【题解】2023 牛客 NOIP 赛前集训营-提高组（第一场）

T1-情景剧

对于 $subtask1$：

- 可以 $O(n^2)$ 暴力枚举每个区间。

对于 $subtask2-3$：

- 以防万一大家是 $O(n\sqrt{n}), O(n\ log\ n)$ 之类的解法。

- 以防万一大家开的 `int` 或者 `long long`。

对于 $subtask4\ \ O(n)$：

- 虽然大家应该基本上会但是还是多啰嗦几句 ?

- 因为要求的是某个达到最值的区间，一个思路是看拿个什么数据结构维护下，能不能得到所有区间的价值，另一个思路是考虑枚举其中一个值或两个值，然后贪心地让乘积最大。

- 这道题我们选择枚举最小值，因为最大值随区间扩大是不降的，所以在枚举最小值的情况下，一定是选取枚举值为最小值的最大区间，这样就只有 n 个可能为答案的区间。

- 对于每个值为最小值的区间，一个常见的方式是拿单调栈维护，然后考虑怎么得到区间的最大值，硬做的话似乎是没有不带 log 的做法 ? 常数比较小的应该是分治。

- 其实这题可以上笛卡尔树，即利用所有 n 个区间是构成一棵树的结构这一特点，所以区间 max 直接从两个子区间取 max 就好了。

- 一个小问题是出题人不知道有没有枚举最小值以外的做法 QaQ。

T2-抽卡

为了方便，下面用 1 表示奇数，0 表示偶数，空段表示连续一段待填位置。

对于 subtask1 $O(q2^n)$：

- 枚举每个位置的奇偶性。

对于 subtask2 $O(qn^2)$：

- 设 $f_{i,j,1/0}$ 表示考虑前 i 位，第 i 位是奇数/偶数，已经用了 j 个奇数的最小代价。

- 转移就枚举一下当前这一位填奇数还是偶数就好了。

对于 subtask3 $O(qn)$：

- 首先您可以想象一下去放数字的过程，然后您就会突然意识到，如果一个空段的左右分别是 0 和 1，中间填任意数量的 0,1，最优也至少要付出 1 的代价，即 0 这边贴着一段 0,1 那边贴着一段 1。

- 于是现在就只需要考虑左右两边数奇偶性相同的空段了。然后一个比较显然的策略就是直接贪，将空段按长度排序，然后从最短的开始，能整个空段填成一样就填。

- 需要注意的是因为中间的空段填充失败会贡献 2 的代价，而如果左右最边上的空段填充失败只会贡献 1 的代价，所以需要稍微讨论下左右要不要贪，还是直接不合法。

对于 subtask4 $O(qlog\, n\times (2\times 4)^3)$：

- 大概是为了给没发现贪心规律的同学多一档分。

- 因为奇数的数量很少，所以可以用矩阵优化 subtask2 的 dp。

对于 subtask5 $O(q\ log\ n)$：

- 其实贪心的部分并不复杂，即对一个从小到大排序数组 a_i (即空段的长度)，找到最大的 k 使得 $\sum_{i=1}^{k} a_i \leq lim$，并且支持动态修改单个的 a_i，可以用一棵权值线段树来实现。不过 std 用的是 BIT。

- 至于删填数对空 段长度造成的变化可以拿一个 set 维护。

T3-修改 01 序列

智力测试

枚举最终形态就可以了，因为所有 1 之间的距离都是 d 的倍数，说明所有 1 所在的位置对 d 求余得到的值相等。那么我们就枚举这个求余得到的值，假设最终所有的 1 的位置对 d 求余得到的结果都是 2，那么我们需要修改的次数就是所有位置对 d 求余不是 2 的数字 1，这个可以预处理出来（也就是预处理出每个 1 的位置对 d 求余的结果，全部加起来，方便最后计算）。

```
#include<bits/stdc++.h>

using namespace std ;

int main() {

    int n , d;

    cin >> n >> d;

    vector<int> a(n + 1, 0);

    for(int i = 1; i <= n; i++)  cin >> a[i];

    vector<vector<int>> cnt(d, vector<int>(2, 0));

    for(int i = 1 ; i <= n; i++)

        cnt[i % d][a[i]] += 1;

    int mn = n;

    int one = count(a.begin() + 1, a.end(), 1);

    for(int i = 0; i < d; i++)

        mn = min(mn, one - cnt[i][1]);

    cout << mn << '\n';

}
```

T4-虚树

合对于测试点 1-2

每次询问直接暴力枚举选择哪些点，$O(Q \times C_n^k)$ 。

对于测试点 9-10

每次只能选两个点，那么答案应该是区间直径，因为区间直径支持合并，用线段树维护即可

对于测试点 13-14

这些测试点的树是菊花，那么选择的虚树肯定也是菊花，也就是说，每次答案是区间前大的边权之和，可以用主席树实现.

对于测试点 3-6

考虑在较低的时间复杂度内求出区间的 k 点最大虚树。考虑当 $k = 2$ 的时候答案就是区间点集直径，那么当 $k > 2$ 的时候就是把直径找出来，然后以直径的一个端点为根做长链剖分，每次选最长的链即可。时间复杂度

$O(n\, log\, n + nQlog\, n + Qk\backslash og\, k)$

如果每次把区间的虚树建立出来，那么询问和区间长度有关，可以通过 17-18 号测试点。

如果预处理一下区间$[1,n]$ 的答案可以通过 15-16 号测试点。

对于测试点 7-8

考虑快速维护区间的答案。可以使用线段树维护一个区间的答案，由于 $k <$ 100，每个区间我们只要保留不超过 k 个点，那么可以每次建立虚树暴力合并点集 。 实测在不特意卡常数的实现下可以获得 60 分的好成绩。 由于时限较为宽松， 如果实现优秀， 如对线段树进行底层分块并且使用较快速的方式实现虚树， 可能可以通过所有测试点。

对于所有测试点：

考虑优化求解过程。我们发现查询复杂度较高，考虑使用 ST 表减少查询的次数。但是这样预处理的复杂度会变成 $O(nk\, log\, n\, log\, k)$不能接受。

考虑将序列每 k个元素分为一块，块间建立 ST 表，那么预处理的复杂度就是 $O(n \log n \log k)$ 。

另外本题还存在一些离线做法，比如可以回滚莫队然后均摊一下端点移动和查询的复杂度，应该也可以有一个不错的复杂度，可以通过离线数据。