yLOI2023 讲评

一扶苏一
山东大学泰山学堂
上海洛谷网络科技有限公司

洛谷
www.luogu.com.cn

目录

前言

感谢大家参加 yLOI2023。

感谢赞助商常菁数编程，你可以在比赛描述界面找到他们的信息。

恭喜 uwagjaynoi AK！

目录

分解只因数

描述

对一个正整数 n，如果它只含奇质因子，则称它是『只因数』。
有 T 组数据，每次给定 n，请判定 n 是不是只因数。
$T \leq 9$，$2 \leq n \leq 10^{18}$。
难度：红。

分解只因数
灵感来源

CF1775D By [redacted] 博客内查看 | 后台编辑

对应题目：CF1775D Friendly Spiders

提交时间: 2023-01-11 08:30:03 该题目下已有 **2** 篇有效题解。

用 m 表示图的边数

对每个 a_i 分解只因数。

分解只因数
算法 -1

~~根据日常生活经验，『只因』二字在连读时常被空耳成『ji』音。所以合理怀疑只因数就是奇数。~~

分解只因数
算法 1

显然 2 不是只因数，3 是只因数。

类似地，当 $n \leq 10$ 时可以手算所有的数是不是只因数。期望得分 50 分。

分解只因数
算法 2

$n \leq 10^9$ 时，可以暴力分解质因数，然后依题意判断。时间复杂度 $O(T\sqrt{n})$。期望得分 90 分。

分解只因数
算法 3

质数除了 2 以外均为奇数。

所以一个数只含奇质因子等价于这个数不含 2 这个因子。

显然：含有因子 2 的数一定是偶数，不含因子 2 的数一定是奇数。

于是直接判断 n 的奇偶性，奇数是只因数，偶数不是只因数。

时间复杂度 $O(T)$，期望得分 100 分。需要开 long long。

目录

苦竹林
描述

给定一个数列 a，找到最小的 ε，使得 a 存在一个长度为 m 的子数列（可以不连续）b，满足对任何的 $1 \le i, j \le m$，都有 $|b_i - b_j| \le \varepsilon$。

$2 \le m \le n \le 10^5$，$1 \le a_i \le 10^9$。

难度：橙。

苦竹林
算法 1

当 $m = 2$ 时，只要找数列里差值最小的两个数；$m = n$ 时，只要找数列里最大值减掉最小值。期望得分 20 分。

苦竹林
算法 2

枚举所有选 b 数列的方案，共 $\binom{n}{m}$ 个。这里枚举可以用二进制枚举，也可以搜索。

对应的 ε 就是该方案里 b 的最大值减去最小值，可以 $O(m)$ 算出。总时间复杂度 $O(m \times \binom{n}{m})$。期望得分 40 分。

苦竹林
算法 3

当 a 有序时，容易发现答案对应的 b 数列一定是数列里一个连续的子段。

于是 $O(n)$ 枚举 a 里每个长度为 m 的子区间，找到子区间的最大值和最小值，可以算出此时的 ε 并与当前算出的答案作比较。

时间复杂度 $O(n)$ 或 $O(n^2)$，期望得分 60 分。

苦竹林
算法 4

注意到 a 的顺序其实并不影响答案。所以 a 无序时只要排个序就可以转化成算法 3 的问题了。

考虑怎么找区间的最大值和最小值：如果暴力扫一遍找最大最小值，总时间复杂度 $O(n^2)$，期望得分 80 分。

注意到 a 已经有序了，所以最小值就是左端点，最大值就是右端点。所以对每个子区间可以 $O(1)$ 求最值。时间复杂度 $O(n \log n)$，瓶颈在排序，期望得分 100 分。

目录

云梦谣

描述

给定一个 $n \times m$ 的方格阵，每个格子有一个高度 $h_{i,j}$，或者是不能通行的障碍物。且有 k 个格子可以传送。每秒可以做如下三件事之一：

- 移动到相邻四联通的格子。
- 如果当前格子允许传送，则可以传送到任意别的允许传送的格子上，条件是目标格子和当前格子等高。
- 改变当前格子的高度为任意正整数。

