正睿 OI 省队选拔 2020 模拟赛

diamond_duke

题目名称	游戏	石子	划分
可执行文件名	game	stone	divide
输入文件名	标准输入	标准输入	标准输入
输出文件名	标准输出	标准输出	标准输出
时间限制	1s	2s	1s
内存限制	512MB	512MB	512MB
子任务个数	5	10	6
题目类型	传统型	传统型	传统型

请注意： 评测时开启 O2 优化和 C++11 编译选项，栈空间限制同空间限制。

1 游戏

考虑被删掉的位置在初始时的位置，设为 $p_1 < p_2 < p_3 < \cdots < p_k$。

可以发现，若序列 $\{p\}$ 固定，则我们一定将它们先移动到靠近 $p_{\lfloor \frac{k}{2} \rfloor}$ 的位置。因此，考虑枚举这个位置 t，然后分别确定 p 的前一半和后一半。

不妨仅考虑 p 的前一半。可以发现，总代价应当为所有选择的位置移动到 t 需要的代价，再去掉 $\frac{1}{2}\lfloor \frac{k}{2} \rfloor(\lfloor \frac{k}{2} \rfloor - 1) \cdot w_s$。

因此，我们的问题变为在之前选择 $\lfloor \frac{k}{2} \rfloor$ 个数字，以最大化节约的魔法值。我们相当于有两种操作：所有元素的代价加上 w_s，以及新增一个元素。这两者都可以通过带懒标记的堆很轻松地完成，直接维护即可。

时间复杂度：$\Theta(n \log_2 n)$。

2 石子

设最终一次操作的操作者是小 A，而另外一个人是小 B。

可以发现，除了最后一次操作外，双方都不会选择移动位置，即 x 必定是 l,r 之一，因为移动棋子是毫无意义的。

考虑小 A 的最后一次操作，其一定选择剩余的两个石子之间，所有数字的最大值或最小值。因此，对于小 A 而言，最终这个范围越大越好，因此其一定不会删去第一个或最后一个石子。反之，考虑小 B 的操作，其目的与小 A 恰恰相反，故他一定会删去第一个或最后一个石子。

由此，我们可以得到一个简单的算法：设小 A 的操作次数为 t，则小 B 的操作次数为 $k-t$。

不妨设小 A 的目标是最大化结果，则最终结果即要任意相邻的 $t+1$ 个石子之间，所有数字最大值的最小值：因为小 B 可以决定小 A 最后选择的范围是哪 $t+1$ 个，故一定会选择使得结果最小的那个区间。

考虑优化这一过程。考虑直接二分答案，然后使用线段树维护所有石子的位置。注意到 $t>k-t$，故这 $t+1$ 个石子一定跨过所有石子的中点。

考虑线段树上二分，即可求出从中点出发，向左以及向右各可以延伸多远，然后数一下其中的石子个数即可判断是否合法。

时间复杂度：$\Theta(q\log_2^2 n)$，可以将二分答案与线段树二分合并达到时间复杂度 $\Theta(q\log_2 n)$。

3 划分

首先考虑 a_i 互不相同的情况。

定理 3.1. *任给 $1 \le k \le n$，存在一个最优方案中所有组均不为空。*

证明. 反证。若有，则必有一个组含有至少两个数字。设其中某个数字为 a，而该组中剩余数字的 gcd 为 b，注意到 $\gcd(a,b) \le a < a+b$，因此将 a 单独分为一组一定更优。这与方案最优性矛盾。 □

定理 3.2. *任给 $1 \le k \le n$，存在一个最优方案中有至少 $k-1$ 个组有且仅有一个数。*

证明. 反证。考虑两个包含至少两个数字的组 S_1, S_2，设它们的 gcd 分别为 a, b。不妨设 $a < b$，则因为所有 a_i 互不相同，故 S_2 中，最大的数字至少为 $2b$。则将其单独分一组，将 $S_1 \cup S_2$ 中其余所有数字分在另一组，则 gcd 之和至少为 $2b+1 > a+b$。这与方案最优性矛盾。 □

定理 3.3. *任给 $1 \le k \le n$，最大的 k 个数属于的组互不相同。*

证明. 反证。若其中 $x < y$ 属于同一组，设 $x = ag$，$y = bg$，其中 $a \perp b$ 且 $a < b$。根据定理 3.2，我们知道其余组均有且仅有一个元素。考虑其中某个并非最大的 k 个数的元素 c，则此时这两组的 gcd 之和不超过 $g+c$。若将 y 单独分为一组，而将原组的其余元素与 c 分为一组，则 gcd 之和至少为 $y+1 = bg+1 > g+ag > g+c$。这与方案最优性矛盾。 □

由此可得此时的做法：我们考虑枚举 k，则此时前 $n-k$ 个数字必定处于同一个组中。考虑维护它们的 gcd，注意到该 gcd 的取值仅会变化 $\Theta(\log_2 a_i)$ 次，因此我们可以每当其变化时暴力地求出一段 k 的答案，总时间复杂度即为 $\Theta(n \log_2 n \log_2 a_i)$。

然后考虑重复的情况。可以发现，所有重复部分（即每个数字去掉第一次出现）可以被分为两种：单独一组的，以及放入之前某组的，因为两个重复部分分到一组显然不如其中一个一组，而另一个放入之前的组更优。

单独一组的贡献即为这些数字之和，而放入之前某组的显然不会使那些组的 gcd 变大，因此一定与它们的第一次出现放在一组中，故 gcd 不变。

因此，问题变为：我们可以选择一些额外数字，将它们与原来的组的 gcd 相加，用于更新答案。容易发现，我们选择的这样的数字一定是最大的那些。设原答案为序列 a，额外数字从大到小排序后的前缀和为 b，则我们即要求：$c_k = \max\limits_{i=1}^{k}\{a_i + b_{k-i}\}$。

容易发现 b 是凸的，因此我们可以使用决策单调性进行优化。总时间复杂度仍为 $\Theta(n \log_2 n \log_2 a_i)$。

注： 事实上，a 也是凸的，因此可以使用闵科夫斯基和合并，时间复杂度不变。