+1][i+D_i]$
- p_i 是新的次大值，原本的最大值位置是 $i+D_i$，有 $\mathrm{dp}[i][i+D_i] \leftarrow \mathrm{dp}[i+1][i+D_i]$
- p_i 小于原本的次大值，那么什么都不会改变，这样的 p_i 有 $(n-i-1)$ 种填法，则 $\forall j$，我们做转移 $\mathrm{dp}[i][j] \leftarrow (n-i-1) \times \mathrm{dp}[i+1][j]$

这样复杂度是 $O(n^2)$ 的。

F. Second Gap

我们考虑 $\mathrm{dp}[i+1] \to \mathrm{dp}[i]$ 发生了什么变化。

发现只会对 $i, i+D_i$ 这两个位置做单点修改，然后做一个全局乘 $(n-i-1)$。

具体地，假设现在序列 $f = \mathrm{dp}[i+1]$，那么我们做以下的操作：

- 先记录下来原本的 $x = f_{i+D_i}$
- 然后全局乘 $(n-i-1)$
- 然后把 f_i, f_{i+D_i} 两个位置单点 $+x$

操作完之后，有 $f = \mathrm{dp}[i]$。

那么我们只需要维护单点加，全局乘，这个可以维护一个全局的乘法标记 mul，表示 f_i 的真实值是 $f_i \times mul$，然后如果要把 i 位置加上 v 我们就做 $f_i \leftarrow f_i + v \times mul^{-1}$。

这样复杂度就是 $O(n)$ 的。

这里可能需要求逆，以及有 $mul = 0$ 的无法求逆元的情况

对于前者，可以预处理 1 到 n 的逆元，是 $O(n)$ 的。

对于后者，只需要注意到每次只会把 2 个位置修改成非 0，因此直接维护所有的非 0 位置，$\times 0$ 的时候暴力清空即可。

G. Catch All Apples

G. Catch All Apples

有 N 个苹果落在一个数轴上。苹果 i 在时间 T_i 落在坐标 X_i 上。

你希望在数轴上放置一些机器人来收集所有 N 个苹果。机器人可以放置在任意坐标上。

每个机器人从时间 0 开始运行，并可以以最大速度为 1 沿数轴自由移动。多个机器人可以同时占据同一坐标。每个机器人只有在时间 T_i 恰好位于坐标 X_i 时才能收集苹果 i。

求收集所有苹果所需的最少机器人数量。

数据范围：

- $1 \leq N \leq 3 \times 10^5$
- $0 \leq T_i \leq 3 \times 10^5$
- $0 \leq X_i \leq 3 \times 10^5$
- $(T_i, X_i) \neq (T_j, X_j)$ 当 $i \neq j$

G. Catch All Apples

相当于把 $\{(X_1,T_1),(X_2,T_2),\cdots,(X_N,T_N)\}$ 分成最少个数的集合，使得每个集合内部的苹果都能被一个机器人收集。

考虑若干个苹果 $(x_1,t_1),\cdots,(x_k,t_k)$ 能被一个机器人收集的条件，按照 t 排序，发现只需要相邻两个 $(x_i,t_i),(x_{i+1},t_{i+1})$ 都满足 $t_{i+1}-t_i \geq |x_{i+1}-x_i|$。

那么如果两个苹果 $i<j$ 满足 $T_j-T_i \geq |X_j-X_i|$，就连一条有向边 $i \to j$。那么会得到一个 DAG，我们需要求出这个 DAG 的最小链划分。

G. Catch All Apples

Dilworth 定理：最小链划分 = 最长反链

- 一个反链指的是一个集合 S，满足 S 中任意两个不同的元素都互相不可达。

证明:

- 显然如果能选出大小为 k 的反链，那么最小链划分肯定 $\geq k$
- 另一方面，我们可以给出一个大小为 k 的构造：每次从最长反链递归到它的前后两部分，然后在中间拼接起来。如果某一部分为空，说明这个反链全都是 0 入度或者全都是 0 出度，那么我们取一个 0 入度的点 x 和一个 0 出度的点 y，任取一条 $x \to y$ 的路径删掉，就可以让最大反链大小 -1，然后接着递归下去构造就好了。

G. Catch All Apples

那么对于本题，只需要注意到 $T_j - T_i \ge |X_j - X_i|$ 这个条件可以把绝对值拆开，得到

- $X_j - T_j \le X_i - T_i$
- $X_j + T_j \le X_i + T_i$

于是就变成了二位偏序，求 LDS 长度即可。

如果要构造方案，注意到这个题的最长反链是 LIS 的形式，所以有更简单的构造：直接求出 f_i 表示以 i 结尾的最小 LIS，然后把 f_i 相同的分为一类即可。