求从 $(1,1)$ 走到 (n,m) 的最短用时。

$1 \le n, m \le 3 \times 10^3$。$1 \le k \le n \times m$。

难度：黄。

云梦谣
算法 1

当 $n = m = 4$ 时有一万种方法求解，不表。可得 20 分。
特别的，测试点 1 输出 -1 可得 5 分。

云梦谣
算法 2

当 $k = 0$ 时，操作二和三都没有意义。这就是一个简单的 bfs 走迷宫问题。

时间复杂度 $O(nm)$，期望得分 30 分。结合算法一可得 40 分。

云梦谣
算法 3

当 k 比较小且 $h \leq 1$ 时，不需要操作 3。
于是可以进行这样的 bfs：

- **在非传送格子上正常进行四联通 bfs。**
- **在传送格子上时，除了进行四联通 bfs，还枚举所有的其它的传送阵，尝试传送到其它格子上去。**

一共有 $O(nm)$ 个状态，在转移时需要 $O(k)$ 的时间枚举其它的传送阵。总时间复杂度为 $O(nmk)$。期望得分 15 分。结合算法一、二可得 55 分。

云梦谣
算法 4

注意到操作 3 只会在传送之前一秒进行，可以把这两个操作绑定。

注意到操作 2 和 3 一起做需要两秒。为了不破坏 bfs 时『每步时间增加 1』的性质，可以设 $dis_{x,y,0/1}$ 表示走到 (x,y) 格子，且该格子的高度没有改变/刚刚把该格子的高度改编成其他任意正整数的最短用时。

此时的转移是：

1. **正常的四联通转移。**
2. $dis_{x,y,0}$ **转移到** $dis_{x,y,1}$**，表示这一秒改了格子的高度。**
3. **从** $dis_{x,y,0}$ **转移到其他高度相同的传送阵。**
4. **从** $dis_{x,y,1}$ **转移到其他高度不同的传送阵。**

仍然有 $O(nm)$ 个状态，在转移时需要 $O(k)$ 的时间枚举其它的传送阵。总时间复杂度为 $O(nmk)$。期望得分 70 分。

云梦谣
算法 5

本题的 key conclusion 是：传送至多会使用一次，且一定是离起点最近的传送阵传送到离终点最近的传送阵（无论他们的高度是否一样，当然有高度相同的优先用高度相同的）。

云梦谣
算法 5

从 $(1,1)$ 开始 bfs 出起点距所有点的距离，找出离起点最近的所有传送阵；然后从终点再做一次 bfs，同样找出离终点最近的传送阵。

检查离起点最近的阵中和里终点最近的阵中有没有等高的。如果有，则答案就是直接走过去和从起点走到传送阵传送并走到终点的时间取最小值；如果没有，把后者的时间加一取最小值。

$h \leq 1$ 时，无需检查等高，算法时间复杂度为 $O(nm)$，期望得分 65 分。

$h > 1$ 但 k 比较小时，可以 $O(k^2)$ 枚举所有传送阵对作比较。时间复杂度 $O(nmk^2)$。与 $h \leq 1$ 的情况一起共期望得分 85 分。

云梦谣
算法 6

称离起点或终点距离最近的传送阵为『有效传送阵』。

在造数据的时候发现无法造出 $O(nm)$ 个有效传送阵的数据。有一个符合直觉的猜测时有效传送阵的个数只有 $O(n+m)$ 个，但是我无法证明。事实上数据里有效传送阵确实只有 $O(n+m)$ 个。

于是暴力枚举传送阵对检查高度的时间复杂度其实是 $O(n^2)$ 的（认为 n, m 同阶）。设有效传送阵有 t 个，则算法时间复杂度为 $O(nm+t^2)$，可以得到 100 分。

云梦谣
算法 7

事实上存在 $O(k)$ 的检查高度方法：

先扫一遍离起点最近的传送阵，用一个桶记录这些传送阵的高度（$c_x = 1$ 表示离起点最近的传送阵中有一个高度为 x 的）。

然后扫一遍离终点最近的传送阵。对每个传送阵看它的高度在桶里是否出现。如果出现则表示找到了一对同高度的传送阵。

这样就可以做到 $O(nm + k)$ 了。期望得分 100 分。

目录

腐草为萤

描述

数轴上有 n 个点，每个点的初始坐标为 x_i，权重为 a_i。在任何时刻，每个点会向与它相邻的两个点中权重较大的那个点移动，速度为一个单位长度每秒，如果相邻两个点的权重都比它自身权重小，则不移动。两个点相遇时，权重较小的点消失。

求出每个点消失时的坐标。

$1 \leq n \leq 5 \times 10^5$，$1 \leq x_i \leq 10^9$，$a$ 是长度为 n 的排列。

难度：绿 $+$/蓝-（考虑了代码难度）

腐草为萤
算法 1

$n = 2$ 时只要比较两个点谁大即可。期望得分 5 分。

当 $n \leq 100$，$x_i \leq 200$ 时，显然每隔 200 秒至少会有一个点消失。于是逐秒模拟即可。时间复杂度 $O(n^2 x_i)$，期望得分 30 分。

腐草为萤
算法 2

找到离得最近的且距离在缩小的点对 (i,j)。显然它们碰撞前不会有其他点碰撞。所以可以直接把时间跳到 (i,j) 碰撞的时刻，其他点在这段时间内运动方向不变，暴力算出所有点的坐标。

每次修改 $O(n)$ 个坐标，共有 $O(n)$ 次碰撞。时间复杂度 $O(n^2)$，期望得分 60 分。

腐草为萤
算法 3

a_i 单调递增的情况：除了最后一个点，所有的点都向右移动。因为点的移动速度是一样的，所以移动方向相同的两个点的间距是不会变的，不会发生碰撞。大家最后都会和 n 号点碰撞，然后去世。

a_i 单峰的情况：类似的，所有点都会朝着权值最大的点移动，且同向的两个点间距不变。最后大家都和权重最大的点碰撞，去世。

两种情况分别占 5 分，结合算法 2 可得 70 分。

腐草为萤
算法 4

结合算法三、四、五，我们尝试分析运动的形式，得出 key conclusion：

1. 相邻两个同向运动的点（在一方方向改变前）的间距不会改变。
2. 发生一次碰撞后，至多只有两个点的运动方向会改变。
3. 只有运动方向不同的两个相邻点（把静止也看做一个运动方向）会发生碰撞。

根据第三条，考虑维护处所有运动方向不同的相邻点对。在一次碰撞后，根据第二条，只需要进行 $O(1)$ 次修改就能维护新状态。

腐草为萤
算法 4

每次碰撞是找间距最小的两个点对，于是需要一个支持插入、删除、查找最小值的数据结构。

可以用 set 或 priority_queue 维护。

priority_queue 本身不支持删除，但是可以参考 dijkstra 堆优化的做法用时间戳的形式改为可删除的堆。简单起见，下面不妨约定使用 set。

腐草为萤
算法 4

我们需要能找出这些点对里当前间距最小的，这个点对会最先发生碰撞。这里产生了一个问题：每个点对的间距是在加入数据结构的时刻计算的。当时间增加时，不可能暴力修改数据结构里点对的距离。那么如何找出当前间距最小的点对呢？

方法是维护所有点对在 0 时刻的等效间距。

在时刻 T，间距 d 的点对的等效间距为 $T + d$。

腐草为萤
算法 4

最后一个问题是，在插入数据结构时，要求出两点当前的间距。这要求我们能快速求出每个点的坐标。

考虑懒更新，仅当一次碰撞发生后，更新与消失的点相邻的点的坐标。

记 $lastChange_i$ 表示上次更新 i 的坐标的时刻，上次更新后 i 的坐标是 x_i，当前时刻为 T，则 i 当前的坐标就是 $x_i + (T - lastChange_i) \times aspect_i$。其中，当 i 向左移动时，$aspect_i = -1$，i 向右移动时，$aspect_i = 1$，静止不动时 $aspect_i = 0$。这是因为两次更新之间 i 的运动方向不可能变化。这其实也是能使用等效间距维护最小间距的原因。

腐草为萤
算法 4

此外，为了能求出当前与某个点相邻的点，需要按顺序维护当前还存活的点，支持查询上一个、下一个。这部分可以使用链表完成，std 图省事使用了 std::set。

总的来说，一共发生了 $O(n)$ 次碰撞，每次碰撞对数据结构有 $O(1)$ 次修改，单次修改复杂度 $O(\log n)$。于是算法的总时间复杂度为 $O(n \log n)$。期望得分 100 分。

因为需要讨论萤火虫的朝向，所以代码不算非常好写，std 写了 79 行。但是良心的我把对拍时排出的错误数据都加到了样例里。本题一共给了 7 个样例，所以让我们一起赞美良心出题人吧。